PROJECT REPORT

ON

ExploreIT

<u>SUBMITTED BY:</u>

(Vikas Singh	201500785)
(Amit Kumar Yadav	201500081)
(Anshul Chaudhary	201500115)
(Hritik Kuntal	201500576)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GLA

UNIVERSITY,

MATHURA

DECLARATION

We would like to express our special thanks to our project guide **Dr. Manoj Varshney** who gave us the golden opportunity to do this wonderful project called, **ExploreIT** , which also helped us in doing a lot of research and we came to know about so many new things. We are really thankful to them.

Secondly, we would also like to thank our parents and friends who helped us a lot in finalizing this project within the limited time frame.

Candidate's Names:

Amit Kumar Yadav	**201500081**
Vikas Singh	**201500785**
Ritik Kuntal	**201500576**
Anshul Chaudhary	**201500115**

GLA UNIVERSITY
MATHURA
Recognised by UGC Under Section 2(f)
Accredited with A Grade by NAAC

CERTIFICATE

This is to certify that the above statements made by the candidates are correct to the best of my/our knowledge and belief.

Project

Supervisor

Dr.Manoj Varshney

Technical Trainer

Date : 27-5-2023

Table of Contents

INTRODUCTION

ExploreIT refers to a group of online platforms and tools that enable users to create, share or exchange information, ideas, pictures, and videos in virtual communities and networks. ExploreIT allows people to connect with others from all over the world, share their thoughts and experiences, and engage in conversations and discussions. Popular social media platforms include Facebook, Twitter, Instagram, LinkedIn, TikTok, and YouTube, among others. Each platform has its own unique features and audience, but they all serve the purpose of connecting people and fostering communication and collaboration. In recent years, social media has become a ubiquitous part of modern life, with billions of users worldwide. Social media has transformed the way people communicate and interact with one another, and it has had a significant impact on many aspects of society, including politics, marketing, and social activism. However, it has also raised concerns about privacy, mental health, and the spread of misinformation.

WEB DEVELOPEMENT:

Web development refers to the process of creating websites on the Internet.

The term “Web Development” is relatively broad in its application. You could create a single website page from a wix template, or you could painstakingly develop a massive website with thousands of original pages — and technically, both of those would count as web development.

Nowadays web development is the thing which is in boost.

TECHNOLOGY USED:

HTML:

HTML is an acronym which stands for **Hyper Text Markup Language** which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page.

HyperText: HyperText simply means "Text within Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext.

HyperText is a way to link two or more web pages (HTML documents) with each other.

Markup language: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc.

Web Page: A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. **With the help of HTML only, we can create static web pages**.

CSS:

CSS stands for Cascading Style Sheets. It is a style sheet language which is used to describe the look and formatting of a document written in markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces. It can also be used with any kind of XML documents including plain XML, SVG and XUL.

CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications and user interfaces for many mobile applications.

MongoDB:

MongoDB is a popular NoSQL database management system that uses a document-oriented data model. Instead of using traditional tables and rows, MongoDB stores data as JSON-like documents, making it a good fit for handling unstructured or semi-structured data. It offers features such as dynamic schema, horizontal scaling, high availability, and automatic sharding, which make it a scalable and flexible database solution. MongoDB also provides various tools and libraries for interacting with the database, including a command-line interface, GUI tools, and drivers for multiple programming languages.

Express:

Express is a popular web framework for Node.js that simplifies the process of building web applications and APIs. It provides a set of features for handling HTTP requests and responses, routing, middleware, and templating engines. Express allows developers to write server-side JavaScript code that can respond to client requests with data, HTML, or other resources. It also offers a flexible and extensible architecture that allows developers to add third-party modules and plugins to enhance the functionality of their applications. Express is widely used in the Node.js community and has a large ecosystem of tools and libraries that make it easy to build complex web applications quickly.

React :

React is based on a component-based architecture where each UI element is a separate component. These components can be easily composed together to create complex UIs. React also uses a virtual DOM (Document Object Model) which allows for efficient updates to the UI, minimizing the number of actual DOM manipulations needed.React has gained popularity due to its performance, scalability, and ease of use. It also has a large and active community of developers who contribute to its development and provide support through various forums and channels.React can be used with other libraries and frameworks such as Redux, Next.js, and

JavaScript :

JavaScript is a high-level, object-oriented programming language that is primarily used for creating interactive and dynamic web content. It was created by Netscape in 1995 and is now supported by all modern web browsers. JavaScript can be used for a wide range of tasks, including web development, server-side programming, and mobile application development. It is often used in conjunction with HTML and CSS to create interactive web pages. JavaScript is a loosely typed language, meaning that variables do not need to be declared with a specific data type. It is also an interpreted language, meaning that the code is executed at runtime rather than being compiled beforehand.

SYSTEM REQUIREMENTS

Software Requirement-

To build application –

- 64-bit Windows 8/10/11
- Libraries
- Visual Studio code (latest version).

To Run Website –

- Web Browsers (chrome, Edge,Brave)

Hardware Requirement –

- x86_64 CPU architecture;
- 2nd generation Intel Core or newer
- 4 GB RAM or more
- 8 GB of available disk space minimum

IMPLEMENTATION

Final Code:

- **Login Page:**

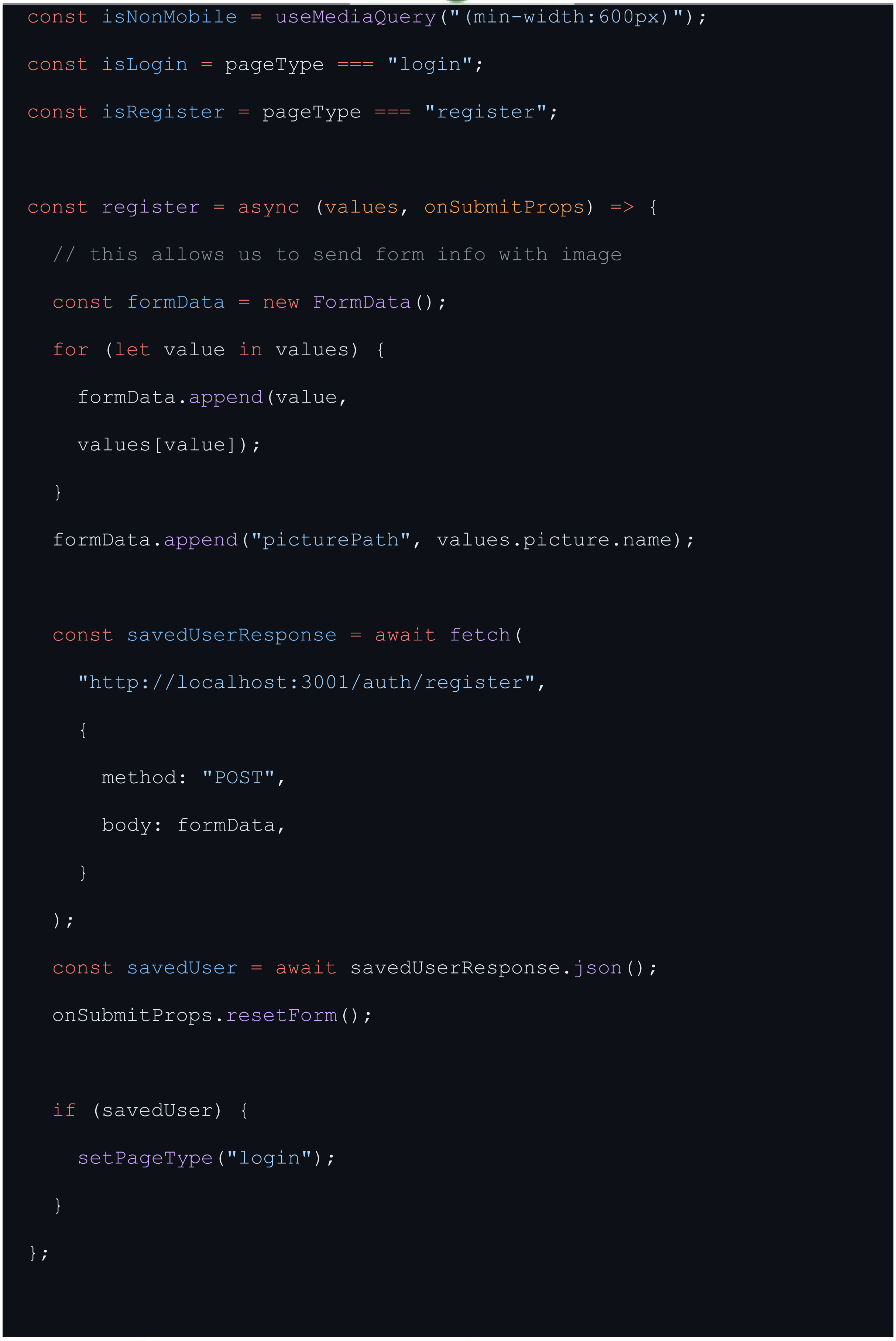
```
import { Box, Typography, useTheme, useMediaQuery } from
"@mui/material";

import bg from './bg.jpg'

import Form from "./Form";


const LoginPage = () => {

  const theme = useTheme();

const isNonMobileScreens = useMediaQuery("(min-width: 1000px)");

  return (

<Box>

      <Box

  sx={{

backgroundImage: `url(${bg})`,

    backgroundPosition: 'center',

    backgroundSize: 'cover',

height: '150px', // or any other height you want
```

```
    display: 'flex',

    justifyContent: 'center',

    alignItems: 'center',

  }}

>

  <Typography fontWeight="bold" fontSize="150px" color=""
fontFamily="Lucida, courier new, monospace">

    Hemut

  </Typography>

</Box>



      <Box

        width={isNonMobileScreens ? "50%" : "93%"}

        p="2rem"

        m="2rem auto"

        borderRadius="1.5rem

        " sx={{

          backgroundImage:'url(${bg})'

        }}



      >

        <Typography fontWeight="500" variant="h5" sx={{ mb: "1.5rem"
}}>

          Welcome to Hemut, Social Media for art Enthusiasts

        </Typography>

        <Form />

      </Box>

    </Box>

  );

};
```

```
export default LoginPage;
```

- **Login Page(Form):**

```
import { useState } from "react";

import {

Box, Button,

  TextField,

useMediaQuery,

  Typography,

  useTheme,

} from "@mui/material";

import EditOutlinedIcon from "@mui/icons-material/EditOutlined";

import { Formik } from "formik";

import * as yup from "yup";

import { useNavigate } from "react-router-dom";

import { useDispatch } from "react-redux";

import { setLogin } from "state";

import Dropzone from "react-dropzone";

import FlexBetween from "components/FlexBetween";


const registerSchema = yup.object().shape({

  firstName: yup.string().required("required"),

  lastName: yup.string().required("required"),

email: yup.string().email("invalid email").required("required"),

  password: yup.string().required("required"),

location: yup.string().required("required"),
  occupation: yup.string().required("required"),

  picture: yup.string().required("required"),

});


const loginSchema = yup.object().shape({

  email: yup.string().email("invalid email").required("required"),

  password: yup.string().required("required"),

});


const initialValuesRegister =

  { firstName: "",

  lastName: "",

  email: "",

  password: "",

  location: "",

  occupation: "",

  picture: "",

};


const initialValuesLogin = {

  email: "",

  password: "",

};


const Form = () => {

  const [pageType, setPageType] = useState("login");

  const { palette } = useTheme();

  const dispatch = useDispatch();

  const navigate = useNavigate();
```

