International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.054 ISSN: 2581-4621

TASK MATE – A TASK MANAGEMENT SYSTEM

A. Vikram Reddy[1], P. Akshitha[2], P. Bhanu Prakash[3], S. Prashanth Yadav[4], K. Kishore[5]
[1,2,3,4]UG Student, Dept of CSE (Data Science), Vidya Jyothi Institute of Technology, Hyderabad, Telangana, India
[5]Associate Professor, Dept of CSE (Data Science), Vidya Jyothi Institute of Technology, Hyderabad, Telangana, India

Abstract: The Task Management System is a role-based web application developed to streamline task allocation, tracking, and management within an organization. The system consists of two primary roles: Admin and User. The Admin has full control over task operations — creating new tasks, assigning them to users, prioritizing them as High, Medium, or Low based on due dates, and managing user accounts.
The User can log in to the system to view their assigned tasks, update task status, and delete completed tasks. This project is implemented using the MERN stack — MongoDB, Express.js, React.js, and Node.js — which provides a powerful and flexible full-stack JavaScript environment. The frontend, built with React.js, offers an interactive and responsive interface, while the backend, developed with Node.js and Express.js, ensures secure communication and efficient data handling. MongoDB is used to store user credentials, task details, priorities, and due dates in a scalable NoSQL database.
By integrating authentication, role-based access control, and task prioritization, the system enhances coordination between administrators and team members. It helps organizations maintain transparency, monitor progress, and ensure timely completion of assigned tasks. This project demonstrates the practical use of modern web technologies to improve task organization and overall productivity in a collaborative environment.
To evaluate the functionality and performance of the Task Manager Web Application, a detailed case study was conducted in a controlled local development environment. The study aimed to assess the system's usability, efficiency, and reliability in managing user authentication and task operations through a web-based interface.
Keywords: User Authentication, Task Management, REST API, Client-Server Architecture, CRUD Operations, Database Integration, Responsive Interface, Localhost Deployment

1. INTRODUCTION

In today's digital era, effective task organization has become essential for improving productivity, time management, and work efficiency. Individuals, students, and professionals often need a simple and reliable system to manage daily activities, track pending work, and organize responsibilities in a structured manner. Traditional methods such as handwritten notes or basic reminder tools are often limited in terms of accessibility, data management, and flexibility. To overcome these limitations, web-based task management systems have emerged as a practical solution for handling tasks in a more organized and efficient way.
The Task Manager project is a web-based application developed to help users create, manage, update, and delete tasks through an interactive and user-friendly interface. The system is designed to simplify task handling by providing a centralized platform where users can securely access their tasks and monitor their work progress. It combines modern web development technologies to deliver a responsive and efficient application that supports real-time interaction between the user interface, server, and database.
The project is implemented using a three-tier architecture consisting of the frontend, backend, and database layers. The frontend is developed using React.js, which provides a dynamic and responsive user interface for user interaction. The backend is built with Node.js and Express.js, which handle business logic, API routing, and request processing. MongoDB is used as the database to store user information and task-related data securely and efficiently. Through HTTP-based communication, the frontend and backend interact seamlessly, allowing users to perform operations such as

DOI:10.46647/ijetms.2026.v10i02.054 ISSN: 2581-4621

registration, login, task creation, task updating, and task deletion.

One of the major features of the Task Manager system is user authentication. This ensures that only registered users can access their accounts and manage their tasks. After successful login, users can view their task dashboard, add new tasks, edit existing ones, mark tasks as completed, and remove unnecessary tasks. This functionality improves personal organization and supports better planning of daily or long-term activities. The application also offers a clean and responsive interface, making it suitable for use on desktops and laptops.

From an academic and technical perspective, this project demonstrates the practical application of full-stack web development concepts. It includes database integration, client-server communication, RESTful API usage, authentication mechanisms, and CRUD operations. The project serves as a useful example of how modern web technologies can be combined to solve real-world problems efficiently.

In conclusion, the Task Manager project is developed as a simple yet effective productivity tool that helps users manage their tasks in a structured way. It not only addresses the need for better task organization but also provides valuable insight into the implementation of modern web applications. The project can further be extended with additional features such as deadlines, reminders, task prioritization, team collaboration, and cloud deployment, making it more powerful and suitable for real-world usage.

2. PROBLEM STATEMENT

The growing dependence on digital tools for personal and professional productivity has increased the need for efficient task management systems. Many individuals still rely on traditional methods such as handwritten notes, scattered reminders, or basic mobile applications that provide limited functionality. These approaches often lack proper organization, secure access, real-time updates, and centralized storage, which makes task tracking difficult and inefficient.

One of the major problems in conventional task handling is the absence of a unified and structured platform for managing daily activities. Users often face difficulty in creating, updating, organizing, and monitoring tasks effectively. In many cases, tasks are forgotten, duplicated, or left incomplete due to the lack of reminders, clear categorization, and proper tracking mechanisms.

