UNIVERSITY of INFORMATION TECHNOLOGY and MANAGEMENT
in Rzeszow, POLAND

FACULTY OF APPLIED INFORMATION TECHNOLOGY

Field of study: INFORMATION TECHNOLOGY Specialty: Programming

Nyengeterayi Mawire,Johnson Mudzingwa No. of student's record book w66964,w66984

Social Media Application

Supervisor: MSc. Marcin Jagieła

Software Development Techniques Project

Rzeszów 2024

Contents

Introduction

The world is all about connectivity and one of the best platforms to keep the world connected is social media. The ability to connect a group of individuals across miles upon miles from each other in an instance. We decided to create a social media app that seeks to implement that connectivity feature, giving users the ability to update one another and communicate with each other.

Chapter 1

Project

Overview

1.1 Project Objectives

- Upload posts:
  We would like for users to be able to upload media of either image or video types
- Like posts:
  The ability for users to be able to like another users post
- Comment on post:
  Allow users to be able to leave a comment on a desired post
- Message other users:
  Allow for users to be able to message and communicate with other users

Chapter 2

Software Requirements and Specifications

2.1 React JS

React.js is JavaScript framework that is fast, secure, and scalable. It provides a fantastic user and developer experience. Its use of components makes programming easier and unlike regular html frameworks its rendering capabilities saves time and processing as only key information and parts of the page are re-rendered instead of requesting for a whole new page.

2.1.1 Libraries

- React-router-dom : used to handle routes for different pages

2.1.2 Functional Requirements

1. Register a new User
2. Login a User
3. Display posts
4. Like a post
5. Comment on a post

6. View other User profiles
7. Follow or Unfollow user
8. Message Users
9. Upload Post
10. Changes user settings e.g Username,Password,etc

2.2 Node JS Back End

2.2.1 Node JS

Node.js is an open source, cross-platform runtime environment and library that allows for the running of web applications outside the client's browser.

2.2.2 Express Framework

Express JS is the most common framework used by Node JS. It is easy to use and is designed for creating web applications and APIs

2.2.3 Express JS API request:

In Express as we are using it as a communication between frontend and backend we are going to need a few requests. Here is a list of the api requests:

- /user/ (Get): Get all users
- /user/{id} (POST): get a user
- /user/login (Post): Login as user
- /user/ (Post): Get all users
- /user/add/{id} (Patch): Add a follower to a single user
- /user/ remove/{id} (Patch): Remove a follower from a single user
- /user/ addcontact/{id} (Patch): Add a contact to user
- /user/ addprofile/{id} (Patch): Add a profile photo for a single user
- /user/ settings/{id} (Patch): Update user details
- /post/ {id} (Get): Get all posts for a particular user
- /post/ {id} (Delete): Delete a post
- /post/post/ (Post): Create a post
- /post/addComment/ {id} (Patch): Add a comment on a post

- /post/addLike/ {id} (Patch): Add a like on a post
- /post/removeComment/ {id} (Patch): Remove comment on a post
- /post/removeLike/ {id} (Patch): Unlike a post
- /post/uplodMedia/ {id} (Patch): Create a post
- /message/sendmessage/ {id} (Post): Send a message to user
- /message/deletemessage/ {id} (Delete): Send a message to user
- /message/ {id} (Get): Get all messages for a single conversation
- /comment/add/{id} (Patch): Add a comment
- /comment/{id} (Get) : Get all comments for a single post
- /comment/delete/{id} : Delete a comment

2.2.4 Mongo DB(Database)

MongoDB is an open-source document-oriented database that is designed to store a large scale of data and also allows you to work with that data very efficiently. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables.

MongoDB ATLAS:

MongoDB atlas is a service provided by MongoDB which allows for the deployment of your database across platforms allowing for interaction whilst providing the same functionalities.

2.2.5 Socket IO

Socket IO is a web socket library used in Javascript used for real time messaging. It offers low latency and full duplex functionalities and has both backend and frontend functionalities and implementations.

Chapter 3

Results and Outputs

3.1 User Interaction

Fig. 4.1 shows the Register page and the result when a user successfully registers

Fig. 4.1: Home Page
Source: Own Study (20.12.2023)

Fig. 4.2 Shows the entering of data for logging in and Fig. 4.3 show the results of successfully logging in

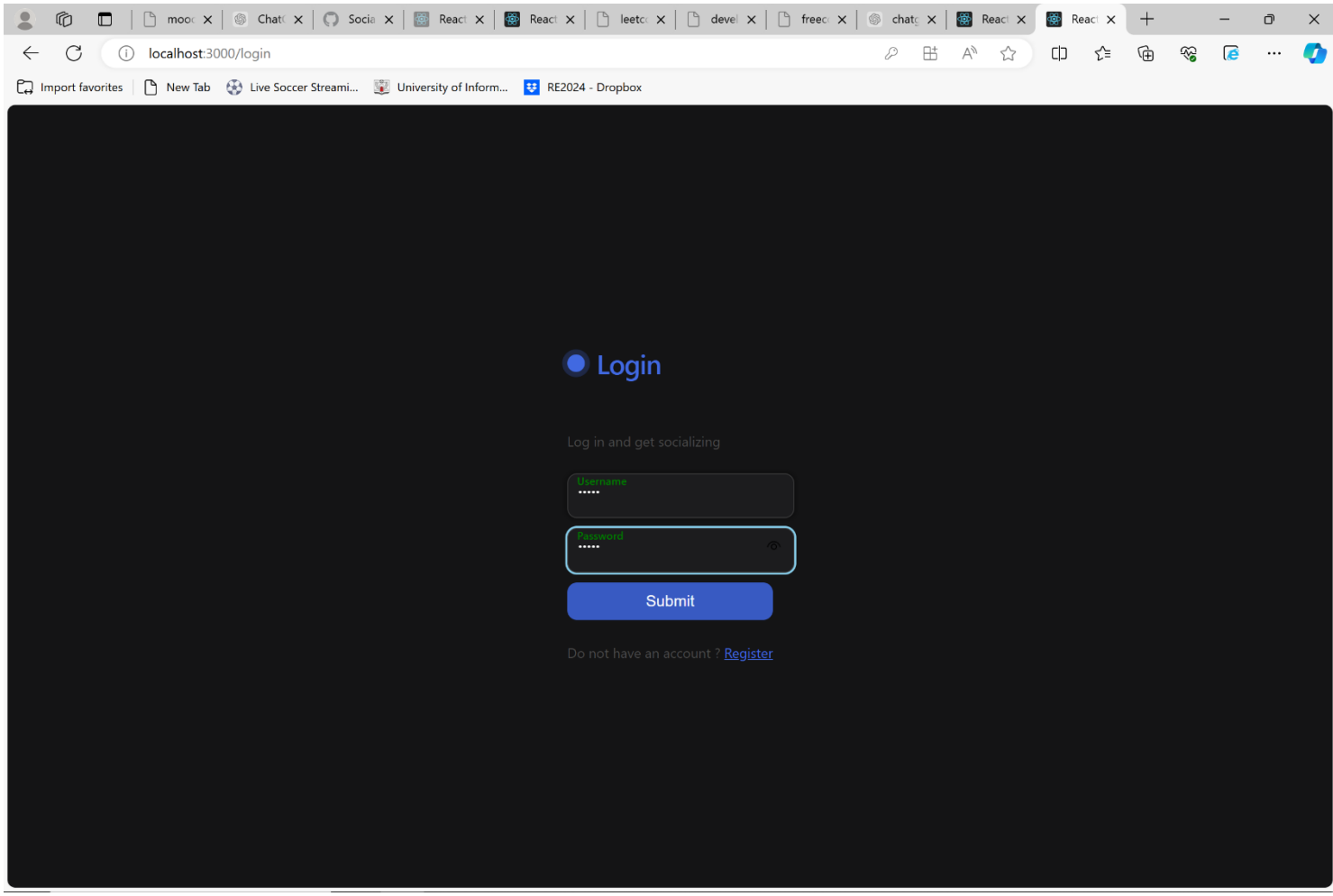

Fig. 4.2: Login page
Source: Own Study (03.06.2024)

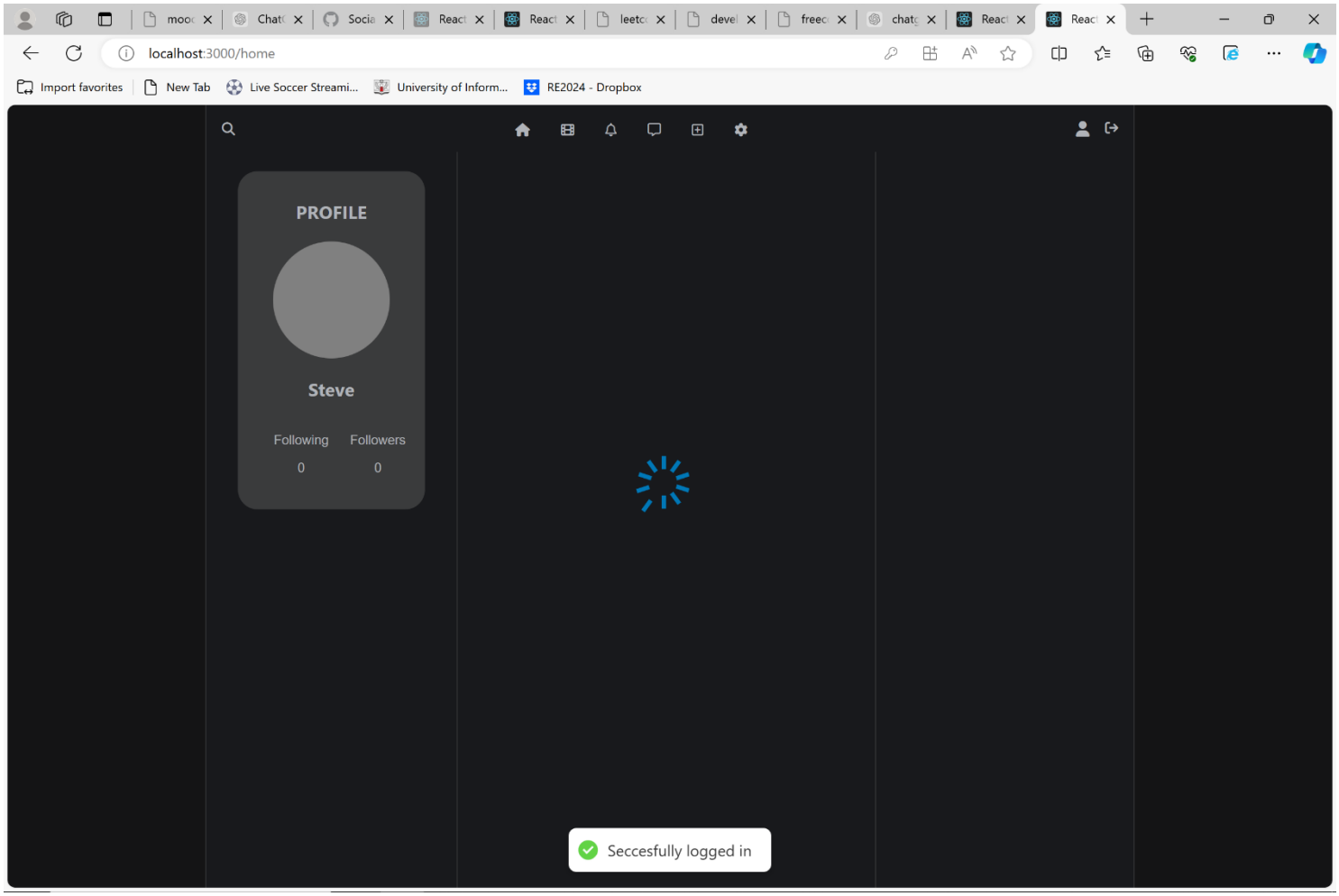

Fig. 4.3: Home page
Source: Own Study (03.06.2024)

