SOCIAL NETWORKING APPLICATION

TEAM DETAILS

[1] Venkata sai meghana nissenkara, UI design/Frontend

[2] Navya Sri Muppera, UX design/Frontend

[3] Sahithya nimmala, Backend Developer

[4] Anjani Racherla, Backend Developer

The story and its details:

These days, everybody utilizes online media applications to interface with others, communicate with them, and offer their substance. Online media, then again, are intelligent innovations that empower the creation, sharing, and trade of data, thoughts, interests, and different sorts of articulation through virtual networks and organizations. It is profoundly huge in everybody's life since correspondence is currently a lot more straightforward because of online media. Be that as it may, while the advantages are various, there are additional downsides, for example, undermining user’s security since client content or individual information is currently accessible on the web. In our venture, we present a web-based media application where individuals can share their substance, associate with others by adding new companions, and present new highlights, for example, sound posts and erase a visit, which eliminates the talk on the two finishes, giving clients more protection and control. For instance, assuming an incidental message is shipped off a senior or notable individual, the person can keep away from false impressions by just erasing the message being referred to. Furthermore the primary designated crowd or clients are the people who need more protection for their profile and are searching for new cool highlights like sound postings and erase choices in a visit with more prominent client control. We likewise utilized MERN technology to finish every one of the errands in our proposed application. Our application is more helpful for clients who require more security, like representatives, finance managers, etc.

Features in our application:

- **Simple Register into the application:** Users can undoubtedly enlist in our application to acquire admittance to all accessible highlights in our application by giving their email address, secret key, and other individual data. Clients would then be able to sign in utilizing their email address and secret key**.**
- **Empower clients to deal with their profile:** Allow clients to deal with their own profiles, like their name, secret phrase, shared posts, and profile picture, in addition to other things.
- **Post content:** Users can make or upload posts on their profiles to impart content to their companions. This substance can be a picture.
- **Delete post:** Users have the ability to delete any posted content from their profile. They can do so by selecting the remove post option from the create post feature.
- **Add new Friends:** Users can check out others' profiles and add them as friends by sending a friend request, which is acknowledged whether the other individual agrees to be friends. Users will see their profile photograph, name, and ID, just as the choice to actually look at their profile.
- **Chat with friends:** Users can chat/connect with other people on their friend's list by going to the app's chats and exchanging messages and starting conversations.
- **Check Friends list:** Users can view their friend's list by selecting the option friends, which displays all of the user's friends.
- **Check notifications:** Our application has a feature for users to check notifications to see if there are any new messages or to see the status of pending requests as well as the requests they have received and check likes and comments to the posted content.
- **Home page:** Users can view all of the feeds on the home page and like or comment on their friends' content.

Requirements:

Hardware Requirements

- Processor : Any processor with 2 core 3 threads or above
- Speed : 2.9 GHz
- RAM : 6 GB RAM
- Hard Disk : 40GB

