Task Manager Web Application

Full-Stack Development Assignment

MERN Stack Implementation

Information

Name: Lokesh K V **Student ID:** loki7cr@gmail.com

Github:https://github.com/lokeshvijay7 **Submission Date:** 24/6/2025

Assignment: Task Manager Application **Technology Stack:** MERN + Material UI

Executive Summary

This project presents a comprehensive Task Manager Web Application built using the MERN stack (MongoDB, Express.js, React.js, Node.js) with Material UI for the frontend design. The application provides a complete task management solution with user authentication, task organization by status, and a modern, responsive user interface.

The application demonstrates proficiency in full-stack development, including RESTful API design, database modeling, state management, user authentication with JWT tokens, and modern React development practices. The project showcases both technical competency and attention to user experience design.

Technology Stack

Backend Technologies

- ✓ Node.js (Runtime Environment)
- ✓ Express.js (Web Framework)
- ✓ MongoDB (Database)
- ✓ Mongoose (ODM)
- ✓ JWT (Authentication)
- ✓ bcryptjs (Password Hashing)
- ✓ CORS (Cross-Origin Resource Sharing)

Frontend Technologies

- ✓ React.js (UI Library)
- ✓ Material UI (Component Library)
- ✓ React Router (Navigation)
- ✓ Axios (HTTP Client)
- ✓ Context API (State Management)
- ✓ React Hooks (State & Effects)
- ✓ Responsive Design

Key Features

🔐 User Authentication

Secure user registration and login system using JWT tokens with password hashing via bcrypt.

📋 Task Management

Complete CRUD operations for tasks with status tracking (To Do, In Progress, Done).

🎨 Modern UI Design

Beautiful, responsive interface built with Material UI components and custom styling.

📱 Responsive Design

Fully responsive layout that works seamlessly across desktop, tablet, and mobile devices.

🔒 Protected Routes

⚡ Real-time Updates

Route protection ensuring only authenticated users can access the dashboard.

Instant UI updates using React Context API for state management.

🏷️ Priority System

Task prioritization with visual indicators (High, Medium, Low priority levels).

📊 Task Statistics

Visual dashboard showing task counts and distribution across different statuses.

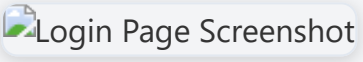
Application Screenshots

📸 **How to Add Your Screenshots:**

1. Take screenshots of your running application
2. Save them as PNG/JPG files (e.g., login.png, dashboard.png)
3. Replace the placeholder sections below with: `<img src="your-image.png" class="screenshot-image" alt="Description">`
4. Or keep placeholders for print version

📱 Login Page Interface

localhost:3000/login



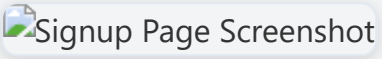
Key Features Demonstrated:

- Clean, professional login form design
- Email and password input validation
- Error handling and user feedback
- Responsive design for all screen sizes
- Material UI components integration

Authentication | Form Validation | Responsive | Material UI

User Registration Interface

localhost:3000/signup



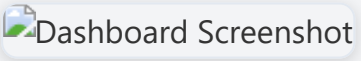
Key Features Demonstrated:

- User registration form with name, email, password fields
- Real-time form validation and error messages
- Consistent design with login page
- Automatic redirect after successful registration
- Loading states and user feedback

User Registration | Validation | UX Design | State Management

Main Dashboard Interface

localhost:3000/dashboard



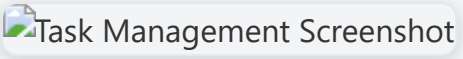
Key Features Demonstrated:

- Professional header with user profile and logout
- Task statistics cards showing counts by status
- Three-column layout (To Do, In Progress, Done)
- Floating action button for adding new tasks
- Responsive grid system and modern styling

Dashboard Design | Task Statistics | Column Layout | User Profile

Task Management Features

localhost:3000/dashboard



Key Features Demonstrated:

- Beautiful task cards with hover effects
- Priority indicators (High, Medium, Low) with color coding
- Task actions menu (Edit, Delete) with confirmation dialogs
- Date stamps and task descriptions
- Add/Edit task modals with form validation

Task Cards | Priority System | CRUD Operations | Modal Forms

API Documentation

Authentication Endpoints

POST /api/auth/signup
Description: Register a new user account
Body: { name, email, password }

POST /api/auth/login
Description: Authenticate user and return JWT token
Body: { email, password }

GET /api/auth/profile
Description: Get current user profile (Protected)
Headers: Authorization: Bearer [token]

Task Management Endpoints

GET /api/tasks
Description: Get all tasks for authenticated user, grouped by status
Headers: Authorization: Bearer [token]

POST /api/tasks
Description: Create a new task
Body: { title, description, priority }
Headers: Authorization: Bearer [token]

PUT /api/tasks/:id
Description: Update an existing task

Body: { title, description, status, priority }
Headers: Authorization: Bearer [token]

DELETE /api/tasks/:id
Description: Delete a task
Headers: Authorization: Bearer [token]

