



PROJECT REPORT

Fullstack Development  
  
LinkedIn-Style Social Media Platform

|  |  |  |  |
| --- | --- | --- | --- |
| **Created By:** | Sachit Pandey | **Approved By:** | <Domain Lead Name> |
| **Created On:** | 28-06-2025 | **Approved On:** | DD-MMM-YYYY |

Page left blank intentionally

**INDEX**

[**1** **PROJECT DETAILS** 2](#_Toc143445375)

[**2** **SUMMARY** 2](#_Toc143445376)

[**3** **INTRODUCTION** 2](#_Toc143445377)

[3.1 Background 2](#_Toc143445378)

[3.2 Stakeholders 2](#_Toc143445379)

[3.3 Objectives 2](#_Toc143445380)

[**4** **METHODOLOGY** 2](#_Toc143445381)

[4.1 Considerations & Assumption 3](#_Toc143445382)

[4.2 Approach 3](#_Toc143445383)

[4.3 Activities 3](#_Toc143445384)

[**5** **TARGETTED V/S ACHIEVED OUTPUT** 3](#_Toc143445385)

[**6** **CONCLUSION** 3](#_Toc143445386)

[**7** **APPENDICES** 4](#_Toc143445387)

[7.1 Appendix A – Title 4](#_Toc143445388)

**General Instructions for using the Live Project Report Template**

* This template and the subsequent document created using this template is a confidential document and is the intellectual property of Cloud Counselage Pvt. Ltd. Circulating it outside of the organisation without the consent of Cloud Counselage Pvt. Ltd. is the breach of company policies and will lead to legal actions
* This template is a guideline document to communicate the implementation of design ideas and the results of the work to the stakeholders.
* The **text between inequality (< >) is to be replaced** by relevant text
* Please **remove the yellow highlight on the Text** between the inequality (< >). This is done to help you notice the text to be changed/replaced
* The text in *italics* highlighted in grey is just for reference and should be removed after adding the relevant text

# **PROJECT DETAILS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Name** | LinkedIn-Style Social Media Platform | | |
| **Project Sponsor** | Tushar Topale | | |
| **Project Manager** | Harshada Topale | | |
| **Start Date** | 04-04-2025 | **Completion Date** | 28-06-2025 |

# **SUMMARY**

The main goal of this project was to build a LinkedIn-style social media web application using the MERN stack – which includes MongoDB, Express.js, React, and Node.js. The idea was to create a platform where users can sign up, manage their profiles, share posts, and connect with other users in a professional setting.

This project was important because it gave me the opportunity to apply what I’ve learned in web development to a real-world use case. It helped me understand how different technologies work together – from building interactive user interfaces in React to handling backend logic in Node.js, storing data in MongoDB, and even integrating tools like Cloudinary for image uploads.

Working on this application taught me how to plan features, debug issues, test APIs, and deploy a full-stack app. In the long run, this project not only helped me gain technical skills but also showed me how to work through a complete development cycle – which will definitely help in my future roles as a software developer.

# **INTRODUCTION**

## Background

In today’s professional environment, networking plays a crucial role in career growth and opportunity building. Existing platforms like LinkedIn provide structured spaces for professionals to connect, share updates, and grow their personal brand. Inspired by this idea, the project aimed to develop a lightweight LinkedIn-style social media platform using the MERN (MongoDB, Express.js, React.js, Node.js) stack.

The concept for the platform was discussed during the initial requirement gathering sessions, where the need for profile creation, post sharing, connection requests, and a clean UI/UX was emphasized. Additionally, integration with image hosting services and secure user authentication was highlighted to ensure the platform's scalability and usability.

## Stakeholders

The stakeholders for this project include:

* **Intern / Developer (Sachit Pandey)** – Responsible for designing, building, testing, and deploying the fullstack application.
* **Domain Guide (Harshada Topale)** – Provided guidance on scope, documentation, and development best practices.
* **Project Evaluators** – Responsible for reviewing the final output, documentation, and learning outcomes.
* **End Users (Fictional)** – Represented by test accounts simulating real users to ensure the platform supports login, post creation, and connection functionality.

## Objectives

The primary objectives as stated in the Project Charter were:

* Build a responsive MERN-based application with user authentication.
* Implement profile creation and editing functionality with image uploads.
* Allow users to create posts and interact with others’ content.
* Provide a connection system (send/accept/reject requests).
* Deploy the application on cloud platforms (Render and Vercel).

**Achievements**:

* All major modules—Authentication, Profile Management, Posts, and Connections—were successfully implemented.
* The application was deployed, tested, and documented as planned.

# **METHODOLOGY**

## Considerations & Assumption

While developing this project, several assumptions and constraints were taken into account:

* It was assumed that MongoDB Atlas and Cloudinary (free-tier) services would remain available and reliable throughout the project duration.
* The application would be primarily accessed on modern desktop or mobile browsers, so mobile responsiveness was prioritized.
* A single developer (Sachit Pandey) would be responsible for both frontend and backend, which meant tasks were planned with clear modular boundaries.
* Given time limitations, only core features like profile editing, posting, and connections were implemented. Advanced features (e.g., real-time chat, notifications) were deferred.
* The system would follow a RESTful architecture using JWT-based authentication and cloud-hosted services for deployment and scalability.

## Approach

A **modular and iterative approach** was adopted, using the **Waterfall methodology** for its clarity and structure in a solo development setting. The overall development followed these conceptual steps:

1. **Requirement Analysis** – Understanding the scope, user flow, and feature list.
2. **Design Planning** – Wireframes, component breakdown, and database schema definition.
3. **Development** – Frontend and backend components were developed independently and integrated via REST APIs.
4. **Testing & Debugging** – Manual testing using Postman for backend and React DevTools for frontend.
5. **Deployment** – The app was deployed using Render, with configurations done via environment variables.
6. **Documentation** – Final reports and logs were created alongside the demo video.