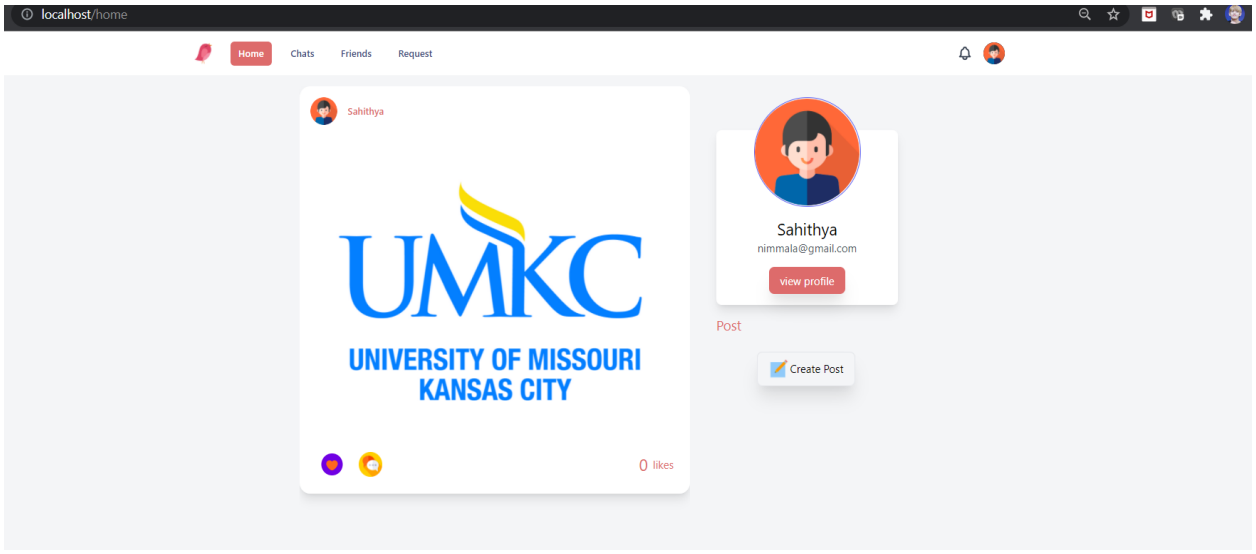
Software Requirements

- Operating System : Ubuntu 18 Lts
- Browser : Chrome/Firefox
- Front End : tailwindCSS, React js
- Back End : Express js, Node.js, Mongoose, docker
- Database : Mongo DB

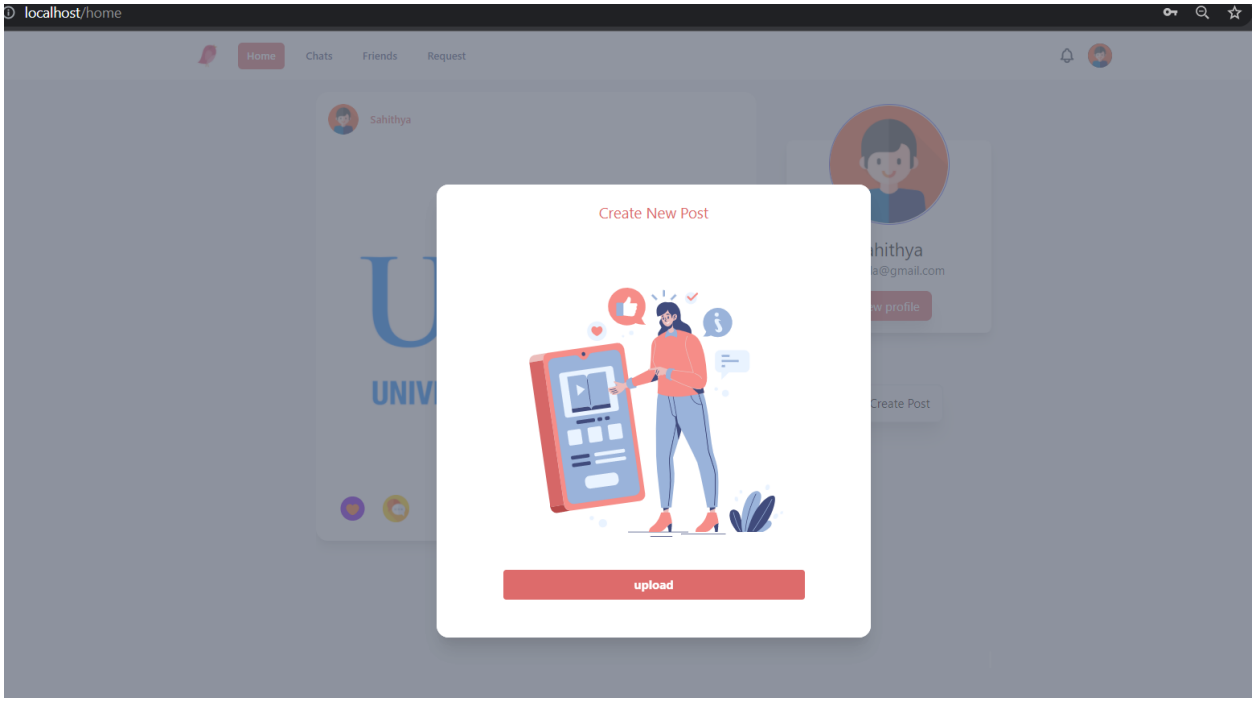
Technologies used:

Our application is developed using MERN technologies. After the four major technologies that make up the stack, MERN stands for MongoDB, Express, React, and Node. MongoDB is a document-oriented database. Node.js web framework Express(.js). React(.js) is a JavaScript framework for the client. The most popular JavaScript web server is Node(.js).and we used tailwind, a CSS framework that focuses on utility for quickly creating custom user interfaces. It's a low-level CSS framework with a lot of customization options. and Axios a Javascript library for making HTTP requests from the browser using node.js or XMLHttpRequests to make API requests

