Day - 11 (July -31)

Some styling and component features adding in the vchat

```
import React, { useState, useContext, useRef, useEffect } from "react";
import Nav from "./Nav";
import { AuthContext } from "../App";
import UploadImage from "../assets/media/upload.png";
import ImageResizer from "./ImageResizer";
import axios from "axios";

const EditProfile = () => {
  const { currentUser } = useContext(AuthContext);
  const [isUserInputClicked, setIsUserInputClicked] = useState(false);
  const [isFirstNameClicked, setIsFirstNameClicked] = useState(false);
  const [isSecondNameClicked, setIsSecondNameClicked] = useState(false);
  const [isBioClicked, setIsBioClicked] = useState(false);
  const [isEmailClicked, setIsEmailClicked] = useState(false);
  const [isPrivateClicked, setIsPrivateClicked] = useState(false);
  const [isImageResizer, setIsImageResizer] = useState(false);
  const [croppedImage, setCroppedImage] = useState(null);

  const userNameRef = useRef(null);
  const bioRef = useRef(null);
  const emailRef = useRef(null);
  const firstNameRef = useRef(null);
  const isPrivateRef = useRef(null);
  const lastNameRef = useRef(null);

  const [editedUserDetails, setEditedUserDetails] = useState({
    avatar: currentUser.avatar,
    bio: currentUser.bio,
    email: currentUser.email,
    first_name: currentUser.first_name,
    is_private: currentUser.is_private,
    last_name: currentUser.last_name,
    username: currentUser.username,
  });

  useEffect(() => {
    setEditedUserDetails({
      avatar: croppedImage || currentUser.avatar,
      bio: bioRef.current ? bioRef.current.value : currentUser.bio,
      email: emailRef.current ? emailRef.current.value : currentUser.email,
      first_name: firstNameRef.current ? firstNameRef.current.value :
currentUser.first_name,
      is_private: isPrivateRef.current ? isPrivateRef.current.value :
currentUser.is_private,
      last_name: lastNameRef.current ? lastNameRef.current.value :
currentUser.last_name,
      username: userNameRef.current ? userNameRef.current.value :
currentUser.username,
    });
  }, [
    croppedImage,
    currentUser.avatar,
    currentUser.bio,
    currentUser.email,
    currentUser.first_name,
    currentUser.is_private,
    currentUser.last_name,
    currentUser.username,
  ]);

  const handleInputChange = (field, value) => {
    setEditedUserDetails((prevDetails) => ({
      ...prevDetails,
      [field]: value,
    }));
  };

  const handleSubmit = () => {
    // Update state with the latest input values
    setEditedUserDetails({
      avatar: croppedImage || currentUser.avatar,
      bio: bioRef.current ? bioRef.current.value : currentUser.bio,
      email: emailRef.current ? emailRef.current.value : currentUser.email,
      first_name: firstNameRef.current ? firstNameRef.current.value :
currentUser.first_name,
      is_private: isPrivateRef.current ? isPrivateRef.current.value :
currentUser.is_private,
      last_name: lastNameRef.current ? lastNameRef.current.value :
currentUser.last_name,
      username: userNameRef.current ? userNameRef.current.value :
currentUser.username,
    });

    console.log(editedUserDetails);
  };

  const handleImageResizer = () => {
    setIsImageResizer((prev) => !prev);
  };

  const isPrivateHandleClick = () => {
    setIsPrivateClicked(true);
  };
  const emailEditableFunc = () => {
    setIsEmailClicked(true);
  };

  const bioEditableFunc = () => {
    setIsBioClicked(true);
  };

  const lastNameEditableFunc = () => {
    setIsSecondNameClicked(true);
  };

  const firstNameEditableFunc = () => {
    setIsFirstNameClicked(true);
  };
  const userNameEditableFunc = () => {
    setIsUserInputClicked(true);
  };
  return (
    <div>
      {isImageResizer ? (
        <ImageResizer
          setCroppedImage={setCroppedImage}
          setIsImageResizer={setIsImageResizer}
        />
      ) : (
        <div className="grid grid-cols-12">
          <div className="col-span-1">
            <Nav avatar={currentUser.avatar} />
          </div>
          <div className="col-span-4 mt-[140px] ml-[70px]">
            <div className="border-4 border-blue-600 p-4 rounded-full ">
              <div className="relative">
                <img
                  src={croppedImage ? croppedImage : currentUser.avatar}
                  alt="user avatar"
                  className="rounded-full"
                  width={350}
                />
                <div
                  onClick={handleImageResizer}
                  className="absolute bottom-0 right-0 mb-2 mr-4 border-4
rounded-full bg-slate-400 border-gray-400"
                >
                  <img width={30} src={UploadImage} alt="upload image icon"
/>
                </div>
              </div>
            </div>
          </div>
          <div className="mt-[150px] col-start-6 col-end-12 ml-[80px]">
