Software Requirements Specification

for

ShikshaHub

Version 1.0 approved

Prepared by Shruti Malode

Komal Punwatkar

Asmi Bhandekar

Bajaj Institute Of Technology

21/04/25

Table of Contents

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document establishes the requirements for the Educational Blog & resource sharing Platform - a web application built with the MERN stack (MongoDB, Express.js, React.js, Node.js).
The system is designed to facilitate the interaction among students, faculties, and administrators within an educational setting through enabling resource sharing, blog writing, review processes, AI-based blog summaries and chatbot. This SRS outlines the full functionality of the system and is not a partial system or single subsystem.
The document is mostly for use by:
Developers, for system functionality understanding and implementation.
Testers, to ensure the system satisfies stated requirements.
The stakeholders, like the educational institutions or learning platforms, for comprehending the system's objectives and capabilities.

1.2 Intended Audience and Reading Suggestions

This Software Requirements Specification (SRS) document is targeted for the following categories of readers:

- Developers – To learn the functional and non-functional requirements and implement the system as such.
- Testers Team – To create test cases and ensure that the software satisfies all the requirements specified.
- Project Managers – To plan, monitor progress, assign resources, and deliver all the requirements.
- UI/UX Designers – To create user interfaces that support the functional flow and system goals.
- System Administrators –user access, and platform maintenance.

Reading Recommendations:

All Readers should start with:

- Section 1: Introduction – Offers an overview of the system, its purpose, scope, and terminology.
- Section 2: Overall Description – Describes user characteristics, product perspective, and system environment.

Developers and Testers are best advised to take special notice of:

- Section 3: Specific Requirements – Includes detailed functional and non-functional requirements.
- Section 5: Use Cases and Diagrams – Provides visual representations of system interactions.

Project Managers and Stakeholders can concentrate on:

- Section 1 and 2 for an overview at a high level.
- Section 4: External Interfaces – Outlines system integration and points of interaction.

UI/UX Designers refer to:

- Section 2.3: User Characteristics and
- Section 3: Functional Requirements in order to design user-friendly experiences.

1.3 Product Scope

The product outlined in this report is a MERN stack-based Educational Blog and Resource sharing Platform, meant to be an interactive and collaborative environment for students, faculties, and administrators at academic institutions. The platform makes use of cutting-edge web technologies and AI tools to facilitate better content creation, learning, and communication.

Purpose and Objectives:

- To create a systematic mechanism for students to develop and submit educational blogs.
- To have a multi-step approval system with faculties and administrators to ensure content quality.
- To facilitate faculties to upload study materials, handle student blogs, and add academic events.
- To provide administrators with the capability to monitor user activities, uploaded content and ensure platform performance.
- To include AI-based functionalities like blog summary and a chatbot for resolving doubts.

Benefits:

- Promotes student participation through writing and peer learning.
- Enhances content quality and academic worth with a transparent review process.
- Facilitates ease of content discovery and access through intelligent search and tagging.
- Lessens the reliance on physical touch by offering a virtual collaboration space.
- Aids institutional objectives of digitization, content management, and improved learning experience.

1.4 References

[1] Gupta, Nikunj Mani, et al. "MERN Stack in Web Development: An Interactive Approach."
[2] Zhu, Haolin, et al. "Online Sharing Platform for Course Modules: Understanding Materials Use and Effectiveness." 2021 ASEE Virtual Annual Conference Content Access. 2021.
[3] Desai, Krutika, and Jinan Fiaidhi. "Developing a Social Platform using MERN Stack." Authorea Preprints (2023).
[4] https://classroom.google.com
[5] https://medium.com

2. Overall Description

2.1 Product Perspective

ShikshaHub is a web-based, centralized platform for classroom collaboration to simplify the process of communication, content posting, and class/group management for students, instructors, and admins. It's an independent product but can integrate with other edu platforms or in-house authentication infrastructures using APIs.

The platform is accessed through role-based access control, where users are categorized into three primary roles: Admin, Teacher, and Student. Each has certain functionalities in order to support secure and effective use of the system.

System Interfaces:

ShikshaHub will provide RESTful APIs as interfaces for data exchange between frontend (React.js) and backend (Node.js with Express), with data persisting in a MongoDB database. It will accommodate integration with institution systems through OAuth or token-based authentication.

User Interfaces:

The platform will include responsive web interfaces that are optimized for desktop and mobile use. Different dashboards and modals will be customized for every user role, providing a clean and simple experience.

Hardware Interfaces:

ShikshaHub, being a web application, will run on any contemporary hardware equipped with a web browser and internet connectivity. No additional hardware is needed.