Fig 4.4 Shows the outcome of successfully adding a comment

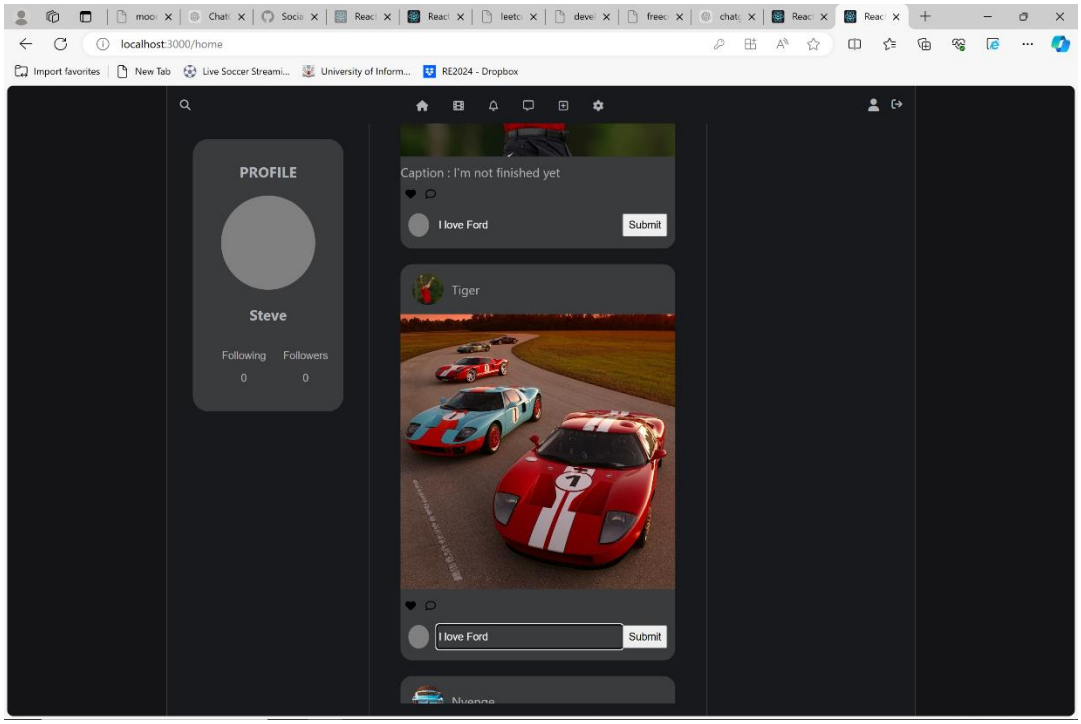

Fig. 4.4: Home page
Source: Own Study (03.06.2024)

Fig.4.5 Shows the notification alert to another user notifying them that someone has commented on their post

Fig 4.6 Shows the outcome of following a user and Fig 4.8 shows the notification sent to the user notifying them they are being followed

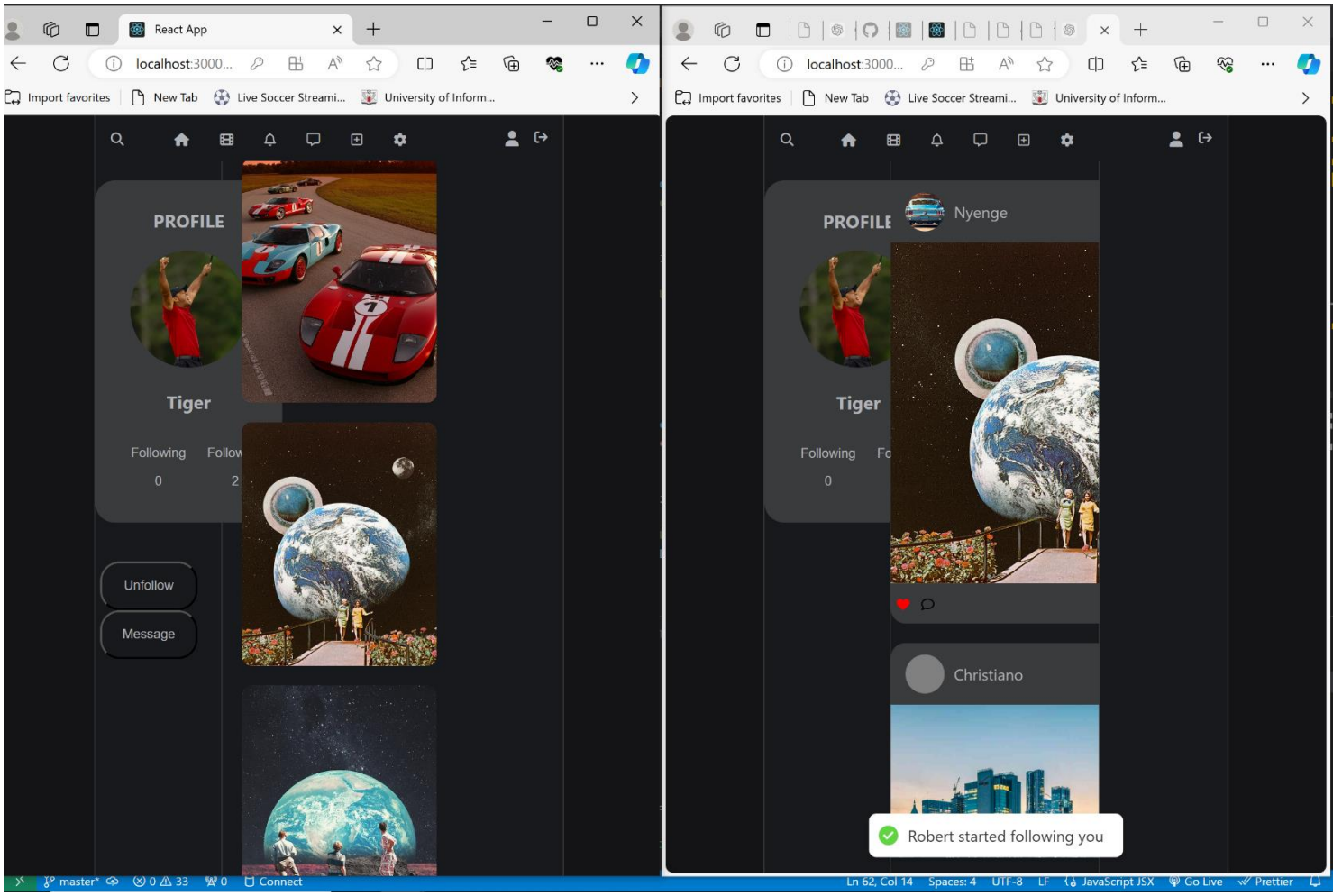

Fig. 4.6: Home page
Source: Own Study (03.06.2024)

Fig 4.7 Shows displays a user liking a post and shows the notification notifying the recipient of the liked post event

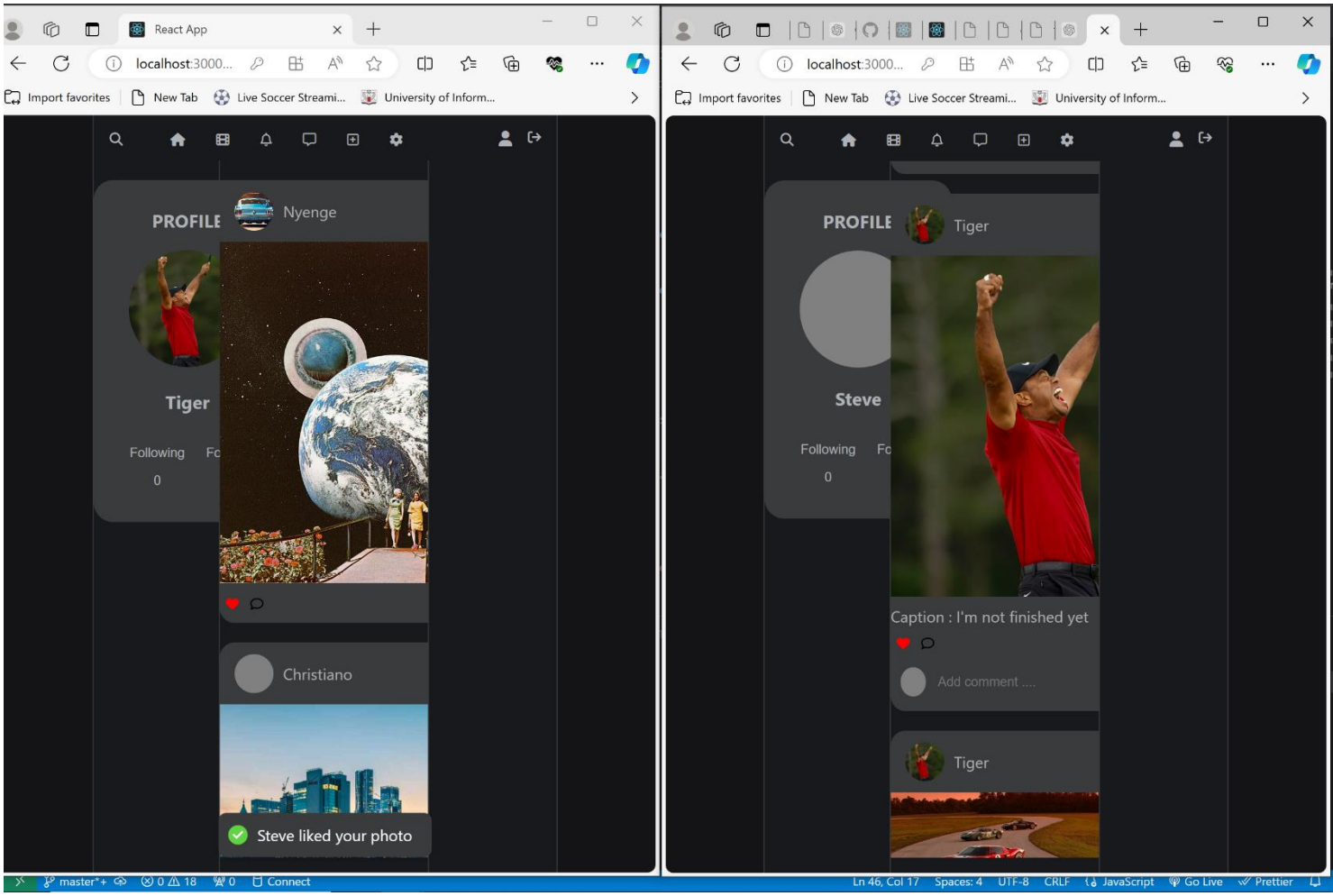

Fig. 4.7: Home page
Source: Own Study (03.06.2024)

Fig 4.8 Shows the outcome when a user sends a message to another user and shows the notification received on the recipients end