            <div className="grid grid-cols-3 mb-4">
              <div>
                <h2 className="text-mono text-lg">Username</h2>
              </div>
              {isUserInputClicked ? (
                <div className="border-solid border-2 border-b-sky-500">
                  <input
                    ref={userNameRef}
                    className=""
                    type="text"
                    defaultValue={currentUser.username}
                    onChange={(e) => handleInputChange('username',
e.target.value)}
                  />
                </div>
              ) : (
                <>
                  <div className="border-solid border-2 border-b-sky-500">
                    <h2 className="text-mono text-lg">
                      {currentUser.username}
                    </h2>
                  </div>
                  <div className="flex justify-center ml-12">
                    <button onClick={userNameEditableFunc}>Edit</button>
                  </div>
                </>
              )}
            </div>
            <div className="grid grid-cols-3 mb-4">
              <div>
                <h2 className="text-mono text-lg">First Name</h2>
              </div>
              {isFirstNameClicked ? (
                <div className="border-solid border-2 border-b-sky-500">
                  <input
                    ref={firstNameRef}
                    className=""
                    type="text"
                    defaultValue={currentUser.first_name}
                    onChange={(e) => handleInputChange('first_name',
e.target.value)}
                  />
                </div>
              ) : (
                <>
                  <div className="flex space-x-12 border-solid border-2
border-b-sky-500">
                    <h2 className="text-mono text-lg">
                      {currentUser.first_name}
                    </h2>
                  </div>
                  <div className="flex justify-center ml-12">
                    <button onClick={firstNameEditableFunc}>Edit</button>
                  </div>
                </>
              )}
            </div>
            <div className="grid grid-cols-3 mb-4">
              <div>
                <h2 className="text-mono text-lg">Last Name</h2>
              </div>
              {isSecondNameClicked ? (
                <div className="border-solid border-2 border-b-sky-500">
                  <input
                    ref={lastNameRef}
                    className=""
                    type="text"
                    defaultValue={currentUser.last_name}
                    onChange={(e) => handleInputChange('last_name',
e.target.value)}
                  />
                </div>
              ) : (
                <>
                  <div className="flex space-x-12 border-solid border-2
border-b-sky-500">
                    <h2 className="text-mono text-lg">
                      {currentUser.last_name}
                    </h2>
                  </div>
                  <div className="flex justify-center ml-12">
                    <button onClick={lastNameEditableFunc}>Edit</button>
                  </div>
                </>
              )}
            </div>
            <div className="grid grid-cols-3 mb-4">
              <div>
                <h2 className="text-mono text-lg">Bio</h2>
              </div>
              {isBioClicked ? (
                <div className="border-solid border-2 border-b-sky-500">
                  <input
                    ref={bioRef}
                    className=""
                    type="text"
                    defaultValue={currentUser.bio}
                    onChange={(e) => handleInputChange('bio',
e.target.value)}
                  />
                </div>
              ) : (
                <>
                  <div className="flex space-x-12 border-solid border-2
border-b-sky-500">
                    <h2 className="text-mono text-lg ">{currentUser.bio}
</h2>
                  </div>
                  <div className="flex justify-center ml-12">
                    <button onClick={bioEditableFunc}>Edit</button>
                  </div>
                </>
              )}
            </div>
            <div className="grid grid-cols-3 mb-4">
              <div>
                <h2 className="text-mono text-lg">Email</h2>
              </div>
              {isEmailClicked ? (
                <div className="border-solid border-2 border-b-sky-500">
                  <input
                    ref={emailRef}
                    className=""
                    type="text"
                    defaultValue={currentUser.email}
                    onChange={(e) => handleInputChange('email',
e.target.value)}
                  />
                </div>
              ) : (