Software Interfaces:

- Frontend: React.js with Formik and TailwindCSS for form management and UI consistency.
- Backend: Node.js using Express, providing REST APIs.
- Database: MongoDB for users, groups, materials, and roles.
- Authentication: JSON Web Tokens (JWT) for secure session login.

2.2 Product Functions

The Learning Platform and Educational Blog offers a formalized set of features intended to assist students, faculties, and administrators. At a high level, the product allows users to create, manage, review, and consume educational content within a collaborative environment augmented by AI tools.

Key Functional Areas:

1. Student Functions:

- Register and log in to the platform.
- Create and upload blogs.
- Access all study materials.
- Summarize blogs with AI-based tools.
- Engage with an AI-driven chatbot for doubt resolution.
- Bookmark blogs and save for later reading.

2. Teacher Functions:

- Check, approve, or reject student-submitted blogs.
- Give structured feedback for rejected blogs.
- Upload and manage study materials.
- Create and manage academic events.

3. Admin Functions:

- Create communities/groups.
- Manage user accounts (faculties and students).
- Check teacher-approved blogs prior to final publication.
- Manage uploaded content including study materials and event data.

4.AI Integration:

- Blog summarizer tool based on Natural Language Processing (NLP).
- Integration with chatbots to manage repetitive asked academic questions or queries.

5. System Functions:

- Role-based access control for Students, Faculties, and Admins.
- UI that is responsive and user-friendly across devices.
- Notification system for blog status notifications, events, and messages.
- Secure authentication and authorization (e.g., JWT-based login).

2.3 User Classes and Characteristics

The Educational Blog and Learning Platform is geared to accommodate three main classes of users: Students, Faculties, and Administrators. Each of these classes has defined roles, rights, and interactions with the system. The user classes differ in their level of technical expertise, task,
and usage frequency on the platform.

1. Students

- Role: Learners and content creators.
- Privileges: Ability to create and post blogs, utilize AI tools (chatbot and summarizer), bookmark items, and search for materials.

Key Characteristics:

- Require simple and intuitive UI.
- Encouraged by feedback and approval of their blogs.
- Get the advantage of AI-based tools to improve learning.

2. Teacher

- Role: Content evaluators and contributors of knowledge.
- Privileges: Ability to review and approve/reject student blogs with comments, post study materials, and create academic events.

Key Characteristics:

- Need dashboard-style access to student work submissions and content tools.
- Responsible for educational guidance and ensuring the quality of content.

3. Administrators

- Role: System managers and content moderators.
- Privileges: Complete access to user management, blog moderation, content monitoring, and platform settings.

Key Characteristics:

- Ensure running smoothly of the platform and adherence to guidelines.
- Approve final content, and administer user roles.
- Need secure, full-access control panels and comprehensive system logs.

Most Critical User Classes:

- Students and Faculties are the system's core focus as they power the content creation and review processes.
- Administrators are needed to sustain the system but are less in number and more backend-oriented.

2.4 Operating Environment

The Learning Platform and Educational Blog will function in a contemporary web-based environment, constructed using the MERN stack (MongoDB, Express.js, React.js, Node.js). The platform will be hosted on a cloud-based system and accessed through standard web browsers on multiple devices.

Client-Side Environment:

- Hardware: Desktop, laptop, tablet, or smartphone with internet access.
- Supported Operating Systems:
    - Windows 10 and later
    - macOS 10.14 and later
    - Linux (Ubuntu, Fedora, etc.)

Server-Side Environment:

- Backend Framework: Node.js using Express.js
- Frontend Framework: React.js
- Database: MongoDB (NoSQL)
- Hosting Platform: Cloud-based (e.g., AWS)

Authentication & Security:

- JWT (JSON Web Tokens) for user authentication
- HTTPS for secure communication
- Role-based access control

Third-Party Integrations:

- AI Services: Blog summarization and chatbot using NLP models (e.g., OpenAI API or custom ML models).

2.5 Design and Implementation Constraints

The building of the Educational Blog and Learning Platform shall be subject to particular constraints and limitations that affect design decisions, technology stack, security policies and maintainability.

1. Technology Stack Constraints

- The application should be developed within the MERN stack:
  - MongoDB as the database.
  - Express.js and Node.js as the backend.
  - React.js as the frontend UI.

2. Platform and Browser Compatibility

- The application should be completely responsive and mobile-friendly.
- It should be compatible with all major browsers (Chrome, Firefox, Edge).

3. AI Tool Integration

- AI functionality such as blog summarization and chatbot should integrate through pre-defined APIs (e.g., OpenAI or in-house models).
- API keys and tokens should be handled securely through environment variables.