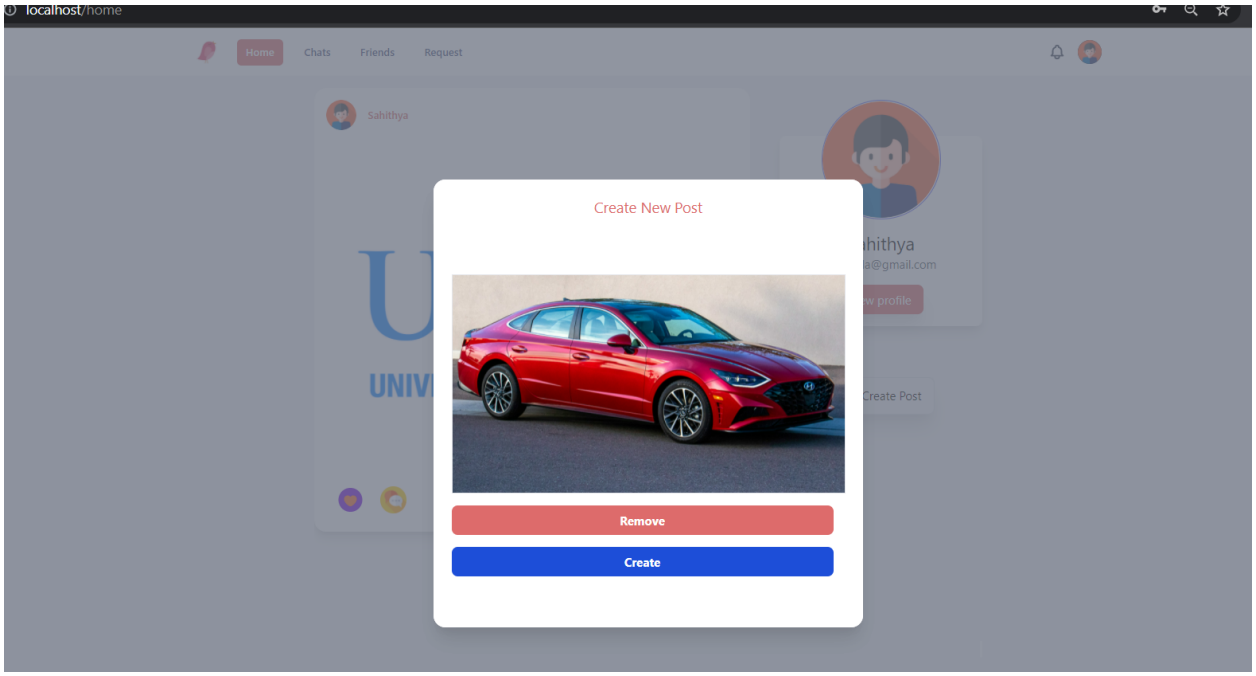
Working screens:



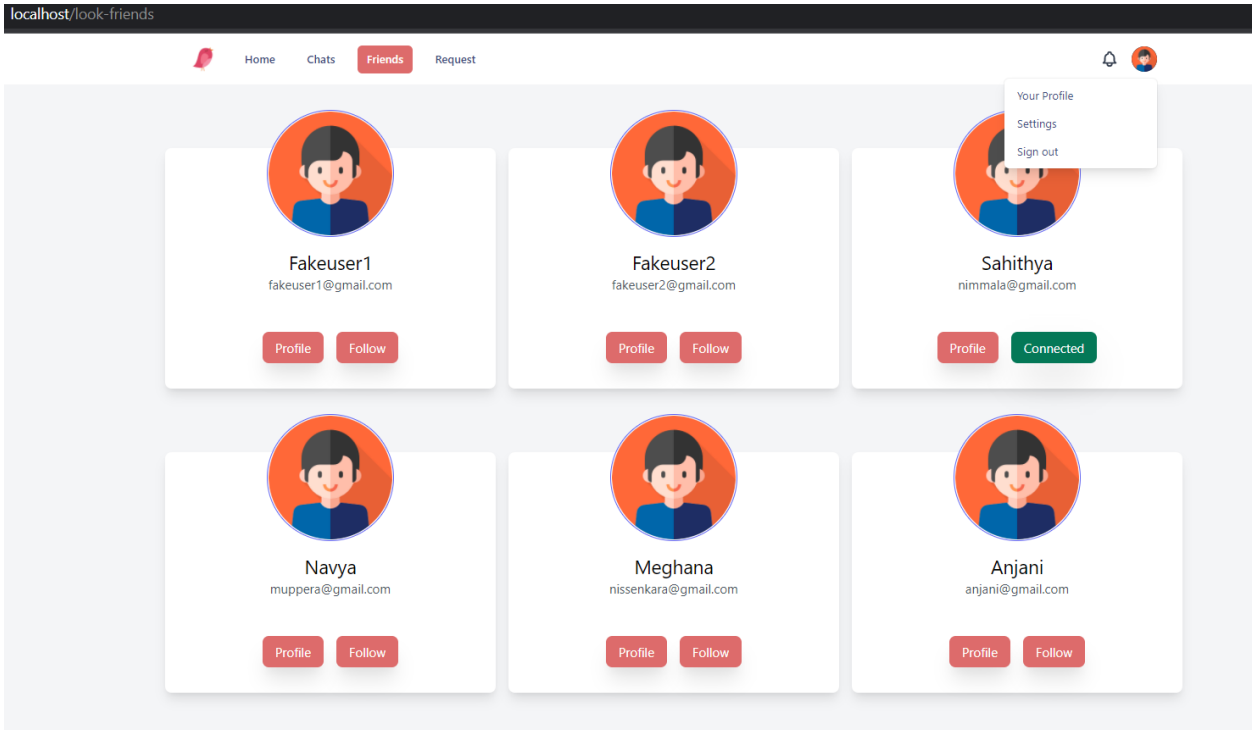
Home Page



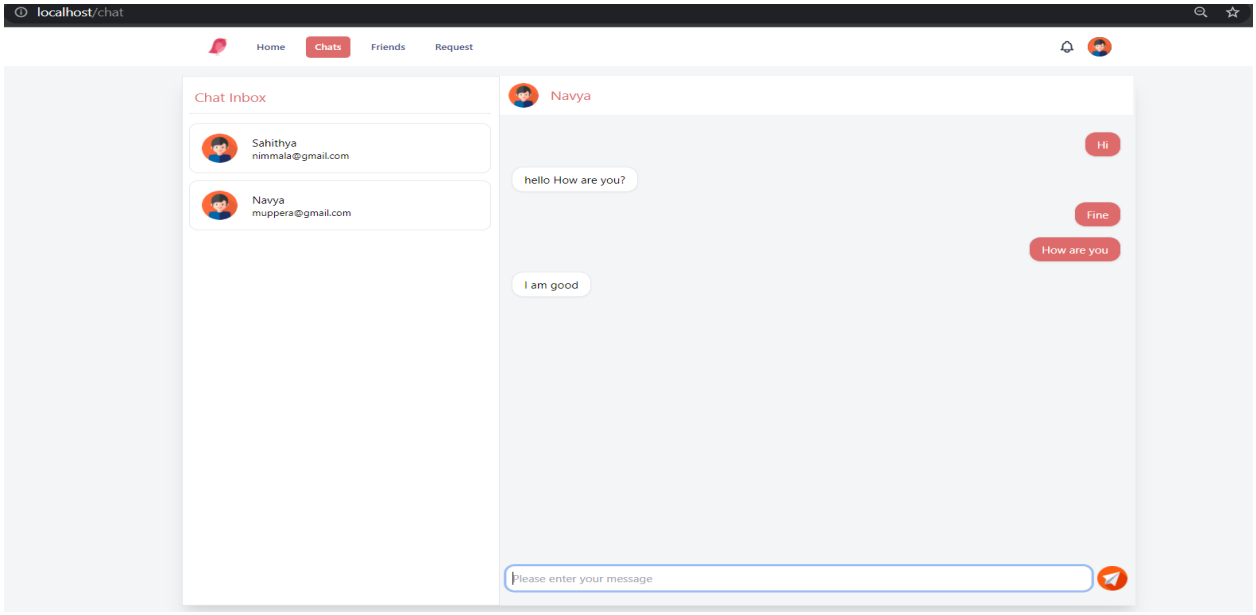
Create New Post



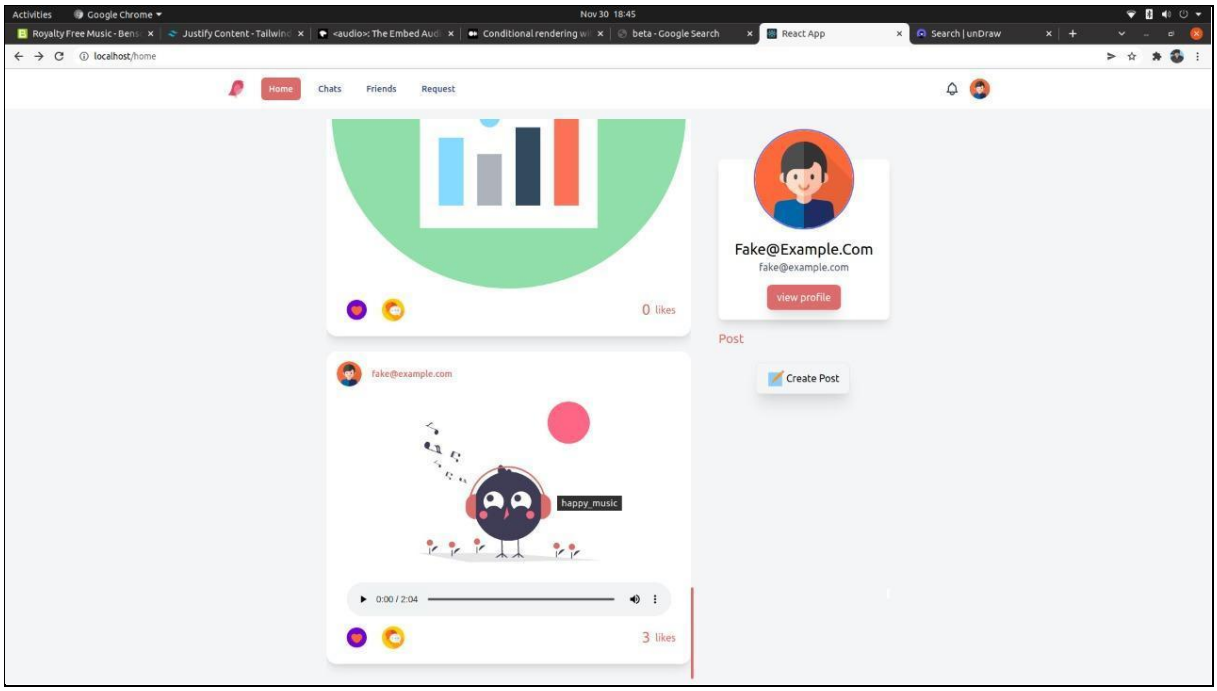
Remove Post



Friends list



Chats



Voice Posts

Improvement from the previous increment:

We completed tasks such as enabling a chat function for users to engage, creating posts such as posting photographs, removing posts, and adding friends by sending friend requests in the previous increment. We provide complete privacy to our users. In the current increment, we added more new features in this increment so that users may get more out of our application. Posting voice postings and deleting chat with more control are two of the new features. If a user deletes a chat, it is deleted on both sides.

Additional Features:

- **Voice Posts:** Users are given a new feature to their application that allows them to publish audio if they want to share anything relevant to the audio file with their friends.
- **Delete Chat:** Users can now delete their chat with more control, and it will be deleted on both ends if they do so.