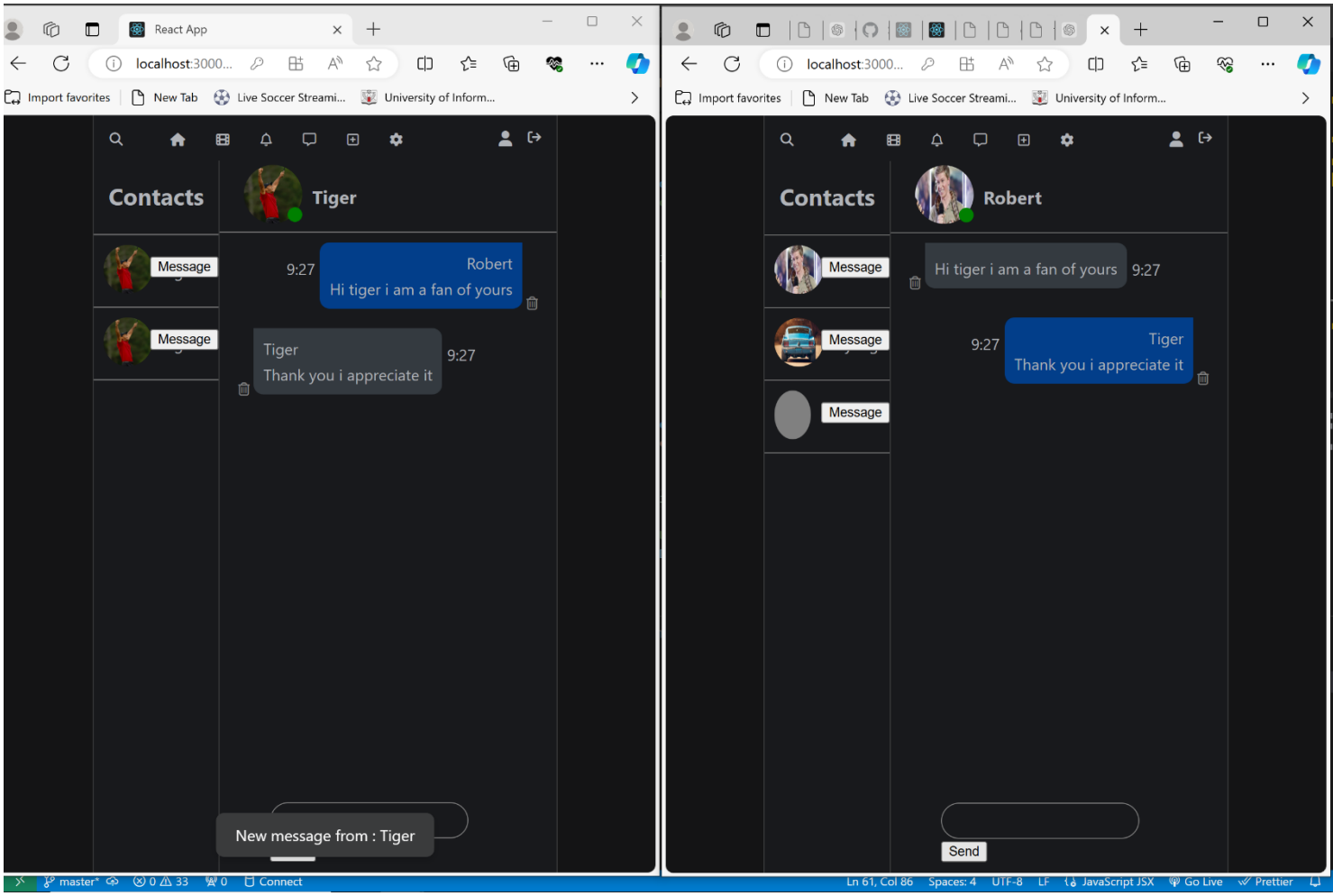

Fig. 4.8: Message page
Source: Own Study (03.06.2024)

Fig 4.9 Shows the results of uploading media and Fig 5.0 shows the success display notification

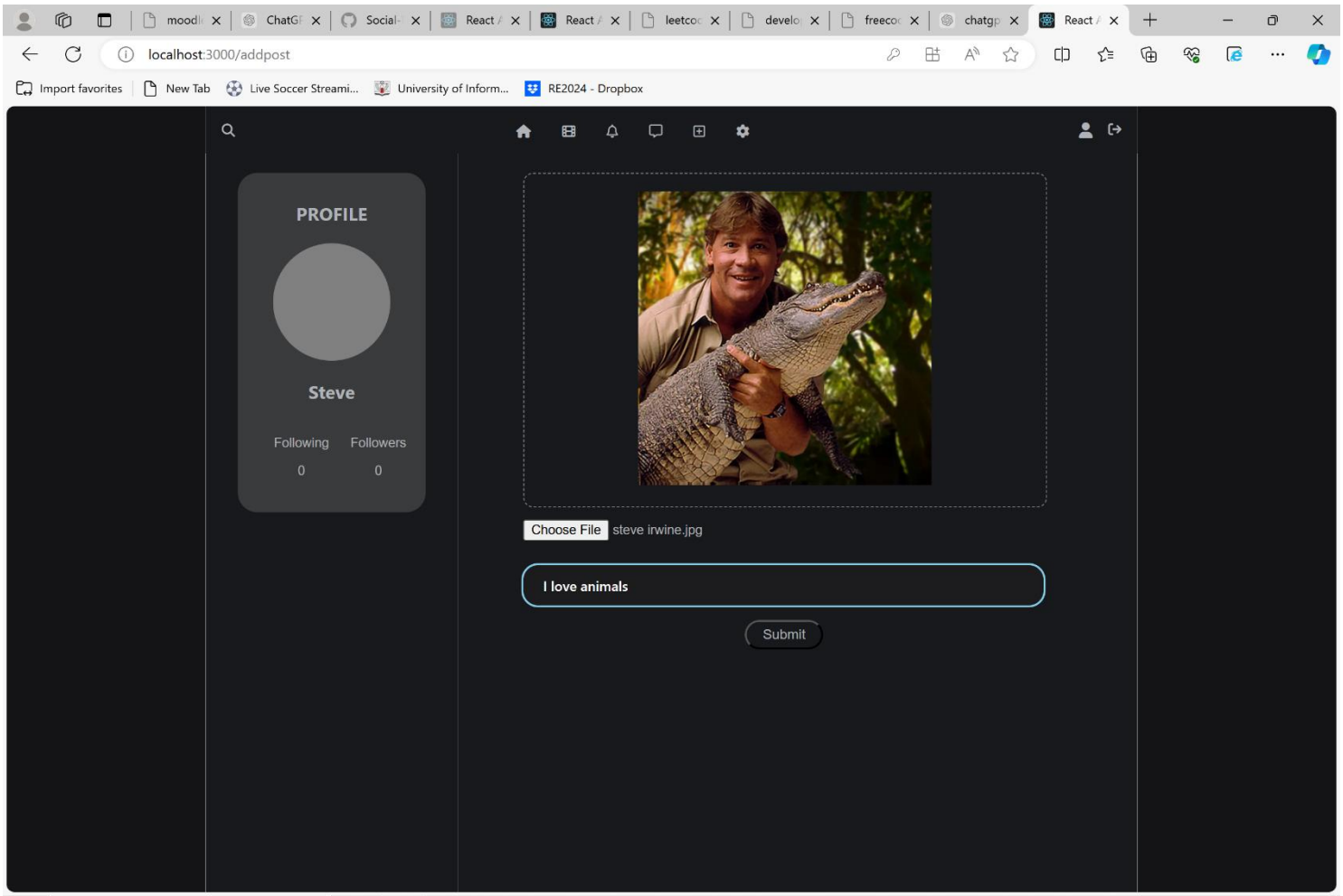

Fig. 4.9: Upload page
Source: Own Study (03.06.2024)

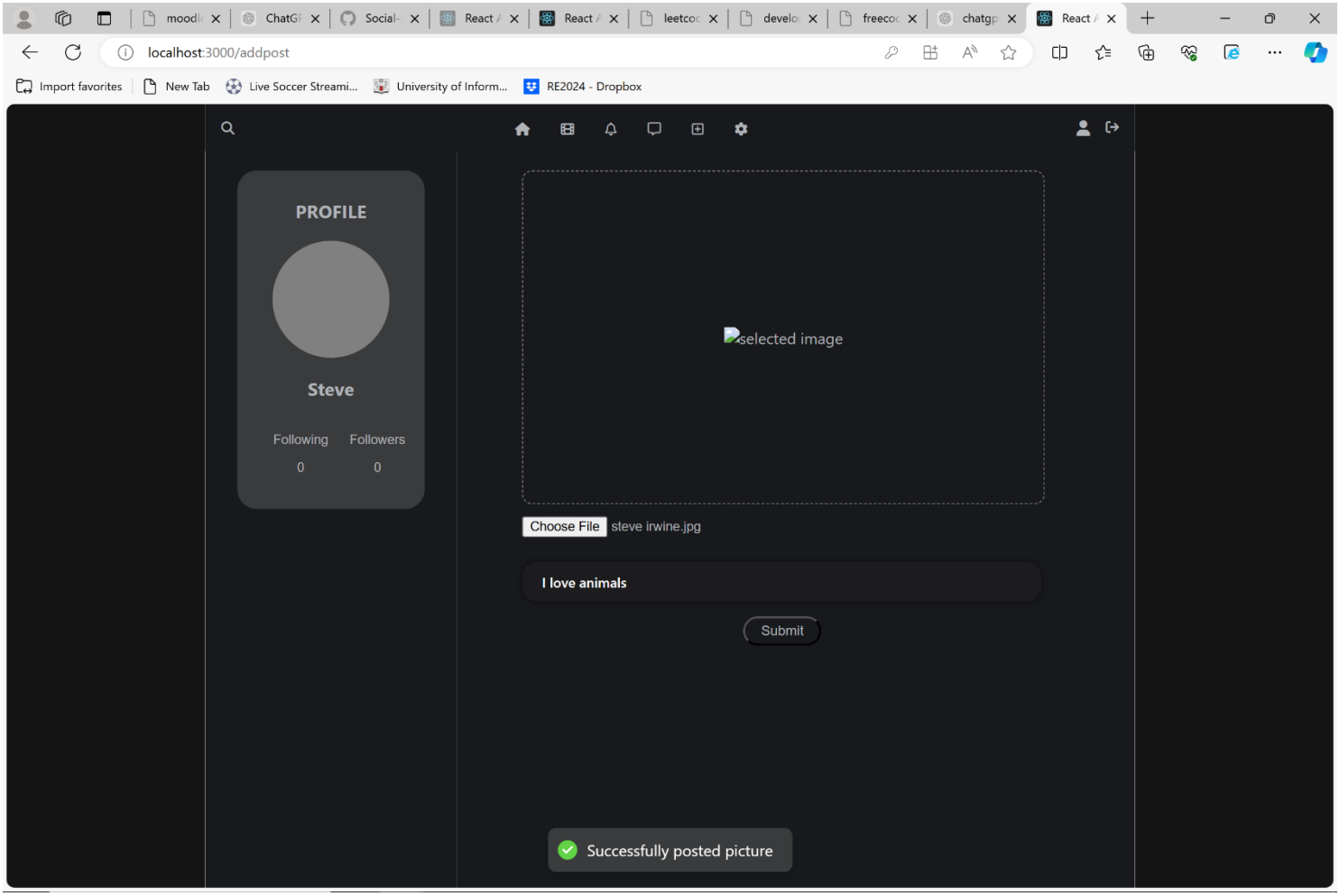

Fig. 5.0: Upload page
Source: Own Study (03.06.2024)

4.1.2 Errors

Fig. 5.1, Fig. 5.2, Fig. 5.2, Fig. 5.3, Fig. 5.4, and Fig. 5.5 shows an error when a user attempts to register with a set of missing fields