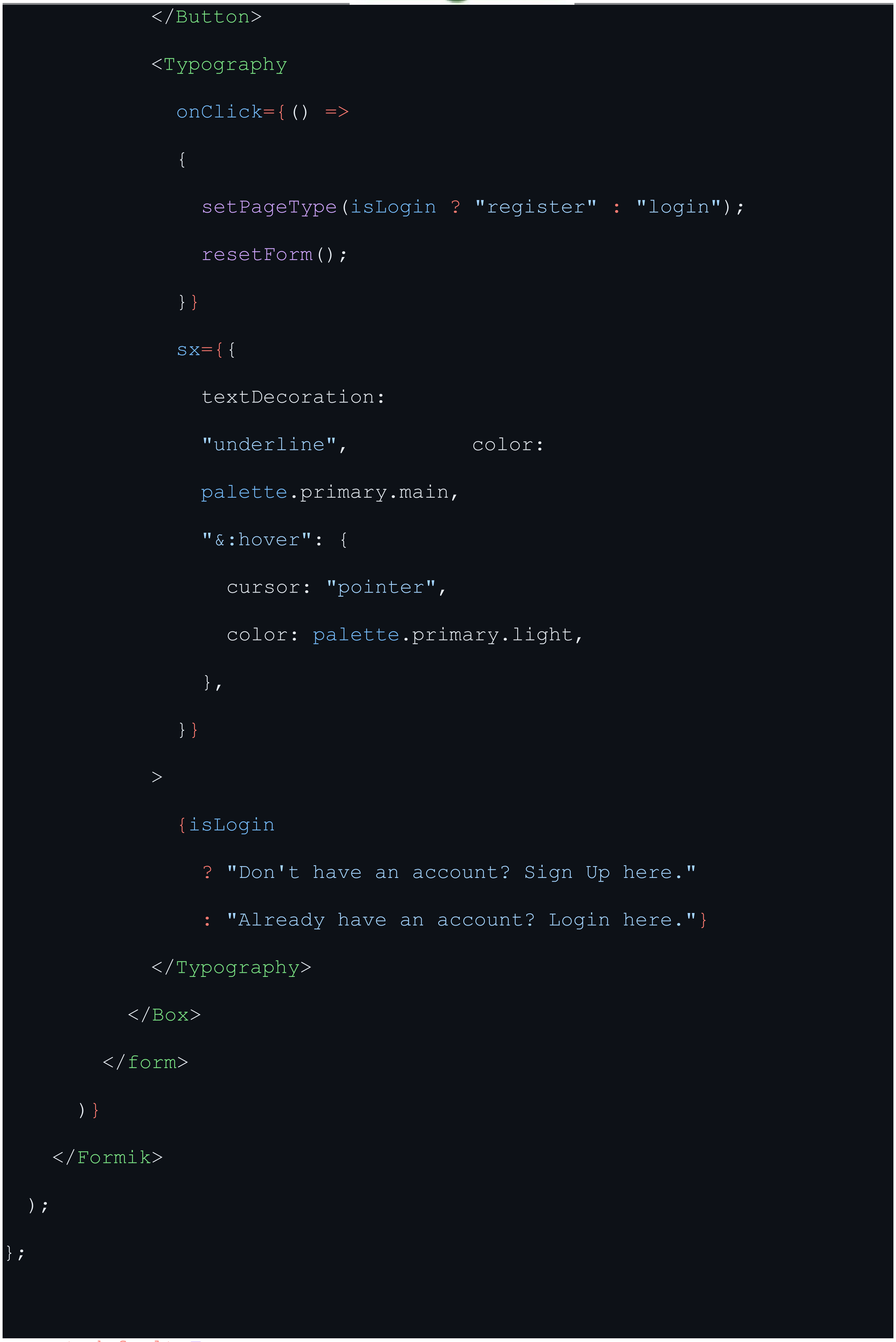


```
const isNonMobile = useMediaQuery("(min-width:600px)");

const isLogin = pageType === "login";

const isRegister = pageType === "register";


const register = async (values, onSubmitProps) => {

  // this allows us to send form info with image

  const formData = new FormData();

  for (let value in values) {

    formData.append(value,

    values[value]);

  }

  formData.append("picturePath", values.picture.name);


  const savedUserResponse = await fetch(

    "http://localhost:3001/auth/register",

    {

      method: "POST",

      body: formData,

    }

  );

  const savedUser = await savedUserResponse.json();

  onSubmitProps.resetForm();


  if (savedUser) {

    setPageType("login");

  }

};


const login = async (values, onSubmitProps) => {
    const loggedInResponse = await
fetch("http://localhost:3001/auth/login",
{

      method: "POST",

      headers: { "Content-Type": "application/json" },

      body: JSON.stringify(values),

    });

    const loggedIn = await loggedInResponse.json();

    onSubmitProps.resetForm();

    if (loggedIn) {

      dispatch(

        setLogin({

          user: loggedIn.user,

          token:

          loggedIn.token,

        })

      );

      navigate("/home");

    }

  };


  const handleFormSubmit = async (values, onSubmitProps) => {

    if (isLogin) await login(values, onSubmitProps);

    if (isRegister) await register(values, onSubmitProps);

  };


  return (

    <Formik

      onSubmit={handleFormSubmit}

      initialValues={isLogin ? initialValuesLogin :
initialValuesRegister}

      validationSchema={isLogin ? loginSchema : registerSchema}
    >
      {({
        values,
        errors,
        touched,
        handleBlur,
        handleChange,
        handleSubmit,
        setFieldValue
        , resetForm,
      }) => (
        <form onSubmit={handleSubmit}>
          <Box
            display="grid"
            gap="30px"
            gridTemplateColumns="repeat(4, minmax(0, 1fr))"
            sx={{
              "& > div": { gridColumn: isNonMobile ? undefined : "span
4" },
            }}
          >
            {isRegister && (
              <>
                <TextField
                  label="First Name"
                  onBlur={handleBlur}
                  onChange={handleChange}
                  value={values.firstName}
                  name="firstName"
                  error={
                    Boolean(touched.firstName)
                    &&
Boolean(errors.firstName)
                  }
                  helperText={touched.firstName && errors.firstName}
                  sx={{ gridColumn: "span 2" }}
                />
                <TextField
                  label="Last Name"
                  onBlur={handleBlur
                  }
                  onChange={handleChange
                  }
                  value={values.lastName
                  } name="lastName"
                  error={Boolean(touched.lastName)
&& Boolean(errors.lastName)}
                  helperText={touched.lastName                  &&
                  errors.lastName} sx={{ gridColumn: "span 2" }}
                />
                <TextField
                  label="Location"
                  onBlur={handleBlur}
                  onChange={handleChange}
                  value={values.location}
                  name="location"
                  error={Boolean(touched.location) &&
Boolean(errors.location)}
                  helperText={touched.location && errors.location}
                  sx={{ gridColumn: "span 4" }}
                />
```

```
<TextField
```

```
          label="Occupation"
          onBlur={handleBlur}
          onChange={handleChange}
          value={values.occupation}
          name="occupation"
          error={
            Boolean(touched.occupation) &&
Boolean(errors.occupation)
          }
          helperText={touched.occupation && errors.occupation}
          sx={{ gridColumn: "span 4" }}
        />
        <Box
          gridColumn="span 4"
          border={`1px solid
          ${palette.neutral.medium}`}
          borderRadius="5px"
          p="1rem"
        >
          <Dropzone
            acceptedFiles=".jpg,.jpeg,.png
            " multiple={false}
            onDrop={(acceptedFiles) =>
              setFieldValue("picture", acceptedFiles[0])
            }
          >
            {({ getRootProps, getInputProps }) => (
              <Box
                {...getRootProps()}
                border={`2px dashed ${palette.primary.main}`}
```

```
                        p="1rem"
                        sx={{ "&:hover": { cursor: "pointer" } }}
                      >
                        <input {...getInputProps()} />
                        {!values.picture ? (
                          <p>Add Picture Here</p>
                        ) : (
                          <FlexBetween>

<Typography>{values.picture.name}</Typography>
                            <EditOutlinedIcon />
                          </FlexBetween>
                        )}
                      </Box>
                    )}
                  </Dropzone>
                </Box>
              </>
            )}

            <TextField
              label="Email"
              onBlur={handleBlur}
              onChange={handleChange}
              value={values.email}
              name="email"
              error={Boolean(touched.email) && Boolean(errors.email)}
              helperText={touched.email && errors.email}
              sx={{ gridColumn: "span 4" }}
            />

            <TextField
              label="Password"
              type="password"
              onBlur={handleBlur}
              onChange={handleChange}
              value={values.password}
              name="password"
              error={Boolean(touched.password)
&& Boolean(errors.password)}
              helperText={touched.password && errors.password}
              sx={{ gridColumn: "span 4" }}
            />
          </Box>

          {/* BUTTONS */}
          <Box>
            <Button
              fullWidth
              type="submit"
              sx={{
                m: "2rem 0",
                p: "1rem",
                backgroundColor: palette.primary.main,
                color: palette.background.alt,
                "&:hover": { color: palette.primary.main },
              }}
            >
              {isLogin ? "LOGIN" : "REGISTER"}
```



```
            </Button>
            <Typography
              onClick={() =>
              {
                setPageType(isLogin ? "register" : "login");
                resetForm();
              }}
              sx={{
                textDecoration:
                "underline",          color:
                palette.primary.main,
                "&:hover": {
                  cursor: "pointer",
                  color: palette.primary.light,
                },
              }}
            >
              {isLogin
                ? "Don't have an account? Sign Up here."
                : "Already have an account? Login here."}
            </Typography>
          </Box>
        </form>
      )}
    </Formik>
  );
};
```

```
export default Form;
```

Home Page:

```
import { Box, useMediaQuery } from
"@mui/material"; import { useSelector } from
"react-redux";
import Navbar from "scenes/navbar";
import UserWidget from "scenes/widgets/UserWidget";
import MyPostWidget from "scenes/widgets/MyPostWidget";
import PostsWidget from "scenes/widgets/PostsWidget";
import AdvertWidget from "scenes/widgets/AdvertWidget";
import FriendListWidget from "scenes/widgets/FriendListWidget";

const HomePage = () => {
  const isNonMobileScreens = useMediaQuery("(min-width:1000px)");
  const { _id, picturePath } = useSelector((state) =>
  state.user);

  return (
    <Box>
      <Navbar />
      <Box
        width="100%"
        padding="2rem
        6%"
        display={isNonMobileScreens ? "flex" : "block"}
        gap="0.5rem"
        justifyContent="space-between"
      >
        <Box flexBasis={isNonMobileScreens ? "26%" : undefined}>
          <UserWidget userId={_id} picturePath={picturePath} />
```

```
<Box m="2rem 0" />
```

```
          <FriendListWidget userId={_id} />
        </Box>
        <Box
          flexBasis={isNonMobileScreens ? "42%" : undefined}
          mt={isNonMobileScreens ? undefined : "2rem"}
        >
          <MyPostWidget picturePath={picturePath} />
          <PostsWidget userId={_id} />
        </Box>
        {isNonMobileScreens && (
          <Box flexBasis="26%">
            <AdvertWidget />



          </Box>
        )}
      </Box>
    </Box>
  );
};
```

Nav Bar:

```
import { useState } from "react";
import {
  Box,
  IconButton,
  InputBase,
  Typography,
  Select,
  MenuItem,
  FormControl,
  useTheme,
  useMediaQuery,
} from "@mui/material";
import {
  Search,
  Message,
  DarkMode,
  LightMode,
  Notifications,
  Help,
  Menu,
  Close,
} from "@mui/icons-material";
import { useDispatch, useSelector } from "react-redux";
import { setMode, setLogout } from "state";
import { useNavigate } from "react-router-dom";
import FlexBetween from
"components/FlexBetween";

const Navbar = () => {
  const [isMobileMenuToggled, setIsMobileMenuToggled] =
useState(false);
```