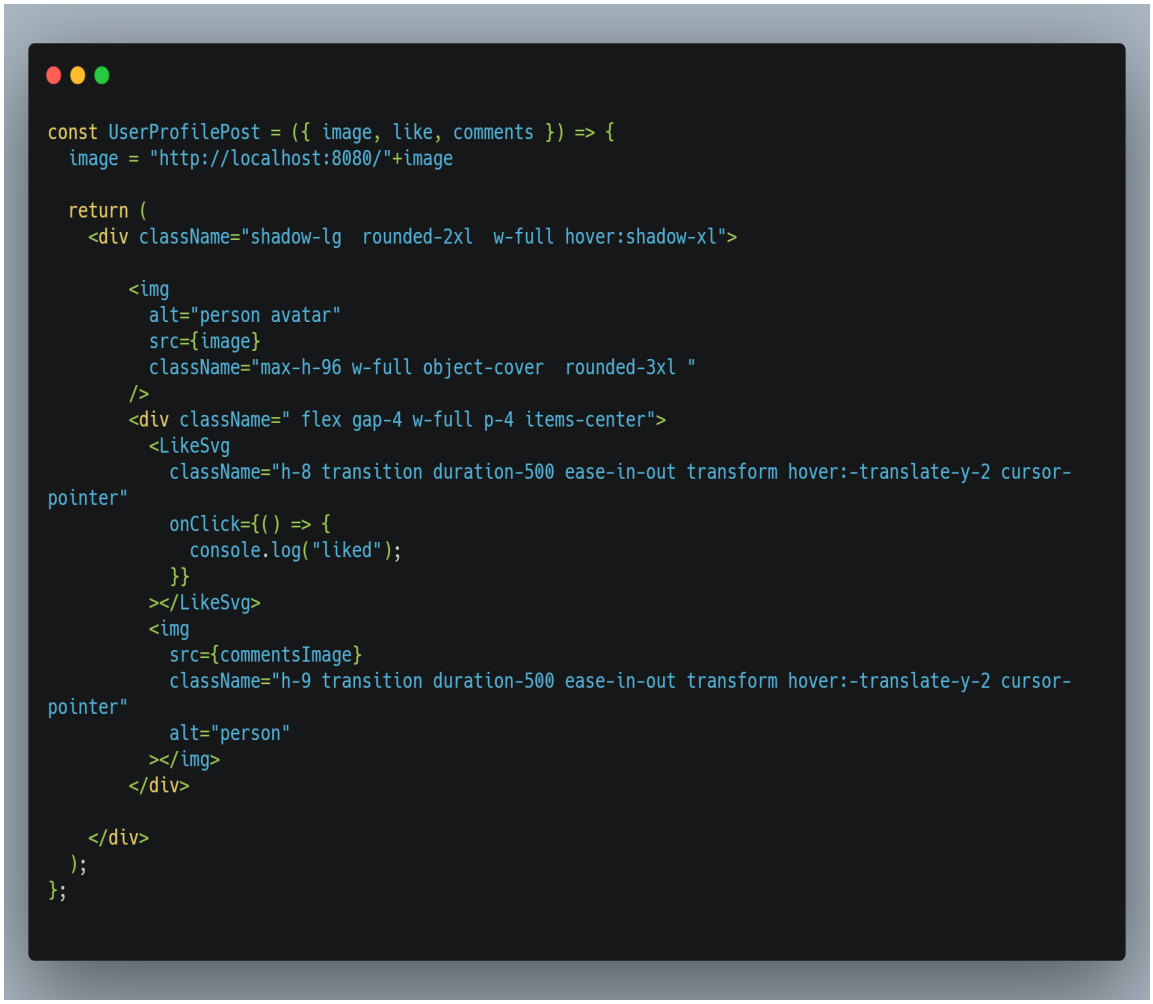
Code Snippets:

```
const UserCard = ({ email, name, image }) => {


  return (
    <div className="w-full border hover:bg-gray-50 shadow-lg flex  items-center bg-white p-4 rounded-
xl">
      <div className="">
      <img className="w-12 h-12 rounded-full " src={image} alt="" width="100" height="100"/>
      </div>
      <div className="flex flex-col ml-5">
        <p className="bold text-md">{name}</p>
        <p className="bold text-sm">{email}</p>
      </div>
    </div>
  );
};

export default UserCard;
```

User Card



```
const UserProfilePost = ({ image, like, comments }) => {
  image = "http://localhost:8080/"+image

  return (
    <div className="shadow-lg  rounded-2xl  w-full hover:shadow-xl">

        <img
          alt="person avatar"
          src={image}
          className="max-h-96 w-full object-cover  rounded-3xl "
        />
        <div className=" flex gap-4 w-full p-4 items-center">
          <LikeSvg
            className="h-8 transition duration-500 ease-in-out transform hover:-translate-y-2 cursor-
pointer"
            onClick={() => {
              console.log("liked");
            }}
          ></LikeSvg>
          <img
            src={commentsImage}
            className="h-9 transition duration-500 ease-in-out transform hover:-translate-y-2 cursor-
pointer"
            alt="person"
          ></img>
        </div>

    </div>
  );
};
```