4. Security Requirements

- The system should have role-based access control (RBAC) with JWT authentication.
- Client-server communication should be done over HTTPS.
- User data privacy should be maintained according to basic data protection principles (e.g., encryption at rest and in transit, password hashing).

5. Development Standards

- The codebase should adhere to JavaScript best practices and naming conventions.
- Usage of Git for code quality control and version management.
- Development has to be compliant with agile with modular and reusable components for scaling.

6. Maintenance and Handoff

- The platform should be thoroughly documented for handover to the maintenance or client team.
- If the client organization is taking over the platform after deployment, there should be clear API documentation and deployment guides.

3. External Interface Requirements

3.1 User Interfaces

The Learning Platform and Educational Blog will have an easy-to-use, responsive, and role-based web interface for administrators, faculties, and students. The UI will be guided by current UI/UX best practice with accessibility and ease of use at the center.

General User Interface Guidelines:

- Same layout and navigation bar across every page.
- Responsive design supporting desktop, tablet, and mobile devices.
- Clean and minimalist interface with a consistent color scheme and font styles.
- Standardized buttons (e.g., Submit, Cancel, Save, Back) throughout all forms.

Common UI Elements Throughout All Roles:

- Top Navigation Bar: Logo, role-specific menu items, notifications, and user profile dropdown.
- Footer: About, Contact, Terms & Policies, and Social Media links.
- Search Bar: Filtered search globally available.

Role-Specific UI Components:

- Student Interface:
  - Dashboard: Rapid view of submitted blogs, status of feedback, bookmarks, and summary history.
  - Blog Editor: Rich text editor with format, insert media, and submit blog options.
  - Summarization Tool: Get AI-generated summary.
- Teacher Interface:
  - Dashboard: View of pending blog submissions, uploaded study materials, and upcoming events.
  - Blog Review Panel: Viewers of the blog with approve/reject buttons and a feedback box.
  - Upload Study Material: Uploading PDFs, documents, and videos with categorization.
  - Event Manager: Academic or community event creation, editing, and publishing.
- Admin Interface:
  - Admin Dashboard: Statistics and summaries (e.g., number of users, blogs, pending reviews).
  - User Management: Add, remove, or deactivate students/faculties with role assignment.
  - Blog Moderation: Last review panel for teacher-approved blogs before publishing.
  - Content Manager: All study material and event overview.

User Interface Technologies:

- Developed with React.js and styled with Tailwind CSS or Material UI.
- interactive components supported through React libraries.
- UI components are accessible, supporting screen readers and keyboard navigation.

3.2 Hardware Interfaces

Since the Educational Blog and Learning Platform is a web application based on the MERN stack, there are no explicit hardware interface dependencies other than those usual for accessing web services.

Client-Side (User Device) Hardware Interfaces:

- **Supported Device Types:**
  - Desktop and laptop computers (x86 and ARM-based processors)
  - Tablets and smartphones (Android and iOS devices)

- **Input Devices Supported:**
  - Keyboard and mouse (primary input)
  - Touchscreen input (for mobile and tablet users)

Minimum Recommended Hardware for Smooth Operation:

- **Desktop/Laptop:**
  - CPU: Dual-core 2 GHz or more
  - RAM: 4 GB or more
  - Display: 1280x720 resolution or better
- **Mobile Devices:**
  - Android 8.0+ or iOS 13+
  - Minimum 2 GB RAM

Data Interaction:

- Communication with backend over HTTPS protocol.
- All interactions are stateless and RESTful using JSON-based APIs.

Server-Side Hardware Interfaces:

- **Hosting Environment:**
  - Cloud-based servers (e.g., AWS, Vercel Functions)
- **Hardware-Related Protocols:**
  - Standard TCP/IP networking protocols.
  - HTTPS for secure client-server communication.

3.3 Software Interfaces

This part describes the various software components, services, and tools that the Educational Blog and Learning Platform interface with. The interfaces enable the system's fundamental functionalities, such as user management, blog processing, AI summarization, and data storage.

1.Operating System Interfaces
Target OS (Server-Side):

- Windows
- Node.js runtime compatibility required

Client-Side Compatibility:

- Windows, macOS, Android, iOS compatible via contemporary browsers (Chrome, Firefox, Edge, Safari)

2. Database Interfaces

- Database Used:
  - MongoDB
  - NoSQL document-oriented database used for storing blogs, summaries, users, feedback, bookmarks, study materials, and events.

Communication Protocol:

- Makes use of Mongoose ORM to interact with MongoDB from Node.js (Express backend)
- Data is sent in JSON format over RESTful API endpoints.

