

PROJECT REPORT
ON
TWITTER CLONE

Submitted to MAHARAJA RANJIT SINGH PUNJAB TECHNICAL
UNIVERSITY in partial fulfillment of the requirement for the award of the
degree of

B. TECH

In

COMPUTER SCIENCE & ENGINEERING

Submitted By

JASVEER SINGH, Roll. No. 190280060

PARSHANT KHICHI, Roll. No. 190280089



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
GIANI ZAIL SINGH CAMPUS COLLEGE OF ENGINEERING &
TECHNOLOGY, MRSPTU, BATHINDA-151001

DEC 2022

PROJECT REPORT
ON
TWITTER CLONE

Submitted to MAHARAJA RANJIT SINGH PUNJAB TECHNICAL
UNIVERSITY in partial fulfillment of the requirement for the award of the
degree of

B. TECH

In

COMPUTER SCIENCE & ENGINEERING

Submitted By

JASVEER SINGH, Roll. No. 190280060

PARSHANT KHICHI, Roll. No. 190280089



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
GIANI ZAIL SINGH CAMPUS COLLEGE OF ENGINEERING &
TECHNOLOGY, MRSPTU, BATHINDA-151001

DEC 2022

PREFACE

Projects are an integral part of B.Tech and each and every student has to create several projects throughout the degree.

This record is concerned about our project created during the 7th semester of our B.Tech. We have created **Twitter Clone** as our project. During this project, we got to learn many new things about the industry and the current requirements of companies. This project proved to be a milestone in our knowledge of present industry. Every day and every moment was an experience in itself, an experience which theoretical study can't provide.

ACKNOWLEDGEMENT

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced our thinking, behavior and acts during the course of study.

We express our sincere gratitude to ***Prof. Jyoti Rani*** worthy HOD, ***Dr. Dinesh Kumar*** and ***Er. Manpreet Kaur***, Training & Placement In-charge for providing us an opportunity to work on this wonderful project.

We are especially thankful to **Dr. Dinesh Kumar** for his support, cooperation, and motivation provided to us during the project for constant inspiration, presence and blessings.

Lastly, we would like to thank the almighty and my parents for their moral support and our friends with whom we shared our day-to-day experience and received lots of suggestions that our quality of work.

Jasveer Singh,
Parshant Khichi

CANDIDATE’S DECLARATION

We, **Jasveer Singh**, Roll No. 190280060 and **Parshant Khichi**, Roll No. 190280089, B.Tech (Semester-VII) of the **Gaini Zail Singh Campus College of Engineering & Technology, Bathinda** hereby declare that the Project Report entitled “**TWITTER CLONE**” is an original work and data provided in the study is authentic to the best of my knowledge. This report has not been submitted to any other Institute for the award of any other degree.

Jasveer Singh

(Roll No. 190280060)

Parshant Khichi

(Roll No. 190280089)

Place: Bathinda

Date: Dec 2022

Table of Contents

| | |
|--|----|
| CHAPTER 1: Introduction..... | 1 |
| CHAPTER 2: Features of Twitter Clone..... | 2 |
| 2.1 Sign-In user | 2 |
| 2.2 Register new account..... | 4 |
| 2.3 Post tweets..... | 5 |
| 2.4 Display Tweets..... | 7 |
| 2.5 Like Tweets..... | 8 |
| 2.6 Retweet..... | 8 |
| 2.7 Comments | 8 |
| 2.8 Search Posts..... | 9 |
| 2.9 Search Users..... | 9 |
| 2.10 Profiles | 10 |
| 2.11 Notifications | 11 |
| 2.12 Inbox | 12 |
| CHAPTER 3: Softwares Used | 14 |
| CHAPTER 4: Technologies Used | 15 |
| 4.1 Frontend Technologies | 15 |
| 4.2 Backend Technologies | 17 |
| CHAPTER 5: Conclusion and future scope | 20 |

CHAPTER 1: INTRODUCTION

The goal of this project is to implement a Twitter Clone. As it is very obvious that it is not practically possible to include all the features of the original Twitter app, our goal here is to just implement all the basic but important features that the original Twitter provides. So this twitter-clone is what came up with the short time span that we had.

We have copied all the styles from the original twitter. This app is responsive which means this can be viewed on the small screen like mobiles as well as bigger screens like desktop PCs.

We are using MongoDB atlas as a database store all the information of users, posts, messages etc. MongoDB is not good enough for uploading the images, so we uploaded the actual image to amazon AWS bucket and just the link of it stored in the MongoDB document. That made the posting and retrieving of images very easy. We've the free plan for AWS.

We've used Heroku to deploy our app. Here too we've the free plan of Heroku as we right now have only a small number of users.

Here is the link to our web-app: <https://twitter2.up.railway.app/>

and code is uploaded on the GitHub: <https://github.com/Parshant7/twitter-clone/>

CHAPTER 2: Features of Twitter Clone

2.1 SIGN-IN USERS:

When users arrive at the home page they are required to either login using existing account or signup if they don't have any account registered. It is an essential process for a user for further usage of the app.