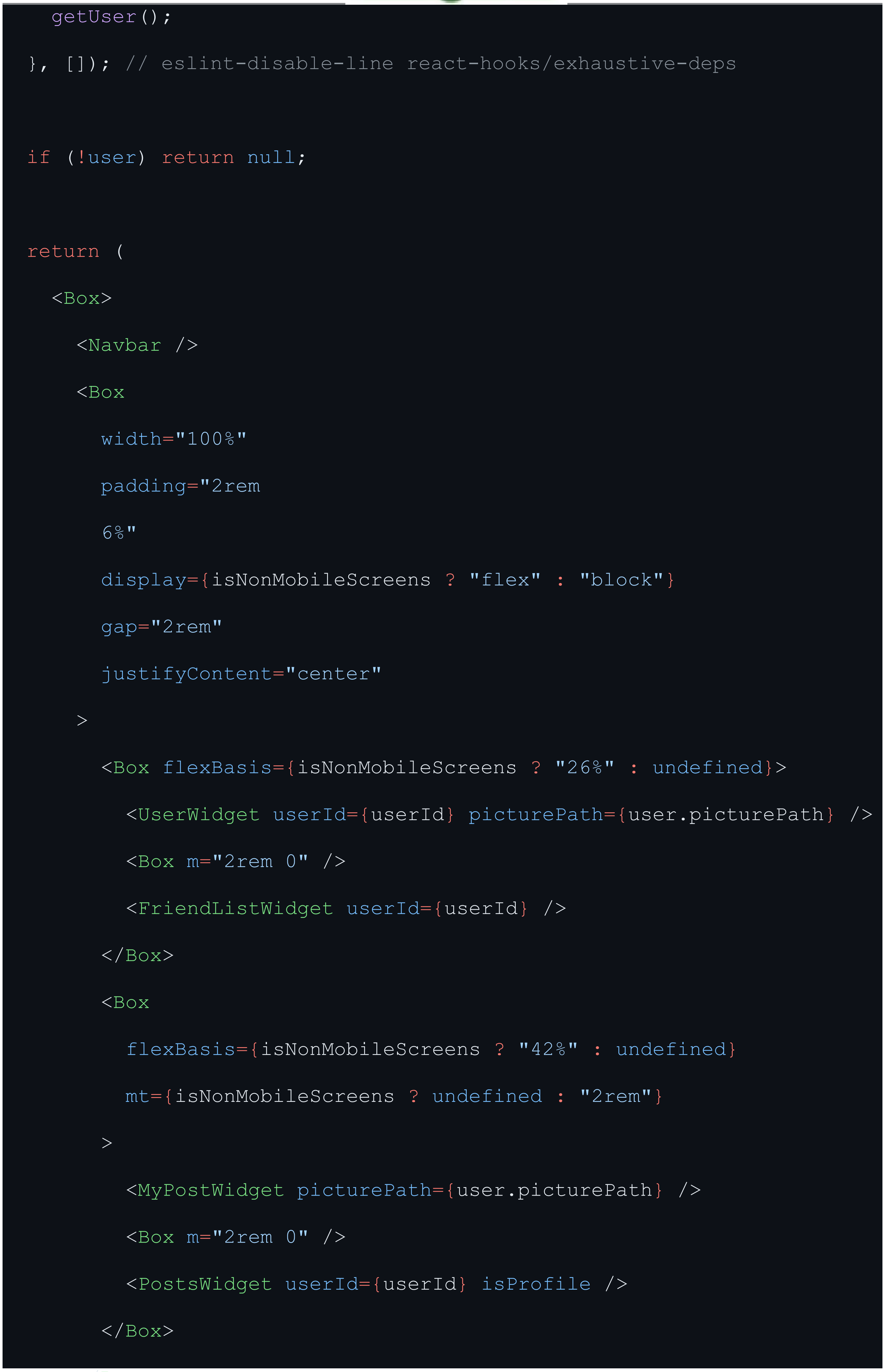
```
const dispatch = useDispatch();

const navigate = useNavigate();

const user = useSelector((state) => state.user);

const isNonMobileScreens = useMediaQuery("(min-width: 1000px)");


const theme = useTheme();

const neutralLight = theme.palette.neutral.light;

const dark = theme.palette.neutral.dark;

const background = theme.palette.background.default;

const primaryLight = theme.palette.primary.light;

const alt = theme.palette.background.alt;


const fullName = `${user.firstName} ${user.lastName}`;


return (

  <FlexBetween padding="1rem 6%" backgroundColor='#1B3D81'>

    <FlexBetween gap="1.75rem">

      <Typography

        fontWeight="bold"

        fontSize="clamp(3rem, 5rem,

        2.5rem)" color=""

        fontfamily="Lucida, courier new, monospace"

        onClick={() => navigate("/home")}

        sx={{

          "&:hover": {

            color: primaryLight,

            cursor: "pointer",

          },

        }}
      >
        Hemut
      </Typography>
      {isNonMobileScreens && (
        <FlexBetween
          backgroundColor={neutralLight}
          borderRadius="9px"
          gap="3rem"
          padding="0.1rem 1.5rem"
        >
          <InputBase placeholder="Search..." />
          <IconButton>
            <Search />
          </IconButton>
        </FlexBetween>
      )}
    </FlexBetween>

    {/* DESKTOP NAV */}
    {isNonMobileScreens ? (
      <FlexBetween gap="2rem">
        <IconButton onClick={() => dispatch(setMode())}>
          {theme.palette.mode === "dark" ? (
            <DarkMode sx={{ fontSize: "25px" }} />
          ) : (
            <LightMode sx={{ color: dark, fontSize: "25px" }} />
          )}
        </IconButton>
        <Message sx={{ fontSize: "25px" }} />
```

```
          <Notifications sx={{ fontSize: "25px" }} />

          <Help sx={{ fontSize: "25px" }} />

          <FormControl variant="standard" value={fullName}>

            <Select

              value={fullName}

              sx={{

                backgroundColor: neutralLight,

                width: "150px",

                borderRadius: "0.25rem",

                p: "0.25rem 1rem",

                "& .MuiSvgIcon-root": {

                  pr: "0.25rem",

                  width: "3rem",

                },

                "& .MuiSelect-select:focus": {

                  backgroundColor:

                  neutralLight,

                },

              }}

              input={<InputBase />}

            >

              <MenuItem value={fullName}>

                <Typography>{fullName}</Typography>

              </MenuItem>

              <MenuItem onClick={() => dispatch(setLogout())}>Log
Out</MenuItem>

            </Select>

          </FormControl>

        </FlexBetween>
      ) : (
        <IconButton

          onClick={() => setIsMobileMenuToggled(!isMobileMenuToggled)}

        >

          <Menu />

        </IconButton>

      )}


      {/* MOBILE NAV */}

      {!isNonMobileScreens && isMobileMenuToggled && (

        <Box

          position="fixed-top"

          right="0"

          bottom="0"

          height="100%"

          zIndex="10"

          maxWidth="500px

          "

          minWidth="300px

          "

          backgroundColor={background}

        >

          {/* CLOSE ICON */}

          <Box display="flex" justifyContent="flex-end" p="1rem">
            <IconButton
              onClick={()
              =>
setIsMobileMenuToggled(!isMobileMenuToggled)}

            >

              <Close />

            </IconButton>

          </Box>
```

```
{/* MENU ITEMS */}
<FlexBetween
  display="flex"
  flexDirection="column"
  justifyContent="center"
  alignItems="center"
  gap="3rem"
>
  <IconButton
    onClick={() =>
    dispatch(setMode())} sx={{
    fontSize: "25px" }}
  >
    {theme.palette.mode === "dark" ? (
      <DarkMode sx={{ fontSize: "25px" }} />
    ) : (
      <LightMode sx={{ color: dark, fontSize: "25px" }} />
    )}
  </IconButton>
  <Message sx={{ fontSize: "25px" }} />
  <Notifications sx={{ fontSize: "25px" }} />
  <Help sx={{ fontSize: "25px" }} />
  <FormControl variant="standard" value={fullName}>
    <Select
      value={fullName}
      sx={{
        backgroundColor: neutralLight,
        width: "150px",
        borderRadius: "0.25rem",
        p: "0.25rem 1rem",
                  "& .MuiSvgIcon-root": {
                    pr: "0.25rem",
                    width: "3rem",
                  },
                  "& .MuiSelect-select:focus": {
                    backgroundColor:
                    neutralLight,
                  },
                }}
                input={<InputBase />}
              >
                <MenuItem value={fullName}>
                  <Typography>{fullName}</Typography>
                </MenuItem>
                <MenuItem onClick={() => dispatch(setLogout())}>
                  Log Out
                </MenuItem>
              </Select>
            </FormControl>
          </FlexBetween>
        </Box>
      )}
    </FlexBetween>
  );
};

export default Navbar;
```

Profile Page:

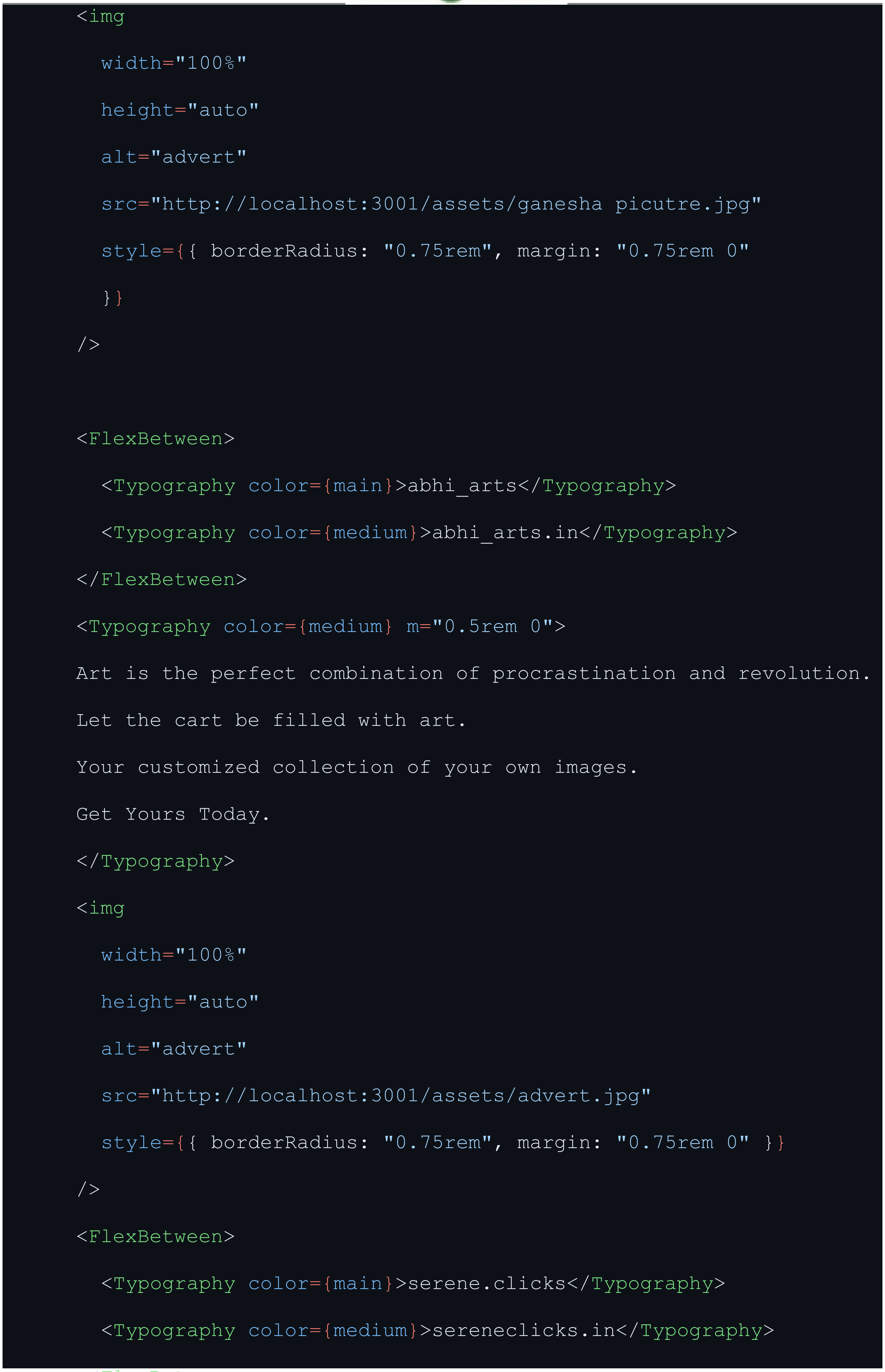
```
import { Box, useMediaQuery } from
"@mui/material"; import { useEffect, useState }
from "react";
import { useSelector } from "react-redux";
import { useParams } from "react-router-dom";
import Navbar from "scenes/navbar";
import FriendListWidget from "scenes/widgets/FriendListWidget";
import MyPostWidget from "scenes/widgets/MyPostWidget";
import PostsWidget from "scenes/widgets/PostsWidget";
import UserWidget from "scenes/widgets/UserWidget";


const ProfilePage = () => {
  const [user, setUser] = useState(null);
  const { userId } = useParams();
  const token = useSelector((state) => state.token);
  const isNonMobileScreens = useMediaQuery("(min-width:1000px)");

  const getUser = async () => {
    const response = await
fetch(`http://localhost:3001/users/${userId}`, {
      method: "GET",
      headers: { Authorization: `Bearer ${token}` },
    });
    const data = await response.json();
    setUser(data);
  };


  useEffect(() => {
```



