CMR UNIVERSITY

Private University Established in Karnataka State by Act No. 45 of 2013

SCHOOL OF ENGINEERING AND TECHNOLOGY

Case Study

On

SNIPIFY: BLOGGING WEBSITE

For the requirements of the 6th semester

(49CSPL3121-Web technology framework)

Bachelor of Technology

Computer Science Engineering

Of CMR University, Bangalore

Submitted by:

ELIGINTI AJAY KUMAR RED(21BBTCS081)

GOWTHAM N(21BBTCS089)

ITHA VENKATA SAIRAM(21BBTCS098)

Submitted to

Prof.Chethana R M

Assistant Professor,

Dept of CSE, CMR University

Department of COMPUTER SCIENCE AND ENGINEERING

Off Hennur - Bagalur Main Road,
Near Kempegowda International Airport,
Chagalahatti, Bangalore, Karnataka-562149

Academic Year- 2023-24

Table of Contents

Project Proposal:

Abstract:

This project aims to create a web platform that replicates the core features and functionalities of a popular online publishing and blogging platform. This platform will provide users with the ability to create, publish, and engage with articles on various topics. It comes with user-friendly interface and the project seeks to offer a familiar experience to both content creators and readers.

Plan Of Work And Product Ownership:

1. User Authentication and Profiles:

- Users can sign up, log in, and manage their profiles.

- Profile customization options, including profile pictures, bios.

2. Article Creation and Publishing:

- Users can create and edit articles using a rich text editor.

- Support for various formatting options, including text styles, headings, images, and embedded media.

- Option to save articles as drafts and publish them when ready.

3. Engagement and Interaction: (DARSHAN JS)

- Readers can like, comment on, and share articles.

- Authors receive notifications about interactions on their articles.

4. Search Functionality: (DARSHAN JS)

- Provide a search bar for users to find articles by keywords, tags, or authors.

5. User Dashboard: (DARSHAN V)

- Users have access to a personalized dashboard displaying their articles, drafts, statistics (e.g., views, likes), and notifications.

Snipify aims to deliver a feature-rich platform that replicates the essence of Medium while also introducing potential enhancements. By prioritizing user experience, engaging features, and a robust recommendation system, the platform intends to attract both content creators and avid readers.

PROJECT PLAN:

1. Project Lifecycle:

Lifecycle Model: Agile (Scrum)

Justification: Agile allows for iterative development, which is well-suited for web applications like a blog app. It enables flexibility in responding to changing requirements and provides frequent opportunities for user feedback.

- **Quality Control:** We can concentrate on developing and testing a few tiny components at a time, making sure that each one functions properly before moving on to the next. This might lead to a more trustworthy end result.

- **User feedback:** Even in the absence of numerous stakeholders, we may use user feedback to enhance the usability and functionality of the Blog app. So we can take user feedback into account as you create using the iterative paradigm.

- **Early Progress:** While developing, the team will be inspired and helped by being able to monitor progress and have a functioning version of the Blog app rather quickly.

2. Tools:

Flexibility: When we are unsure of all the project specifics up front, the iterative model's ability to adapt and make modifications to your ATM simulator as you go along can be useful.

Design Tool: Figma for wireframing and UI design.

Version Control: Git and GitHub for source code management.

Development Tool: Visual Studio Code for coding and development.

Testing Tool: Jest for unit testing.

3. Deliverables:

Reusable Components

User Authentication Module:

Description: This component is responsible for managing user authentication and authorization. It allows users to sign up, log in, reset passwords, and manage their accounts securely.

Key Features: User registration, login, password reset, roles and permissions management, and secure authentication methods (e.g., OAuth, JWT).

Commenting System Module:

Description: The commenting system enables users to interact with articles or other content on the platform by leaving comments. It can include features such as nested comments, upvoting/downvoting, and moderation tools.

Key Features: Comment creation, editing, deletion, threaded/nested comments, user notifications, and moderation capabilities.

User Profile Management Module:

Description: This module allows users to create and manage their profiles. Users can update their personal information, upload avatars, and set preferences.

Key Features: Profile creation and editing, avatar upload, privacy settings, and user preferences customization.

Build Components Backend Server:

Description: The backend server is the core of your application. It handles data storage, retrieval, and business logic. It exposes APIs for the frontend to communicate with and is responsible for processing requests and serving data to the client.

Key Features: database integration, server-side business logic, security features, and scalability.

Frontend User Interface:

Description: The frontend is the user-facing part of your application. It's responsible for rendering the user interface and interacting with the backend server to display data and enable user interactions.

Key Features: Responsive design, user-friendly UI, interactive components, user registration and login screens, and integration with the backend APIs.

Article Creation and Management Module:

Description: This module allows users to create, edit, and manage articles or content on your platform. It includes features like rich text editors, version control, and publication settings.

Key Features: Article creation, editing, version history, scheduling, drafts, and content management.

Search Functionality:

Description: Search functionality enables users to find specific articles or content quickly. It should provide relevant search results, including filters and sorting options.

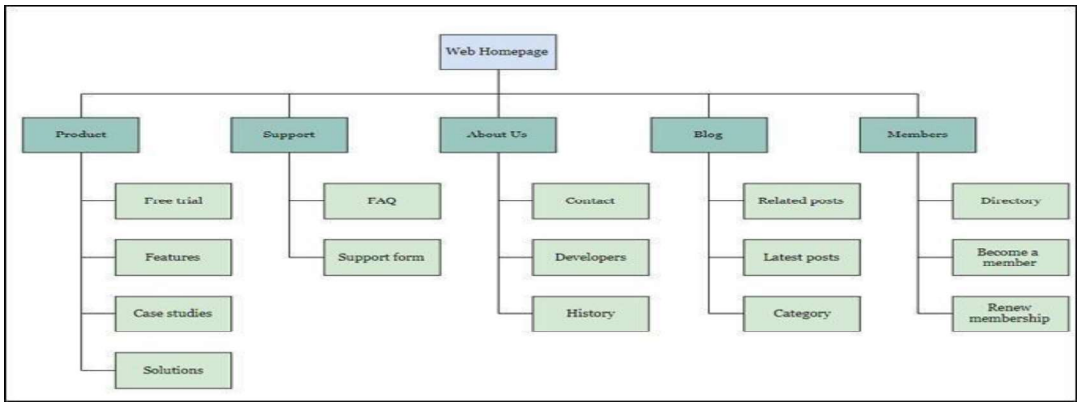
Key Features: Keyword-based search, filters (e.g., date, category), sorting options, and search result ranking.

Documentation and User Guides:

Description: This is essential for helping users, administrators, and developers understand how to use and maintain the system. Documentation includes user guides, installation guides, API documentation, and system architecture explanations.

Key Features: Clear and comprehensive documentation, including how-to guides, troubleshooting guides, and code documentation for developers.