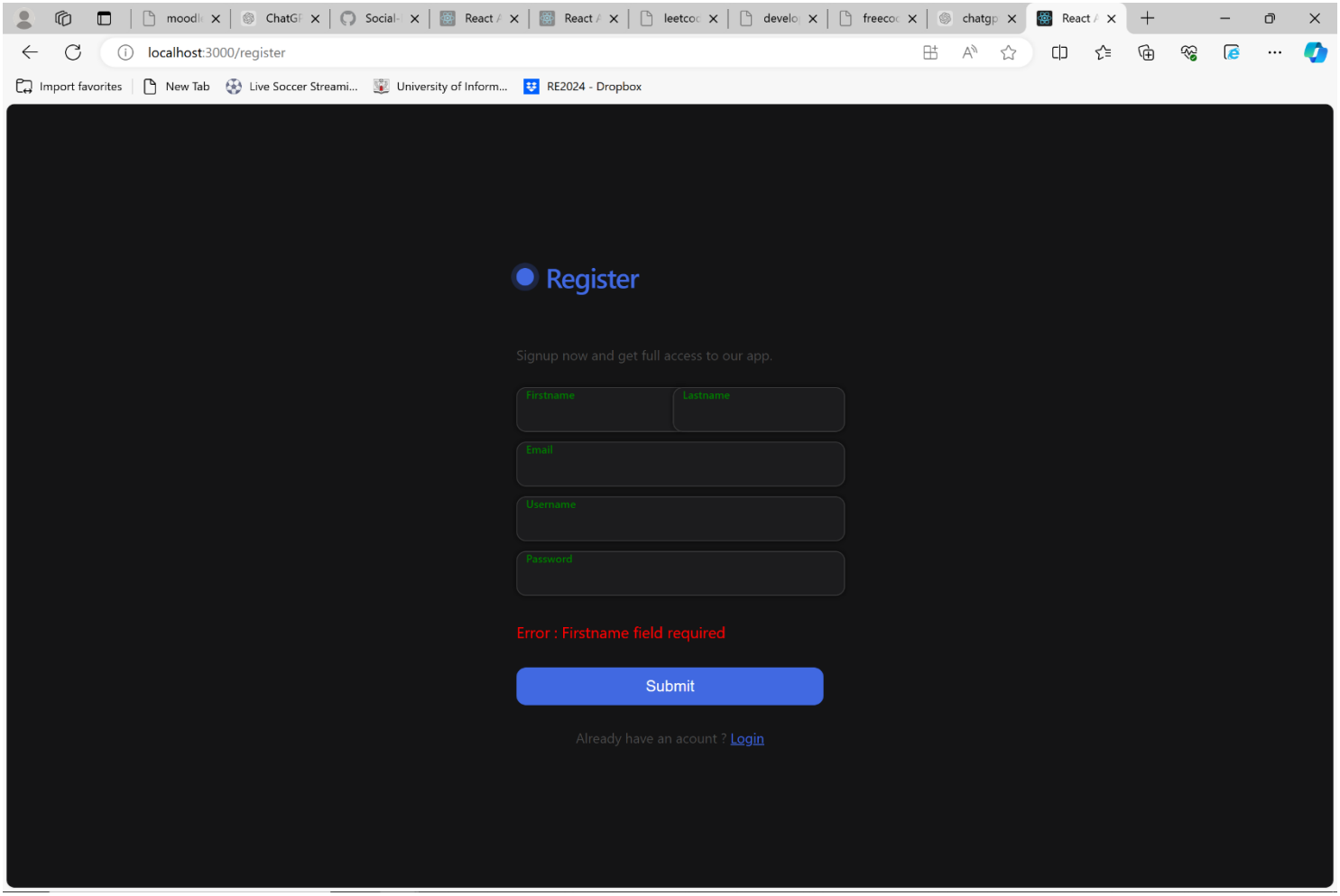

Fig. 5.1: Register page
Source: Own Study (03.06.2024)

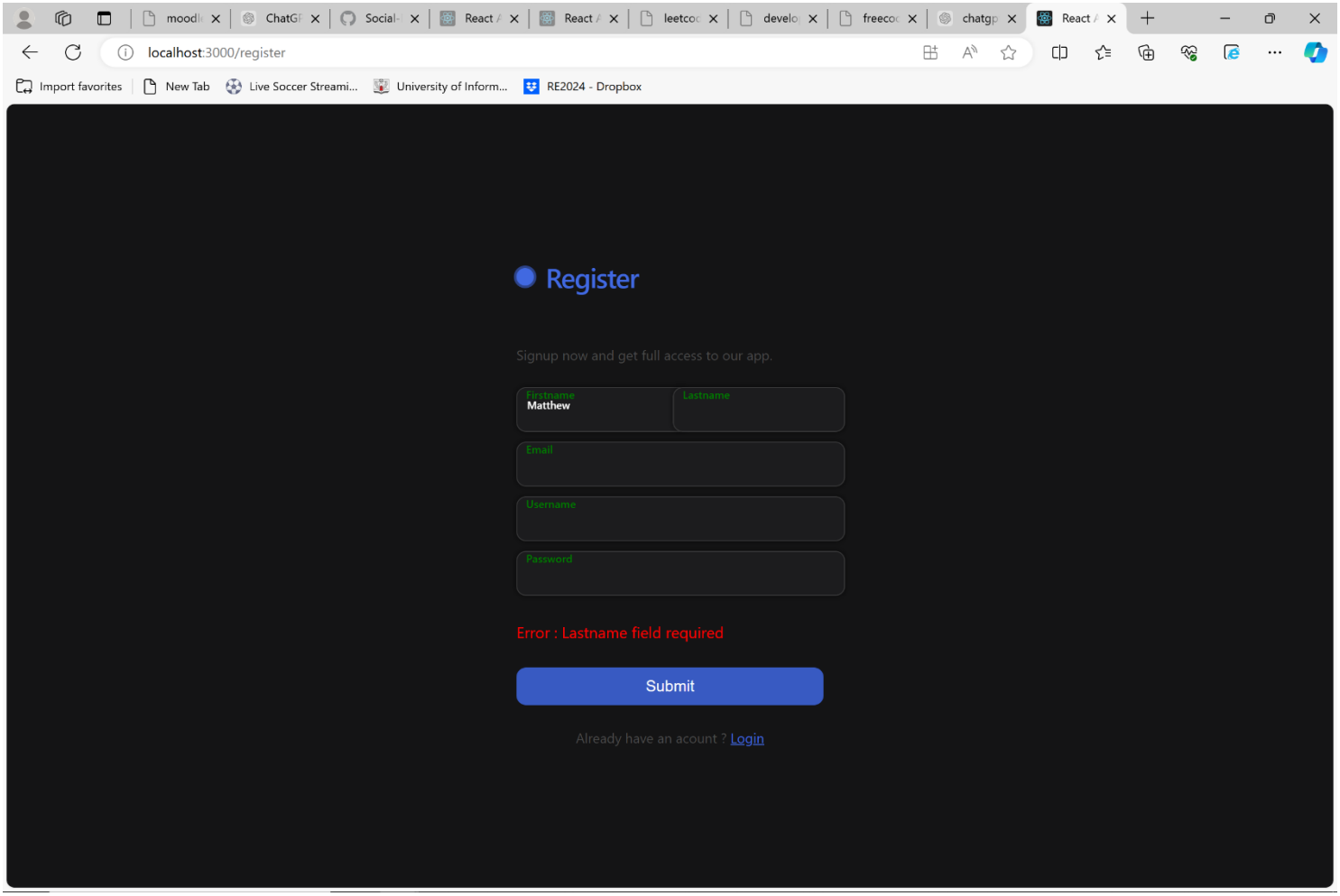

Fig. 5.2: Register page
Source: Own Study (03.06.2024)

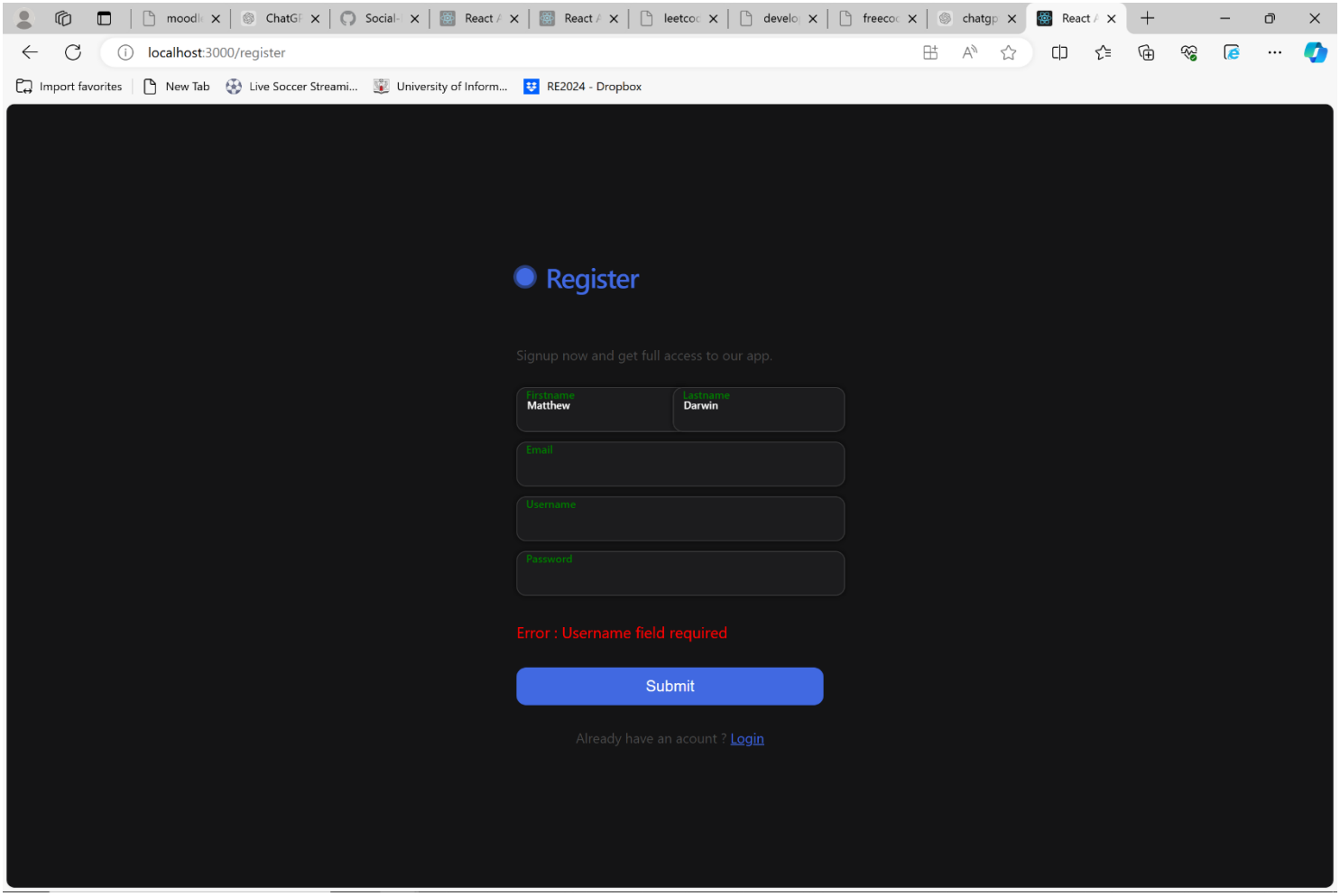

Fig. 5.3: Register page
Source: Own Study (03.06.2024)