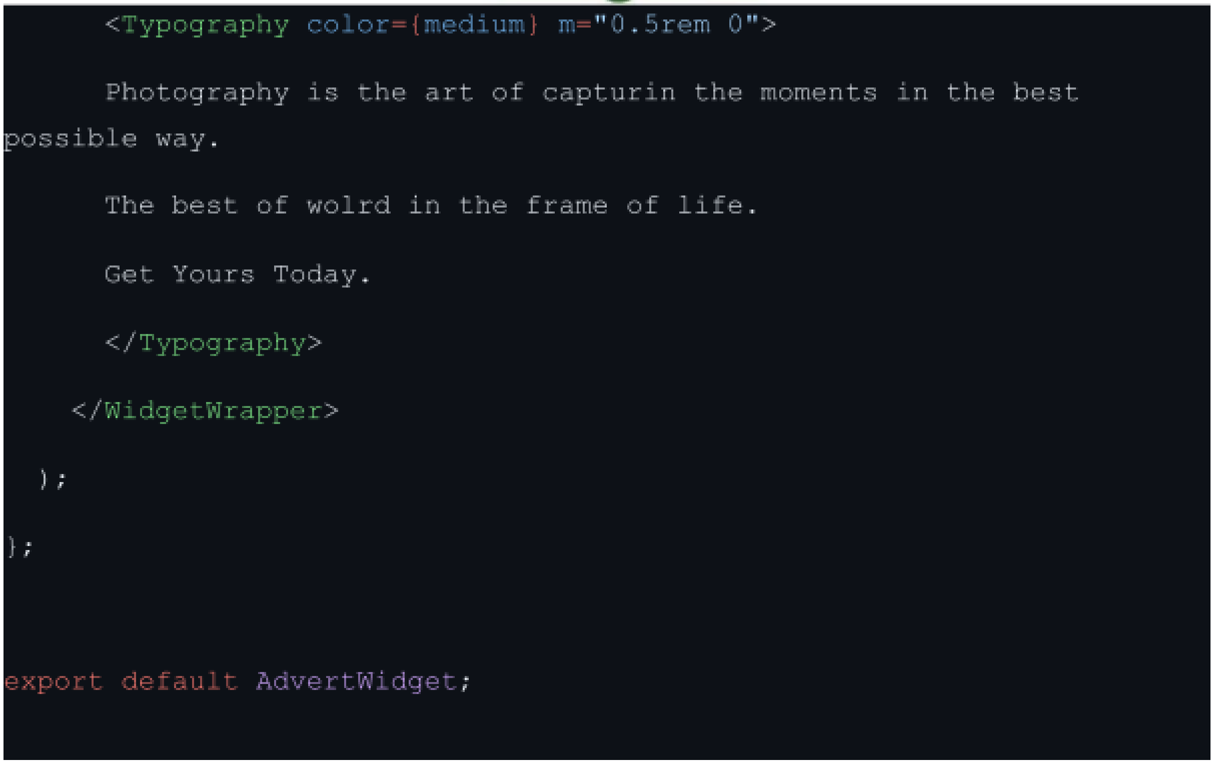
```
    getUser();
  }, []); // eslint-disable-line react-hooks/exhaustive-deps

  if (!user) return null;

  return (
    <Box>
      <Navbar />
      <Box
        width="100%"
        padding="2rem
        6%"
        display={isNonMobileScreens ? "flex" : "block"}
        gap="2rem"
        justifyContent="center"
      >
        <Box flexBasis={isNonMobileScreens ? "26%" : undefined}>
          <UserWidget userId={userId} picturePath={user.picturePath} />
          <Box m="2rem 0" />
          <FriendListWidget userId={userId} />
        </Box>
        <Box
          flexBasis={isNonMobileScreens ? "42%" : undefined}
          mt={isNonMobileScreens ? undefined : "2rem"}
        >
          <MyPostWidget picturePath={user.picturePath} />
          <Box m="2rem 0" />
          <PostsWidget userId={userId} isProfile />
        </Box>
      </Box>
    </Box>

  );

};


export default ProfilePage;
```

Advert Page:-

```
import { Typography, useTheme } from "@mui/material";

import FlexBetween from "components/FlexBetween";

import WidgetWrapper from "components/WidgetWrapper";


const AdvertWidget = () => {

  const { palette } = useTheme();

const dark = palette.neutral.dark; const

  main = palette.neutral.main; const

  medium = palette.neutral.medium;


return (

    <WidgetWrapper>

      <FlexBetween>

        <Typography color={dark} variant="h5" fontWeight="500">

          Sponsored

</Typography>

<Typography color={medium}>Create Ad</Typography>

  </FlexBetween>
```

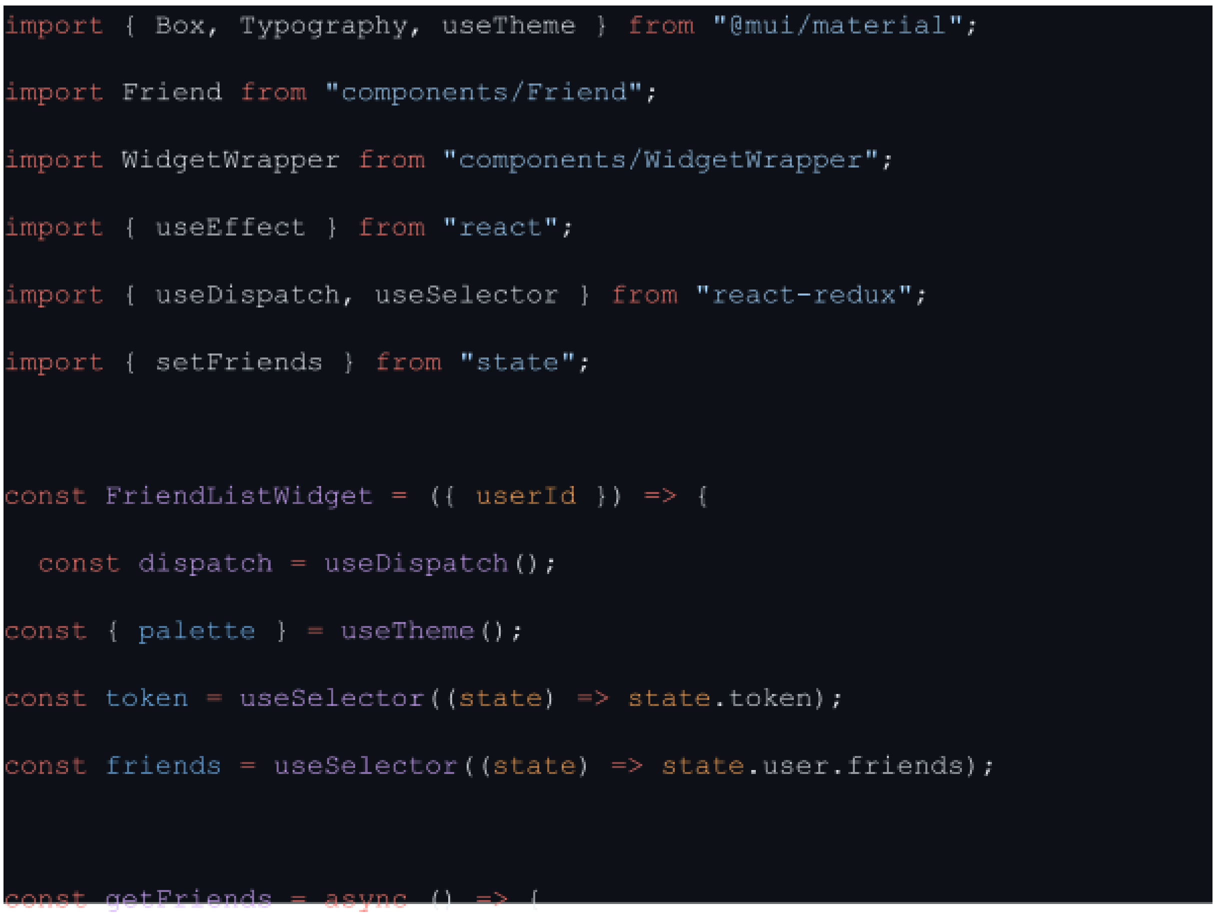


```
<img
  width="100%"
  height="auto"
  alt="advert"
  src="http://localhost:3001/assets/ganesha picutre.jpg"
  style={{ borderRadius: "0.75rem", margin: "0.75rem 0"
  }}
/>

<FlexBetween>
  <Typography color={main}>abhi_arts</Typography>
  <Typography color={medium}>abhi_arts.in</Typography>
</FlexBetween>
<Typography color={medium} m="0.5rem 0">
Art is the perfect combination of procrastination and revolution.
Let the cart be filled with art.
Your customized collection of your own images.
Get Yours Today.
</Typography>
<img
  width="100%"
  height="auto"
  alt="advert"
  src="http://localhost:3001/assets/advert.jpg"
  style={{ borderRadius: "0.75rem", margin: "0.75rem 0" }}
/>
<FlexBetween>
  <Typography color={main}>serene.clicks</Typography>
  <Typography color={medium}>sereneclicks.in</Typography>
</FlexBetween>
```



```
      <Typography color={medium} m="0.5rem 0">

      Photography is the art of capturin the moments in the best
possible way.

      The best of wolrd in the frame of life.

      Get Yours Today.

      </Typography>

    </WidgetWrapper>

  );

};


export default AdvertWidget;
```