Work Breakdown Structure (WBS):



EFFORT ESTIMATION

Using Cocomo model (Organic),

Effort = α × (KLOC) ^β

Time = γ × (Effort) ^ δ

where α, β, γ, and δ are constants with values 2.4, 1.05, 2.5 and

0.38

KLOC > Kilo Lines of Code

With 10001500 expected lines of code we can say that,

Effort = 2.4 × (1 or 1.5)

1.05

= 2.4 to 3.6737 Person months

Time = 2.5 × (2.4 or 3.6737)

0.38

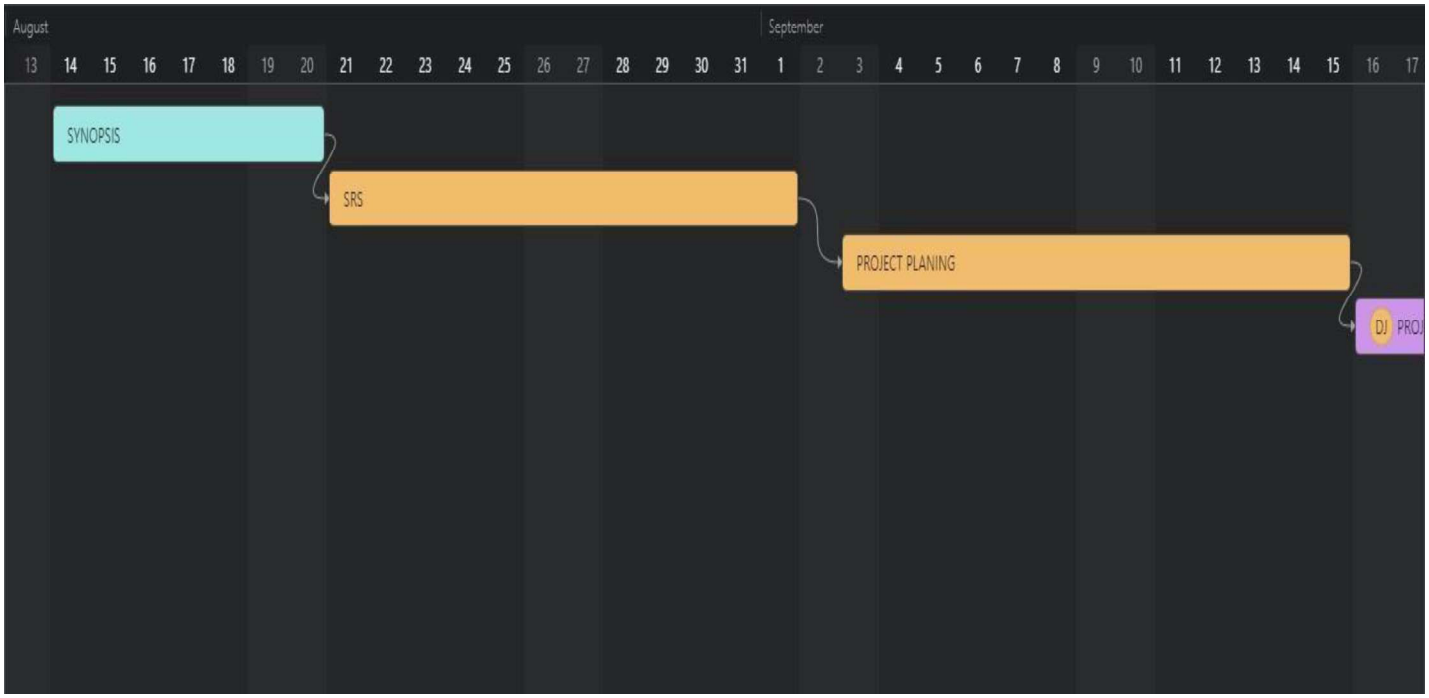
= 3.4867 or 4.099 months

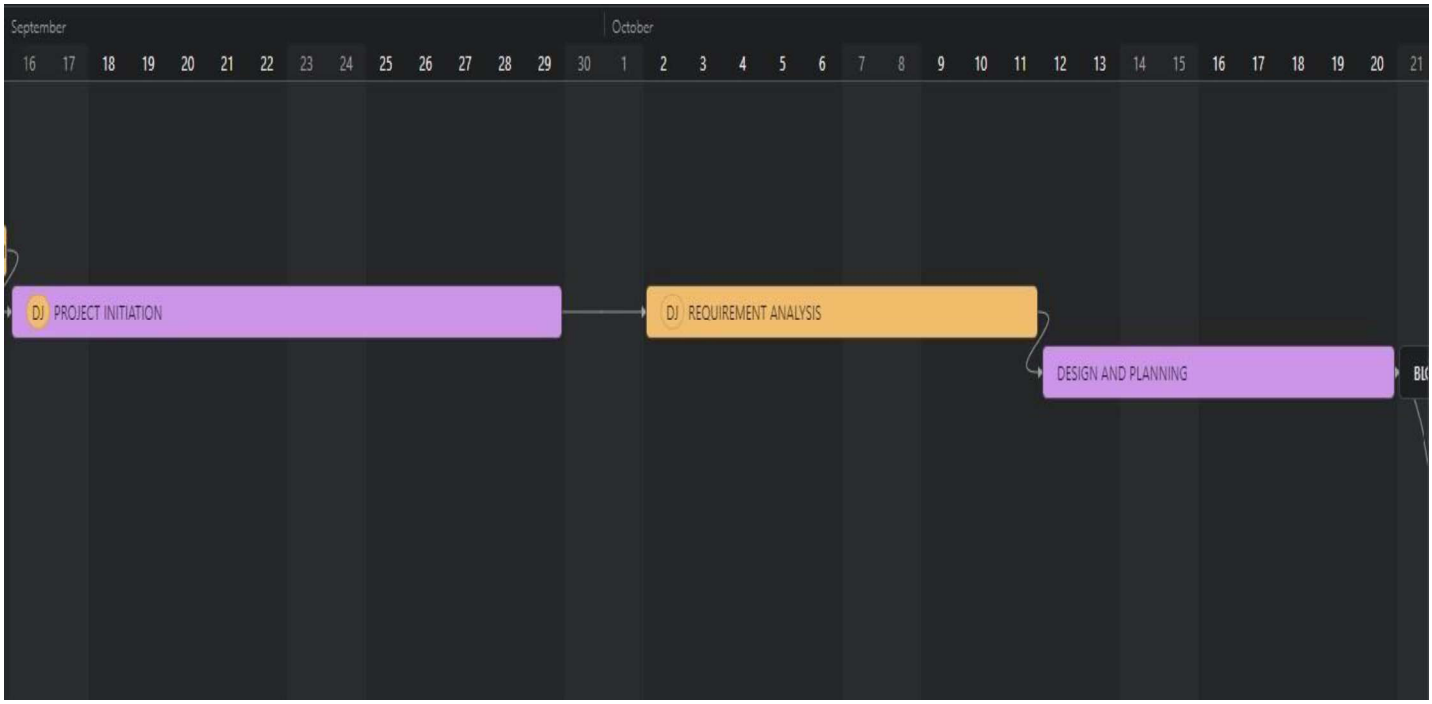
With 4 people in team,

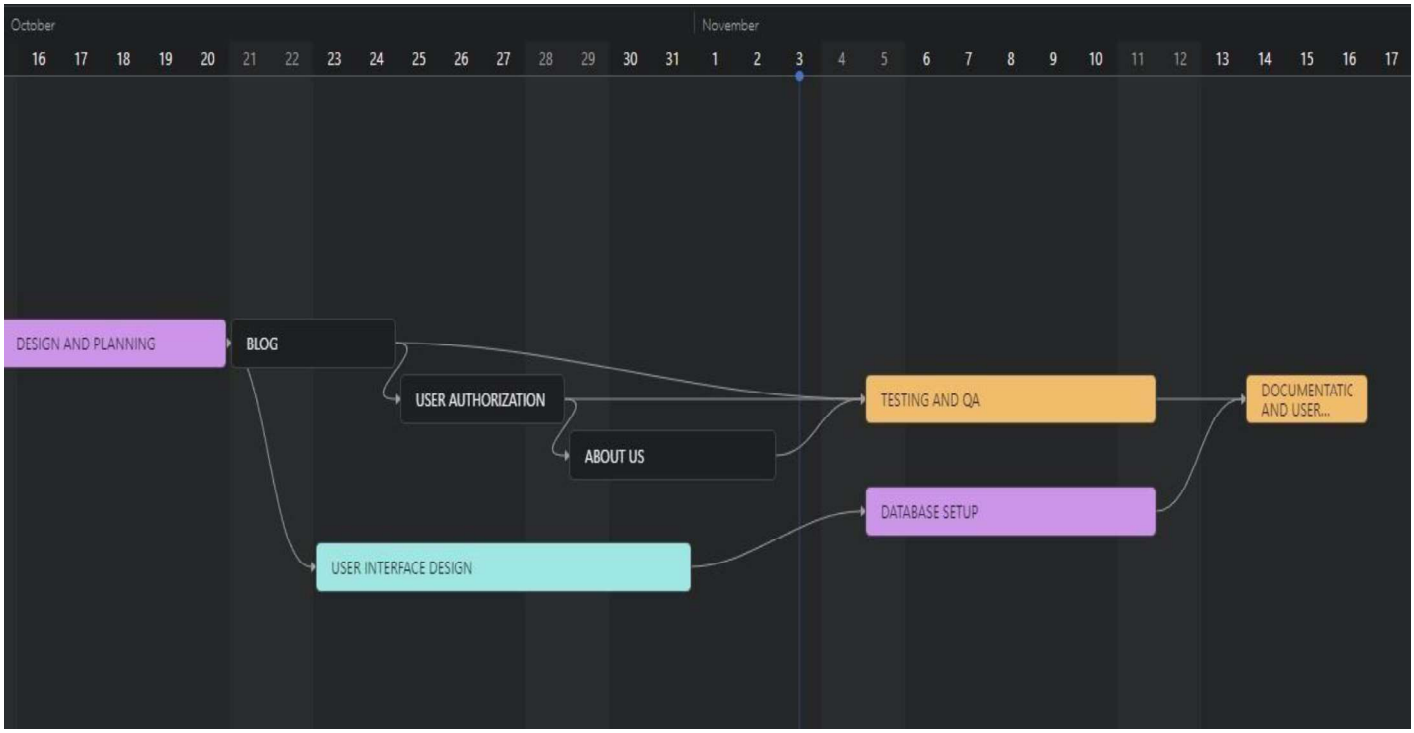
Time taken = 3.4867/4 or 4.099/4

= 0.8716 or 1.02475 months

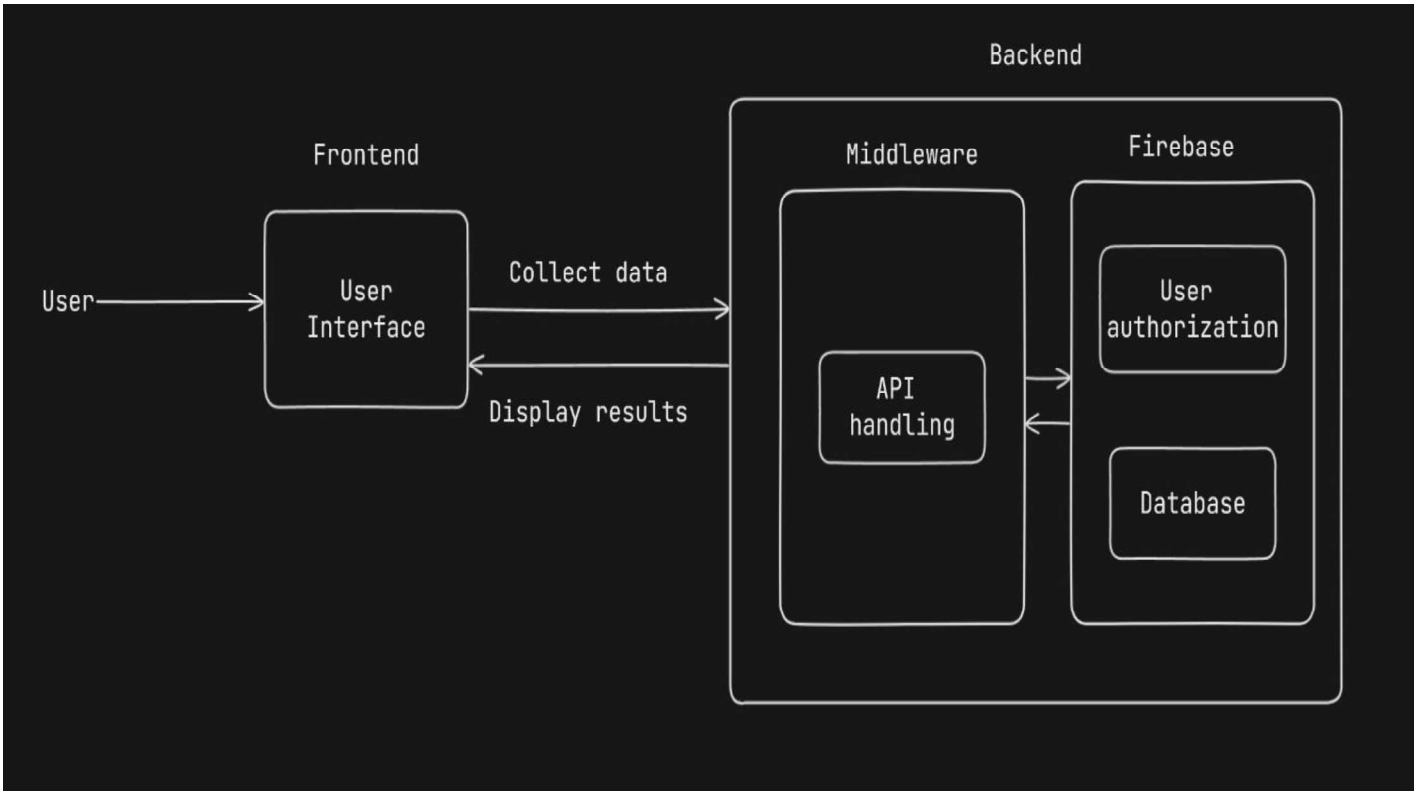
Gantt Chart:




September
October
16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
DJ PROJECT INITIATION
DJ REQUIREMENT ANALYSIS
DESIGN AND PLANNING


October
November
16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
DESIGN AND PLANNING
BLOG
USER AUTHORIZATION
ABOUT US
TESTING AND QA
DOCUMENTATIC AND USER...
DATABASE SETUP
USER INTERFACE DESIGN

Architecture Diagram:



Software Requirements Specification:

Introduction

Purpose

Our platform's purpose is twofold. First, it offers writers and content creators a powerful stage to showcase their expertise, creativity, and unique perspectives. Second, it provides readers with a curated selection of high-quality articles and stories on a myriad of topics. Whether you're here to publish your work or embark on a reading journey, Snipify aims to enrich your online experience.

Intended Audience

Our platform caters to a diverse audience, including but not limited to writers, bloggers, researchers, readers, small businesses, academics, and enthusiasts. We invite anyone with a passion for sharing knowledge, storytelling, or exploring new ideas to become a part of our vibrant community.

Product Scope

Snipify encompasses an array of features, including user-friendly article creation and editing tools, personalized user profiles, a robust recommendation system, and a responsive design for seamless reading across devices. Our aim is to provide a comprehensive online publishing and reading experience that caters to both content creators and readers alike.

References

To ensure the highest quality of content and user experience, Snipify draws inspiration from industry leaders like Medium, WordPress, and other reputable online publishing platforms. We are committed to incorporating the best practices and innovations that have shaped the digital publishing landscape

Overall Description

Product Perspective

Snipify is a sophisticated online publishing platform that operates in the digital content creation and sharing ecosystem. It serves as an ecosystem where writers, bloggers,

content creators, and readers converge to create, share, and consume diverse content. The platform offers a holistic environment for content publication, discovery, and engagement.

Product Functions

Article Creation and Editing: Users can create, edit, and format articles using a feature-rich text editor.
Content Categorization: Articles are organized by tags or categories for easy navigation.
User Interaction: Readers can engage with content by liking, commenting, and sharing articles.
User Profiles: Users can customize profiles, manage preferences, and follow their favorite authors. Recommendation Engine: A sophisticated recommendation system suggests content based on user preferences.
Notifications: Real-time notifications inform users about interactions on their content. Search Functionality: A robust search feature allows users to discover content by keywords or categories.

User Classes and Characteristics

Writers and Bloggers: Content creators who contribute articles and stories, ranging from professionals to hobbyists.
Readers: Individuals who consume content, from avid readers to occasional visitors.
Moderators and Admins: Responsible for content moderation, user management, and platform oversight

Operating Environment

Snipify operates in a web-based environment, accessible through modern web browsers across various devices and operating systems. It relies on server-side technologies to manage user accounts, content storage, and interactions. The platform employs secure authentication protocols and data encryption to ensure user privacy and data integrity.

Design and Implementation Constraints

Scalability: The system must handle a growing user base and content library efficiently. Performance: Ensuring fast response times for article rendering and user interactions. Security: Robust security measures must be in place to protect user data and content. Compatibility: Compatibility with various web browsers and devices must be maintained. Regulatory Compliance: Adherence to data protection regulations (e.g., GDPR, CCPA) and intellectual property laws is crucial.

Assumptions and Dependencies

Snipify assumes that users will adhere to community guidelines and acceptable use policies. The platform depends on reliable web hosting infrastructure and may require third-party services for features like payment processing.
User-generated content is assumed to be owned by the creators, and proper attribution and copyright adherence are expected.
Continuous improvement and feature development are dependent on user feedback and technological advancements.
In summary, Snipify is a comprehensive digital content platform designed to facilitate content creation, sharing, and engagement. It caters to a diverse user base, operates securely in a webbased environment, and is subject to various design constraints and dependencies. The success of the platform hinges on user participation, adherence to policies, and adaptability to evolving technologies and user needs.

External Interface Requirements

User Interfaces

Web-Based User Interface (UI): Users will access Snipify through a web browser on desktop and mobile devices. The UI should be intuitive, responsive, and visually appealing, providing an excellent user experience.
User Profiles: Users can customize their profiles, including profile pictures, bios, and links to social media accounts. The profile interface should be user-friendly, allowing easy editing and management.
Article Creation and Editing: A rich text editor should enable writers to create and edit articles efficiently. It should support formatting, media embedding, and easy saving of drafts.

Content Discovery: The interface for discovering content should include category/tag navigation, a search bar, and a recommendation feed tailored to user preferences.
User Notifications: Users should receive notifications about interactions, comments, and updates via a user-friendly notification center.
Messaging and Comments: The platform should provide a user-friendly interface for private messaging and commenting on articles.

Software Interfaces

Backend Server: Snipify will interact with a backend server that manages user accounts, content storage, and communication between users. This server will handle authentication, content storage, and data retrieval.
Third-Party Services: Integration with third-party services may be required for functions such as payment processing, analytics, and social media sharing. These services should seamlessly interact with the platform.
APIs: The platform may expose APIs for third-party developers to integrate their applications or services. API documentation and endpoints should be clearly defined for external developers.

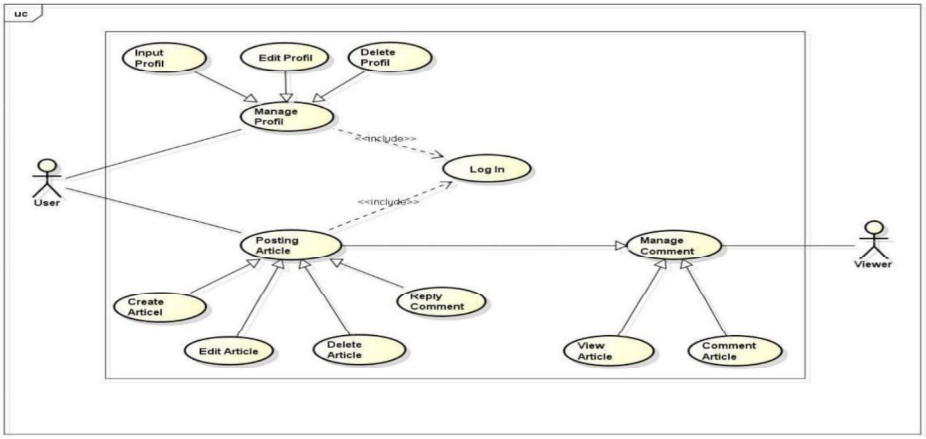
Communications Interfaces

User Communication: The platform should facilitate real-time communication between users, including private messaging and article comments. This requires robust communication protocols and secure data transmission.
Email Notifications: Email notifications will be sent to users for account-related activities, such as password reset, account verification, and content updates. Email templates and SMTP server interfaces should be configured.
External Services: Communication with third-party services (e.g., payment gateways, social media sharing) should follow industry-standard protocols to ensure data security and integrity.
API Integration: If third-party developers are allowed to build integrations, a RESTful API or similar interface should be provided, along with clear documentation for API endpoints and usage

Analysis

Models

Use case diagram:



System Features

Feature 1: User Registration and Authentication

Description and Priority

This feature allows users to create an account and authenticate themselves on the platform. It is of high priority as it forms the foundation for user engagement and content creation.