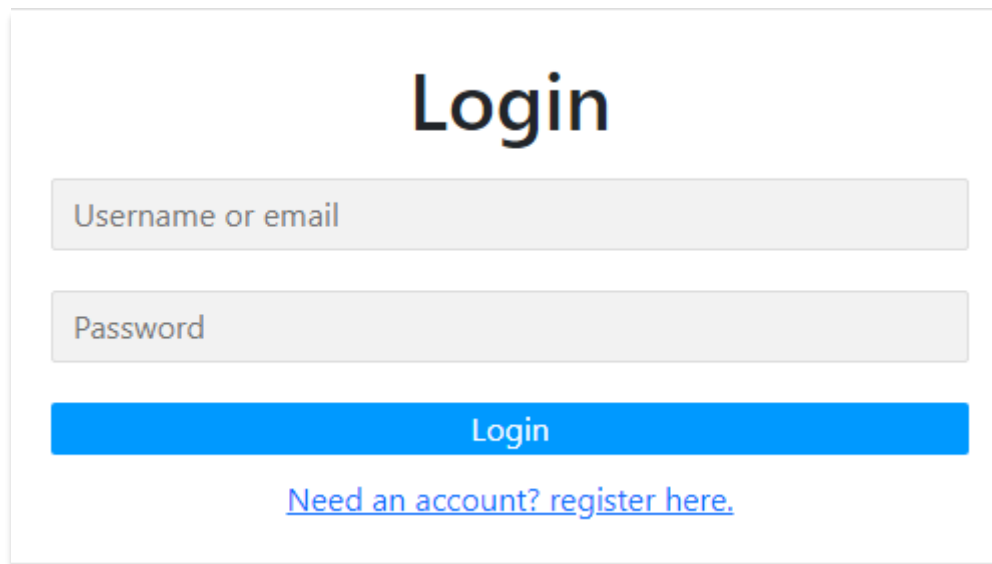
A login form with a white background and a light gray border. At the top, the word "Login" is displayed in a large, bold, black font. Below it, there are two input fields: the first is labeled "Username or email" and the second is labeled "Password". Both labels are in a light gray font. Below the input fields is a blue button with the word "Login" in white. At the bottom, there is a blue hyperlink that says "Need an account? register here."

Figure 2.1.1 login page

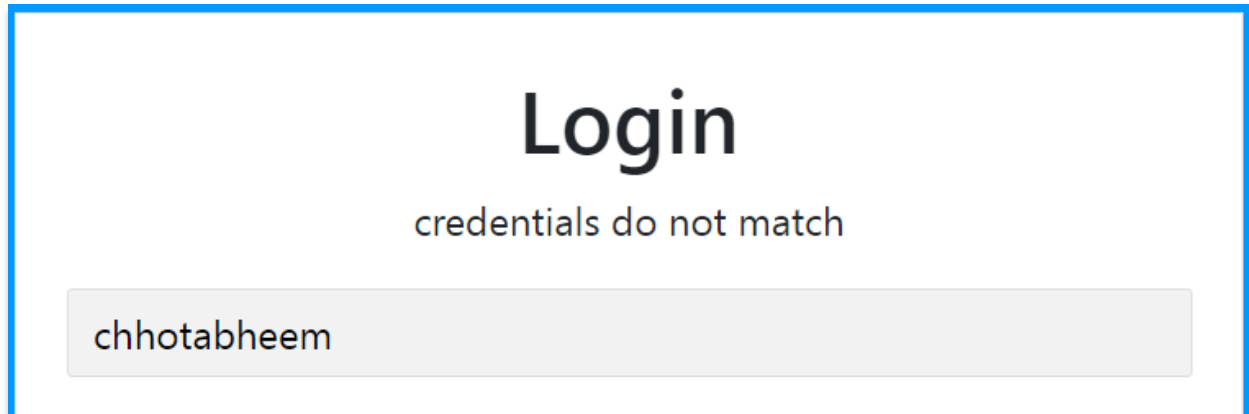
This is also the home page of the twitter-clone webapp. It includes following features with respect to the sign in process.

Users are allowed to sign in using their username or email address, which they have entered while creating their account.

It has a placeholder for both the fields: username and password which makes it easier for a user to identify where to enter his username and password. Both the fields are required fields.

A click handler is attached to the login button which when clicked checks for credentials by making a GET request to server and obtain the results.

In case of wrong username or password entered a message “*credentials do not match*” is displayed on the screen.



The image shows a login interface within a blue-bordered box. At the top, the word "Login" is displayed in a large, bold, black font. Below it, the text "credentials do not match" is shown in a smaller, black font. Underneath the error message is a light gray rectangular input field. Inside this field, the text "chhotabheem" is entered in a black font.