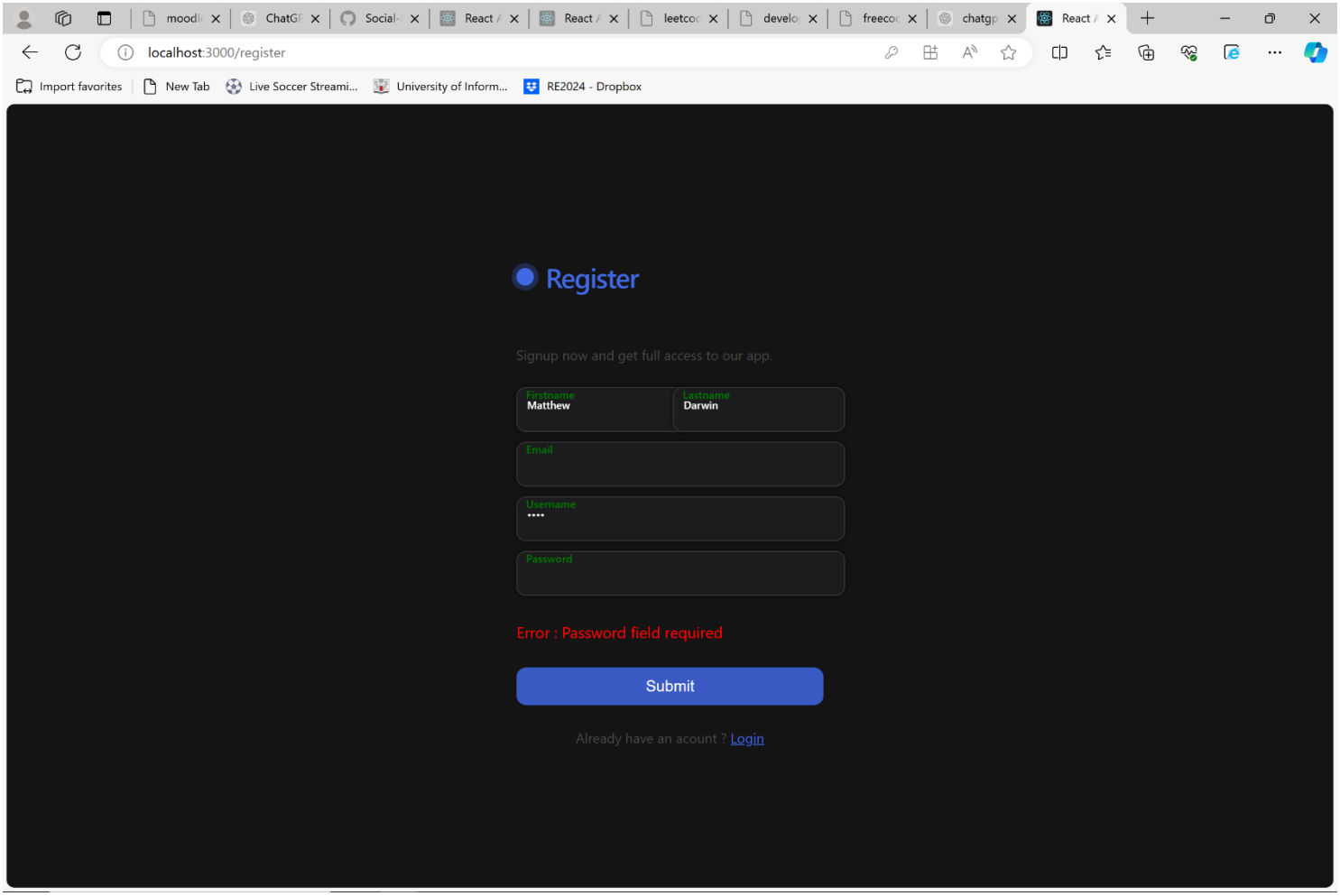

Fig. 5.4: Register page

Source: Own Study (03.06.2024)

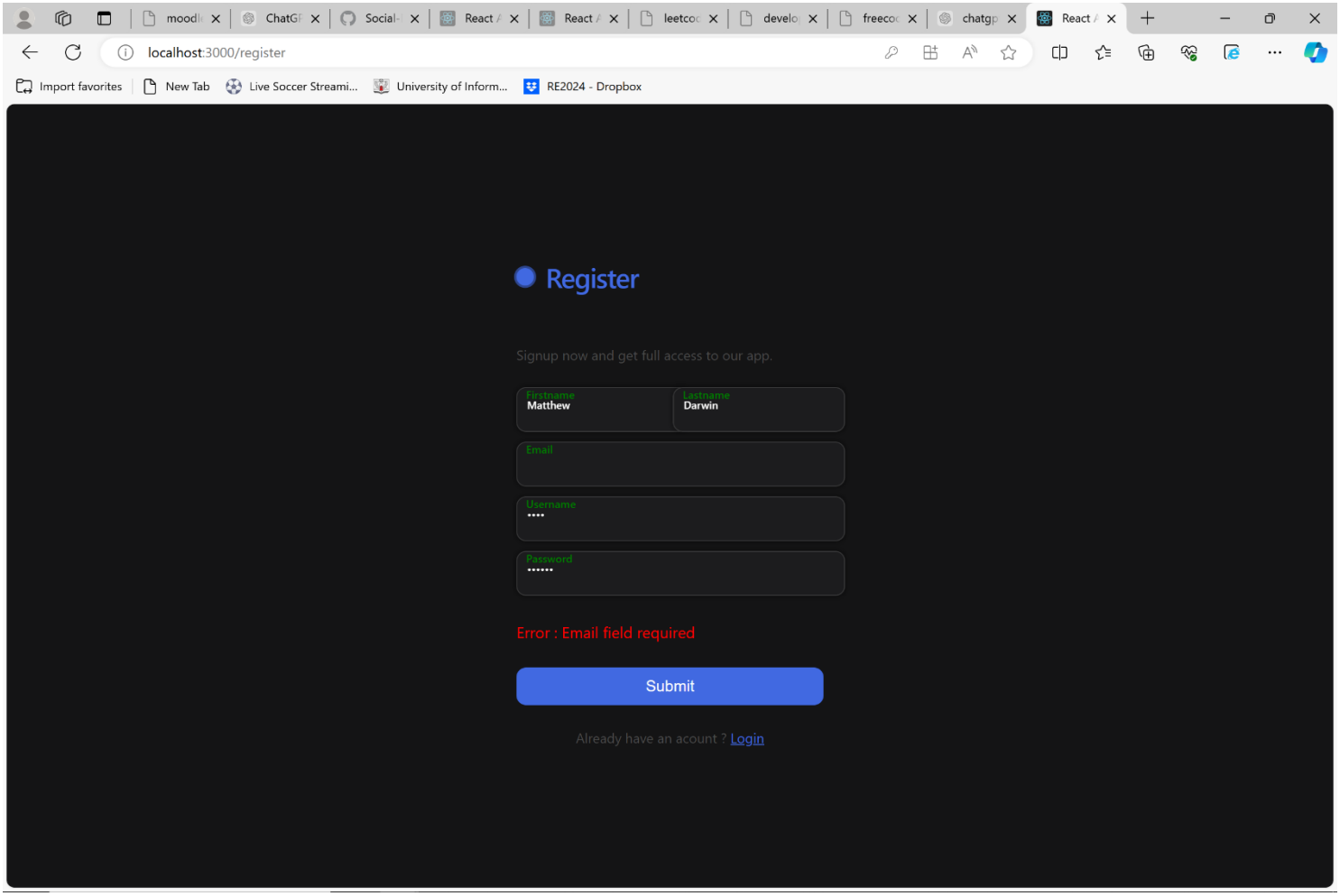

Fig. 5.5: Register page
Source: Own Study (03.06.2024)

Fig. 5.6 and Fig. 5.7 Shows the error that occurs when a user attempts to login with missing data fields.

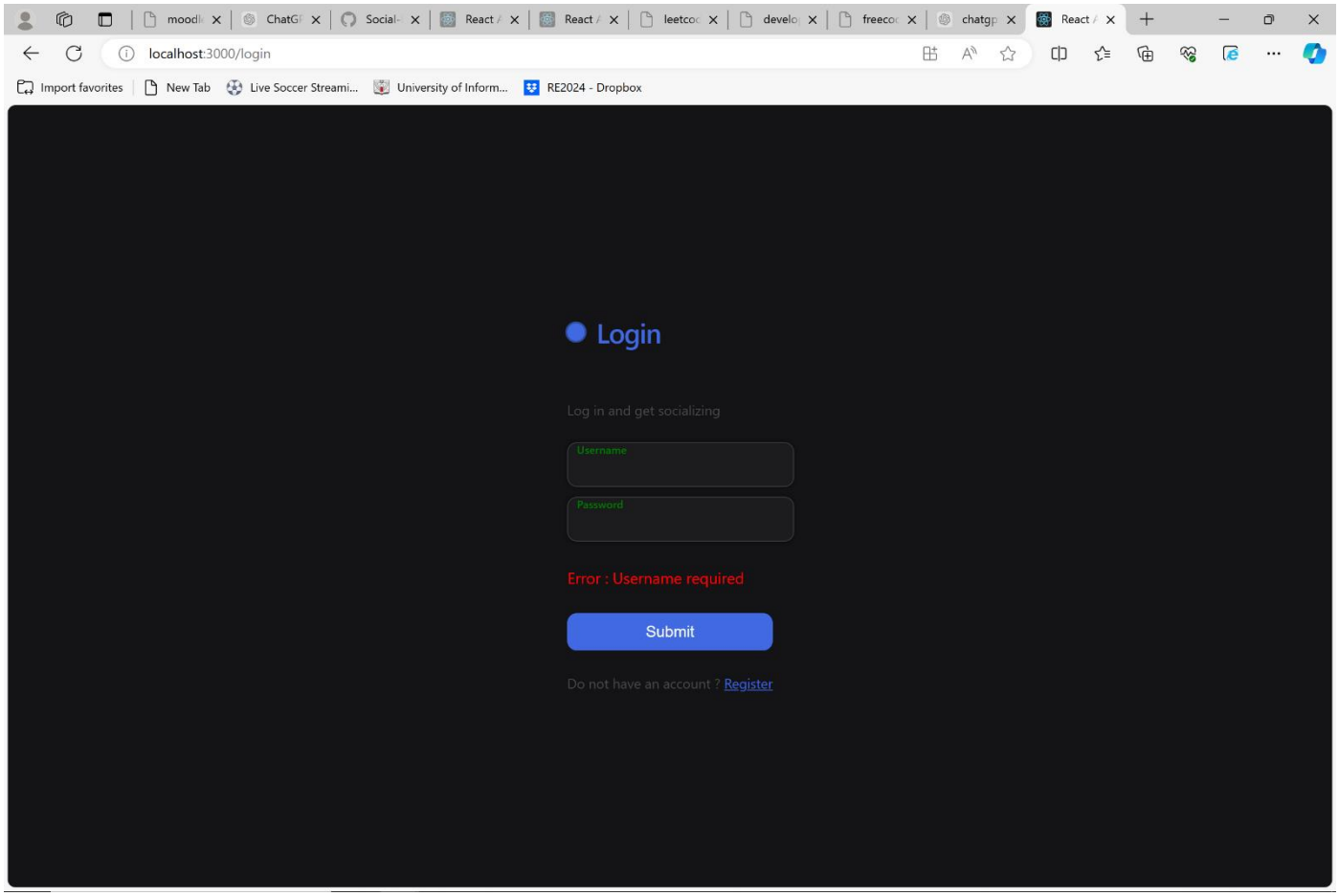

Fig. 5.6: Login page
Source: Own Study (03.06.2024)