Stimulus/Response Sequences

- User navigates to the registration page.
- User provides a username, email, and password.
- User submits the registration form.
- System validates the information provided.
- If validation is successful, the system creates the user account and sends a verification email.
- User clicks the verification link in the email.
- System verifies the email and activates the user account.

Functional Requirements

- REQ-1: Users must provide a unique username, a valid email address, and a strong password during registration.
- REQ-2: Usernames and emails must be unique across the platform.
- REQ-3: Passwords must meet predefined strength criteria (e.g., length, complexity). - REQ-4: The system must securely store user credentials (e.g., hashed passwords) and other registration data.
- REQ-5: Users must be able to request a password reset via email in case of forgotten passwords.

Feature 2: Article Creation and Editing

Description and Priority

This feature empowers authors to create and edit articles using a rich text editor. It is of high priority as it's fundamental to the core functionality of content creation on the platform.

Stimulus/Response Sequences

- User navigates to the article creation page.
- User uses the rich text editor to compose and format an article.
- User can add text, headings, images, videos, and other media.
- User saves the article as a draft or publishes it.
- System validates and saves the article, notifying the user of successful publication or saving as a draft.

Functional Requirements

- REQ-1: The system must provide a user-friendly rich text editor for article creation and editing. - REQ-2: Authors should be able to format text, add headings, insert images, embed videos, and create hyperlinks within articles.
- REQ-3: The editor must support draft saving and auto-save functionality to prevent data loss.
- REQ-4: Authors should be able to update and edit published articles.
- REQ-5: Authors should receive notifications upon successful publication and when others.

Feature 3: Content Categorization and Discovery

Description and Priority
This feature improves content discoverability by categorizing articles into topics or themes and providing a user-friendly search function. It is of high priority to enhance user experience.

Stimulus/Response Sequences
User navigates to the category/tag section and selects a topic of interest.
User uses the search bar to look for articles related to a specific keyword.
System displays a list of articles within the selected category or based on the search query.

Functional Requirements

REQ-1: Authors must categorize their articles by selecting relevant tags or categories during the publishing process.
REQ-2: Users can click on a category or tag to view articles related to that topic.
REQ-3: The search function should provide real-time suggestions as users type their query.
REQ-4: Search results must be relevant and ranked based on factors like relevance, publication date, and user interactions.
REQ-5: The platform should provide an option for users to save and follow specific categories or tags for personalized content recommendations.

Other Non-Functional Requirements

Performance Requirements

Response Time: The system should respond to user interactions (e.g., loading pages, submitting forms) within 2 seconds, ensuring a responsive user experience.
Scalability: The platform must be capable of handling a minimum of 100,000 concurrent users without a significant degradation in performance.
Data Retrieval: Database queries for common operations (e.g., fetching articles, user profiles) should complete within 500 milliseconds on average.

Safety Requirements

Data Backup: User data, articles, and system configurations must be regularly backed up to ensure data recovery in case of system failures or data loss.
Error Handling: The system should gracefully handle errors and exceptions to prevent data corruption and ensure the stability of the platform.

Security Requirements

Data Encryption: User passwords, sensitive user data, and communication between users and the system must be encrypted using industry-standard encryption algorithms.
Authorization: Access to user data and functionalities should be role-based and follow the principle of least privilege, ensuring that users can only access what they are authorized to.
User Data Protection: Compliance with data protection regulations (e.g., GDPR, CCPA) is mandatory.
User data should be anonymized or deleted upon user request.

Software Quality Attributes

Usability: The platform should be designed with a user-friendly interface, intuitive navigation, and accessible design to ensure a positive user experience.
Reliability: The system must be available 99.9% of the time, with minimal downtime for maintenance.
Maintainability: Code should be well-documented, modular, and adhere to coding standards, making it easy for developers to maintain and extend the system.

Business Rules

Content Moderation: Content that violates community guidelines or contains prohibited materials must be flagged and subject to moderation. Repeated violations can result in account suspension.

Appendix A: Glossary

Blog Post: An article or piece of content created by a user on the site.

Comment: A response to a blog post or another user’s comment.

Category: A classification system used to organize content based on topic or theme.

Content: Any type of media created by the user, including text, images, and videos.
Authentication: The process of verifying a user’s identity

User Profile: A user's personal space on the platform where they can customize their information, share their bio, and display their contributions.

Followers: Users who subscribe to another user's profile to receive updates on their new content.

Notification: A message or alert that informs a user about interactions on their content or relevant platform activities.

Draft: An unpublished version of an article that is still being edited and refined by the author.
Tag: A keyword or label associated with an article to categorize and make it easier to find based on topics.

Appendix B: Field Layouts

User Profile

Field Name	Data Type	Description
User ID	Integer	Unique identifier for each user.
Username	String	User's chosen username for identification.
Email	String	User's email address for communication.
Password	String	Encrypted password for authentication.
Profile Picture	Image	User's chosen profile picture.
Biography	Text	User's self-written bio or description.

Article/Post

Field Name	Data Type	Description
Article ID	Integer	Unique identifier for each article.
Title	String	Title of the article.
Content	Text	Main content of the article.
Author ID	Integer	ID of the user who authored the article.
Publication Date	Date/Time	Date and time when the article was published.
Tags	Array	Keywords or tags associated with the article.
Likes	Integer	Count of likes or upvotes received.
Comments	Integer	Count of comments on the article.