Shared Data Elements:

- User profiles (name, email, role, password hash)
- Blog metadata (title, author, status)

- AI summaries (with reference to original blog ID)
- Study material and file references

API Format:
- RESTful APIs over HTTPS
- Request/response bodies in JSON format

3. Frontend Framework
- React.js (v18.x)
- Utilizes Axios for API requests, Tailwind CSS for styling, and potentially React Router for navigation.

4. AI Integration

Third-party AI Services:
These are used for:
- Blog summarization
- Chatbot interaction for doubt-resolution

Interface Details:
- API request made with blog text as input
- Result includes summary text or chatbot response
- Authentication handled through API key

3.4 Communications Interfaces

This section describes the communication protocols, structure, standards, and security measures by which various components of the system and the system and users interact.

1. Web Communication
Protocol Used:
- HTTPS is employed for all interactions between the client and server to provide secured communication and data integrity.

Browsers Supported:
- Google Chrome
- Mozilla Firefox
- Microsoft Edge

Message Format:
- JSON format is used for all requests and responses.
- HTTP methods employed: GET, POST, DELETE

2. Server-Client Communication
Frontend-Backend Interaction:
- The frontend (React.js) sends requests to the backend (Node.js/Express.js) with Axios being utilized for async HTTP requests.

3. Email Communication
Purpose:

- Blog approval, rejection notifications, event invites, and account notifications.

AI Services (e.g., Blog Summarizer, Chatbot):

- Communication with third-party AI services through APIs
- Uses Bearer Token Authentication for secured API access
- Requests have blog content or questions; responses have summaries or chatbot responses

5. File Transfer and Media Handling

Upload Mechanism:

- Study materials and blog attachments are uploaded through multipart/form-data
- Files stored through local server storage.

Download Mechanism:

- Files served over HTTPS with proper headers for secure and proper handling

6. Security and Synchronization

Security Measures:

- JWT-based user authentication and role authorization
- HTTPS provides data encryption in transit
- Input validation and sanitation on client and server
- CORS policy enabled for secure cross-origin requests

Session Management:

- Stateless sessions through tokens (sessionStorage or cookies)
- Token expiration and refresh logic implemented to ensure security

4. System Features

This section explains the essential functions offered by the system, divided by user types (Student, Teacher, Admin) and AI-driven features. Each feature has a brief explanation of its function and purpose.

1.Student Features

- **Blog Upload**
  - Description: Enables the student to create and submit blogs through a rich-text editor.
  - Inputs: Title of the blog, content, tags, optional images/files.
  - Outputs: Blog goes into a pending status for teacher review.
- **Blog Summarization (AI-Powered)**
  - Description: Automatically produce a summary of a written blog.
  - Inputs: Blog content.
  - Outputs: AI-generated summary presented to the user.
- **Use Chatbot for Doubts**
  - Description: Enables students to pose questions concerning their studies.
  - Inputs: Text question or topic.
  - Outputs: Chatbot reply.
- **Bookmark & Read Later**
  - Description: Allows students to bookmark blogs or study materials.
  - Inputs: Blog ID or material ID.
  - Outputs: Bookmark stored in student profile.

2.Teacher Features

- **Blog Review and Feedback**
  - Description: Permits faculties to review submitted blogs, approve or reject them with feedback.
  - Inputs: Blog ID, approval status, feedback comments.
  - Outputs: Updated blog status and optional feedback message.

- **Upload Study Materials**
  - Description: Faculties can upload PDFs, presentations, or notes for use by students.
  - Inputs: File upload, description, subject, grade level.
  - Outputs: Study material accessible by students.

- **Add Events**
  - Description: Faculties can post forthcoming academic or campus events.
  - Inputs: Event title, description, date/time.
  - Outputs: Event posted on the student dashboard.

3 Admin Features

- **User Management**
  - Description: Enables admins to see, add, remove, or block users (students or faculties).
  - Inputs: User details, action type.
  - Outputs: User data updated within the system.

- **Final Blog Approval**
  - Description: Admins review and approve/reject teacher-approved blogs prior to publication.
  - Inputs: Blog ID, approval status.
  - Outputs: Blog published or rejected.

- **Manage Study Materials**
  - Description: Admins are able to view and manage study materials uploaded by faculties.
  - Inputs: Material ID, action type.
  - Outputs: Material visibility or availability updated.

4. Common Features for All Users

- **User Authentication**
  - Description: Secure signup and login with role-based access (JWT-based).
  - Inputs: Email, password, user role.
  - Outputs: Authenticated session/token.

- **Search & Filter Blogs/Materials**
  - Description: Offers filtered and keyword-based search of blogs, materials, or events.
  - Inputs: Search keyword, filters (author, tags, date).
  - Outputs: List of matching content.