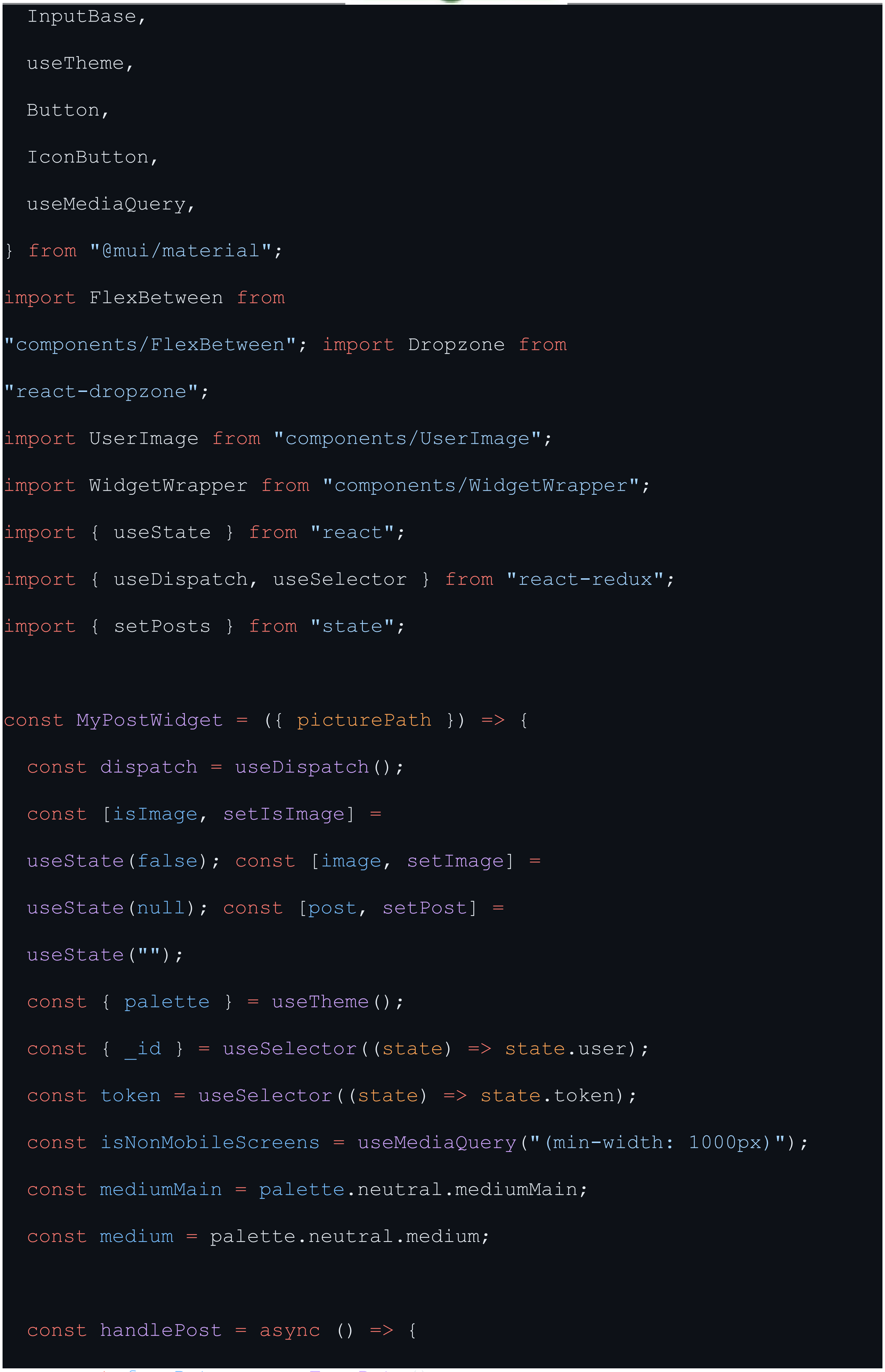
Friend List Widget:



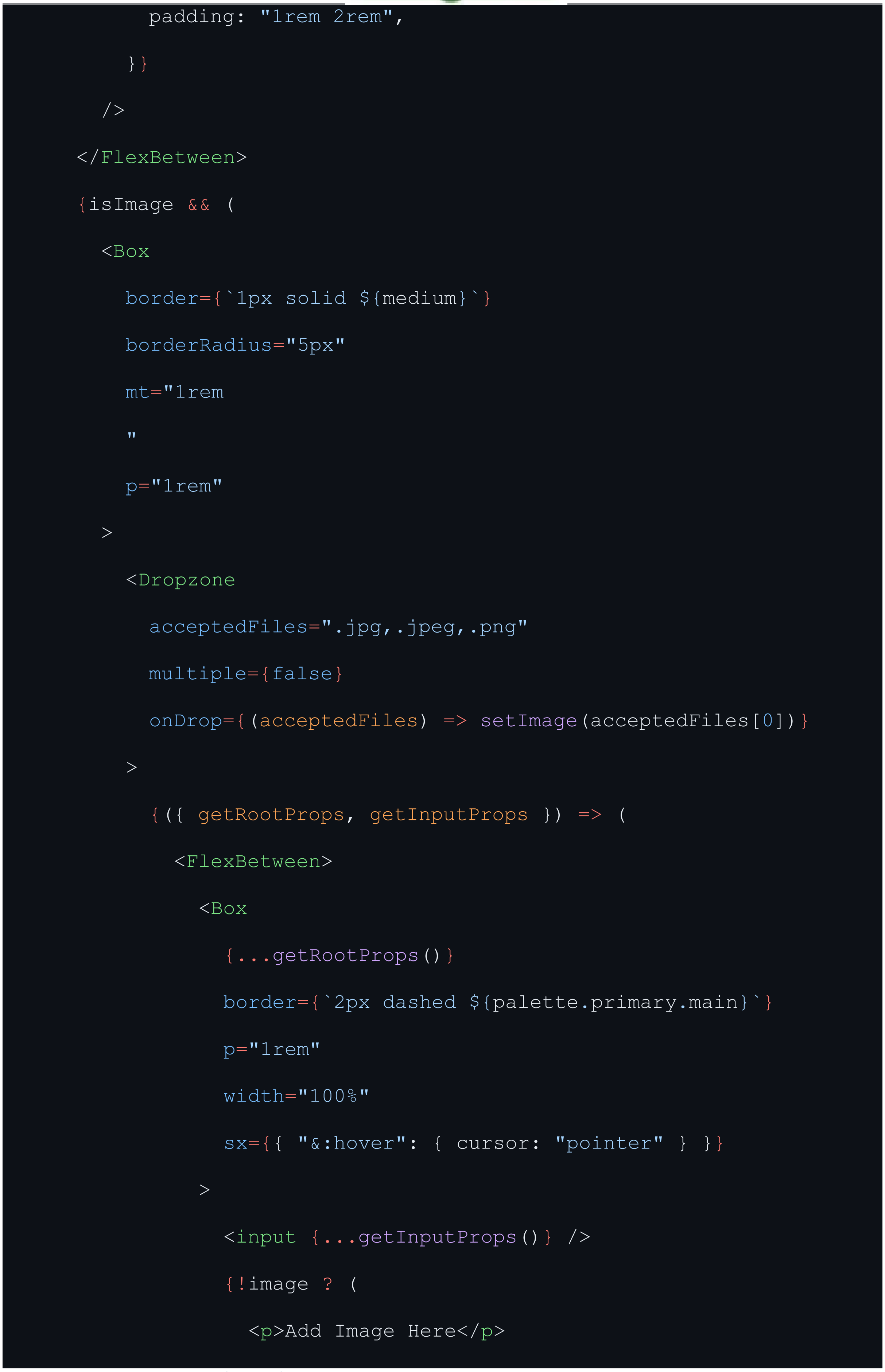
```
import { Box, Typography, useTheme } from "@mui/material";

import Friend from "components/Friend";

import WidgetWrapper from "components/WidgetWrapper";

import { useEffect } from "react";

import { useDispatch, useSelector } from "react-redux";

import { setFriends } from "state";


const FriendListWidget = ({ userId }) => {

  const dispatch = useDispatch();

const { palette } = useTheme();

const token = useSelector((state) => state.token);

const friends = useSelector((state) => state.user.friends);


const getFriends = async () => {
    const response = await fetch(
      `http://localhost:3001/users/${userId}/friends`,
      {
        method: "GET",
        headers: { Authorization: `Bearer ${token}` },
      }
    );
    const data = await response.json();
    dispatch(setFriends({ friends: data
    }));
  };

  useEffect(() => {
    getFriends();
  }, []); // eslint-disable-line react-hooks/exhaustive-deps

  return (
    <WidgetWrapper>
      <Typography
        color={palette.neutral.dark
        } variant="h5"
        fontWeight="500"
        sx={{ mb: "1.5rem" }}
      >
        Friend List
      </Typography>
      <Box display="flex" flexDirection="column" gap="1.5rem">
        {friends.map((friend) => (
          <Friend
            key={friend._id}
            friendId={friend._id}
            name={`${friend.firstName} ${friend.lastName}`}
            subtitle={friend.occupation}
            userPicturePath={friend.picturePath}
          />
        ))}
      </Box>
    </WidgetWrapper>
  );
};

export default FriendListWidget;
```

My Post Widget:

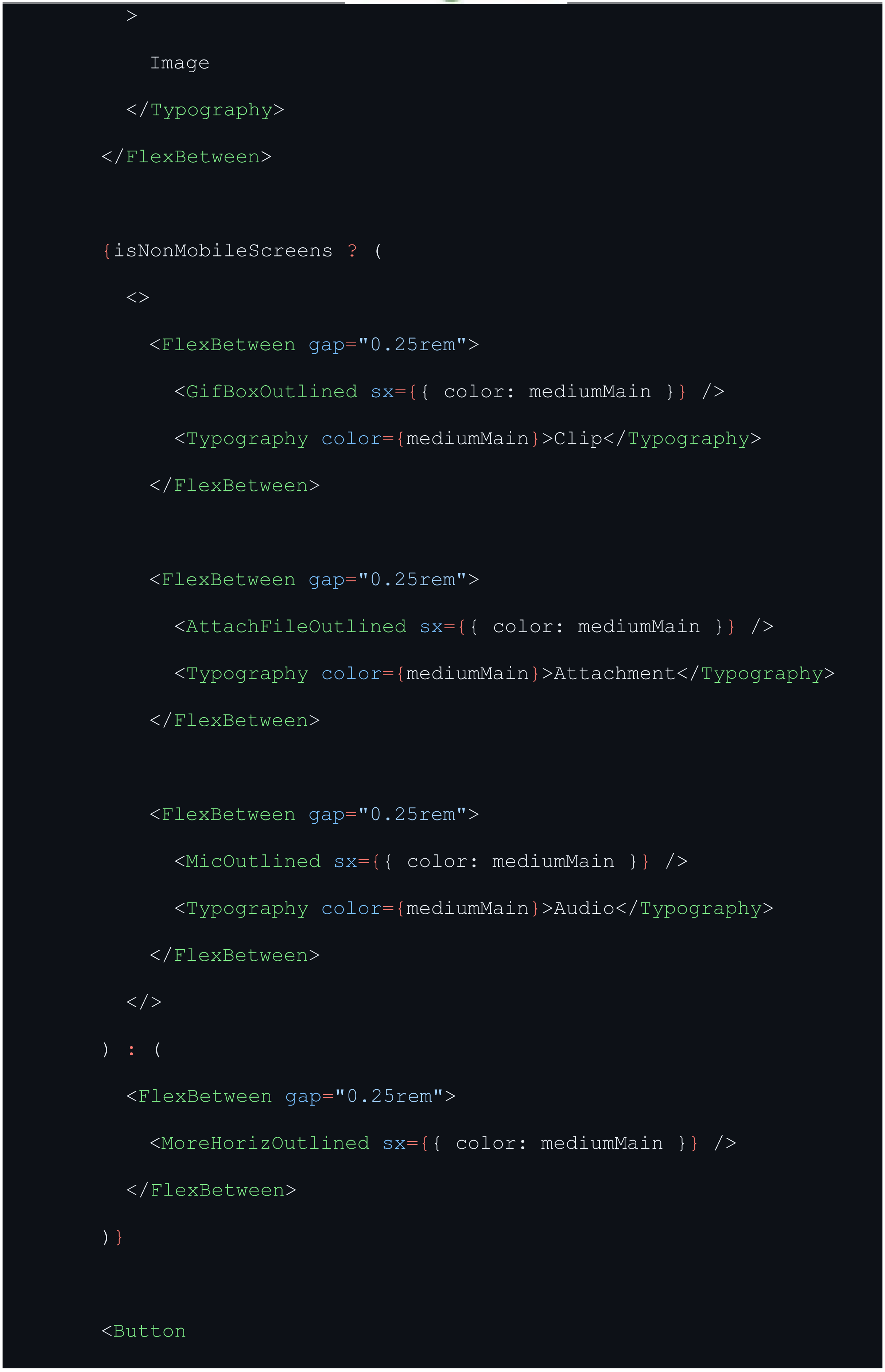
```
import {
  EditOutlined,
  DeleteOutlined,
  AttachFileOutlined,
  GifBoxOutlined,
  ImageOutlined,
  MicOutlined,
  MoreHorizOutlined,
} from "@mui/icons-material";
import {
Box, Divider,
  Typography,
```