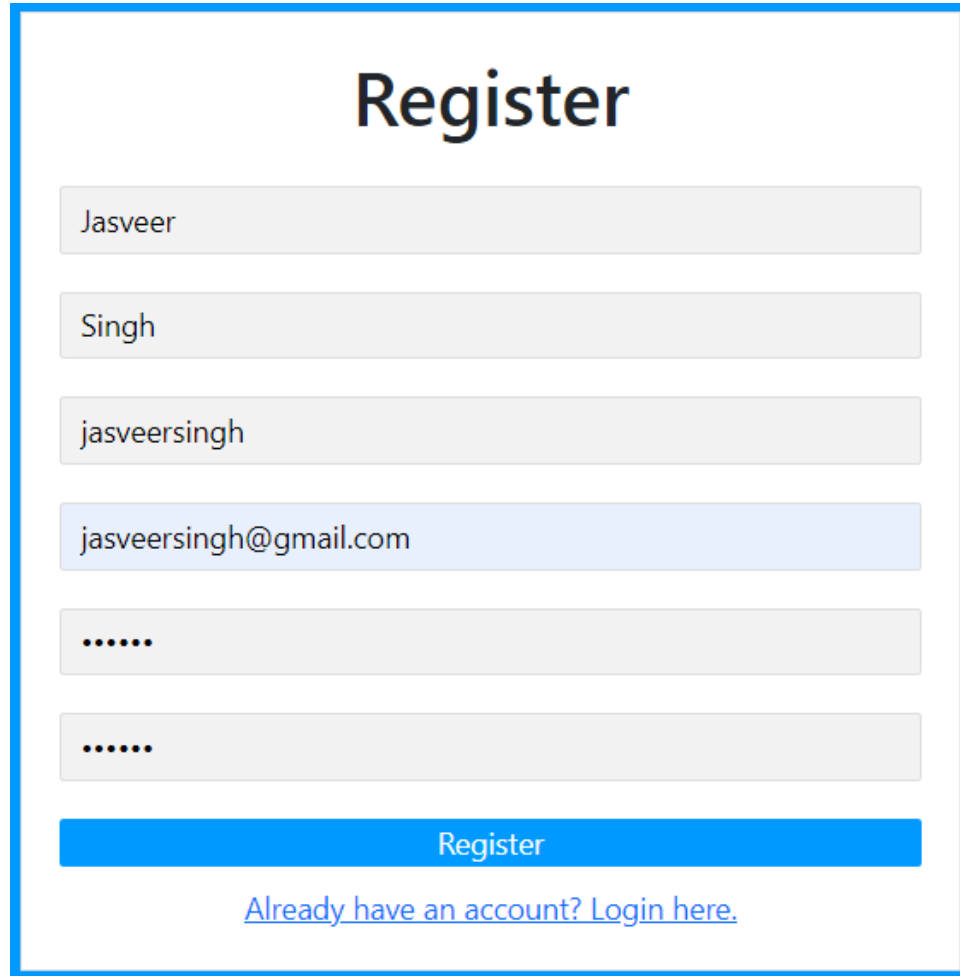
Figure 2.1.2 Authentication

Username or email address entered will be automatically entered in the respective field.

If the users does not have any existing account, a link namely “*Need an account? Register here*” will take them to the register page.

2.2 REGISTER NEW ACCOUNT:

This page allows users to create an account for twitter-clone webapp if they have not already an existing account.



The registration form is titled "Register" in a large, bold, dark blue font. It contains six input fields: "Jasveer" (first name), "Singh" (last name), "jasveersingh" (username), "jasveersingh@gmail.com" (email), and two password fields (both containing "....."). The email field is highlighted with a light blue background. Below the input fields is a blue "Register" button. At the bottom, there is a link that says "Already have an account? Login here." in blue text.

Figure 2.2.1 Sign up page

It includes following functions and features :

This register form includes 6 input fields.

It includes register button to which a click handler is attached.

When register button is clicked it will first check if all the fields have some value in it, secondly it will check if the password and confirm password fields have the same value. In case if the both the conditions are not fulfilled an alert will be shown on the register page “passwords do not match. Please try again”.

It will then make a POST request to the server. The server will check if there exist a user with the same name or password or not. In case there exist a user, it will show us a message “*user already exists*”.

After successfully registering the new user the page will be redirected to the home page.



Figure 2.3.2 Mongodb atlas collection

2.3 POST TWEETS :

A small form on the top of the home page allows users to post tweets as well as upload images together.

A small image icon will trigger the bootstrap modal which will allow users to select files from there local storage and crop them.

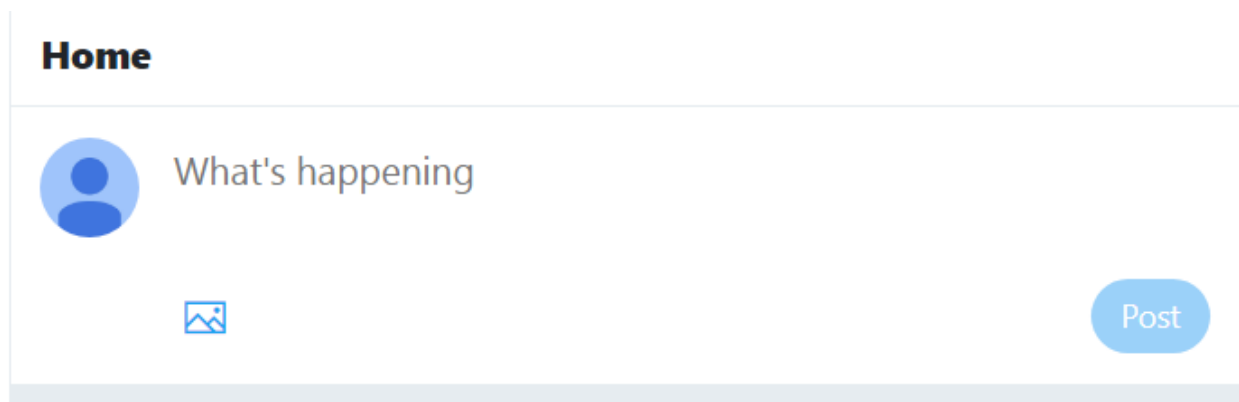


Figure 2.3.1 Post tweet form

After clicking on the post button a post request will be triggered which will also contain the post data. The image will be stored in aws bucket and the text will be stored inside the mongodb atlas database with all the information regarding the posts like posted by, likes, comments, time etc.



