IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

CodeHub: Smart Repository and Version Control Platform

Asst. Prof. V. C. Patil, Shubham Daulati Kamble, Prof. M. S. Bhandigare
Master of Computer Applications (MCA)
Head of Department
Industry Sponsor: QUESTIT PVT. LTD.
Sant Gajanan Maharaj College of Engineering (SGMCOE), Mahagaon
Shivaji University, Kolhapur, Maharashtra, India
vaibhavpatil8743@gmail.com, kambleshubham8975@gmail.com

Abstract: *With the increasing demand for collaborative software development, developers require efficient platforms that support version control, code management, and team coordination in a structured manner. Traditional file-sharing methods fail to provide proper version tracking, collaboration support, and transparency, leading to issues such as code conflicts, data inconsistency, and reduced productivity. This paper presents CodeHub — a web-based repository hosting and version control platform that integrates core development functionalities with an intelligent user guidance system to improve usability and workflow efficiency. The platform includes modules for repository management, file handling, commit tracking, issue management, and activity monitoring, all accessible through a unified dashboard. In addition, a context-based guidance module assists users by providing real-time suggestions, usage hints, and step-by-step instructions based on their interactions within the system. This helps new users understand platform operations quickly and reduces errors during development tasks. The system is developed using Node.js, Express.js, and MongoDB, ensuring scalability and efficient data management. Evaluation shows improved user interaction, faster task completion, and reduced learning time for beginners. These results establish CodeHub as a practical and user-friendly solution for modern collaborative software development environments.*

Keywords: Repository Hosting, Version Control, Developer Collaboration, Commit Tracking, Issue Management, Node.js, MongoDB, Web Application, User Guidance System

I. INTRODUCTION

The software development industry has come under increasing pressure as collaborative coding practices have expanded rapidly and expectations around efficient code management and transparency have grown considerably. Today's developers require more than simple file storage — they expect structured platforms that provide clear version tracking, real-time updates on project activity, and reliable mechanisms for managing contributions from multiple users. However, many small teams and student developers are not equipped to meet these expectations. Their workflows often rely on manual file sharing, unorganized storage, and limited collaboration tools, which leads to version conflicts, loss of previous code, and difficulty in tracking development progress. Additionally, beginners frequently struggle to understand system workflows, and a significant amount of time is spent resolving confusion rather than focusing on development tasks.

CodeHub was developed to address these specific gaps. The project was completed during the 2025–26 academic year at Sant Gajanan Maharaj College of Engineering (SGMCOE), Mahagaon, under the guidance of Asst. Prof. V. C. Patil. The system is designed as a web-based repository hosting and collaborative version control platform that simplifies development workflows while improving usability through

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 18, April 2026 Impact Factor: 8.2

an integrated intelligent user guidance feature. All modules are combined into a single platform that supports every stage of the development process — from repository creation to project collaboration and version tracking.

The primary contributions of this work are:

• Repository Management System — structured creation and organization of repositories with controlled access and efficient storage.

• Version Control (Commit Tracking) — complete tracking of code changes with commit history, timestamps, and user details.

• Collaboration & Issue Management — support for multi-user development, issue tracking, and task coordination.

• Intelligent User Guidance System — real-time suggestions and step-by-step assistance to help users understand and use the platform effectively.

All modules share a unified MongoDB database and Node.js application layer, ensuring seamless data flow across repository management, version tracking, collaboration, and user guidance without duplication.

II. RELATED WORK

Prior research on version control systems and repository hosting platforms provides a strong foundation for the design decisions made in CodeHub. Smith [1] demonstrated that distributed version control systems significantly improve collaboration efficiency by allowing multiple developers to work on the same codebase without conflicts, supporting parallel development and faster integration. Johnson [2] found that repository hosting platforms with integrated issue tracking and collaboration tools enhance project transparency and improve coordination among team members, leading to more reliable development outcomes. Gupta [3] identified code management and version tracking as critical aspects of modern software development, emphasizing that structured repositories and commit histories reduce errors and improve maintainability. Patel [4] presented a technical model for web-based collaboration systems using RESTful APIs and scalable database architectures, showing that efficient data handling and asynchronous communication can support real-time updates and smooth user interaction. Brown [5] provides a detailed study of secure authentication mechanisms, including token-based systems and role-based access control, which informed the secure login and user management design of the CodeHub platform.

Several earlier web-based development platforms — including those reviewed in [6][7][8] — identified recurring limitations such as complex user interfaces, lack of proper guidance for beginners, and heavy reliance on external documentation for understanding workflows. CodeHub directly addresses these limitations by integrating an intelligent user guidance system along with repository management, version tracking, and collaboration features, providing a more accessible and user-friendly development environment.