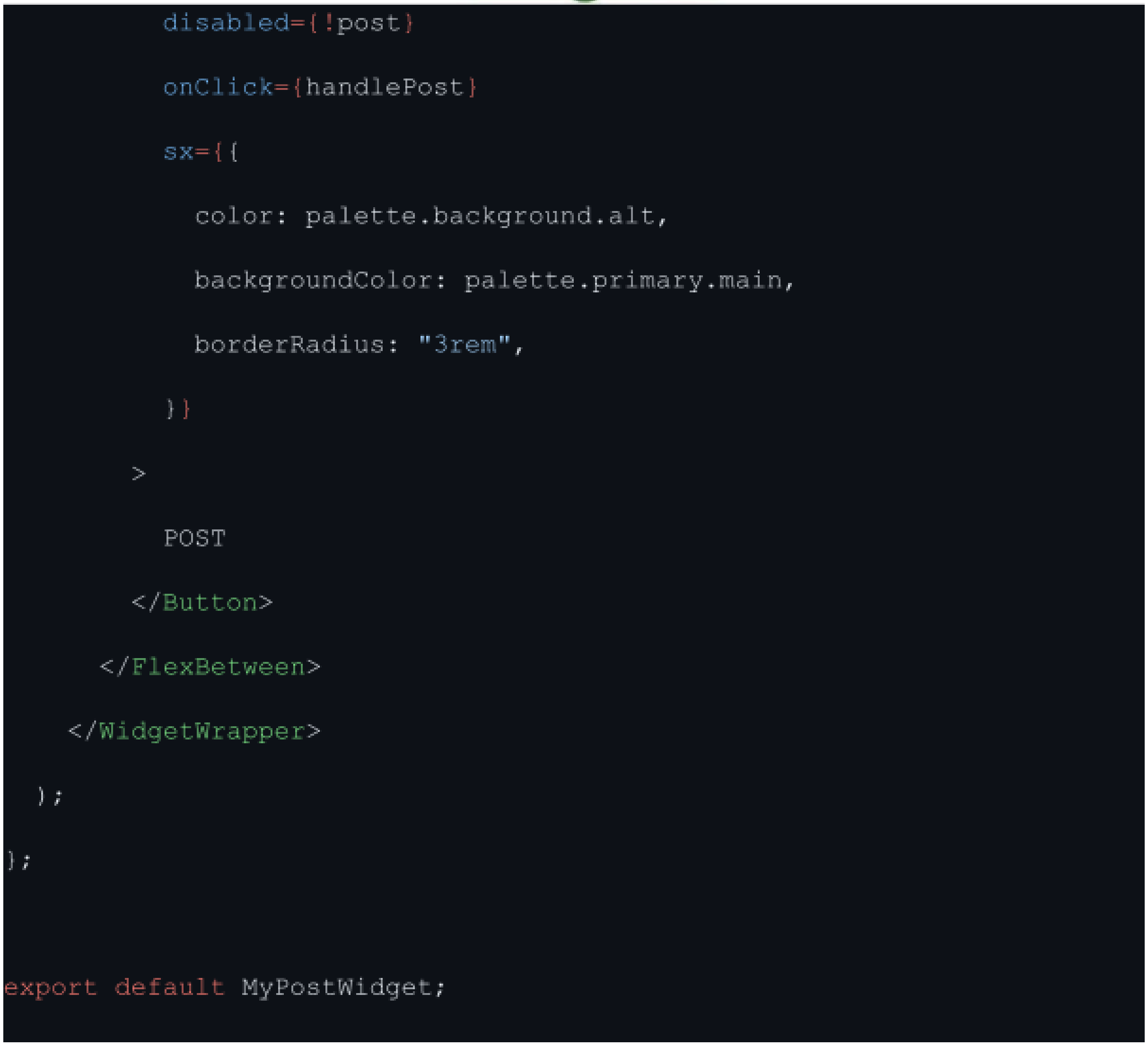
```
  InputBase,

  useTheme,

  Button,

  IconButton,

  useMediaQuery,

} from "@mui/material";

import FlexBetween from

"components/FlexBetween"; import Dropzone from

"react-dropzone";

import UserImage from "components/UserImage";

import WidgetWrapper from "components/WidgetWrapper";

import { useState } from "react";

import { useDispatch, useSelector } from "react-redux";

import { setPosts } from "state";


const MyPostWidget = ({ picturePath }) => {

  const dispatch = useDispatch();

  const [isImage, setIsImage] =

  useState(false); const [image, setImage] =

  useState(null); const [post, setPost] =

  useState("");

  const { palette } = useTheme();

  const { _id } = useSelector((state) => state.user);

  const token = useSelector((state) => state.token);

  const isNonMobileScreens = useMediaQuery("(min-width: 1000px)");

  const mediumMain = palette.neutral.mediumMain;

  const medium = palette.neutral.medium;


  const handlePost = async () => {

    const formData = new FormData();

    formData.append("userId", _id);
    formData.append("description", post);
    if (image) {
      formData.append("picture", image);
      formData.append("picturePath",
      image.name);
    }

    const response = await fetch(`http://localhost:3001/posts`, {
      method: "POST",
      headers: { Authorization: `Bearer ${token}` },
      body: formData,
    });
    const posts = await response.json();
    dispatch(setPosts({ posts }));
    setImage(null);
    setPost("");
  };

  return (
    <WidgetWrapper>
      <FlexBetween gap="1.5rem">
        <UserImage image={picturePath} />
        <InputBase
          placeholder="What's on your mind..."
          onChange={(e) =>
          setPost(e.target.value)} value={post}
          sx={{
            width: "100%",
            backgroundColor: palette.neutral.light,
            borderRadius: "2rem",
```



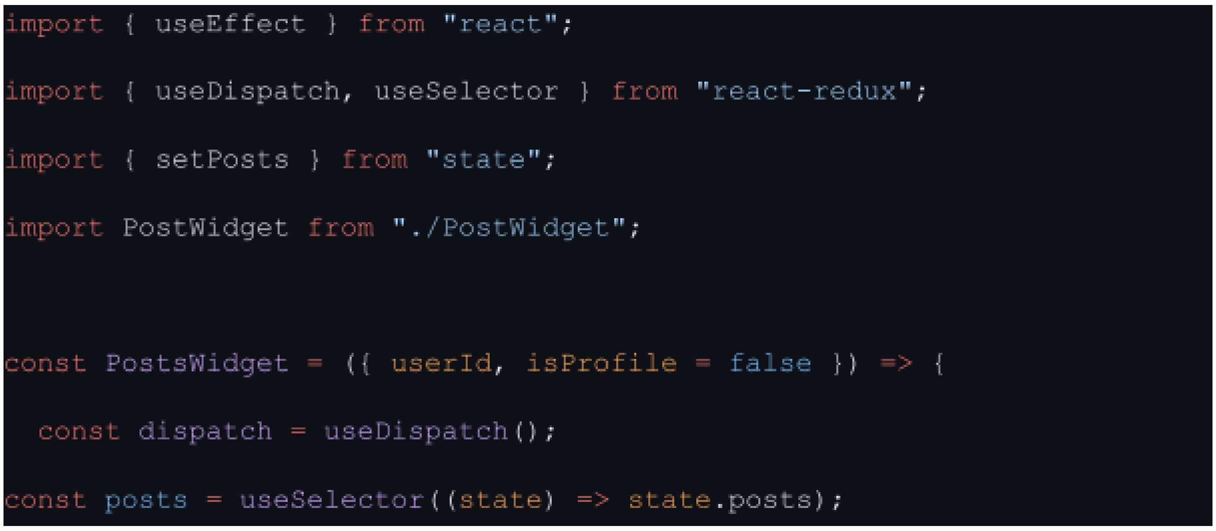
```
      padding: "1rem 2rem",
    }}
  />
</FlexBetween>
{isImage && (
  <Box
    border={`1px solid ${medium}`}
    borderRadius="5px"
    mt="1rem
    "
    p="1rem"
  >
    <Dropzone
      acceptedFiles=".jpg,.jpeg,.png"
      multiple={false}
      onDrop={(acceptedFiles) => setImage(acceptedFiles[0])}
    >
      {({ getRootProps, getInputProps }) => (
        <FlexBetween>
          <Box
            {...getRootProps()}
            border={`2px dashed ${palette.primary.main}`}
            p="1rem"
            width="100%"
            sx={{ "&:hover": { cursor: "pointer" } }}
          >
            <input {...getInputProps()} />
            {!image ? (
              <p>Add Image Here</p>
            ) : (
                      <FlexBetween>
                        <Typography>{image.name}</Typography>
                        <EditOutlined />
                      </FlexBetween>
                    )}
                  </Box>
                  {image && (
                    <IconButton
                      onClick={() => setImage(null)}
                      sx={{ width: "15%" }}
                    >
                      <DeleteOutlined />
                    </IconButton>
                  )}
                </FlexBetween>
              )}
            </Dropzone>
          </Box>
      )}

      <Divider sx={{ margin: "1.25rem 0" }} />

      <FlexBetween>
        <FlexBetween gap="0.25rem" onClick={()
=> setIsImage(!isImage)}>
          <ImageOutlined sx={{ color: mediumMain }} />
          <Typography
            color={mediumMain}
            sx={{ "&:hover": { cursor: "pointer", color: medium } }}
```