Figure 2.3.2 Image preview on the form after being chosen by the user

2.4 DISPLAY TWEETS :

Once the post gets uploaded successfully home page will be refresh and the post will be shown to user in the post container.



Figure 2.4.1 Tweet uploaded and shown in the feed

A post contains :

- Name of the user (first name + last name) in the bold letters
- Username of the user and the relative time when the tweet was posted
- It also includes a pin button, post delete button, tweet text and uploaded image if any with the post.
- Footer includes three buttons comment, retweet and like button.

2.5 LIKE TWEETS :

The like button turns to red and also shows the count of the likes of the respective post. A request is made when the user clicks the like button to the server and in the server the id of the user who likes the post gets added to the *likedby* array of the post.

2.6 RETWEETS :

Just like original tweeter we're allowed to retweet a post by clicking on retweet button. This retweet button is displayed in the green color. When clicked a post is created just like the original post but it has *isRetweet* value as true of the post which indicates that it is a retweet and not a normal tweet.



Figure 2.6.1 Retweets indication

2.7 COMMENTS:

A comment button attached to the bottom of the post allows us to comment on the post. A bootstrap modal is triggered when this button gets clicked and allows user to comment on the particular post.

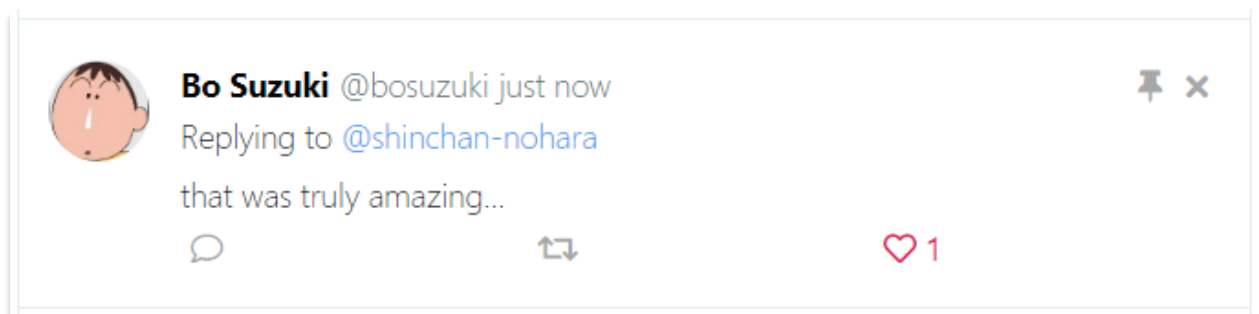


Figure 2.7.1 Comments

All these posts are clickable and clicking on them will take us to the post page where we can see the post and each comment on this post together.

2.8 SEARCH POSTS:

A search page is also provided to search for the posts and users.

User can switch through posts and users by clicking on those two buttons. Whenever the key down event is triggered by typing a term in the search bar, after 1sec the term inside the search box is evaluated and searched through the database by making a get request to the server. As a result posts will be displayed in the result container.