The current task management environment suffers from several key limitations:

- Lack of centralized task organization - Users often depend on multiple tools or manual methods to track tasks.
- Inefficient task tracking - It becomes difficult to monitor pending, completed, or updated tasks systematically.
- Limited accessibility - Traditional methods do not provide easy access across digital platforms.
- Poor data persistence - Without database support, task records may be lost or difficult to retrieve.
- Weak user security - Many simple systems do not provide authentication or secure access to personal task data.

Another important issue is the lack of personalized access control. Without authentication, task records cannot be securely associated with individual users, which reduces privacy and reliability. Users require a system where they can log in securely and manage their own tasks independently.

From a technical perspective, there is a need for a web-based task management system that integrates frontend interaction, backend processing, and database storage into a single platform. Such a system should support secure user authentication, efficient CRUD operations, and responsive user experience.

Therefore, there is a clear need for a lightweight, secure, and user-friendly Task Manager application that allows users to register, log in, and manage tasks efficiently through a modern web interface.

3. METHODOLOGY

The development of the Task Manager application follows a structured and modular methodology

DOI:10.46647/ijetms.2026.v10i02.054 ISSN: 2581-4621

that integrates frontend development, backend API design, database connectivity, and user authentication. The methodology is based on a three-tier web application model, enabling organized development and smooth interaction between all system components.

3.1 Design Philosophy

The core design philosophy of the proposed system is based on building a simple, responsive, and secure web-based task management platform. The system is designed to ensure:

- Simplicity for easy task handling
- Modularity for independent frontend and backend development
- Data persistence through database integration
- Secure access through authentication
- Responsiveness across devices

The project adopts a layered architecture where the frontend, backend, and database work together through APIs.

3.2 Layered System Model

The proposed methodology follows a multi-layered system model, where each layer performs a specific function in the application:

1. PresentationLayer

The frontend provides the user interface for registration, login, and task operations.

2. ApplicationLayer

The backend processes requests, applies business logic, and handles authentication and task management functions.

3. DataLayer

MongoDB stores user information and task data securely and persistently.

3.3 Communication Model

The system employs a request-response communication model to handle user interactions efficiently:

3.3.1 Frontend requests are sent to the backend using HTTP methods such as GET, POST, PUT, and DELETE.

3.3.2 Backend services process the incoming requests and communicate with MongoDB for storing and retrieving data.

3.3.3 Responses are returned in JSON format, enabling smooth integration between the client and server.

This communication model ensures efficient and reliable data exchange between the system components.

3.4 Data Management and Processing

The system uses MongoDB as the primary database for storing application data.

3.4.1 User records such as account details and login credentials are stored securely.

3.4.2 Task records including title, description, status, and updates are stored in the database.

3.4.3 Data processing includes authentication, task creation, retrieval, updating, and deletion.

This structure ensures proper organization, persistence, and retrieval of data.

4. IMPLEMENTATION THEORY

The implementation of the Task Manager application translates the proposed design into a functional and interactive system. The implementation is based on a full-stack web development approach, integrating React for the frontend, Node.js and Express.js for the backend, and MongoDB for the database.

4.1 System Implementation Overview

The system is implemented using a client-server model, where the frontend interacts with the backend through API requests. The backend processes business logic and communicates with MongoDB to store and retrieve user and task data.

The implementation focuses on:

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.054 ISSN: 2581-4621

4.1.1 SecureUserAuthentication
4.1.2 EfficientTaskManagementOperations
4.1.3 ReliableDatabaseConnectivity
4.1.4 Responsive user interface design