III. PROBLEM STATEMENT

The challenges faced by developers and small development teams in managing software projects can be grouped into several observable problem areas. First, project files shared through traditional methods are often unorganized and lack proper version tracking, leading to confusion and accidental overwriting of important code. Second, without a centralized repository system, developers have no structured way to manage their projects or maintain a clear history of changes, making it difficult to track progress and revert to previous versions when needed. Third, collaboration among multiple contributors is often inefficient due to the absence of integrated tools for issue tracking and task coordination, resulting in communication gaps and delays in development. Fourth, many existing platforms are complex and require prior knowledge, causing beginners to struggle with understanding workflows such as commits, repository management, and collaboration processes. Fifth, the lack of real-time guidance within the system forces users to depend on external documentation or tutorials, increasing the time required to perform even basic operations.

CodeHub addresses each of these limitations in a structured manner. Users can create and manage repositories, track changes through commits, collaborate with team members, and receive real-time guidance while performing tasks —

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

all within a single platform. Developers gain a clear view of project activity, improved coordination, and a more efficient and user-friendly development workflow

IV. PROPOSED SYSTEM OVERVIEW

The platform connects developers, contributors, and administrators through a browser-based interface backed by Node.js and MongoDB. All data is stored in a centralized database that every module can access. Each development activity moves through a defined workflow:

• Step 1: Registration & Authentication — Users create accounts and log in through a secure authentication system using encrypted credentials and token-based session management.

• Step 2: Repository Creation — Users create repositories by providing project details; the system assigns a unique repository structure automatically.

• Step 3: File Management — Users upload, update, or delete files within repositories, ensuring proper organization and storage of project data.

• Step 4: Version Control (Commit Tracking) — Changes made to files are recorded as commits with messages, timestamps, and user details, maintaining a complete history of development.

• Step 5: Collaboration & Issue Management — Contributors can work together, report issues, assign tasks, and track project progress within the system.

• Step 6: Intelligent User Guidance — The system provides real-time suggestions and contextual hints to assist users in performing tasks and understanding workflows.

• Step 7: Repository Dashboard — Repository activity, recent commits, issues, and user contributions are displayed in a structured and user-friendly dashboard.

V. SYSTEM ARCHITECTURE

The platform is structured across multiple tiers, each with a clearly defined responsibility. The presentation tier delivers the user interface through responsive web pages built using HTML, CSS, JavaScript, and Bootstrap, with asynchronous requests maintaining smooth interaction between user actions and system responses. The application tier processes HTTP requests through Node.js and Express.js, applying all business logic before interacting with the database or returning responses to the client. The repository management tier houses the core functional modules, including user authentication, repository management, file handling, version control, collaboration, and issue tracking. The guidance tier provides real-time user assistance by analysing user actions and generating contextual suggestions to improve usability and workflow understanding. Finally, the database tier persists all system data in MongoDB, ensuring efficient storage and retrieval of user information, repositories, commits, and activity logs. The complete system architecture is illustrated in Fig. 1 below.

Tier	Layer	Technology
1	Presentation	HTML5, CSS3, JavaScript, Bootstrap, React.js
2	Application	Node.js, Express.js
3	Repository Management	User Authentication, User Profile, Repository, File Management, Version Control, Collaboration, Issue Tracking, Activity Monitoring, Dashboard Modules
4	Guidance System	Intelligent User Guidance Module (Rule-Based Suggestions, Contextual Help System)
5	Database	MongoDB, Mongoose ODM