Comment

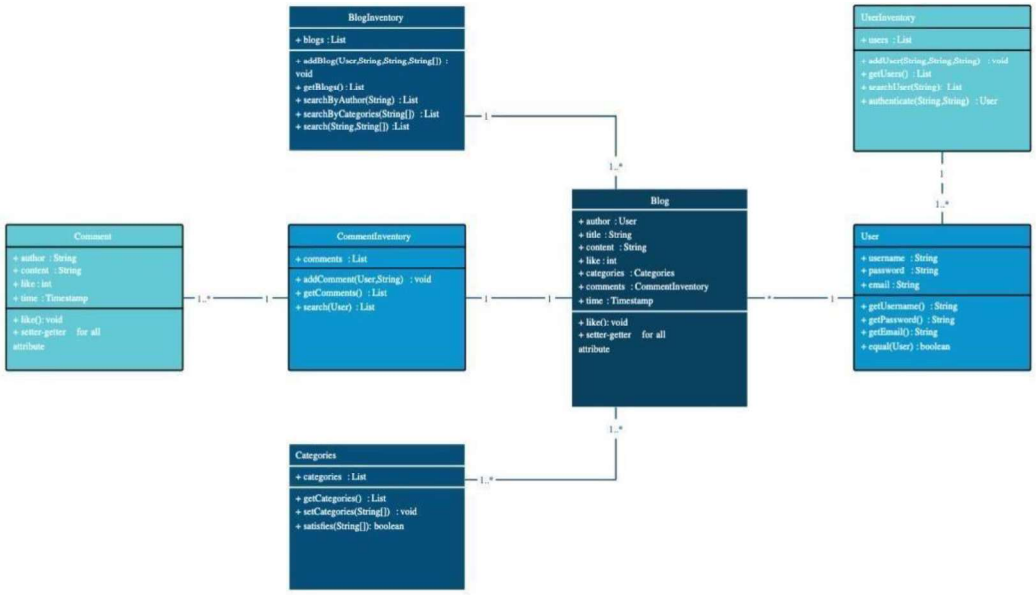
Field Name	Data Type	Description
Comment ID	Integer	Unique identifier for each comment.
Article ID	Integer	ID of the article the comment belongs to.
User ID	Integer	ID of the user who wrote the comment.
Comment Text	Text	Text content of the comment.
Comment Date	Date/Time	Date and time when the comment was posted.
Likes	Integer	Count of likes or upvotes received on the comment.

Appendix C: Requirement Traceability Matrix

Sl. No	Requirement ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID
1	REQ-001	User authentication	A-001	D-001	I-001	T-001	STC-001
2	REQ-002	Create a new blog post	A-002	D-002	I-002	T-002	STC-002
3	REQ-003	Comment on blog post	A-003	D-003	I-003	T-003	STC-003
4	REQ-004	Search functionality	A-004	D-004	I-004	T-004	STC-004

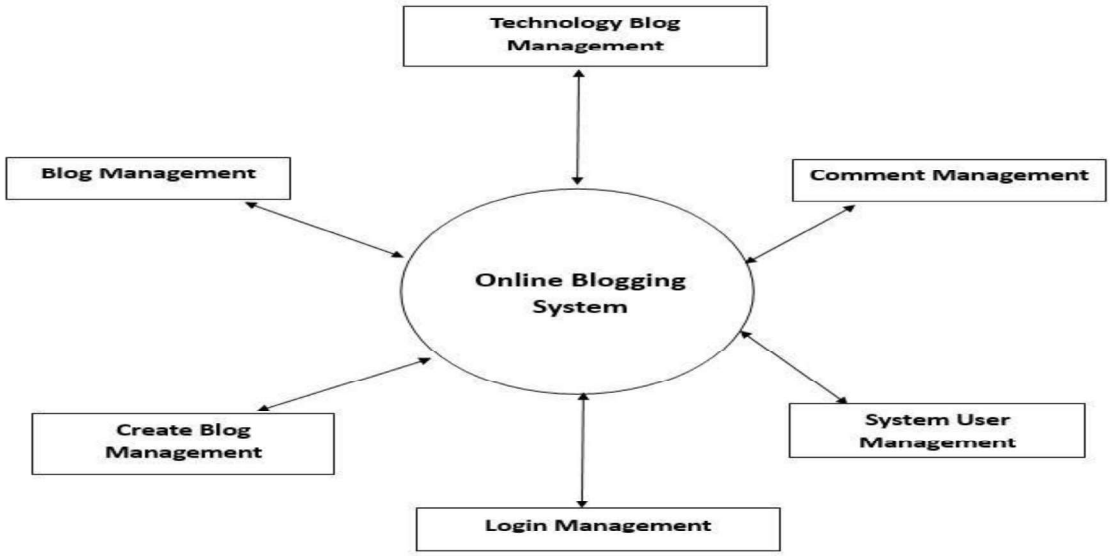
Design Diagrams:

1. CLASS DIAGRAM



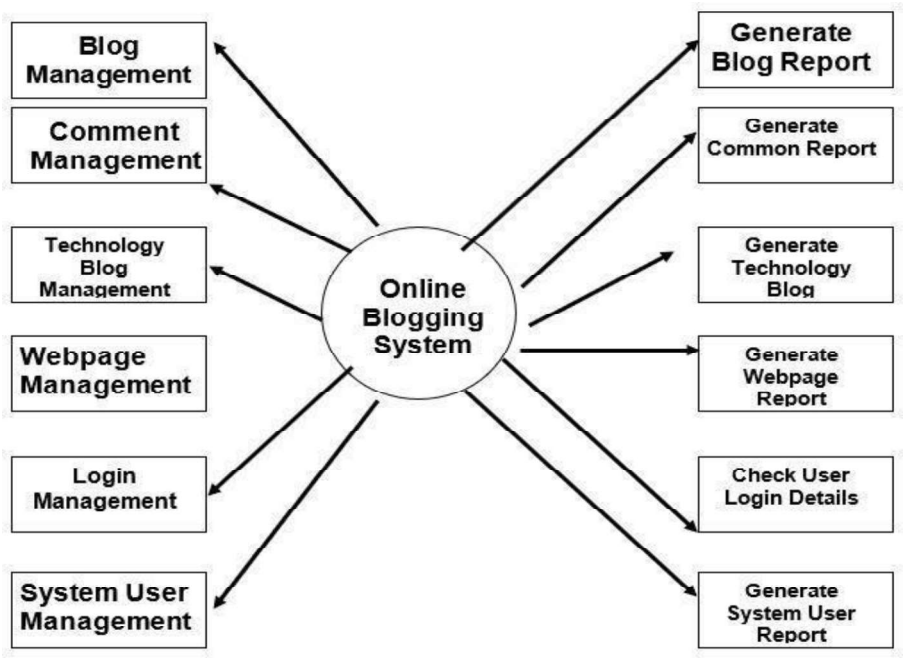
2. DATA FLOW DIAGRAM

a. 0 level DFD

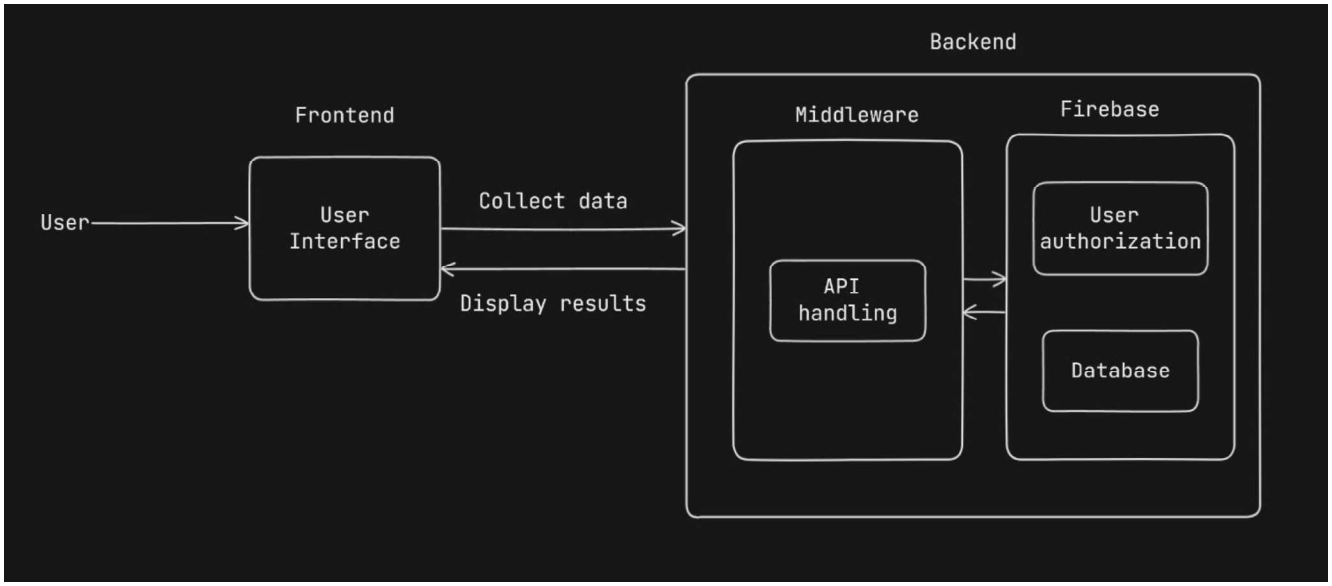


Zero Level DFD – Online Blogging System

b. 1 level DFD



3. Architectural Diagram

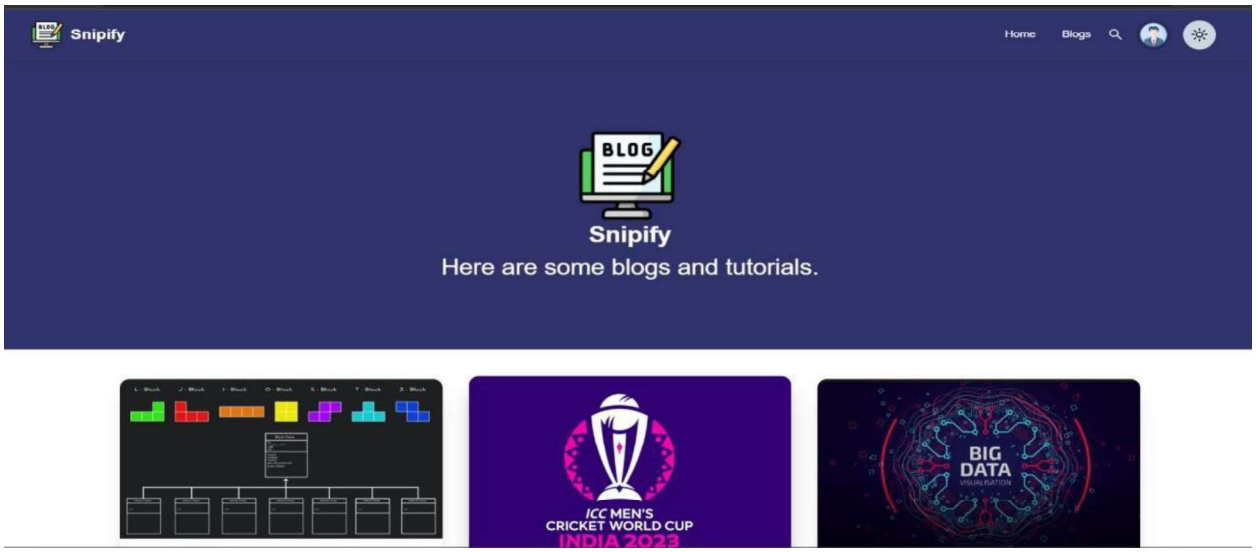


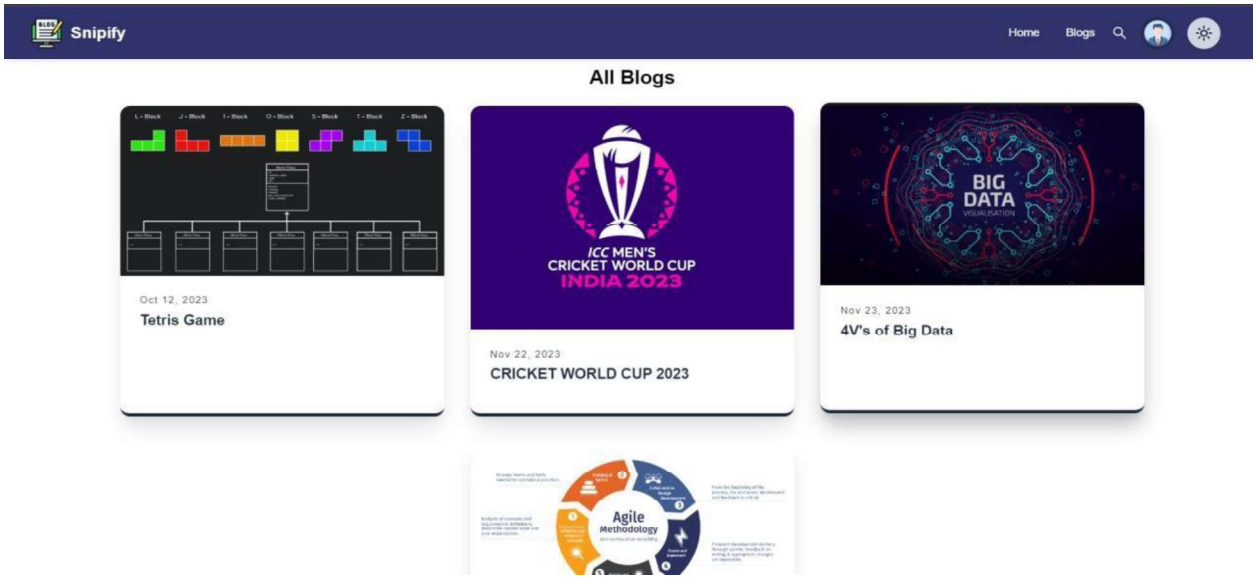
Test Plan, Cases:

Test Case ID	Name of Module	Test Case Description	Pre-conditions	Test Steps	Test Data	Expected Results	Actual Results	Test Result
BA-01	**User Login**	**Test user login functionality**	**User account exists**	**1. Navigate to the login page. 2. Enter valid user credentials. 3. Click "Login."**	**Valid user credentials.**	**User should be logged in successfully.**	**User logged in successfully**	**PASS**
BA-02	**Create Blog Post**	**Test the ability to create a new blog post**	**User is logged in**	**1. Navigate to the "Create Post" page. 2. Enter valid blog post details. 3. Click "Publish."**	**Valid blog post details (title, content, etc.).**	**Blog post should be created and visible on the user's profile.**	**Blog post created successfully**	**PASS**
BA-03	**Delete Blog Post**	**Test the ability to delete a blog post**	**User has at least one published blog post**	**1. Navigate to the "Delete Post" page. 2. Select the blog post to delete. 3. Confirm the deletion.**	**NA**	**Blog post should be deleted and no longer visible on the user's profile.**	**Blog post deleted successfully**	**PASS**

BA-04	**View Blog Post**	**Test the ability to view a blog post**	**At least one blog post exists**	**1. Navigate to the "View Post" page. 2. Select a blog post to view.**	**NA**	**The selected blog post should be displayed with its details.**	**Blog post displayed as expected.**	**PASS**
BA-05	**Search Blog Posts**	**Test the ability to search for blog posts**	**At least two blog posts exist**	**1. Navigate to the search bar. 2. Enter keywords related to blog posts. 3. Click "Search."**	**Keywords related to existing blog posts.**	**Blog posts matching the keywords should be displayed.**	**Relevant blog posts displayed.**	**PASS**
BA-06	**Comment on Blog Post**	**Test the ability to add comments to a blog post**	**User is logged in, and the blog post allows comments**	**1. Navigate to a blog post. 2. Scroll down to the comment section. 3. Enter a valid comment. 4. Click "Submit Comment."**	**Valid comment text.**	**Comment should be added to the blog post.**	**Comment added successfully**	**PASS**

Screenshots of Test Output:





Make Comment

Enter Full Name

Write a comment...

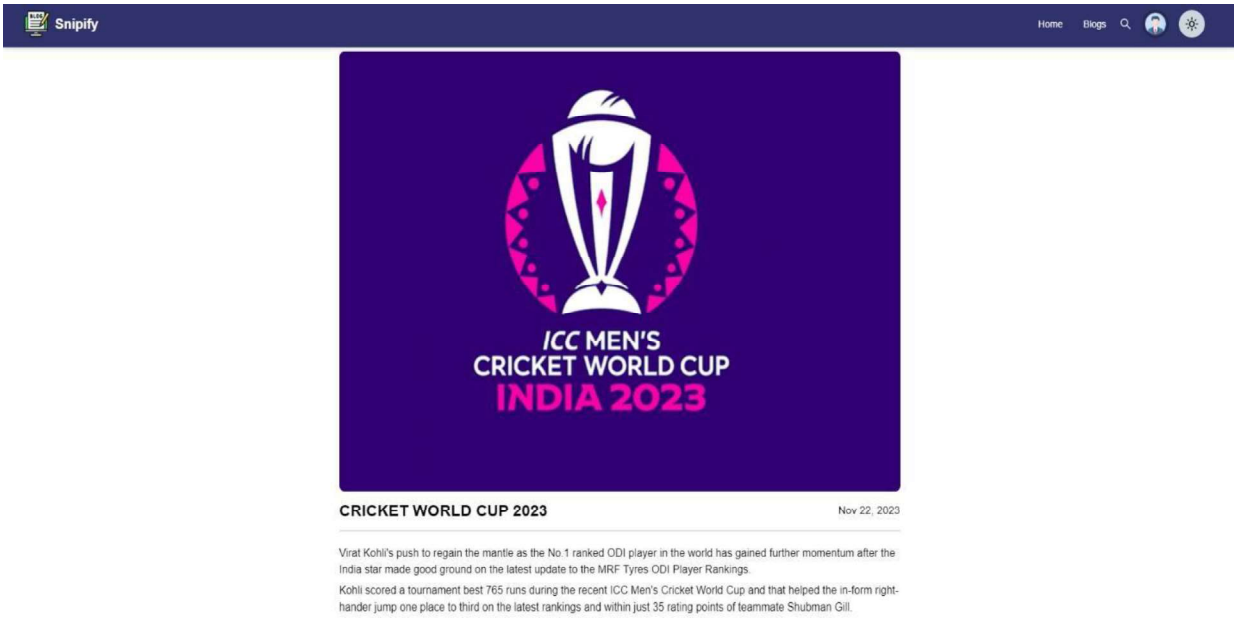
POST COMMENT

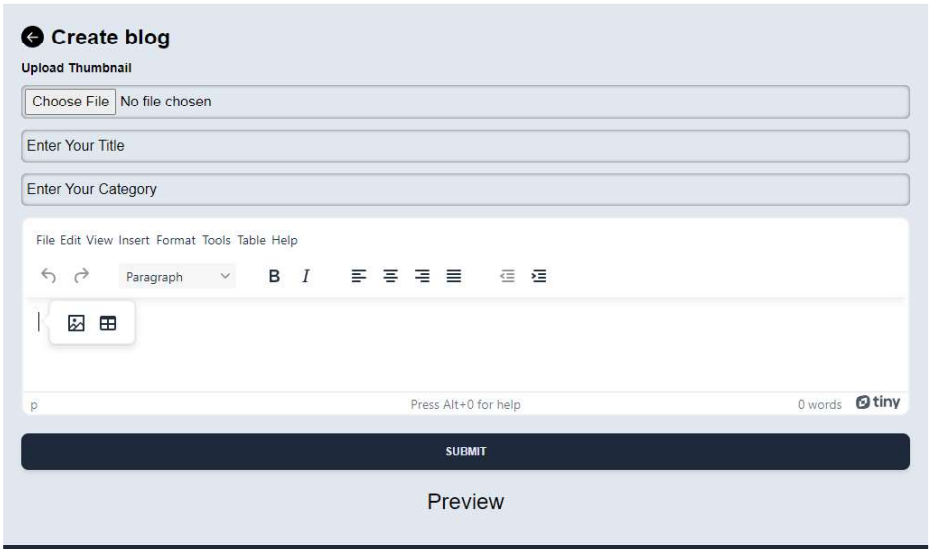
Harsha Nov 23, 2023

👑 Kohli

Darshan Nov 23, 2023

Number 1 coming soon




Create blog
Upload Thumbnail
Choose File
No file chosen
Enter Your Title
Enter Your Category
File Edit View Insert Format Tools Table Help
Paragraph
Press Alt+0 for help
0 words
tiny
SUBMIT
Preview


Admin
Software Developer
admin@gmail.com
Total Blog : 15
CREATE BLOG
LOGOUT


S.No
Thumbnail
Title
Category
Date
Action
1.
CRICKET WORLD CUP 2023
SPORTS
Nov 22, 2023
Delete
2.
4V's of Big Data
Big Data
Nov 23, 2023
Delete
3.
Agile
SE
Nov 23, 2023
Delete