User Profile Post

```
const PostCard = ({ image, likes, user, comments, id }) => {
  const current_user = useUserState();
  const [postLikes, setPostLikes] = useState(likes.length);
  const [postComments, setPostComments] = useState(comments);
  const [viewComments, setViewComments] = useState(false);

  if (!user.image) {
    user.image = defaultAvatar;
  }

  image = "http://localhost:8080/" + image;

  function likePost() {
    console.log("clicked");
    axios
      .put("/post", { user: current_user.id, id: id, type: "like" })
      .then((msg) => setPostLikes(postLikes + 1));
  }

  function addComment(e) {
    e.preventDefault();

    axios
      .put("/post", {
        user: current_user.id,
        id: id,
        type: "comment",
        text: e.target.text.value,
      })
      .then((msg) =>
        setPostComments(
          [
            {
              text: e.target.text.value,
              postedBy: { name: current_user.name },
            },
          ].concat(postComments)
        )
      );
  }
```

Post Card

Work sharing /module sharing between teammates:

Our app has three different modules: posts, chat, and friend requests. Every member of our team participates in all tasks, but we divided our project into three modules, with each member focusing on one of the three.

- **Post Module:** The goal in the posts module is to design posts, which includes actions such as creating new posts, removing posts, and more. our teammates were tasked to design this.
- **Chats Module:** The aim in the chats module is to design and create a chat server for interacting with friends and storing their chats in a database that can be accessed at any time.
- **Friends Module:** Our teammates are tasked with designing and developing a friends tab for our application, where users will be able to check all of their friends, including those who have made pending requests.

We are four in our team and divided the above modules based on the role that is assigned to our teammates.

- **Venkata sai meghana nissenka** our front-end developer was assigned to and participated in the design and development of the aforementioned modules Posts, Chats, and Friends in our application.
- **Navya Sri Muppera**, who worked as a front-end developer, was assigned to and participated in the design and development of the aforementioned modules Posts, Chats, and Friends in our application.
- Our project's backend developer, **Sahithya nimmala** was entrusted with finishing backend work for our project, which included the server, adding functionality to the application, databases, and the structure of our application.

- **Anjani Racherla**, who worked as a Backend Developer, was tasked with finishing backend work for our project, which included the server, adding functionality to the application, databases, and the structure of our application.

Any blockages /issues with the project:

Throughout the course of our project, we had many issues/challenges in various modules, such as the Chat module, where we encountered issues with queries and filter rooms, which we overcame. A complex query was presented to us in the post-module, which was later resolved using aggregation.

Github link for our project:

https://github.com/SahithyaNimmala/Rasa/tree/dev

Video link for our project:

https://pro.panopto.com/Panopto/Pages/Viewer.aspx?tid=7c7cc9c1-c625-4c69-bdd7-ade1004c0664&start=0

Presentation link for our project:

https://github.com/SahithyaNimmala/Rasa/blob/dev/Team%20Survivors%20Presentation.pdf

References:

1. *Dickey, Jeff (2014-09-24). Write Modern Web Apps with the MEAN Stack: Mongo, Express, AngularJS, and Node.js. Peachpit Press. ISBN 9780133962376.*
2. *^ "LAMP vs MEAN, Deciding the right stack for your startup". www.linkedin.com. Retrieved 2020-02-16.*
3. *^ "The MEAN Stack: MongoDB, ExpressJS, Angular and Node.js". Tumblr. Apr 30, 2013.*
4. *^ "Mean Stack". LinkedIn.*
5. *^ "The most popular database for modern apps". MongoDB. Retrieved 2020-02-16.*
6. *^ "Express - Node.js web application framework". expressjs.com. Retrieved 2020-02-16.*
7. *^ II, Thomas Hunter (2019-03-28). "Why should I use a Reverse Proxy if Node.js is Production-Ready?". Medium. Retrieved 2020-02-16.*
8. *^ "Features - Server Side Rendering | Next.js". nexts.org. Retrieved 2020-02-16.*

9. ^ *holfener, Frys. "Offshore MEAN Stack Development". Imenso Software. Retrieved 17 October 2019.*
10. ^ *"JavaScript Everywhere and the Three Amigos (WebSphere: Into the wild BLUE yonder!)". 2013-11-14. Archived from the original on 2013-11-14. Retrieved 2020-02-16.*