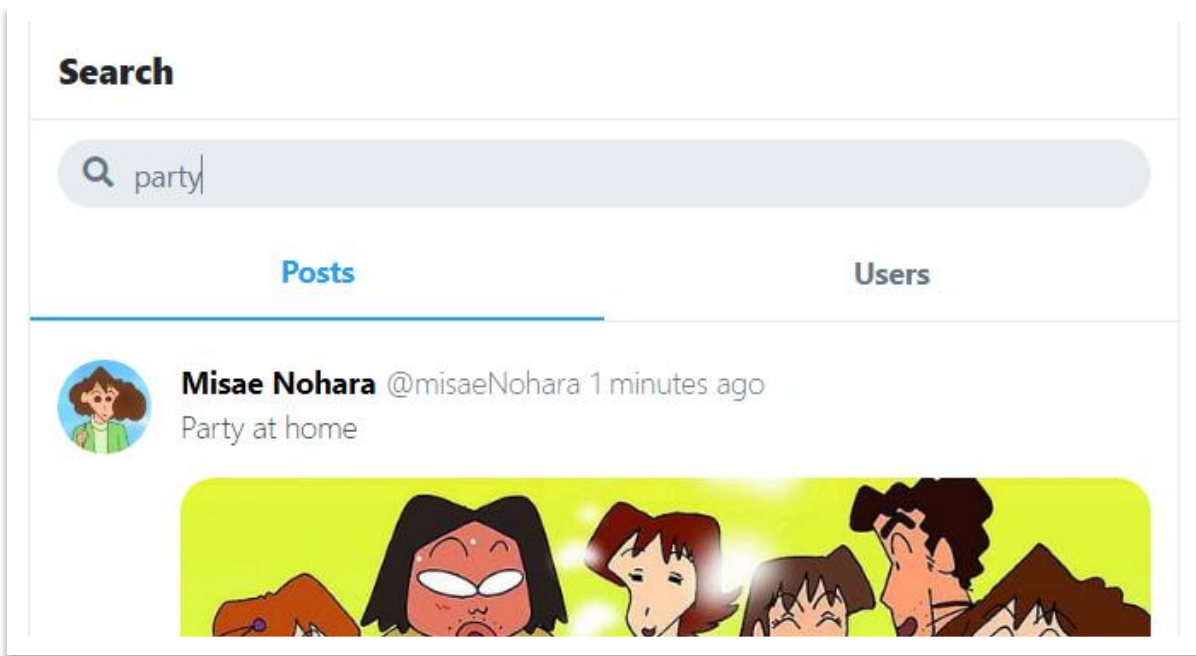


Figure 2.8.1 Searching posts through keywords

2.9 SEARCH USERS:

Similar way a user can search another users by typing their names inside the search bar and results will be resplayed like the image shown below.

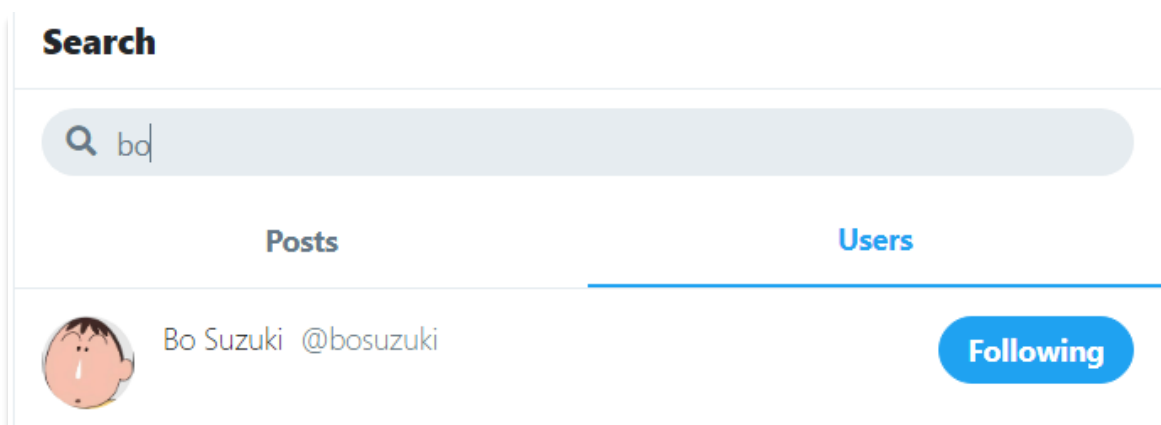


Figure 2.8.1 Searching users through usernames

2.10 PROFILE PAGE:

It shows the whole profile of the user. The profile page of a user includes following things:

Profile Picture: when the user account is created, a default profile picture is assigned to every user which is present in “/images/profilePic.jpeg” location.

On hover on the profile picture a camera icon is displayed if user is on his own profile. By clicking on the camera button a user can upload his/her image just like they upload images in the posts. A bootstrap modal is triggered when clicked on the button. The profile picture ratio is fixed at 1:1.

Cover Picture: By default it just the blue background which can be changed by the users just like profile picture.

Full name and Username: first name and last name is retrieved from the server and it is combined to show the full name and username with different style format.

Followers and Following: Count of the followers and followings is displayed. These are the clickable links which takes us to page which shows us the complete list of the followers and following users.