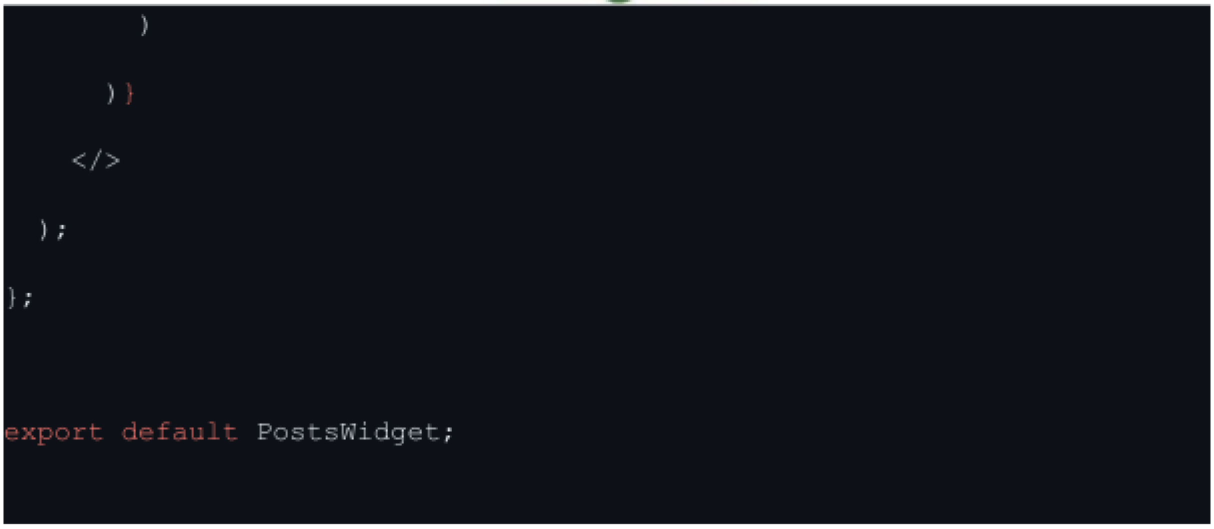
```
  >
    Image
  </Typography>
</FlexBetween>

{isNonMobileScreens ? (
  <>
    <FlexBetween gap="0.25rem">
      <GifBoxOutlined sx={{ color: mediumMain }} />
      <Typography color={mediumMain}>Clip</Typography>
    </FlexBetween>

    <FlexBetween gap="0.25rem">
      <AttachFileOutlined sx={{ color: mediumMain }} />
      <Typography color={mediumMain}>Attachment</Typography>
    </FlexBetween>

    <FlexBetween gap="0.25rem">
      <MicOutlined sx={{ color: mediumMain }} />
      <Typography color={mediumMain}>Audio</Typography>
    </FlexBetween>
  </>
) : (
  <FlexBetween gap="0.25rem">
    <MoreHorizOutlined sx={{ color: mediumMain }} />
  </FlexBetween>
)}

<Button
```


```
          disabled={!post}
          onClick={handlePost}
          sx={{
            color: palette.background.alt,
            backgroundColor: palette.primary.main,
            borderRadius: "3rem",
          }}
        >
          POST
        </Button>
      </FlexBetween>
    </WidgetWrapper>
  );
};

export default MyPostWidget;
```

Post Widget:

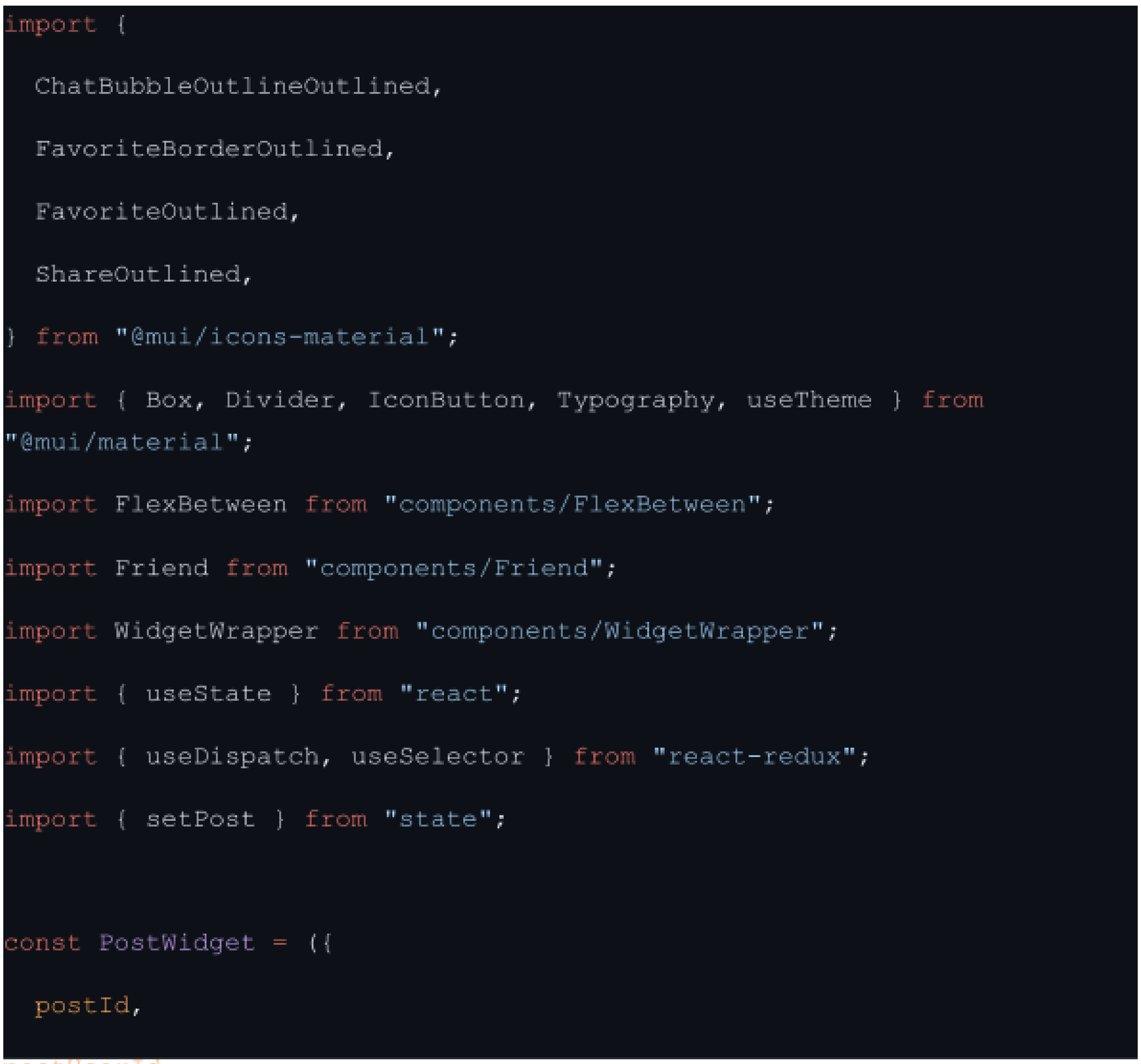

```
import { useEffect } from "react";
import { useDispatch, useSelector } from "react-redux";
import { setPosts } from "state";
import PostWidget from "./PostWidget";

const PostsWidget = ({ userId, isProfile = false }) => {
  const dispatch = useDispatch();
const posts = useSelector((state) => state.posts);
const token = useSelector((state) => state.token);

const getPosts = async () => {
  const response = await fetch("http://localhost:3001/posts", {
    method: "GET",
    headers: { Authorization: `Bearer ${token}` },
  });
  const data = await response.json();
  dispatch(setPosts({ posts: data
  }));
};

const getUserPosts = async () => {
  const response = await fetch(
    `http://localhost:3001/posts/${userId}/posts`,
    {
      method: "GET",
      headers: { Authorization: `Bearer ${token}` },
    }
  );
  const data = await response.json();
  dispatch(setPosts({ posts: data
  }));
};

useEffect(() => {
  if (isProfile)
  {
    getUserPosts();
  } else {
    getPosts();
```

```
}, []); // eslint-disable-line react-hooks/exhaustive-deps


return (

  <>

    {posts.map

      ( ({

        _id,

        userId,

        firstName,

        lastName,

        description,

        location,

        picturePath,

        userPicturePath

        , likes,

        comments,

      }) => (

        <PostWidget

          key={_id}

          postId={_id}

          postUserId={userId}

          name={`${firstName} ${lastName}`}

          description={description}

          location={location}

          picturePath={picturePath}

          userPicturePath={userPicturePath}

          likes={likes}

          comments={comments}

        />
```



```
        )
      )}
    </>
  );
};

export default PostsWidget;
```

Page 3:



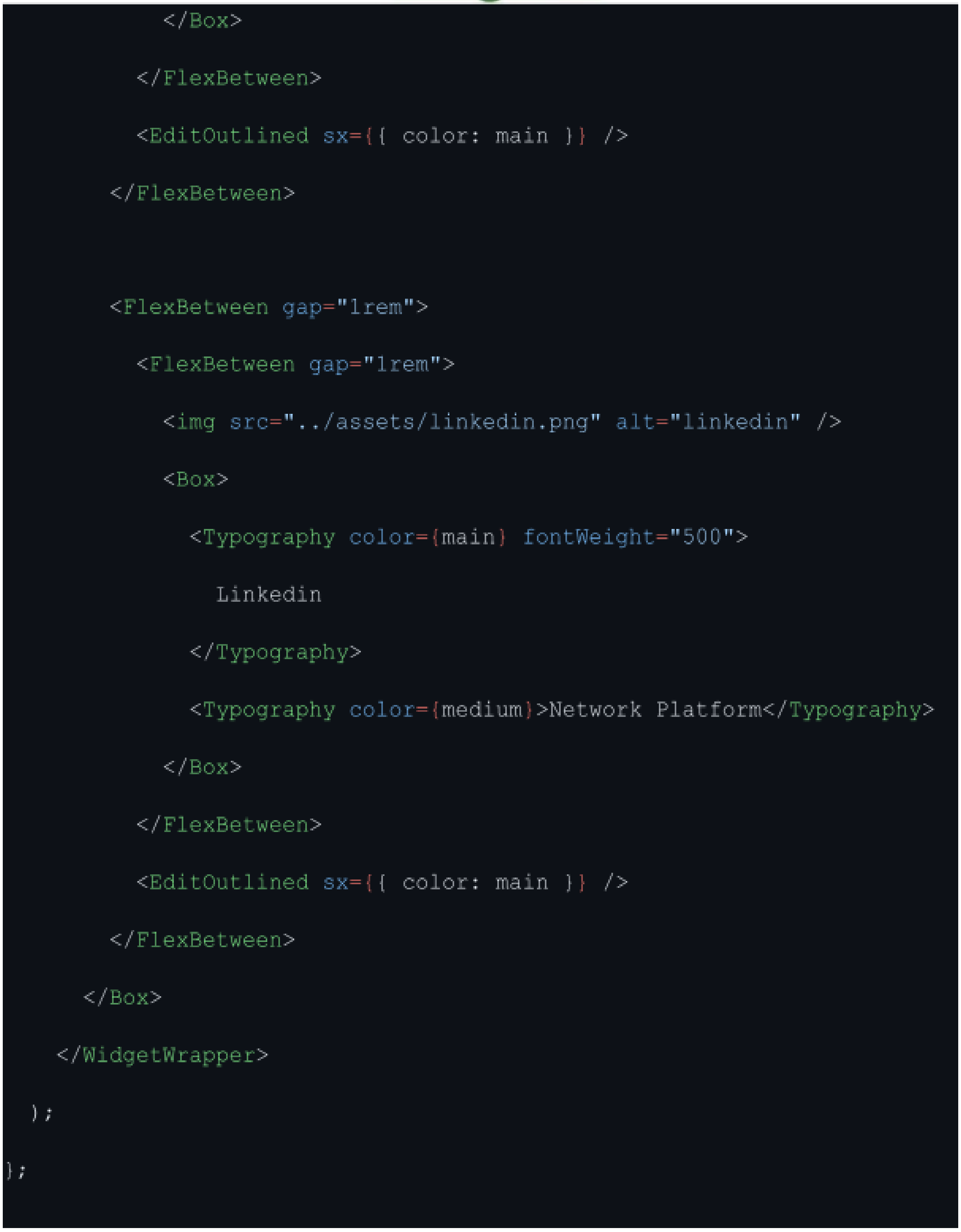
```
import {
  ChatBubbleOutlineOutlined,
  FavoriteBorderOutlined,
  FavoriteOutlined,
  ShareOutlined,
} from "@mui/icons-material";
import { Box, Divider, IconButton, Typography, useTheme } from
"@mui/material";
import FlexBetween from "components/FlexBetween";
import Friend from "components/Friend";
import WidgetWrapper from "components/WidgetWrapper";
import { useState } from "react";
import { useDispatch, useSelector } from "react-redux";
import { setPost } from "state";

const PostWidget = ({
  postId,
postUserId,
  name,

  description,

  location,

  picturePath,

  userPicturePath,

  likes,

  comments,

}) => {

  const [isComments, setIsComments] = useState(false);

  const dispatch = useDispatch();

  const token = useSelector((state) => state.token);

  const loggedInUserId = useSelector((state) => state.user._id);

  const isLiked = Boolean(likes[loggedInUserId]);

  const likeCount = Object.keys(likes).length;


  const { palette } = useTheme();

  const main =

  palette.neutral.main;

  const primary = palette.primary.main;

  const patchLike = async () => {
    const response = await
fetch(`http://localhost:3001/posts/${postId}/like`, {

      method: "PATCH",

      headers: {

        Authorization: `Bearer ${token}`,

        "Content-Type":

        "application/json",

      },

      body: JSON.stringify({ userId: loggedInUserId }),

    });
    const updatedPost = await response.json();
    dispatch(setPost({ post: updatedPost }));
  };

  return (
    <WidgetWrapper m="2rem 0">
      <Friend
        friendId={postUserId
        } name={name}
        subtitle={location}
        userPicturePath={userPicturePath}
      />
      <Typography color={main} sx={{ mt: "1rem" }}>
        {description}
      </Typography>
      {picturePath && (
        <img
          width="100%"
          height="auto"
          alt="post"
          style={{ borderRadius: "0.75rem", marginTop: "0.75rem" }}
          src={`http://localhost:3001/assets/${picturePath}`}
        />
      )}
      <FlexBetween mt="0.25rem">
        <FlexBetween gap="1rem">
          <FlexBetween gap="0.3rem">
            <IconButton onClick={patchLike}>
              {isLiked ? (
                <FavoriteOutlined sx={{ color: primary }} />
              ) : (
                <FavoriteBorderOutlined />
              )}
            </IconButton>
            <Typography>{likeCount}</Typography>
          </FlexBetween>

          <FlexBetween gap="0.3rem">
            <IconButton onClick={() => setIsComments(!isComments)}>
              <ChatBubbleOutlineOutlined />
            </IconButton>
            <Typography>{comments.length}</Typography>
          </FlexBetween>
        </FlexBetween>

        <IconButton>
          <ShareOutlined />
        </IconButton>
      </FlexBetween>
      {isComments && (
        <Box mt="0.5rem">
          {comments.map((comment, i) => (
            <Box key={`${name}-${i}`}>
              <Divider />
              <Typography sx={{ color: main, m: "0.5rem 0", pl: "1rem"
}}>
                {comment}
              </Typography>
            </Box>
          ))}
          <Divider />
        </Box>
      )}
    </WidgetWrapper>
  );
};

export default PostWidget;
```