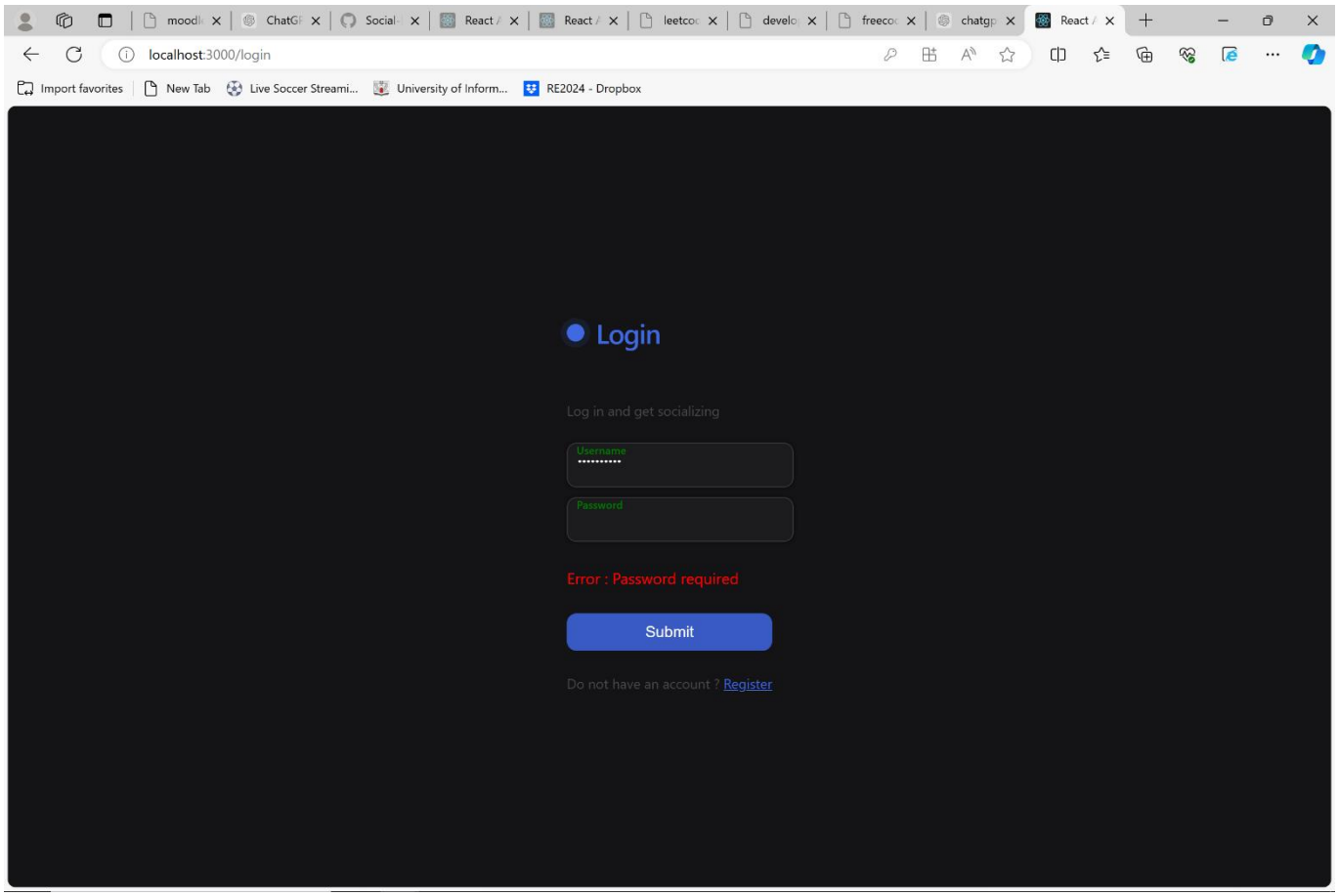

Fig. 5.7: Login page

Source: Own Study (03.06.2024)

Fig 5.8 and Fig. 5.9 Shows the error that occurs when a user enters an incorrect username or password.

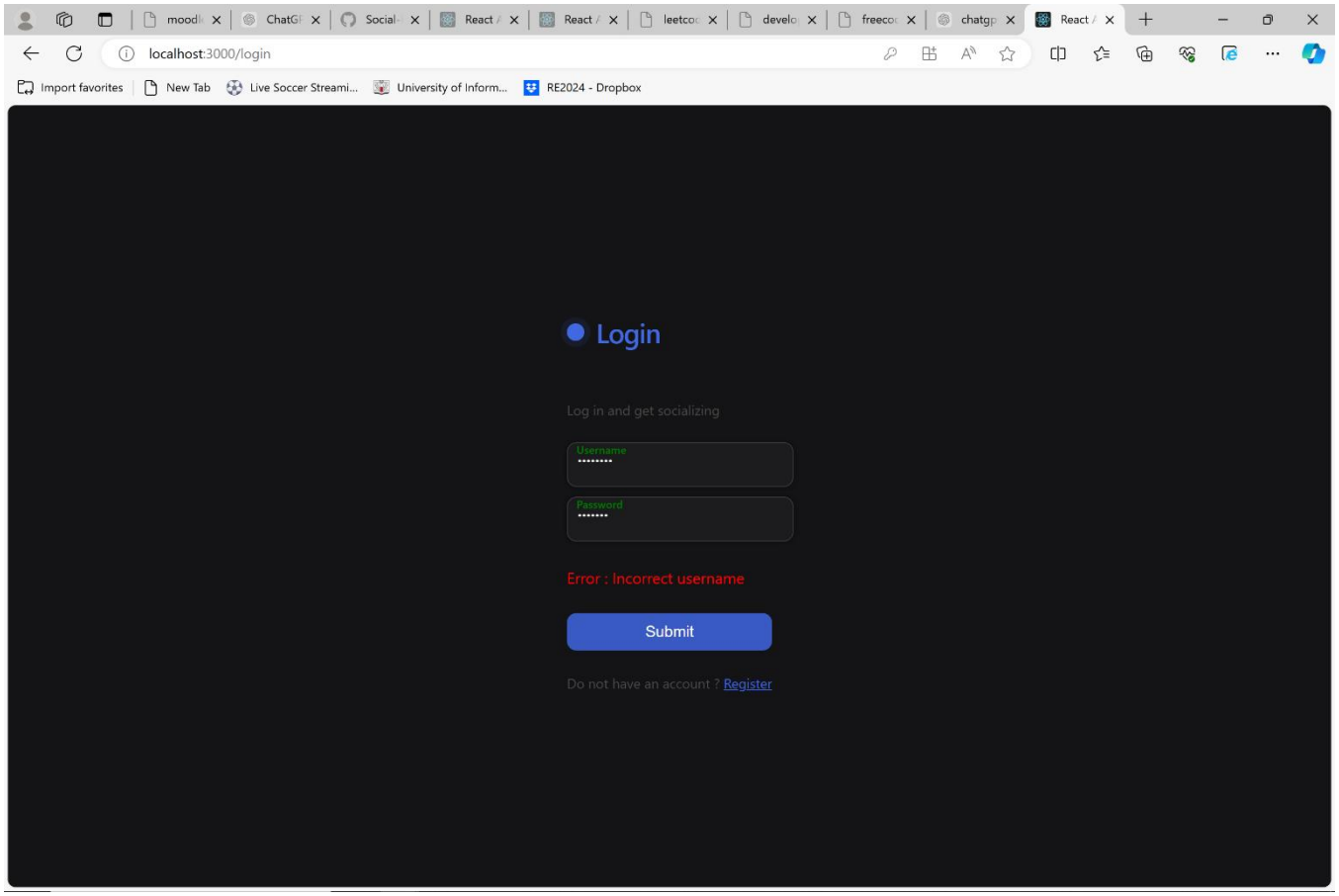

Fig. 5.8: Login page

Source: Own Study (03.06.2024)

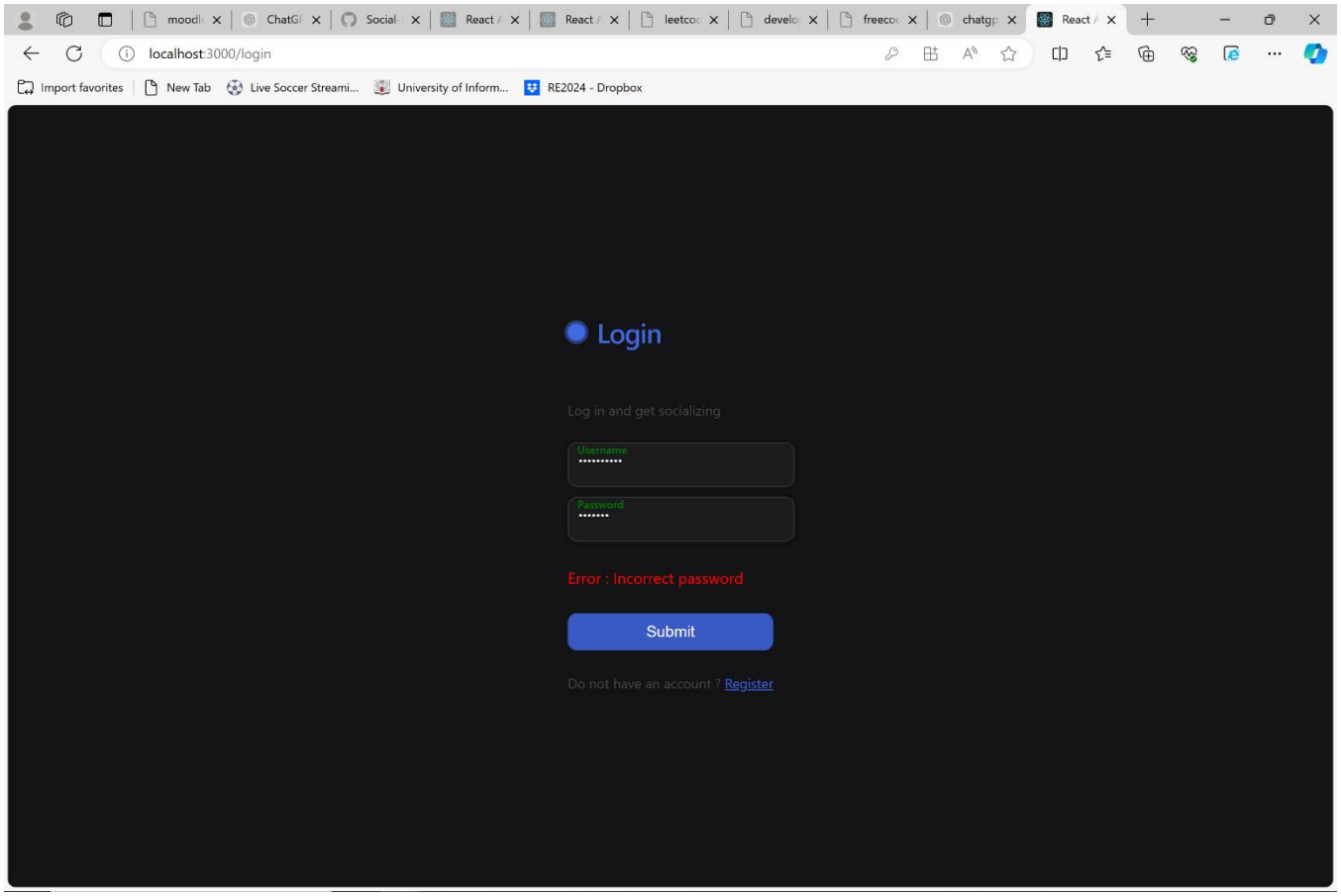

Fig. 5.9: Login page

Source: Own Study (03.06.2024)

3.2Database Table

3.2.1 Database Diagram

Fig 6.0 shows the ERD diagram for the application and how each table communicates with each other