4.1 System Feature 1

4.1.1Description and Priority

Description:

This functionality allows students to author, edit, and submit blogs for review. After submission, the blog is assigned a "pending" state and passed to an instructor for acceptance or rejection.

4.1.2 Stimulus/Response Sequences

Sequence 1: Uploading a New Blog

1. Student logs into the system.
2. Student goes to the "Create Blog" area.
3. Student enters blog title, content, tags, and optionally adds images or files.
4. Student submits.
5. System checks the input.

Sequence 2: Processing Invalid Input

1.Student provides a blog title but does not fill in content.
2.System shows an error: "Blog content cannot be empty."

4.1.3 Functional Requirements

- REQ-1: The system will have a rich-text editor for students to write blog content.
- REQ-2: The system will enable students to enter a blog title, text, tags, and upload optional media files (image/documents).
- REQ-3: The system will check all fields for validity before submission (e.g., the title and text should not be blank).
- REQ-4: The system will save submitted blogs as "Pending" with metadata (author ID, timestamp, status).
- REQ-5: The system should limit submission of blogs to only authenticated users with a "Student" role.

4.2 System Feature 2

Teacher Review and Feedback of Blog

4.2.1 Description and Priority

Description:

Faculties can review blogs posted by students, accept or reject them, and optionally leave feedback. Accepted blogs are sent to the admin for final approval, whereas rejected blogs can be edited and reposted by the student.

4.2.2 Stimulus/Response Sequences

Sequence 1: Approving a Blog

1. Teacher logs in and goes to the "Pending Blogs" area.
2. Teacher opens a submitted blog to view.
3. Teacher clicks on "Approve" and optionally adds a comment.
4. System updates the blog status to "Approved by Teacher".

Sequence 2: Rejecting a Blog with Feedback

- Teacher opens a pending blog.
- Teacher chooses "Reject" and leaves a comment (e.g., make suggestions for improvements).
- System updates the blog status to "Rejected".
- System sends the feedback notification to the student.

4.2.3 Functional Requirements

- REQ-6: The system shall enable faculties to see all blogs in the "Pending" state.
- REQ-7: The system shall enable faculties to approve or reject a blog.

- REQ-8: The system shall enable faculties to input feedback while checking a blog.
- REQ-9: The system shall change the status of the blog to "Approved by Teacher" or "Rejected" accordingly.
- REQ-10: The system shall inform the admin when a blog is approved.

4.3 System Feature 3

4.1.1 Description

Description:

This feature enables efficient sharing and access of study materials among students and faculty. Users can upload, download, and organize notes or study documents.

4.1.2 Stimulus/Response Sequences

Sequence 1: Uploading Study Material

1. Authenticated user logs into the system.
2. User navigates to the "Upload Material" section.
3. User uploads file(s), enters title, description, and selects tags (e.g., subject, topic).
4. User clicks submit.
5. System validates and saves the material.

Sequence 2: Accessing Study Material

1. User logs in and goes to the "Study Materials" section.
2. User uses filters (e.g., subject, semester) or search bar.
3. System displays relevant study materials.
4. User downloads or views the selected material.

4.1.3 Functional Requirements

- REQ-1: The system shall allow uploading of study materials with metadata (title, subject, description, tags).
- REQ-2: The system shall store files (PDFs, DOCs, PPTs, etc.) and associate them with the uploader's ID.
- REQ-3: The system shall allow authenticated users to search/filter materials.
- REQ-4: The system shall only allow verified users to upload content.
- REQ-5: The system shall display study materials in a structured, searchable list.

4.4 System Feature 4

4.2.1 Description

Description:

This feature incorporates a chatbot that assists students in clearing doubts related to subjects, topics, or system usage. The chatbot provides instant responses and can escalate queries to faculty if needed.

4.2.2 Stimulus/Response Sequences

Sequence 1: Student Using Chatbot for Doubt Clearing

1. Student logs in and opens chatbot window.
2. Student types a question (e.g., "Explain Newton’s second law").

3. Chatbot processes the query and provides an answer.
4. If the chatbot is unable to resolve, it provides an option to ask a faculty member.

4.2.3 Functional Requirements

- REQ-1: The system shall integrate an AI-powered chatbot.
- REQ-2: The chatbot shall handle common academic and technical queries.
- REQ-3: The chatbot shall redirect unanswered queries to assigned faculty or support staff.
- REQ-4: The chatbot shall log all interactions for review.
- REQ-5: The chatbot shall be accessible only to authenticated users.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- REQ-NF-1: The system should handle a minimum of 1000 concurrent users (students, faculties, admins) without compromising performance.
- REQ-NF-2: Any user action's average response time (e.g., loading a blog, submitting a form, navigating pages) should not be more than 1 ms for normal load.
- REQ-NF-3: The AI summarization feature in the blog should provide summarized content within 1 ms for a blog.
- REQ-NF-4: Blog submission and review status updates will be displayed in the system within seconds of action completion.
- REQ-NF-5: The system will be available most of the time each month, except during planned maintenance times.