                <>
                  <div className="flex space-x-12  border-solid border-2
border-b-sky-500">
                    <h2 className="text-mono text-lg">{currentUser.email}
</h2>
                  </div>
                  <div className="flex justify-center ml-12">
                    <button onClick={emailEditableFunc}>Edit</button>
                  </div>
                </>
              )}
            </div>
            <div className="grid grid-cols-3 mb-4">
              <div>
                <h2 className="text-mono text-lg">Is Private</h2>
              </div>
              {isPrivateClicked ? (
                <div className="border-solid border-2 border-b-sky-500">
                  <input
                    ref={isPrivateRef}
                    className="border-none"
                    type="text"
                    defaultValue={currentUser.is_private}
                    onChange={(e) => handleInputChange('is_private',
e.target.value)}
                  />
                </div>
              ) : (
                <>
                  <div className="flex space-x-12 border-solid border-2
border-b-sky-500">
                    <h2 className="text-mono text-lg ">
                      {currentUser.is_private ? 'Yes' : 'No'}
                    </h2>
                  </div>
                  <div className="flex justify-center ml-12">
                    <button onClick={isPrivateHandleClick}>Edit</button>
                  </div>
                </>
              )}
            </div>
            <div className=" pt-8 flex justify-center">
              <button onClick={handleSubmit} className="bg-blue-500 rounded-
full px-4 text-white text-xl">
                Submit
              </button>
            </div>
          </div>
        </div>
      )}
    </div>
  );
};

export default EditProfile;
```

ImageResizer

```
import React, { useRef, useState } from "react";
import Cropper from "react-cropper";
import "cropperjs/dist/cropper.css";

const ImageResizer = ({ setCroppedImage, setIsImageResizer }) => {
  const [image, setImage] = useState(null);
  const [cropData, setCropData] = useState("#");
  const cropperRef = useRef(null);

  const onImageChange = (e) => {
    const file = e.target.files[0];
    if (file) {
      const reader = new FileReader();
      reader.onloadend = () => {
        setImage(reader.result);
      };
      reader.readAsDataURL(file);
    }
  };

  const handleSelectImage = () => {

setCroppedImage(cropperRef.current.cropper.getCroppedCanvas().toDataURL());
    setIsImageResizer(false);
  };
  const getCropData = () => {
    if (typeof cropperRef.current?.cropper !== "undefined") {

setCropData(cropperRef.current.cropper.getCroppedCanvas().toDataURL());
    }
  };

  return (
    <div className="flex flex-col items-center p-4 bg-gray-100 min-h-
screen">
      <div className="mb-4 p-4 border border-gray-300 bg-white shadow-md
rounded-md">
        <input
          className="block w-full text-sm text-gray-500 file:mr-4 file:py-2
file:px-4 file:rounded-full file:border-0 file:text-sm file:font-semibold
file:bg-blue-50 file:text-blue-700 hover:file:bg-blue-100"
          type="file"
          accept="image/*"
          onChange={onImageChange}
        />
      </div>
      {image && (
        <div className="w-full max-w-lg bg-white p-4 rounded-md shadow-md">
          <div className="mb-4">
            <Cropper
              ref={cropperRef}
              src={image}
              style={{ height: 400, width: "100%" }}
              aspectRatio={1}
              guides={false}
              viewMode={1}
              scalable={true}
              cropBoxMovable={false}
              cropBoxResizable={false}
              dragMode="move"
            />
          </div>
          <button
            onClick={getCropData}
            className="w-full py-2 px-4 bg-blue-500 text-white font-semibold
rounded-md shadow-md hover:bg-blue-600"
          >
            Get Crop Data
          </button>
          <div className="mt-4">
            <h2 className="text-lg text-center font-semibold mb-2">
              Cropped Image
            </h2>
            <div className="flex justify-center">
              <img
                src={cropData}
                alt="cropped"
                className="max-w-full  rounded-full border border-gray-300
shadow-sm"
              />
            </div>
            <div className=" mt-4">
              <button
                className="w-full py-2 px-4 bg-blue-500 text-white font-
semibold rounded-md shadow-md hover:bg-blue-600"
                onClick={handleSelectImage}
              >
                Select This Image
              </button>
            </div>
          </div>
        </div>
      )}****
    </div>
  );
};

export default ImageResizer;
```