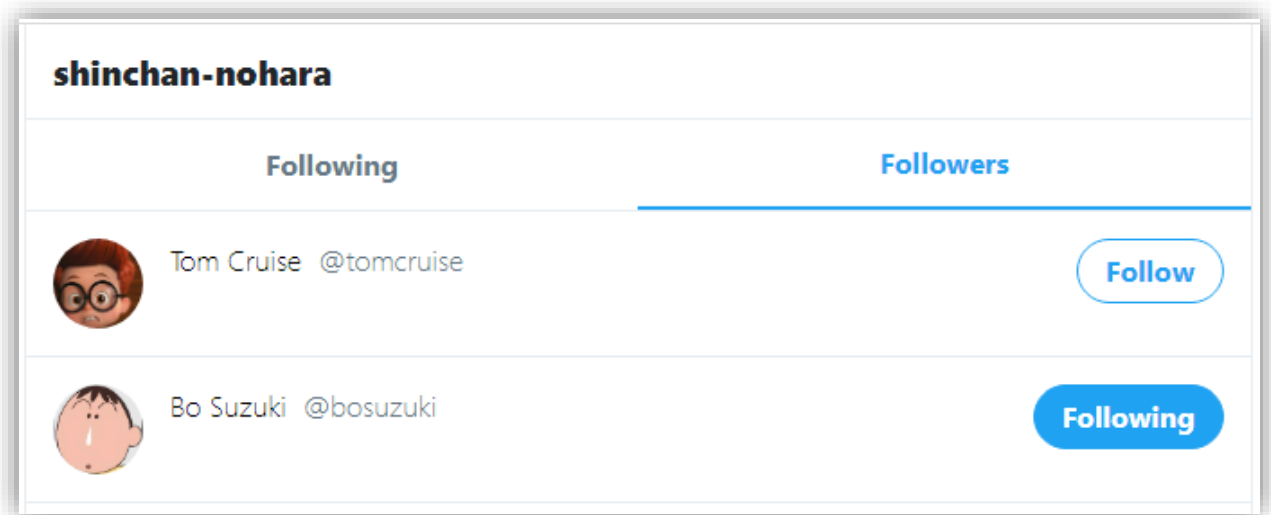


Figure 2.10.1 Followers list

Posts and replies: all the posts and replies of the user are shown in the order of upload time.

Follow and Unfollow button: based on the condition if that user is followed by person logged in or not and is it our own profile or not, a follow button is displayed.

Message button: A message button is also shown to chat with that person if the user is not viewing its own profile.

2.11 NOTIFICATIONS:

This notification page shows all the notifications of the users. A notification is sent to user when any one like, retweet, reply on their posts as well as as when a user follows that person.

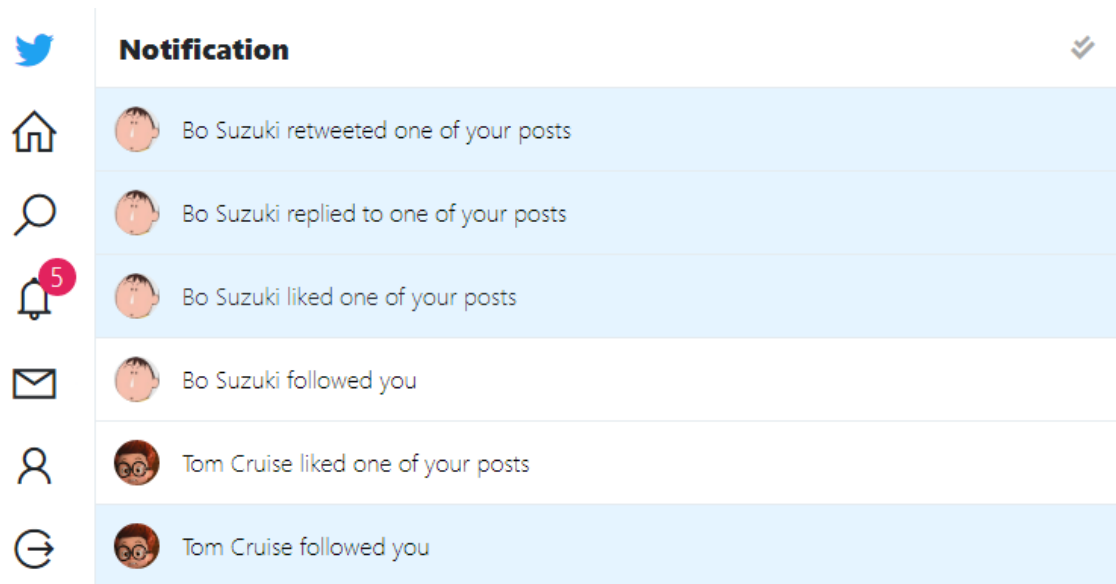


Figure 2.11.1 Notifications

On the left side the bell icon shows no. of unchecked notifications.

A check button is given on the right-top corner. This button will mark all the notifications as read when clicked and the page will be refreshed.

All these notifications are clickable and will take the user on posts or profile based on the notification type.

Notifications are displayed in real time. These notifications are popped up at the right-top corner of the screen. These are clickable and will direct the user to the respective page when clicked. These popup will disappear after 5 sec.



Figure 2.11.2 Real-time notifications

2.12 INBOX :

Inbox page consists of all the chats in which the user loggedIn is involved. When the message button is clicked a get request is made to the server. At the server end, all the chats related to user are searched from the database by accessing the userId through `req.body.session.userLoggedIn` and all the results are sent back to the browser.

The inbox page includes one to one chat and group chats.