TABLE I. Five-Tier Architecture

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

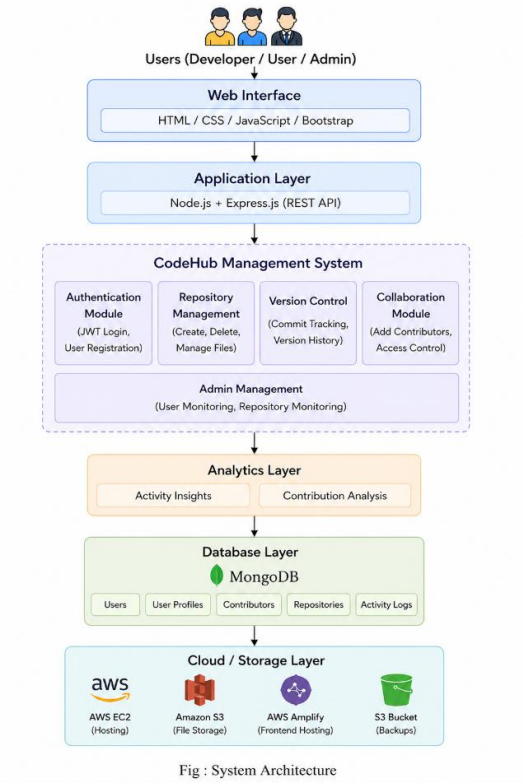

Fig : System Architecture

Fig. 1. System Architecture of CodeHub

VI. MODULE DESCRIPTIONS

A. User Authentication & Security Module — S. D. Kamble

Instead of allowing open access to the platform, this module ensures that every user interacting with the system is properly verified and authenticated. When a new user registers, their credentials such as username, email, and password are securely stored after encryption. During login, the system validates the credentials and generates a JSON Web Token (JWT), which is used to maintain secure sessions across the platform. This approach prevents unauthorized access and protects sensitive repository data. By integrating token-based authentication, the module provides a reliable security layer that supports safe collaboration among developers while maintaining data privacy throughout the system.

B. Repository & File Management Module — S. D. Kamble

Managing project files manually across different systems often leads to confusion and data loss. This module provides a centralized space where users can create repositories and manage all their project files efficiently. Each repository acts as a structured container where users can upload, update, or delete files as needed. All file operations are recorded systematically, ensuring that no changes are lost during development. The system maintains proper organization of source code and related resources, making it easier for developers to handle complex projects. This module forms the backbone of the platform by enabling structured storage and seamless file handling within repositories.

C. Version Control & Commit Tracking Module — S. D. Kamble

Tracking changes in a project is essential, especially when multiple updates are made over time. This module simulates version control functionality by recording every change as a commit. Each commit includes details such as commit

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

message, timestamp, and user information, allowing developers to maintain a complete history of their work. Instead of overwriting files, the system preserves previous versions, making it possible to review or restore earlier states of the project. As development progresses, this module provides clear visibility into how the project evolves, helping developers manage changes effectively and avoid conflicts or accidental data loss.

D. Collaboration & Issue Management Module — S. D. Kamble

Software development is rarely a solo process, and effective collaboration is key to project success. This module allows multiple users to work together within the same repository environment. Contributors can raise issues, report bugs, and suggest improvements through a structured issue management system. Each issue includes a title, description, status, and assigned user, ensuring that tasks are properly tracked and resolved. Communication between contributors becomes more organized, reducing misunderstandings and improving workflow efficiency. This module creates a collaborative environment where development activities are transparent and well-coordinated.

E. Activity Tracking & Analytics Module — S. D. Kamble

Understanding how a project is progressing requires proper tracking of user activities. This module records actions such as repository creation, file updates, commits, and issue handling. The collected data is then analysed to provide meaningful insights into user contributions and overall project activity. Instead of manually checking progress, users can view summarized information through the system dashboard. These insights help in identifying active contributors, monitoring development patterns, and improving productivity. Over time, the analytics become more valuable as more data is collected, offering a clearer picture of system usage and performance.

F. Dashboard & System Overview Module — S. D. Kamble

Rather than navigating through multiple pages, this module provides a centralized dashboard where users can access all important information in one place. It displays repositories, recent commits, issues, and activity logs in a simple and organized format. Users can quickly monitor their projects and track ongoing development work without unnecessary complexity. The dashboard is designed to improve usability by presenting key data clearly, helping users make quick decisions and stay updated with their work. This module enhances the overall user experience by making the platform more interactive and efficient.

G. Deployment & Cloud Integration Module — S. D. Kamble

To ensure that the platform is accessible and scalable, this module handles deployment on cloud infrastructure. The frontend of the application is deployed using AWS Amplify, while the backend runs on AWS EC2, and file storage is managed through AWS S3. This setup ensures high availability and reliable performance even as the number of users grows. Instead of running the system locally, cloud deployment allows users to access the platform from anywhere with an internet connection. This module plays a crucial role in making the system production-ready and capable of handling real-world usage.

H. Admin Monitoring & Control

Module — S. D. Kamble

Maintaining system integrity requires continuous monitoring, which is handled by the module. The administrator has access to a dedicated panel where user activities, repositories, and system operations can be observed and managed. Admins can monitor user accounts, review repository content, and track issues and collaboration activities. If any unusual behaviour is detected, appropriate actions can be taken to maintain platform security. This module ensures that the system operates smoothly while enforcing proper usage policies and maintaining a controlled development environment.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

VII. IMPLEMENTATION

A. Architecture and Stack

The platform follows a structured client-server architecture where the frontend and backend communicate through REST APIs.