5.2 Security Requirements

The site deals with sensitive educational content and user information (e.g., personal information, study materials, blogs, reviews). Thus, there should be strong security measures to avoid unauthorized access, data exposure, and abuse of privilege.

- REQ-NF-SEC1: The system should ensure secure user authentication through email and password. Passwords should be stored in hashed and salted encryption techniques (e.g., bcrypt).
- REQ-NF-SEC2: The system will use role-based access control (RBAC) to limit system functionality according to user roles (student, teacher, admin).
- REQ-NF-SEC3: All sensitive operations (e.g., login, content submission, admin operations) will be done over HTTPS to provide data encryption in transit.
- REQ-NF-SEC4: The system will offer two-factor authentication (2FA) for admin accounts to provide higher account protection.

5.3 Software Quality Attributes

This section identifies fundamental software quality attributes essential to the platform's success both on a user's and development side.

- **Usability:**
  The platform should be easy to use with straightforward navigation. Every user role (students,

faculties, admins) should be able to accomplish their respective tasks without much training. Interfaces should have a standard and clean design approach.

- **Availability:**
  The system should be accessible round the clock to users for viewing, posting blogs, or adminstering the site.
- **Maintainability**
  The system should be modular and documented, supporting simple bug fixing, feature improvements, and integration with third parties.
- **Scalability:**
  The platform should be horizontally scalable to process increasing numbers of users without affecting performance.
- **Security:**
  Security controls should safeguard against unauthorized use, data breaches, and general web vulnerabilities. Sensitive activity should be tracked and monitored.

5.4 Business Rules

The following business rules state the operational policies and restrictions that control the system behavior. The rules capture the way various user roles use the platform and under what circumstances various operations are possible.

- BR-1: It is permitted for users to submit and create blogs. Once submitted, the blog becomes pending and waits for teacher review.
- BR-2: Faculties may only view, approve, or reject student-submitted blogs. They are required to leave feedback if a blog is rejected.
- BR-3: Admins are the ultimate reviewers who can publish approved blogs. A blog will not be published with or without admin approval after being approved by a teacher.
- BR-4: When a blog is rejected by a teacher or an admin, the student is required to edit and resubmit the blog for another review cycle.

6. Other Requirements

This section summarizes other technical, legal, and operational requirements not covered elsewhere in the Software Requirements Specification.

6.1 Database Requirements

- The site will employ MongoDB as the main NoSQL database to hold user information, blog posts, study materials, feedback, bookmarks, and activity logs.
- All user information will be stored encrypted where appropriate (e.g., passwords with bcrypt).
- The database should be capable of supporting indexing to enable quick search queries (e.g., blog titles, tags).

6.2 Legal and Compliance Requirements

- The system has to follow data privacy regulations, particularly in dealing with and storing user data.
- All the conditions of use, privacy policy, and content policy have to be agreed to by users during account registration.
- Students have to ensure that content uploaded (blogs) is original and not plagiarized.

6.3 Reusability and Extensibility

- The platform will be architectured to enable modular development, such that components like the authentication module, blog editor, and chatbot can be easily reused in subsequent projects.

Appendix A: Glossary

Term	Definition
MERN Stack	full-stack web framework consisting of MongoDB, Express.js, React.js, and Node.js. It is employed in the development of full-stack web applications.
SRS	Software Requirements Specification – a document that states the software system to be implemented, including its functional and non-functional requirements.
User Roles	The user categories interacting with the system: Student, Teacher, and Admin.
Blog	A digital post or content piece written and posted by students to be reviewed and published on the site.
Blog Summarization	An artificial intelligence-based function that creates shorter summaries of blogs to make reading and comprehension easier.
Bookmark	A function for saving a blog to read in the future.
Study Material	Educational materials (PDFs, notes, links, etc.) posted by faculties or admins and made available to students.
Event	Planned educational events (e.g., webinars, announcements) created by faculties or admins for student participation.
Pending State	The status given to a blog post once it is submitted by a student, pending approval by a teacher.
Approved/Rejected Blog	Status given to a blog post upon review. Approved blogs are sent to admin for final review; rejected blogs are sent back to the student with comments.
AI Chatbot	Integrated tool that gives immediate responses to students' academic questions based on natural language processing (NLP).
MongoDB	A NoSQL database utilized for platform data storage like user profiles, blogs, feedback, and events.
Frontend	The component of the platform directly accessed by users, developed using React.js.
Backend	The server-side logic of the platform, developed using Node.js and Express.js.