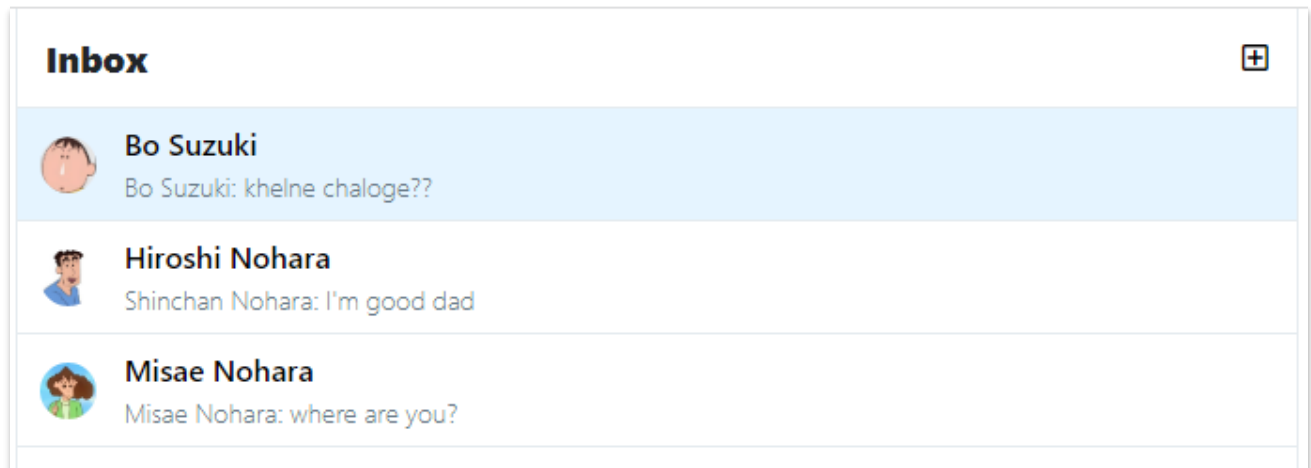


Figure 2.12.1 Inbox

All the unread chats have with light blue color background. Clicking on them will take the users to the chat page.

The button present at the top right corner allows to create a new new group chat

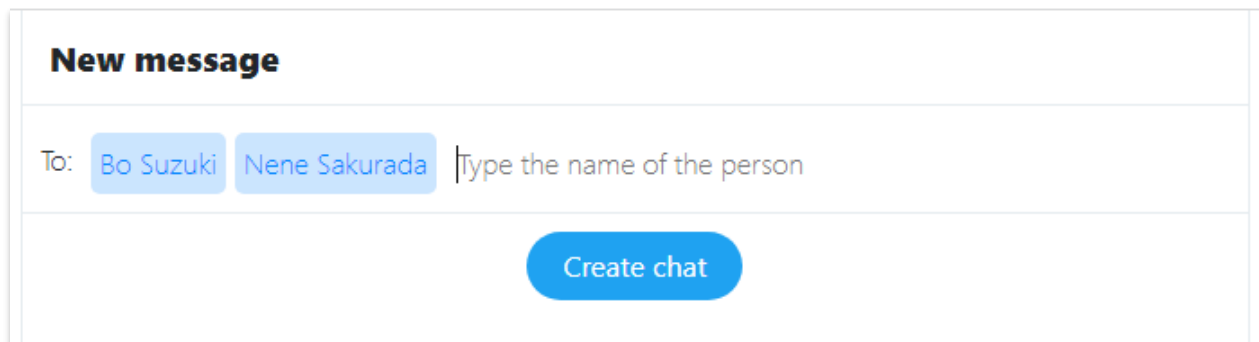


Figure 2.12.2 Creating group chat

User can search for other users by there names. Typing anything will trigger an event which will search for the name from the database and show that user in below bar. From there user can click on that bar and that particular user will be added to list of the group chat. These users can be removed from the list just by typing backspace.

Then user can click on the create chat button and a new chat will be created with no name.

By default a group chat name is a combination of the users present in the group. A group chat's name can be changed by clicking on the name of chat. A bootstrap modal will be created which will allow us to enter the chat name.

Entry of this chat looks something like this in the mongodb Atlas.

```
_id: ObjectId("62c971483fd56d91613ef50b")
isGroupChat: true
> users: Array
  createdAt: 2022-07-09T12:15:04.430+00:00
  updatedAt: 2022-07-09T13:08:40.990+00:00
  __v: 0
chatName: "Kasukabe defence group"
latestMessage: ObjectId("62c97dd83fd56d91613ef57b")
```

Figure 2.12.3 Chat's representation as database document

There's a typing indicator which is just a gif placed above chat textbox and trigger when typing socket event is emitted.



Figure 2.12.4 User typing indicator

A message can be send through that nice looking airplane button as well as by hitting enter key.

A message stored in the database looks like this.

```
> _id: ObjectId("62c97f4a3fd56d91613ef5d5")
  sender: ObjectId("62c95845dd2636693e1205aa")
  content: "hi dosto"
  chat: ObjectId("62c971483fd56d91613ef50b")
  readBy: Array
  createdAt: 2022-07-09T13:14:50.049+00:00
  updatedAt: 2022-07-09T13:23:30.560+00:00
  __v: 0
```



Figure 2.13.1 Message's stored as database document

CHAPTER 3: Softwares used

VS Code : Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, .NET).

Node.js : is a JavaScript runtime built on Chrome's V8 JavaScript engine. It is available to download from its official website and used to incorporate the facility of Node.js into our project.

Git : Git is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows.

Postman: Postman is application that allows us to do Api testing.It is like browser that doesn't render html.In browser we can hit only get HTTP request but here we can hit get,post,put,patch,delete and many more HTTP request in api.

Chrome DevTools : Chrome DevTools is a set of web developer tools built directly into the Google Chrome browser. DevTools can help you edit pages on-the-fly and diagnose problems quickly, which ultimately helps you build better websites, faster.

CHAPTER 4: Technologies used

4.1 Frontend Technologies:

HTML: The Hyper Text Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

CSS: Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media. CSS is among the core languages of the open web and is standardized across Web browsers according to W3C specifications.