The backend is developed using Node.js with Express.js, which handles all application logic, request processing, and data flow between modules. The frontend is built using HTML, CSS, JavaScript, and Bootstrap, providing a responsive and user-friendly interface. All modules of the system were developed using

Backend is developed using Node.js with Express.js which handles all application logic, request processing, and data flow between modules. The frontend is built using HTML, CSS, JavaScript, and Bootstrap, providing a responsive and user-friendly interface. All modules of the system were developed using Visual Studio Code and tested on Windows 10 environment. MongoDB is used as the database for storing user data, repositories, commits, and activity logs. The system is deployed on cloud

infrastructure to ensure scalability and accessibility, AWS EC2 is used for backend hosting, AWS S3 is used for storing repository files, and AWS Amplify is used to deploy the frontend. The interface is designed to respond dynamically using JavaScript and AJAX, reducing page reloads and improving performance. This architecture ensures smooth communication between components and supports efficient handling of multiple users and repositories.

Component	Specification
OS	Windows 10 / Linux
Backend	Node.js, Express.js (REST API)
Frontend	HTML5, CSS3, Bootstrap, JavaScript, AJAX
Database	MongoDB
Cloud	AWS EC2, AWS S3, AWS Amplify
IDE / Tools	Visual Studio Code
Client HW	Min. 2 GHz CPU, 4 GB RAM
Server HW	4 GB RAM, 20 GB storage
Security	JWT Authentication, Password Encryption

TABLE II. Hardware and Software Specifications

B. System and Integration Components

The system is designed to manage repositories and track development activities efficiently without relying on external platforms. All repository data, commit history, and user activities are stored within MongoDB, ensuring fast data retrieval and flexibility. The version control mechanism records every file change as a commit, maintaining a structured history of the project. This allows users to track modifications and manage development progress effectively.

File storage is handled through AWS S3, where repository files and related data are securely stored and retrieved when required. The authentication system uses JSON Web Tokens (JWT) to manage user sessions securely and prevent unauthorized access. Activity tracking is implemented to monitor user actions such as commits, file updates, and issue handling, which helps in analysing user contributions and system usage. The integration between frontend and backend is handled through REST APIs, ensuring smooth data exchange and real-time updates. AJAX is used to send and receive data without reloading pages, improving user experience. Overall, the system components work together to provide a reliable, scalable, and efficient platform for repository management and collaborative development.

VIII. SYSTEM ANALYSIS

1) Efficient Repository Management and Visibility:

The system integrates all repository-related operations such as file handling, commits, and issue tracking into a centralized database and dashboard. Instead of manually checking different sections, users can view repository activity, recent commits, and project updates in one place. This improves visibility of ongoing development work and helps

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

users quickly understand project status. By providing a structured overview, the system reduces confusion and makes repository management more organized and efficient.

2) Accuracy in Version Tracking and Data Consistency:

The version control mechanism ensures that every change made to a file is recorded as a commit with proper details such as message, timestamp, and user information. During testing, the system maintained consistent tracking of file updates without data loss or duplication. This reduces the chances of overwriting important changes and helps in maintaining a reliable history of the project. Compared to manual file management, this approach improves accuracy and ensures better control over project versions.

3) Secure Authentication and Access Control:

The platform uses JWT-based authentication to ensure that only authorized users can access the system. Each user can securely log in and manage their own repositories and data. Passwords are encrypted before storage, and session handling prevents unauthorized access. The admin module has additional privileges to monitor users and repositories without interfering with normal user operations. This controlled access mechanism ensures data security and prevents misuse of the platform.

4) Scalability and Modular Design:

The system is designed in a modular way where each component, such as authentication, repository management, and collaboration, works independently while sharing a common database. This allows easy modification or enhancement of individual modules without affecting the entire system. The use of Node.js and cloud deployment ensures that the system can handle multiple users and repositories efficiently. Testing with multiple operations showed stable performance without data inconsistency, indicating that the system can scale as usage increases.

IX. RESULTS AND DISCUSSION

The platform was evaluated based on functionality, performance, and usability across all core modules. The results show that the system effectively improves repository management, version tracking, and collaboration compared to traditional file-sharing methods. The centralized dashboard reduces the effort required to monitor project activity, while the version control system ensures proper tracking of changes without data loss. Secure authentication and cloud deployment further enhance reliability and accessibility of the platform.

The system performs efficiently in handling repository operations, file updates, and commit tracking with minimal delay. User interactions such as repository creation, file upload, and issue management were executed smoothly during testing. The use of AJAX improves responsiveness by reducing full page reloads, resulting in a better user experience. Overall, the system demonstrates stable performance under multiple user operations and maintains consistency in data handling.