Appendix B: Analysis Models

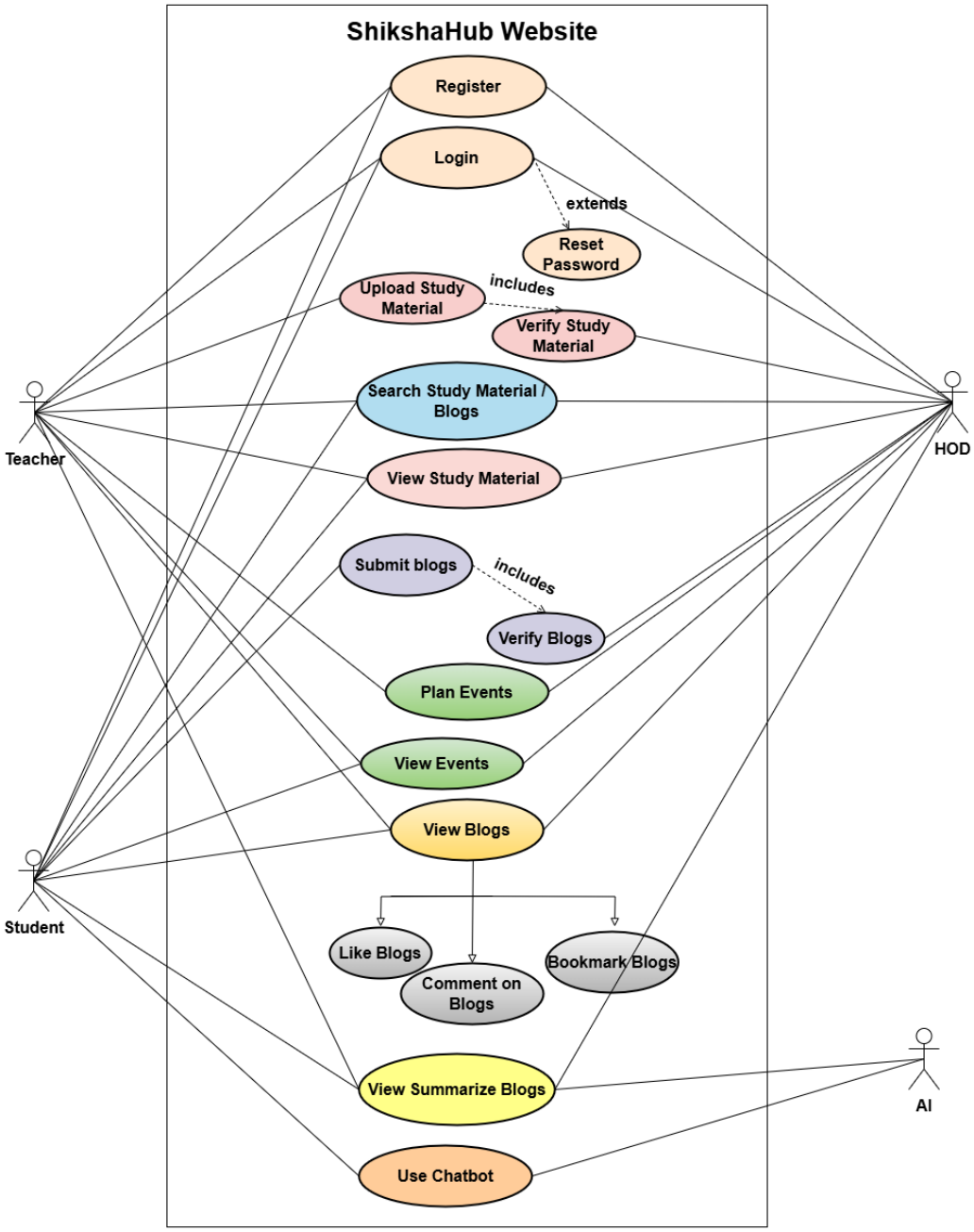
This section holds crucial analysis models that depict the system structure, flow, and behavior. Such models assist developers and stakeholders in understanding the system architecture, interaction, and data relationships.

1. Use Case Diagram
Description: Depicts the key operations of the system and user role interactions.
Actors:

- Student
- Teacher
- Admin

Use Cases:

- Student: Register/Login, Upload Blog, Summarize Blog, Bookmark Blog, Chat with AI, View Study Material, Edit Rejected Blog
- Teacher: Review Blog, Approve/Reject Blog, Upload Study Material, Create Event
- Admin: Manage Users, Review Approved Blogs, Final Publish, Manage Study Materials
- (Let me know if you'd like a visual diagram for this.)