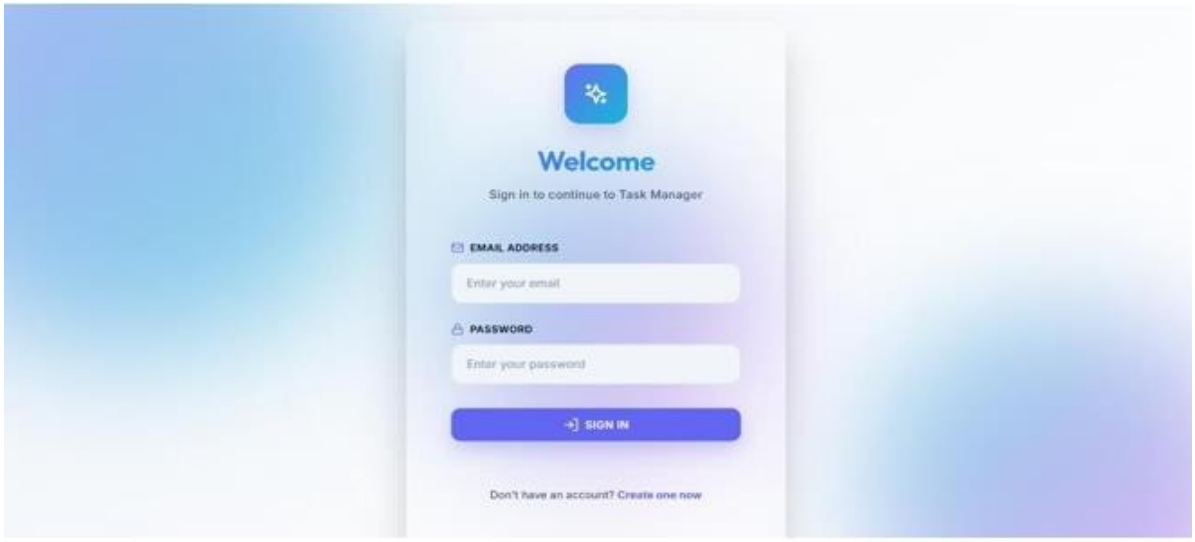
4.2 Frontend Implementation

The frontend layer is developed using React.js. It provides a dynamic and responsive user interface that allows users to interact with the application easily.

Key frontend components include:

4.2.1 Registrationpage
4.2.2 Loginpage
4.2.3 Taskdashboard
4.2.4 Taskcreationandeditinginterface
4.2.5 Task deletion and display modules

The frontend communicates with the backend using HTTP requests and updates the user interface dynamically based on responses.



4.3 Backend Implementation

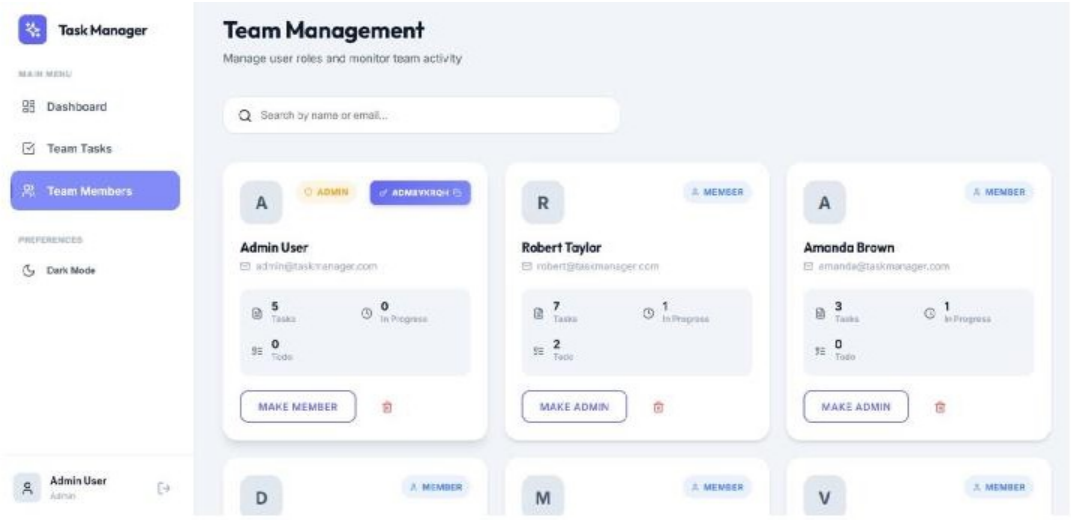
The backend is implemented using Node.js with Express.js. It manages routing, authentication, request handling, and communication with the database.

Core backend modules include:

- User Authentication Module - Handles registration and login validation
- Task Management Module - Handles task creation, update, deletion, and retrieval
- Database Connectivity Module - Manages interaction with MongoDB
- Error Handling Module - Handles invalid input, failed requests, and connection issues

Each module is implemented to ensure modularity, maintainability, and proper data flow.