Module	Metric	Result
Repository Management	Repository creation time	< 2 seconds
File Management	File upload latency	< 1.5 seconds
Version Control	Commit recording time	< 1 second
Dashboard	Load time	< 2 seconds (moderate data)
Authentication	Login response time	< 1 second
Activity Tracking	Log update delay	Real-time
Database	Query response	Fast with no data loss
Concurrency	Simultaneous users	40–50 users (stable)

TABLE IV. Performance and Evaluation Summary

The modular design of the system allows gradual enhancement and easy maintenance. Individual components such as repository management, version control, and authentication can be improved or extended without affecting the overall

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 18, April 2026 Impact Factor: 8.2

system. This flexibility makes the platform suitable for future upgrades and scaling based on user requirements. The integration of cloud services ensures that the system remains accessible and reliable even with increasing data and user load.

Future improvements can focus on enhancing collaboration features, adding advanced analytics for user activity, and developing a mobile-friendly interface for better accessibility. The system can also be extended with additional features such as real-time notifications and improved visualization dashboards to further enhance user experience and productivity.

X. PROJECT TIMELINE

Month	Activity
Dec 2025	Problem definition, literature survey, and topic finalization
Jan 2026	Requirement gathering, system analysis, and architecture design
Feb 2026	Frontend (HTML, CSS, JavaScript) and backend (Node.js, Express.js) development, database setup (MongoDB)
Mar 2026	Implementation of repository management, version control, authentication, and collaboration modules, integration and testing
Apr 2026	Deployment on AWS (EC2, S3, Amplify), documentation, presentation preparation, and final submission

TABLE V. Project Timeline — Academic Year 2025–26

XI. CONCLUSION

CodeHub demonstrates that the common challenges faced in software development — such as lack of proper version tracking, difficulty in collaboration, data inconsistency, and inefficient project monitoring — can be effectively addressed through a well-structured, web-based repository management platform. The system developed provides a centralized environment where users can manage repositories, track changes through commits, and collaborate efficiently without relying on traditional file-sharing methods.

The results achieved through this project are significant in terms of usability and performance. The system ensures accurate version tracking of files, secure user authentication using JWT, and smooth handling of repository operations with minimal delay. The dashboard provides clear visibility of project activity, while activity tracking helps in understanding user contributions. Together, these features improve overall development efficiency and maintain a structured workflow for managing projects.

The modular design of the platform allows it to be extended and improved without affecting existing functionality. New features can be added gradually, such as advanced analytics, real-time notifications, or enhanced collaboration tools, depending on user requirements. The use of cloud deployment further ensures scalability and accessibility, making the system suitable for real-world applications.

Future enhancements may include integrating more advanced visualization for analytics, developing a mobile-friendly interface, and improving collaboration features for larger development teams. Overall, the project establishes a strong foundation for a scalable and efficient repository management system that supports modern software development practices.

REFERENCES

[1] R. S. Pressman and B. R. Maxim, Software Engineering: A Practitioner's Approach, 9th ed. McGraw-Hill, 2020.
[2] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994.
[3] S. Chacon and B. Straub, Pro Git, 2nd ed. Apress, 2014.
[4] D. Flanagan, JavaScript: The Definitive Guide, 7th ed. O'Reilly Media, 2020.
[5] M. Cantelon, M. Harter, T. Holowaychuk, and N. Rajlich, Node.js in Action. Manning Publications, 2017.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 18, April 2026
Impact Factor: 8.2

[6] K. Banks, "MongoDB Basics and Application Development," Int. J. Database Systems, vol. 10, no. 2, pp. 45–52, 2021.
[7] G. Reese, Cloud Application Architectures: Building Applications and Infrastructure in the Cloud. O'Reilly Media, 2019.
[8] Amazon Web Services, "AWS Documentation," 2024 [Online]. Available: https://docs.aws.amazon.com
[9] OpenJS Foundation, "Node.js Documentation," 2024. [Online]. Available: https://nodejs.org
[10] MongoDB Inc., "MongoDB Documentation," 2024 [Online]. Available: https://www.mongodb.com/docs/
[11] Bootstrap Team, "Bootstrap 5 Documentation," 2024. [Online]. Available: https://getbootstrap.com
[12] GitHub Inc., "GitHub Documentation," 2024. [Online]. Available: https://docs.github.com
[13] QuestIT Pvt. Ltd., "Project Sponsorship and Development Support — CodeHub," 2026.