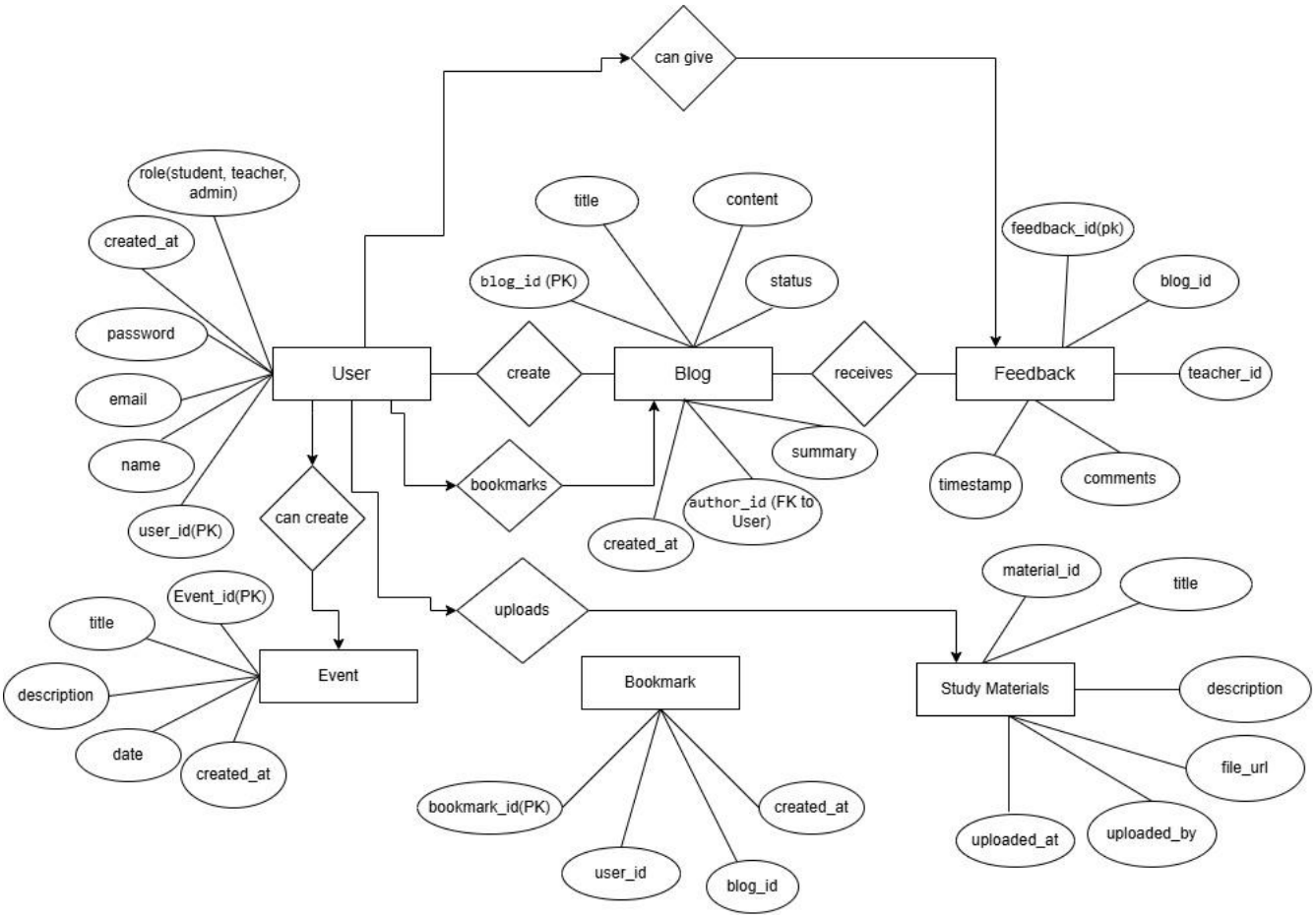


2. Entity-Relationship Diagram (ERD)
Entities:

- User (userID, name, email, password, role)
- Blog (blogID, title, content, userID, status, summary, timestamp)
- StudyMaterial (materialID, title, description, fileURL, uploadedBy)
- Event (eventID, title, date, description, createdBy)
- Feedback (feedbackID, blogID, facultyID, comment, timestamp)
- Bookmark (bookmarkID, userID, blogID, createdAt)

Relationships:

- One User can write many Blogs
- One Blog may have one or more Feedback entries
- One Teacher may upload numerous StudyMaterials and Events
- One User may bookmark numerous Blogs



3. Process Flow Diagram