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.054 ISSN: 2581-4621

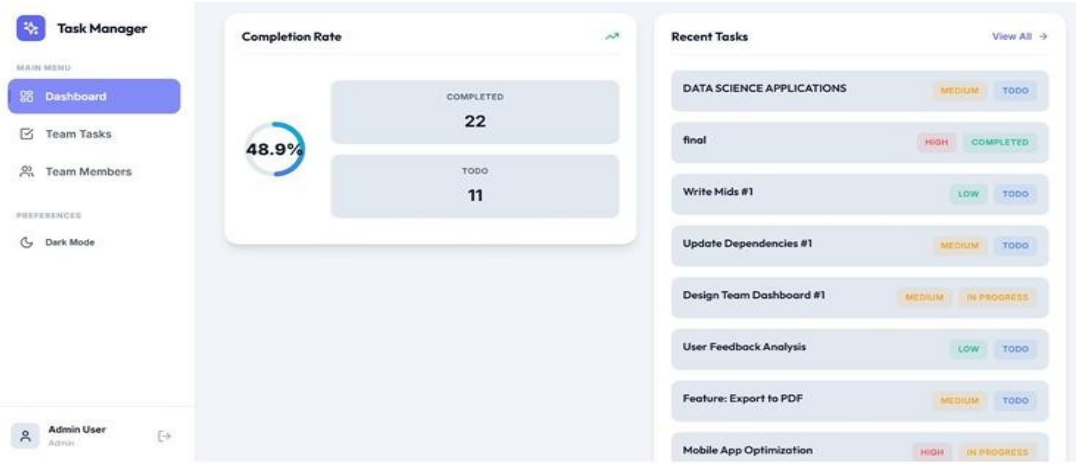


4.4 Integration of Client-Server Communication

The interaction between the client and server is facilitated through HTTP requests and responses.

- Registration and login are performed using POST requests
- Task retrieval is handled using GET requests
- Task updates are handled using PUT requests
- Task deletion is handled using DELETE requests

Express.js manages routing, request parsing, and response generation, ensuring stateless communication between the frontend and backend.



5. RESULTS

The performance of the Task Manager application was evaluated through local testing and practical user interaction scenarios. The evaluation focused on functional correctness, response behaviour, system reliability, and user experience.

5.1 Experimental Setup

The experimental evaluation was conducted using the following setup:

- Deployment on localhost environment
- Frontend tested through web browsers such as Chrome and Edge
- Backend executed using Node.js and Express.js
- MongoDB used as the database
- Testing performed on Windows environment
- Operations tested with different user inputs and task scenarios

DOI:10.46647/ijetms.2026.v10i02.054 ISSN: 2581-4621

This setup ensured that the application was evaluated under realistic local usage conditions.

5.2 Performance Metrics

The system was evaluated based on the following metrics:

5.2.1 Response time - Time taken for frontend-backend interaction

5.2.2 Functional correctness - Accuracy of login and task operations

5.2.3 Reliability - Consistency of system behaviour

5.2.4 Data persistence - Correct storage and retrieval from MongoDB

5.2.5 User experience - Ease of interaction with the interface

5.3 Authentication Performance

The authentication module demonstrated correct behaviour under normal conditions:

5.3.1 Users were able to register successfully with valid credentials

5.3.2 Duplicate accounts were detected and prevented

5.3.3 Registered users were able to log in and access their task dashboard This confirmed the reliability of the authentication process.

5.4 Task Management Performance

The task module was evaluated for task operations:

5.4.1 Tasks were added successfully and displayed correctly

5.4.2 Existing tasks were updated accurately

5.4.3 Deleted tasks were removed properly from both UI and database

5.4.4 Task retrieval remained consistent after repeated operations

These results indicate that the system can manage task data effectively.

5.5 User Interface Evaluation

The user interface was found to be simple, responsive, and easy to use. Users could navigate between registration, login, and task management pages without difficulty. The layout supported clear interaction and improved usability.

5.6 Outcome

The results confirm that the Task Manager application effectively meets its objectives of providing a secure, reliable, and user-friendly task management platform. The system demonstrated correct frontend-backend integration, stable database connectivity, and efficient handling of user and task operations.

6. CONCLUSION

This project presented a Task Manager web application designed to address the problem of inefficient and unstructured task handling. By integrating user authentication, task creation, updating, deletion, and retrieval into a single web platform, the proposed system provides a simple and effective solution for personal productivity and task organization.

The system adopts a three-tier architecture with React.js for the frontend, Node.js and Express.js for the backend, and MongoDB for the database. This architecture ensures modularity, maintainability, and reliable communication between system components. Experimental testing confirmed that the application performs task operations correctly and provides a smooth user experience in a local environment.

In addition to helping users manage tasks efficiently, the project also demonstrates the practical application of full-stack web development concepts such as authentication, REST API communication, CRUD operations, and database integration.

Although the current implementation is functional, it can be enhanced further in future versions. Possible improvements include task prioritization, reminders, deadline notifications, collaboration features, and cloud deployment for broader accessibility.

In conclusion, the Task Manager application provides a strong foundation for further development and serves as an effective academic and practical web development project.

DOI:10.46647/ijetms.2026.v10i02.054 ISSN: 2581-4621

7. REFERENCES

[1] Vashishtha et al., "Empowering Student Success: A Comprehensive Task Management and Notification System" (2024)
[2] Raksha Thakur et al., "Task Management System using AI Prioritizations" (2024)
[3] Mr. May Sila et al., "Task Management System (Student Project)" (2024)
[4] M. Sravanth & R. Dhanush, "Intelligent Task Management System" (2023)
[5] M. Spezie, "The Development of a Task Management Software" (2023)
[6] Shah et al., "Web-Based Task Management System for Improving Group Work Collaboration" (2022)