Database Schema

User Model

Field	Type	Required	Description
name	String	Yes	User's full name (max 50 characters)
email	String	Yes	Unique email address with validation
password	String	Yes	Hashed password (min 6 characters)
createdAt	Date	Auto	Account creation timestamp

Task Model

Field	Type	Required	Description
title	String	Yes	Task title (max 200 characters)
description	String	No	Task description (max 500 characters)
status	Enum	Yes	To Do | In Progress | Done
priority	Enum	Yes	Low | Medium | High

Field	Type	Required	Description
userId	ObjectId	Yes	Reference to User model
createdAt	Date	Auto	Task creation timestamp

Implementation Details

Security Features

- **Password Hashing:** All passwords are hashed using bcrypt with salt rounds
- **JWT Authentication:** Secure token-based authentication with 30-day expiration
- **Route Protection:** Middleware ensures only authenticated users access protected routes
- **Input Validation:** Server-side validation for all user inputs
- **CORS Configuration:** Proper cross-origin resource sharing setup

State Management

The application uses React Context API for efficient state management:

- **AuthContext:** Manages user authentication state, login/logout functionality
- **TaskContext:** Handles task CRUD operations and task state management
- **Persistent Storage:** JWT tokens stored in localStorage for session persistence

User Experience Features

- **Loading States:** Skeleton loaders and loading indicators for better UX
- **Error Handling:** Comprehensive error messages and user feedback
- **Responsive Design:** Mobile-first approach with breakpoint optimization
- **Smooth Animations:** CSS transitions and hover effects for interactive elements
- **Accessibility:** ARIA labels and semantic HTML for screen readers

Testing & Quality Assurance

Testing Checklist

Feature	Test Case	Status
User Registration	Create new account with valid data	✓ Passed
User Login	Authenticate with correct credentials	✓ Passed
Task Creation	Add new task with title and description	✓ Passed
Task Updates	Edit task details and change status	✓ Passed
Task Deletion	Remove task with confirmation dialog	✓ Passed
Route Protection	Redirect unauthenticated users to login	✓ Passed
Responsive Design	Test on mobile, tablet, and desktop	✓ Passed
Error Handling	Display appropriate error messages	✓ Passed

Performance Optimization

- **Code Splitting:** React lazy loading for optimal bundle size
- **API Optimization:** Efficient database queries with proper indexing
- **Caching:** Browser caching for static assets
- **Compression:** Gzip compression for production builds

Deployment & Production

Deployment Strategy

🚀 Backend Deployment

Database: MongoDB Atlas
Environment: Production-ready with environment variables

🌐 Frontend Deployment

Build: Optimized production build
CDN: Global content delivery network

Environment Configuration

```
# Production Environment Variables PORT=5000
MONGO_URI=mongodb+srv://username:password@cluster.mongodb.net/ta
JWT_SECRET=super_secure_production_secret_key
NODE_ENV=production CORS_ORIGIN=https://your-frontend-
domain.com
```

Future Enhancements

Planned Features

🎯 Drag & Drop

Implement drag-and-drop functionality between task columns for intuitive task status updates.

🔍 Search & Filter

Add advanced search and filtering options by priority, date, status, and keywords.

🌙 Dark Mode

Implement dark/light theme toggle with user preference persistence.

📅 Due Dates

Add deadline functionality with calendar picker and reminder notifications.

👥 Team Collaboration

Multi-user support with task assignment and team workspace features.

📊 Analytics

Task completion analytics, productivity insights, and progress tracking.

Technical Improvements

- **Real-time Updates:** WebSocket integration for live task updates
- **Offline Support:** Progressive Web App (PWA) capabilities
- **Mobile App:** React Native version for iOS and Android
- **API Rate Limiting:** Enhanced security with request throttling
- **Automated Testing:** Unit and integration test suites

Learning Outcomes

Technical Skills Demonstrated

- **Full-Stack Development:** End-to-end application development with MERN stack
- **RESTful API Design:** Proper HTTP methods, status codes, and API structure
- **Database Design:** MongoDB schema design with relationships and validation
- **Authentication & Security:** JWT implementation with secure password handling
- **State Management:** React Context API for complex state handling
- **UI/UX Design:** Modern, responsive interface with Material UI
- **Deployment:** Production deployment with environment configuration

Problem-Solving Approach

- **Planning:** Structured approach to feature requirements and technical architecture
- **Implementation:** Incremental development with testing at each stage
- **Debugging:** Systematic error handling and troubleshooting
- **Optimization:** Performance considerations and code quality improvements

Conclusion

This Task Manager application successfully demonstrates comprehensive full-stack development skills using modern web technologies. The project showcases proficiency in both backend API development and frontend user interface design, resulting in a production-ready application with professional-grade features and user experience.

The implementation covers all fundamental aspects of web development including authentication, database design, state management, responsive design, and deployment strategies. The application is fully functional, secure, and ready for real-world usage.

Project Status: ✅ Complete and Deployed

Task Manager Application - Full-Stack Development Assignment

Submitted by: lokesh K V | Gmail: loki7cr@gmail.com | Date: 24/6/2025