JS: JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles. Read more about JavaScript.

pug.js: also known as PUG, is a Javascript library that was previously known as JADE. It is an easy-to-code template engine used to code HTML in a more readable fashion. One upside to PUG is that it equips developers to code reusable HTML documents by pulling data dynamically from the API.

Bootstrap: Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains HTML, CSS and JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components. Bootstrap CDN is incorporated into the project to use its classes and functionality.

Jquery: jQuery is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax. It is free, open-source software using the permissive MIT License. As of May 2019, jQuery is used by 73% of the 10 million most popular websites. Web analysis indicates that it is the most widely deployed JavaScript library by a large margin, having at least 3 to 4 times more usage than any other JavaScript library. jQuery CDN is included into the project for usage.

Font Awesome: Font Awesome is a font and icon toolkit based on CSS and Less. As of 2020, Font Awesome was used by 38% of sites that use third-party font scripts, placing Font Awesome in second place after Google Fonts.

Line Awesome: Line Awesome is a free alternative for Font Awesome 5.11. 2. It consists of ~1380 flat icons that offer complete coverage of the main Font Awesome icon set. This icon-font is based off of the Icons8 Windows 10 style, which consists of over 4,000 icons.

4.2 Backend Technologies:

Node.js: Node.js is primarily used for non-blocking, event-driven servers, due to its single-threaded nature. It's used for traditional web sites and back-end API services, but was designed with real-time, push-based architectures in mind.

Express.js: Express.js, or simply Express, is a back end web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs.

NPM: npm is a package manager for the JavaScript programming language maintained by npm, Inc. npm is the default package manager for the JavaScript runtime environment Node.js. It consists of a command line client, also called npm, and an online database of public and paid-for private packages, called the npm registry.

NPM packages used:

cookie-session: This is a Node.js module available through the npm registry. The session cookie is a server-specific cookie that cannot be passed to any machine other than the one that generated the cookie. The session cookie allows the browser to re-identify itself to the single, unique server to which the client had previously authenticated. Installation is done using the npm command:

```
$ npm install cookie-session
```

aws-sdk: The AWS SDK for JavaScript simplifies use of AWS Services by providing a set of libraries that are consistent and familiar for JavaScript developers. It provides support for API lifecycle consideration such as credential management, retries, data marshaling, serialization, and deserialization. The AWS SDK for JavaScript also supports higher level abstractions for simplified development.

```
$ npm install aws-sdk
```

bcrypt: The bcrypt npm package is a JavaScript implementation of the bcrypt password hashing function that allows you to easily create a hash out of a password string . Unlike encryption which you can decode to get back the original password, hashing is a one-way function that can't be reversed once done.

```
$ npm install bcrypt
```

body-parser: Express body-parser is an npm library used to process data sent through an HTTP request body. It exposes four express middlewares for parsing text, JSON, url-encoded and raw data set through an HTTP request body. These middlewares are functions that process incoming requests before they reach the target controller.

```
$ npm install body-parser
```

dotenv: The dotenv is a zero-dependency module that loads environment variables from a .env file into process.env. Storing configuration in the environment separate from code is based on the Twelve-Factor App methodology.

```
$ npm install dotenv
```

mongodb: it allows the MongoDB Node.js driver to be installed, making it easier for developers to work with MongoDB from inside a Node.js application.

```
$ npm install mongodb
```

mongoose: Mongoose is a JavaScript object-oriented programming library that creates a connection between MongoDB and the Express web application framework.

```
$ npm install mongoose
```

multer: Multer is a node.js middleware for handling multipart/form-data, which is primarily used for uploading files. It is written on top of busboy for maximum efficiency.

```
$ npm install multer
```

multer-s3: Streaming multer storage engine for AWS S3.

```
$ npm install multer-s3
```

pug: Pug in node.js is a template engine that uses case sensitive syntax to generate html, in other words it returns a string of html rendered as per data specified in a pug file. We can say that pug is the middleman who plays a role to convert the injected data and translate it into html syntax.

```
$ npm install pug
```


socket.io: Socket.IO is an event-driven library for real-time web applications. It enables real-time, bi-directional communication between web clients and servers. It has two parts: a client-side library that runs in the browser, and a server-side library for Node.js. Both components have a nearly identical API.

```
$ npm install socket.io
```

uuid: A UUID (Universal Unique Identifier) is a 128-bit value used to uniquely identify an object or entity on the internet. Depending on the specific mechanisms used, a UUID is either guaranteed to be different or is, at least, extremely likely to be different from any other UUID generated.

```
$ npm install uuid
```

CHAPTER 5: Conclusion and Future Scope

To conclude we created a simple clone of the famous social media app twitter that consists of the all the basic features which makes this app ready to use by any user. The development process was not less than any adventure journey we're on through out these couple of weeks. We hope that this app will be liked by others. We'll keep adding new features to the app for making it more attractive and user friendly. We'll be continuously updating this app. Our focus will be on adding the essential features first like:

Allowing users to send photos or other media files to other users.

Making users to be able to edit their profile.

Adding email confirmation mechanism.

Integrating password recovery system.

Making the chats end-to-end encrypted.

Structrizing mongodb database for ease of handing large set of data.

and much more.....