This structure helped keep each module self-contained, track progress effectively, and simplify debugging.

## Activities

Throughout the internship, the following key activities were performed:

* **Requirement Gathering**: Based on project expectations and guidance from the mentor.
* **Planning**: Task breakdown, feature prioritization, and module scheduling.
* **Frontend Development**: Built React components using Vite and Tailwind CSS.
* **Backend Development**: Created REST APIs with Express.js and connected to MongoDB Atlas.
* **Image Handling**: Integrated Cloudinary for image uploads in profile and post features.
* **Testing**: Verified functionality using Postman and browser-based testing.
* **Deployment**: Hosted on Render; managed environment configs.
* **Documentation**: Prepared all supporting documents (Test Plan, WBS, RAID Log, etc.).
* **Demo Preparation**: Recorded app functionality and organized ZIP for final submission.

# **TARGETTED V/S ACHIEVED OUTPUT**

At the start of the internship project, the goal was to build a LinkedIn-style social media platform that includes core features such as user authentication, profile editing, post creation, and a professional connection system.

The table below summarizes the targeted outcomes versus what was achieved during the project:

| **Targeted Output** | **Achieved Output** | **Remarks / Deviations** |
| --- | --- | --- |
| User registration and login system with JWT auth | Implemented using React + Express + JWT | — |
| Profile creation and editing with image upload | Implemented with Cloudinary + EditProfile UI | — |
| Create and view posts (text and image) | Successfully created post API and frontend form | — |
| Like and comment on posts | Like system implemented Comment feature skipped | Simplified due to time constraints and focus on core modules |
| Connection request system (send/accept/reject) | Fully implemented using modular API routes | — |
| Clean responsive UI using Tailwind CSS | Tailwind integrated throughout | — |
| Real-time notifications and chat | Not implemented | Marked as a future enhancement; deprioritized to focus on stability |
| Deployment on cloud platforms | Render | Initial CORS and env issues resolved successfully during deployment phase |
| Project documentation and demo | All reports completed and app demo recorded | — |

# **CONCLUSION**

The LinkedIn-style social media platform developed during this internship has successfully demonstrated the integration of modern full-stack technologies (MongoDB, Express.js, React, and Node.js) into a real-world application. The system provides essential features such as user authentication, profile management, content sharing, and connection building—all within a user-friendly, responsive interface.

This platform proves valuable for stakeholders by:

* Showcasing the intern's ability to design, build, and deploy a full-fledged web application.
* Serving as a reusable template for similar networking platforms or portfolio projects.
* Demonstrating a well-structured approach to full-stack development and project documentation.

**Future Scope**

While the current version fulfils its primary objectives, there is scope to extend and improve the platform further:

* Implementing a **comment system** to improve post interaction.
* Adding **real-time features** like notifications or messaging using WebSockets or Socket.io.
* Introducing **admin features** to manage reported users or flagged content.
* Enhancing **UI/UX** with animations, dark mode, and accessibility improvements.

With a solid architectural foundation in place, the application is well-prepared for future feature expansions and scalability.

# **APPENDICES**

## Appendix A – Title

**Title**: Detailed Design Specification of Core Components in the LinkedIn-Style MERN Stack Application

This appendix contains detailed component-level descriptions for key modules implemented in the project. Each component has been outlined with its identification, purpose, dependencies, and data flow based on the design specification.

The following components are included:

* Login.jsx – React form to authenticate users
* ProfileController.js – Handles profile editing and image upload
* PostRoutes.js & PostController.js – APIs for post creation and interaction

Refer to **Appendix A – Detailed Description of Components** for complete specifications in tabular format.

| **Field** | **LoginPage.jsx** |
| --- | --- |
| Identification | src/pages/LoginPage.jsx |
| Type | React Component / Form |
| Purpose | The login screen ensures only registered users can access the platform. It handles user credential input and communicates with the backend to validate login and store JWT tokens. |
| Subordinates | Home Page, Profile Page |
| Dependencies | Auth Context, Axios, /api/auth/login |
| Interfaces | Input fields, Axios POST request, login response handling |
| Resources | JWT Token, localStorage, Express backend |
| Processing | Handles input, sends login request, stores token, redirects user |
| Data | Email, Password, Token |

| **Field** | **ProfileController.js** |
| --- | --- |
| Identification | controllers/user.controller.js |
| Type | Express Controller |
| Purpose | Updates user profile, bio, skills, and uploads images to Cloudinary |
| Subordinates | User Model, Image Upload Utility |
| Dependencies | Cloudinary, Mongoose, Multer |
| Interfaces | PUT /api/user/updateprofile |
| Resources | MongoDB Atlas, Cloudinary |
| Processing | Parses form data, uploads images, updates user document |
| Data | Bio, Skills, Image URLs |

| **Field** | **PostRoutes.js & PostController.js** |
| --- | --- |
| Identification | routes/post.routes.js, controllers/post.controller.js |
| Type | Express Router and Controller |
| Purpose | Handles post creation, likes, and retrieval |
| Subordinates | Post Model, Like/Comment logic |
| Dependencies | JWT Middleware, Mongoose, Cloudinary |
| Interfaces | POST /api/post, GET /api/post/all, PUT /api/post/like/:id |
| Resources | MongoDB Atlas, Cloudinary |
| Processing | Validates request, uploads images, stores in DB |
| Data | Text, Image URL, UserID, Like Count |