Social Media Web App

Rayan Adoum

G00394595

Bachelor of Engineering (Honours) in Software and Electronic Engineering Project Engineering Year 4 2024/2025

Ollscoil Teicneolaíochta an Atlantaigh

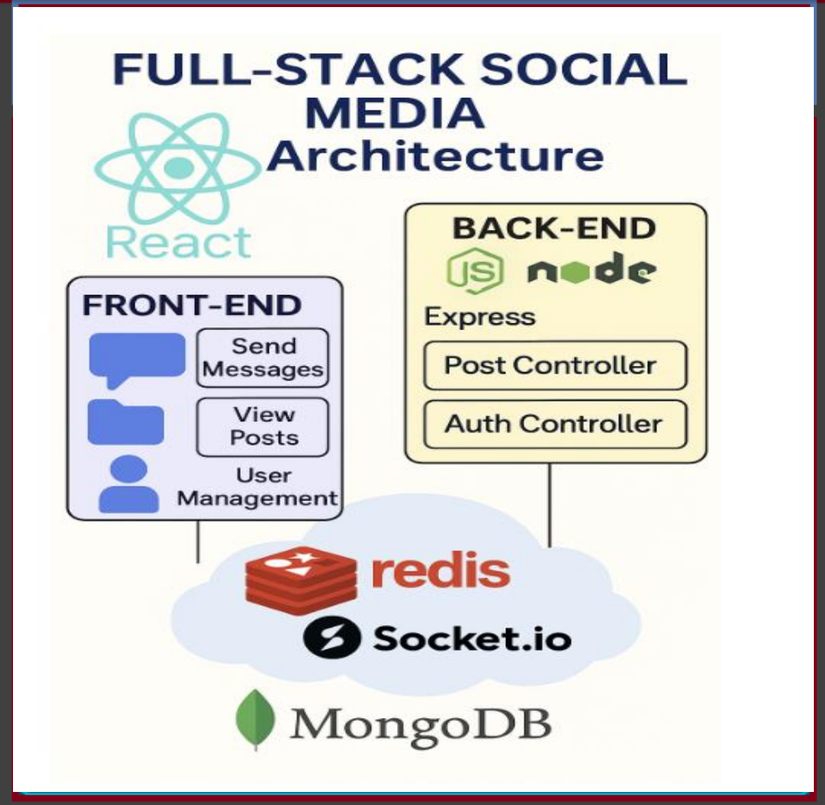
Atlantic Technological University

Introduction

For my final year project, I decided to build a full-stack social media web app using the **MERN stack** — **MongoDB, Express.js, React, and Node.js**. The app allows users to create an account, log in, write posts, comment, react with likes or emojis, and message each other in real time. It also supports notifications, and a basic follow system — similar to what you'd find on any modern social platform.

The backend is powered by **Node.js** and **Express.js**, which handle all API routes. Each route is connected to a controller, which passes the logic to a service class. This setup helps keep the project organized and makes it easier to scale. **MongoDB** is used for storing data like users, posts, comments, and messages.

To enable real-time features like messaging and notifications, I integrated **Socket.IO**. **Redis** is used to help speed up certain features like caching or message delivery.



Features

- Sign up and log in
- Make posts
- View other users' posts
- Real-time chat
- Live notifications
- Follow/unfollow users
- Upload images
- Uses MongoDB for storage
- Uses Socket.IO for live updates

Summary

While working on this project, I quickly realized it wasn't just about getting React to render or Node.js to return data — it was about ensuring all layers of the stack communicated smoothly. It became clear that building a user-friendly experience meant focusing on how the front end and back end interacted in real time. That's why I integrated Socket.io, which allowed users to send and receive messages instantly, making conversations feel fluid without constant page reloads or polling.

To support that, I also incorporated Redis to manage fast, in-memory data — particularly for handling real-time features like notifications and event-based updates. When a user sends a message or reacts to a post, Redis helps deliver those interactions quickly and efficiently, improving responsiveness and reducing strain on the database.

Together, these technologies made the app feel more alive and interactive, far beyond a basic CRUD setup. It also taught me the importance of designing for performance and user experience, especially when building real-time applications that need to scale well and stay reliable under pressure.