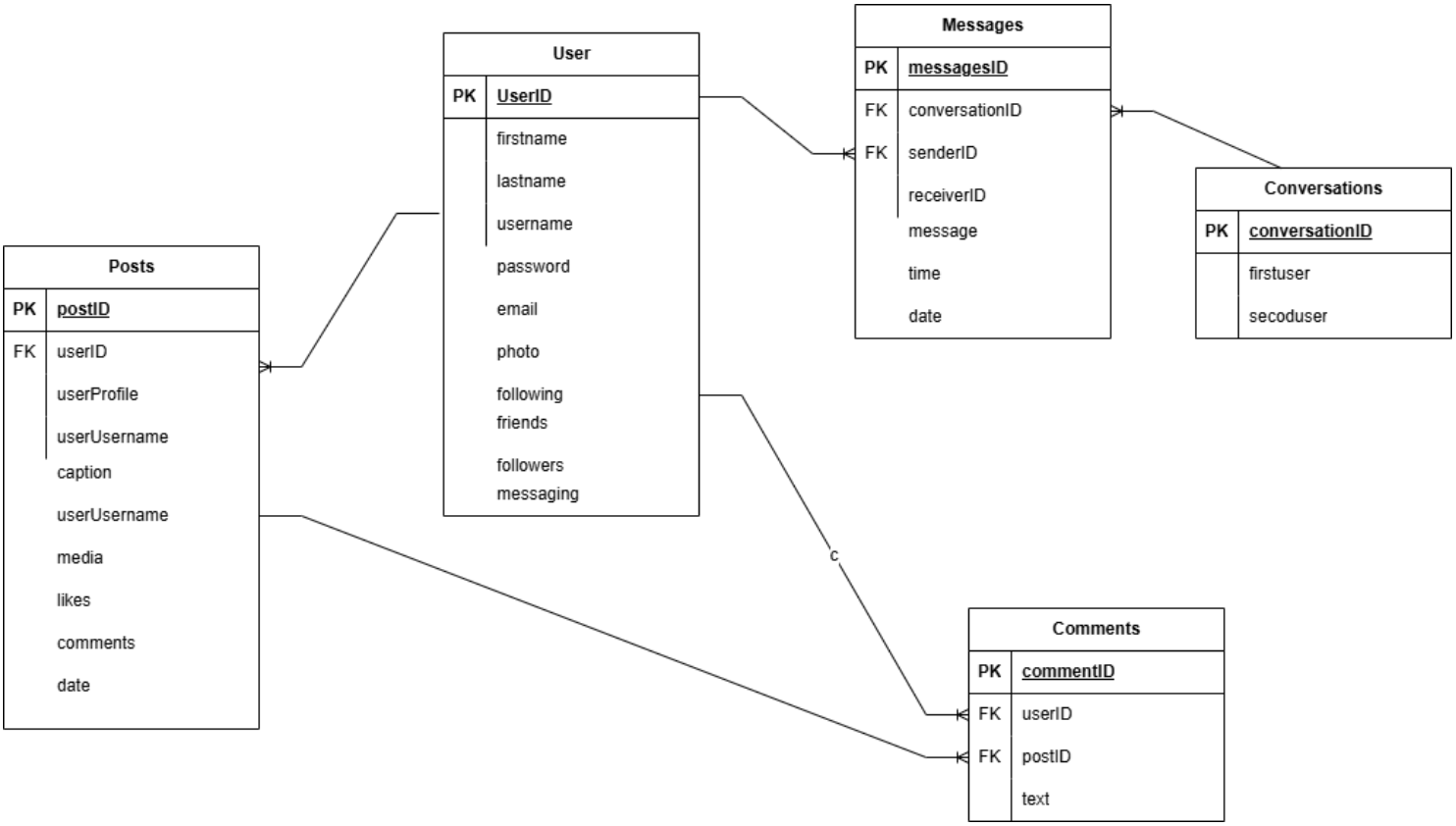

Fig. 6.0: ERD diagram
Source: Own Study (31.12.2023)

Fig.6.1 shows the user collection and how the users data is stored

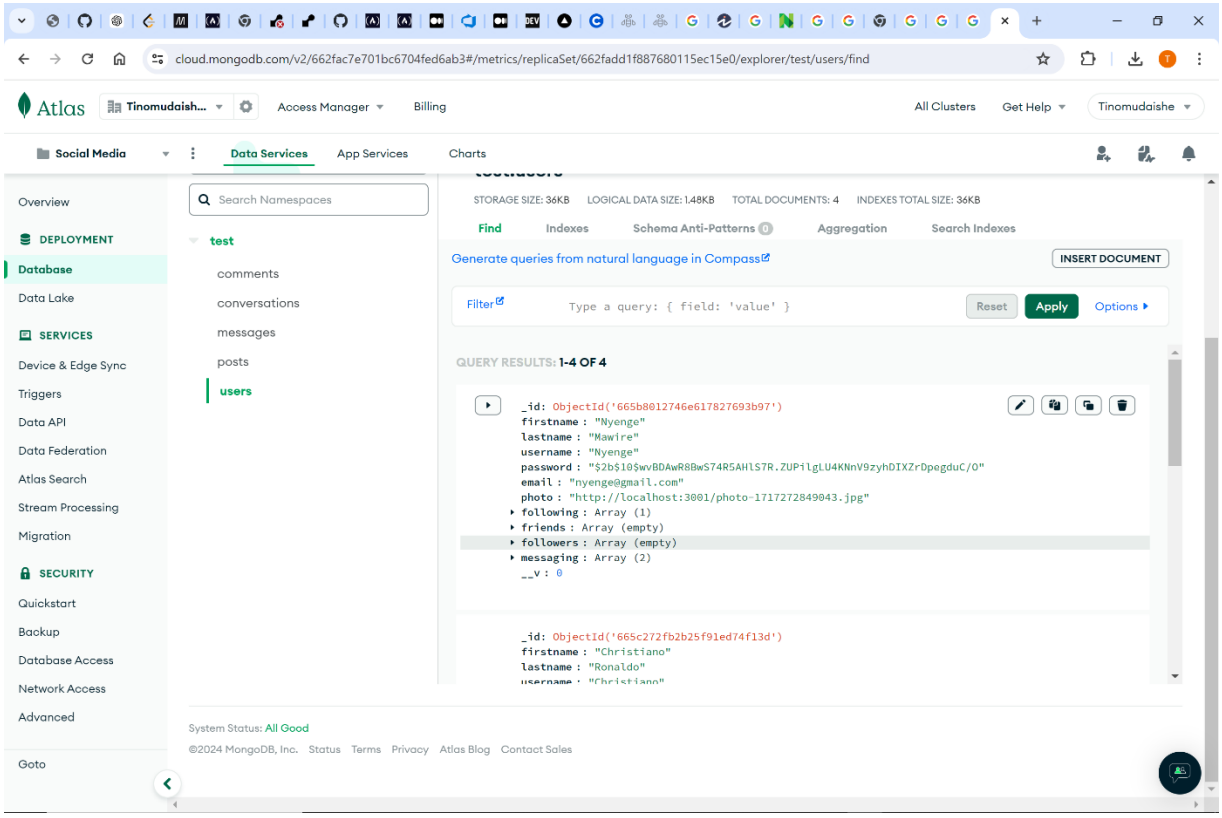

Fig. 6.1: Database Collection: Users
Source: Own Study (04.06.2024)

Fig. 6.2 shows the post collection and how the data is stored for each post

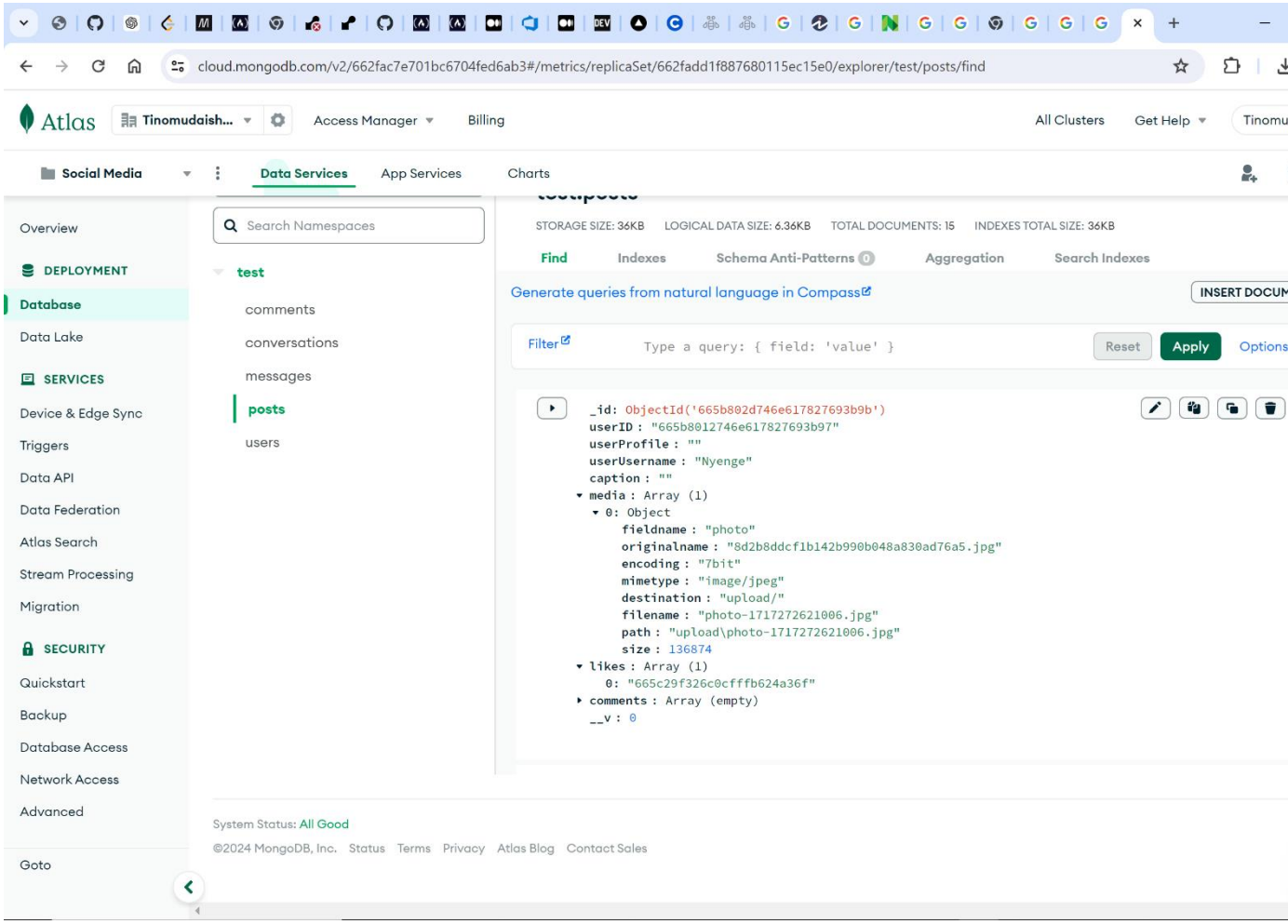

Fig. 6.1: Database Collection: Posts
Source: Own Study (04.06.2024)

Fig. 6.3 shows the conversation collection and how the data is stored for each conversation

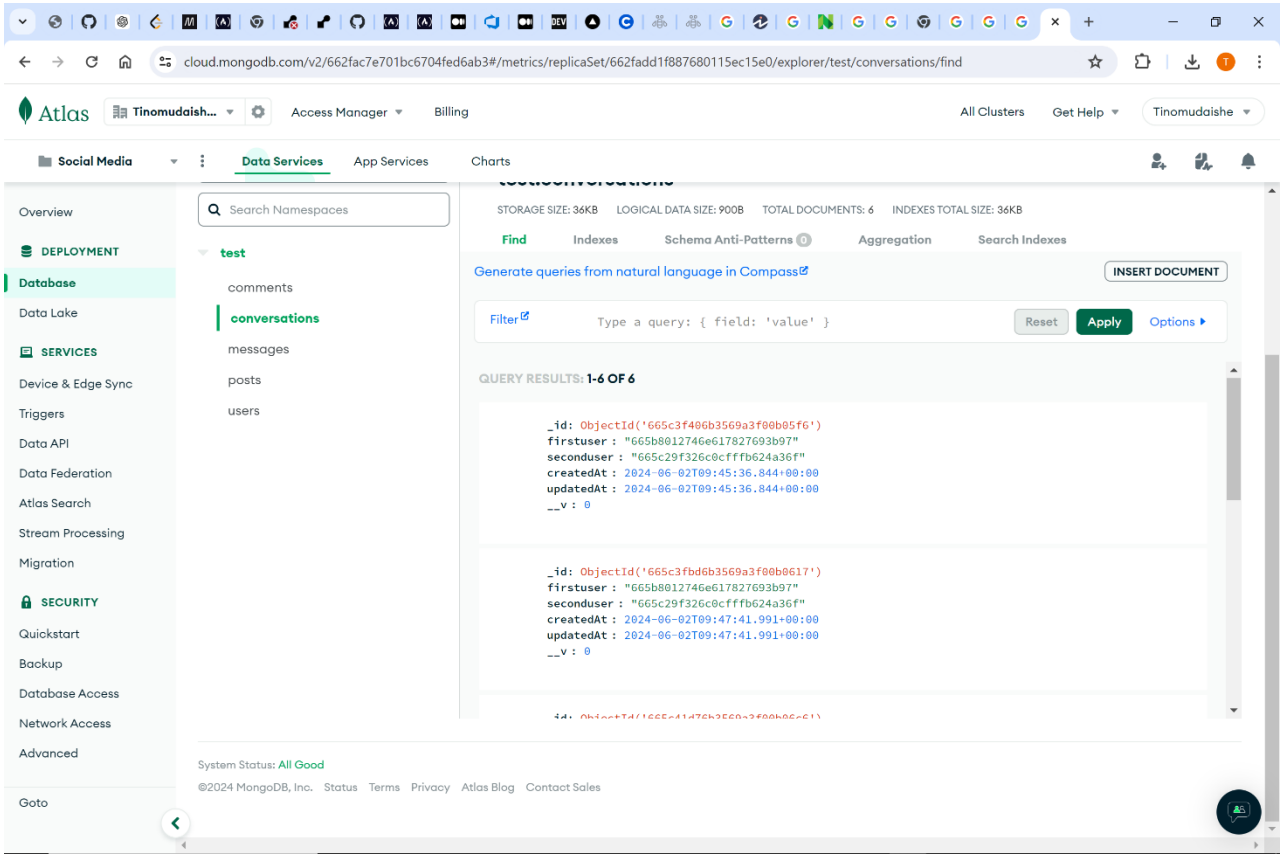

Fig. 6.2: Database Collection: Conversations
Source: Own Study (04.06.2024)

Fig.6.3 shows the messages collection and how the data is stored for each message

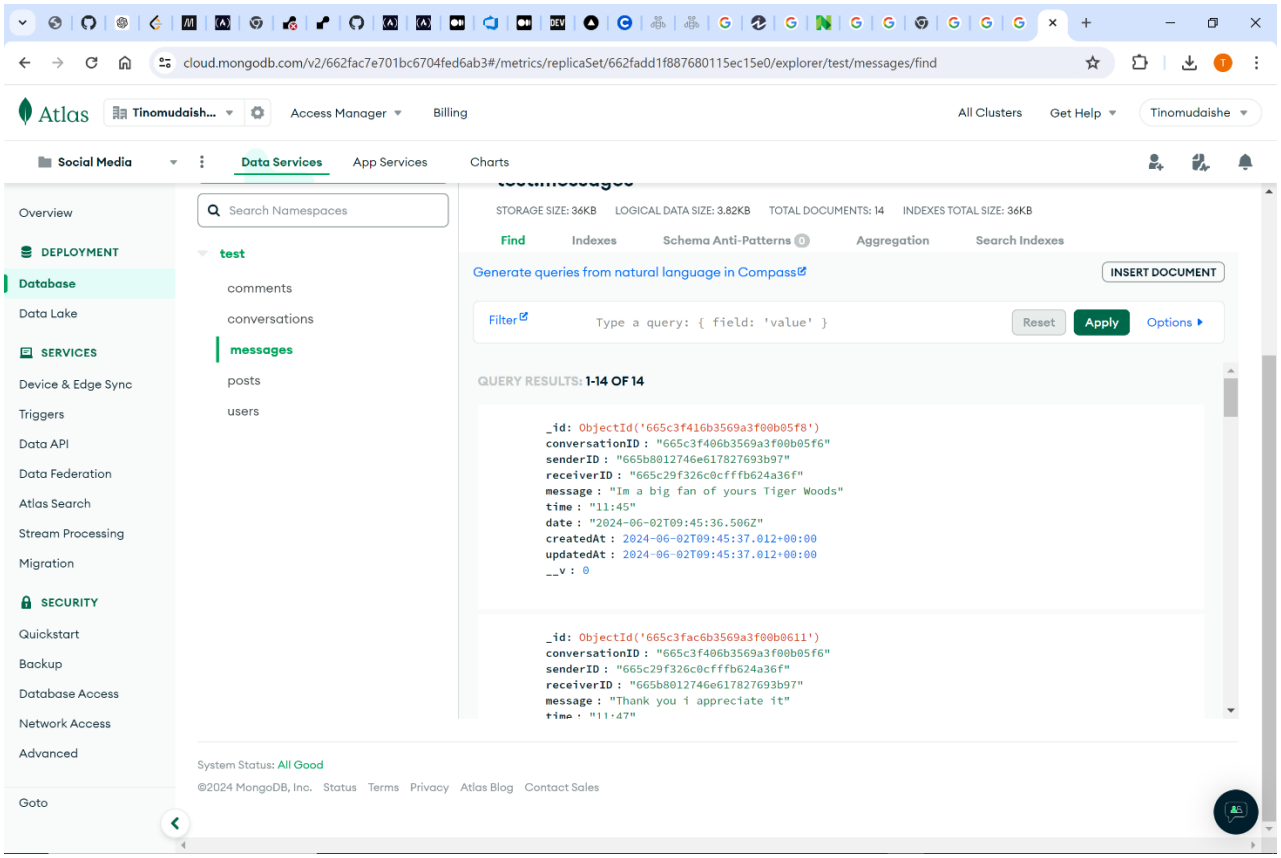

Fig. 6.3: Database Collection: Messages

Source: Own Study (04.06.2024)

Fig.6.4 shows the comments collection and how the data is stored for each comment

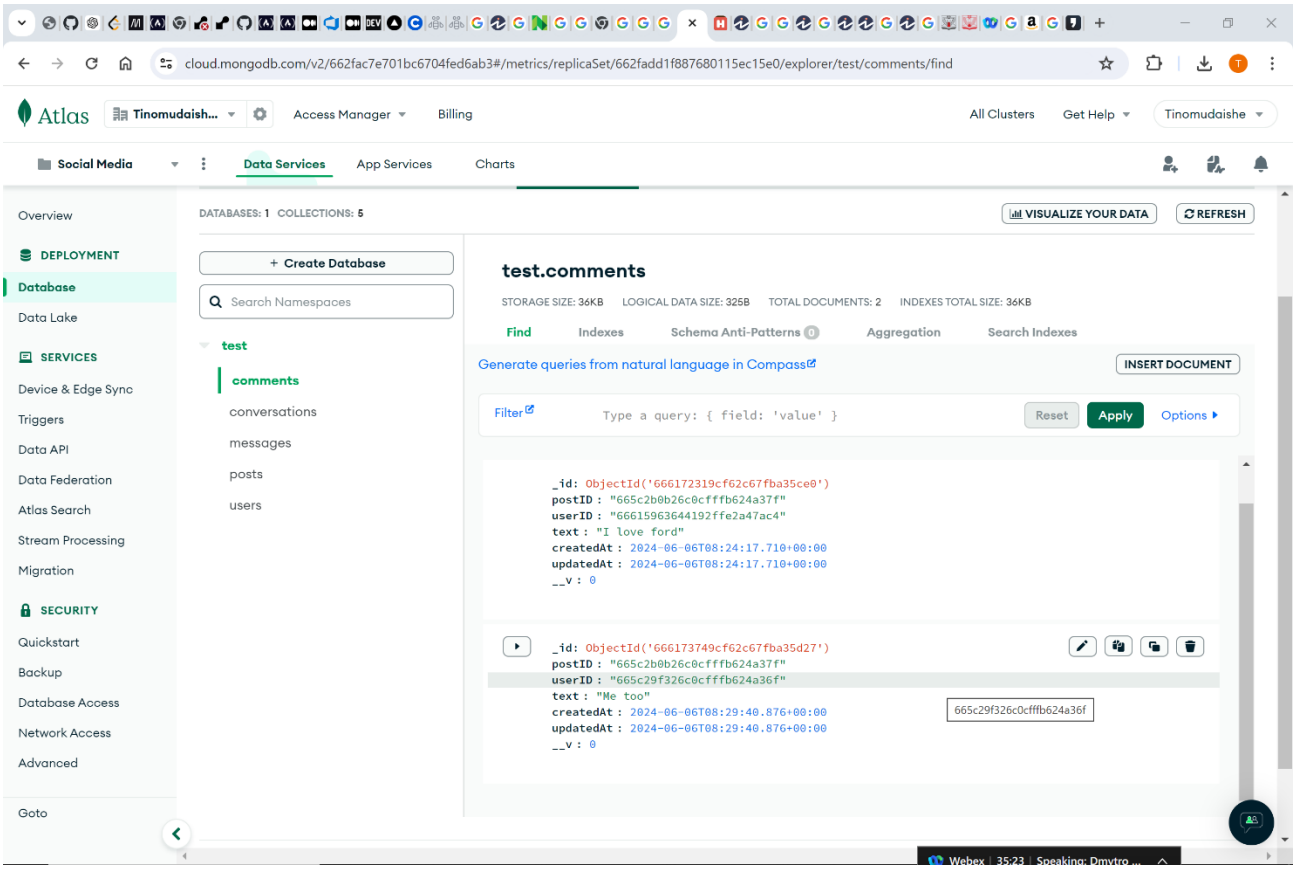

Fig. 6.1: Database Collection: Comments
Source: Own Study (04.06.2024)

Chapter 4

Conclusion

In conclusion we have managed to implement a functional social media platform. It performs the basic needs that are implemented in some of the largest social media platforms today. We have managed to reach and accomplish all of our objectives and features. In as much as the project is good it lacks a few key functionalities that stop it from being great. The lack of a responsive frontend leaves for sloppy design that is distasteful. Lack of video or photo sharing in messaging feature make the communication capabilities really blunt and boring.

These functionalities are features that we can later on add to the application to give it more depth and make it more unique and appealing to the consumers eyes.

Bibliography

[1] React Documenation: https://react.dev/reference/react

[2] NodeJS Documentation :https://nodejs.org/en/learn/command-line/how-to-read-environmentvariables-from-nodejs

[3] React Routing :https://expressjs.com/en/guide/routing.html

[4] MongoDB : https://www.mongodb.com/docs/manual/core/document/

[5] CORS : https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS

[6] React beginners course : https://www.youtube.com/watch?v=f55qeKGgB_M

[7] ExpressJS tutorial: https://www.youtube.com/watch?v=39znK-Yo1o&list=PL_cUvD4qzbkwp6pxx27pqgohrsP8v1Wj2

Summary

Social Media Application

Author: Nyengeterayi Mawire, Johnson Mudzingwa
Supervisor: MSc. Marcin Jagieła

We have made a Social Media Application build on the MERN stack. Its capabilities allow for a user to upload a post in the form of either video or image. Users can like and leave comments on the post, and users can also message one another through the messaging feature incorporated in the application.