User Widget:

```
import {
  ManageAccountsOutlined,
  EditOutlined,
  LocationOnOutlined,
  WorkOutlineOutlined,
} from "@mui/icons-material";
import { Box, Typography, Divider, useTheme } from "@mui/material";
import UserImage from "components/UserImage";
import FlexBetween from "components/FlexBetween";
import WidgetWrapper from "components/WidgetWrapper";
import { useSelector } from "react-redux";
import { useEffect, useState } from "react";
import { useNavigate } from "react-router-dom";

const UserWidget = ({ userId, picturePath }) => {
  const [user, setUser] = useState(null);

  const { palette } = useTheme();

  const navigate = useNavigate();

  const token = useSelector((state) => state.token);

  const dark = palette.neutral.dark;

  const medium = palette.neutral.medium;

  const main = palette.neutral.main;


  const getUser = async () => {
    const response = await
fetch(`http://localhost:3001/users/${userId}`, {

      method: "GET",

      headers: { Authorization: `Bearer ${token}` },

    });

    const data = await response.json();

    setUser(data);

  };



  useEffect(() => {

    getUser();

  }, []); // eslint-disable-line react-hooks/exhaustive-deps



  if (!user) {

    return

    null;

  }



  const {

    firstName,

    lastName,
    location,
    occupation,
    viewedProfile,
    impressions,
    friends,
  } = user;

  return (
    <WidgetWrapper>
      {/* FIRST ROW */}
      <FlexBetween
        gap="0.5rem"
        pb="1.1rem"
        onClick={() => navigate(`/profile/${userId}`)}
      >
        <FlexBetween gap="1rem">
          <UserImage image={picturePath} />
          <Box>
            <Typography
              variant="h4"
              color={dark}
              fontWeight="500"
              sx={{
                "&:hover": {
                  color: palette.primary.light,
                  cursor: "pointer",
                },
              }}
            >
```

```
              {firstName} {lastName}
            </Typography>
            <Typography color={medium}>{friends.length}
friends</Typography>
          </Box>
        </FlexBetween>
        <ManageAccountsOutlined />
      </FlexBetween>

      <Divider />

      {/* SECOND ROW */}
      <Box p="1rem 0">
        <Box display="flex" alignItems="center" gap="1rem" mb="0.5rem">
          <LocationOnOutlined fontSize="large" sx={{ color: main }} />
          <Typography color={medium}>{location}</Typography>
        </Box>
        <Box display="flex" alignItems="center" gap="1rem">
          <WorkOutlineOutlined fontSize="large" sx={{ color: main }} />
          <Typography color={medium}>{occupation}</Typography>
        </Box>
      </Box>

      <Divider />

      {/* THIRD ROW */}
      <Box p="1rem 0">
        <FlexBetween mb="0.5rem">
          <Typography color={medium}>Who's viewed
your profile</Typography>
```

```
          <Typography color={main} fontWeight="500">
            {viewedProfile}
          </Typography>
        </FlexBetween>
        <FlexBetween>
          <Typography color={medium}>Impressions of
your post</Typography>
          <Typography color={main} fontWeight="500">
            {impressions}
          </Typography>
        </FlexBetween>
      </Box>


      <Divider />


      {/* FOURTH ROW */}
      <Box p="1rem 0">
        <Typography fontSize="1rem" color={main} fontWeight="500"
mb="1rem">
          Social Profiles
        </Typography>


        <FlexBetween gap="1rem" mb="0.5rem">
          <FlexBetween gap="1rem">
            <img src="../assets/twitter.png" alt="twitter" />
            <Box>
              <Typography color={main} fontWeight="500">
                Twitter
              </Typography>
              <Typography color={medium}>Social Network</Typography>
```



```
            </Box>
          </FlexBetween>
          <EditOutlined sx={{ color: main }} />
        </FlexBetween>

        <FlexBetween gap="1rem">
          <FlexBetween gap="1rem">
            <img src="../assets/linkedin.png" alt="linkedin" />
            <Box>
              <Typography color={main} fontWeight="500">
                Linkedin
              </Typography>
              <Typography color={medium}>Network Platform</Typography>
            </Box>
          </FlexBetween>
          <EditOutlined sx={{ color: main }} />
        </FlexBetween>
      </Box>
    </WidgetWrapper>
  );
};
```

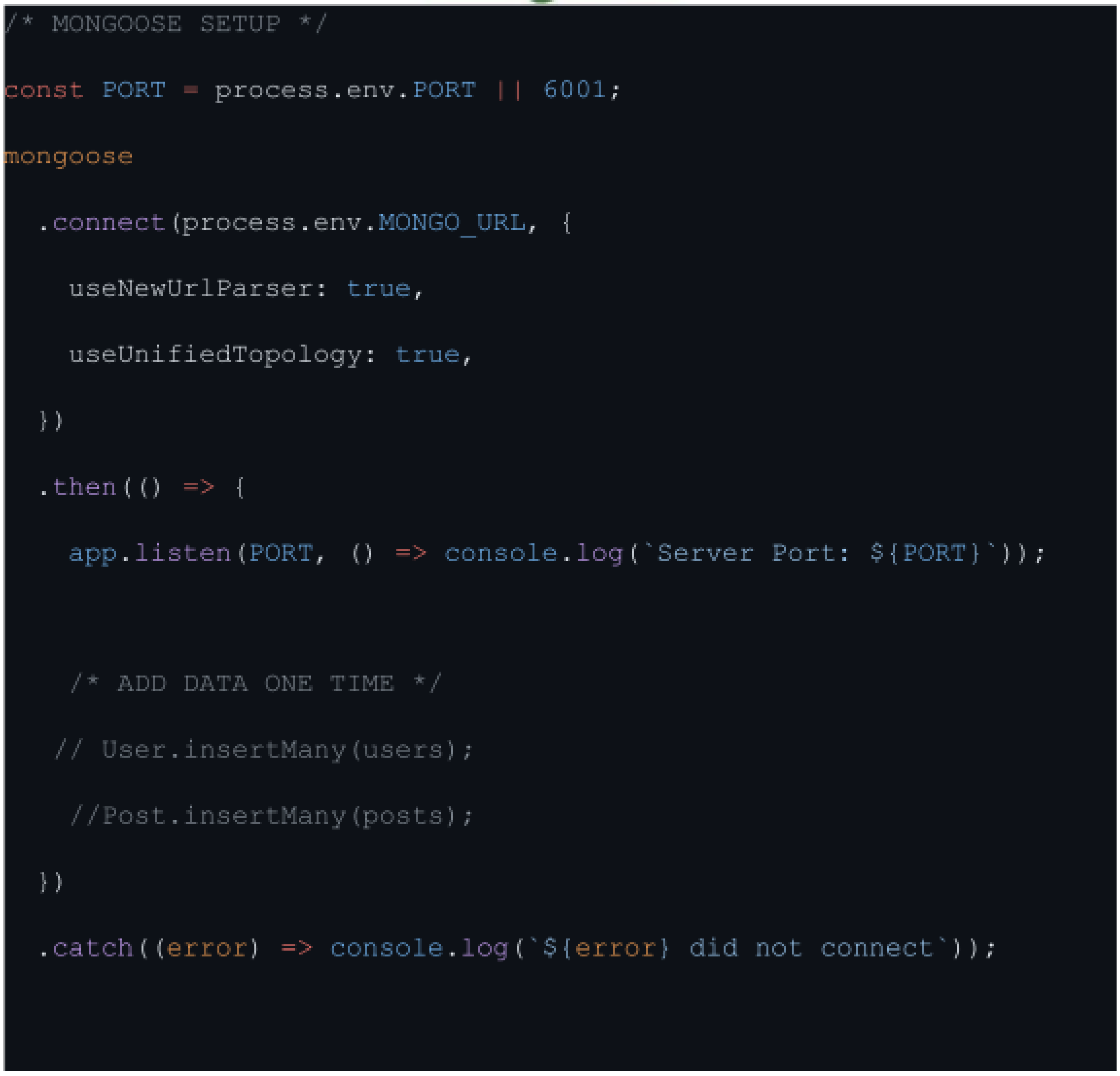
<u>BACKEND CODE : (DATABASE , SIGNUP , LOGIN)-</u>

Index.js:

```
import express from "express";

import bodyParser from "body-parser";

import mongoose from "mongoose";

import cors from "cors";

import      dotenv      from

"dotenv";    import    multer

from     "multer";     import

helmet     from     "helmet";

import      morgan      from

"morgan";  import  path  from

"path";

import { fileURLToPath } from "url";

import authRoutes from

"./routes/auth.js";

import userRoutes from "./routes/users.js";

import postRoutes from "./routes/posts.js";

import { register } from

"./controllers/auth.js";

import { createPost } from "./controllers/posts.js";

import { verifyToken } from "./middleware/auth.js";

import User from "./models/User.js";

import Post from "./models/Post.js";

import { users, posts } from "./data/index.js";


/* CONFIGURATIONS */

const __filename = fileURLToPath(import.meta.url);

const __dirname = path.dirname(_filename);

dotenv.config();

const app = express();
```

```
app.use(express.json())

; app.use(helmet());
```

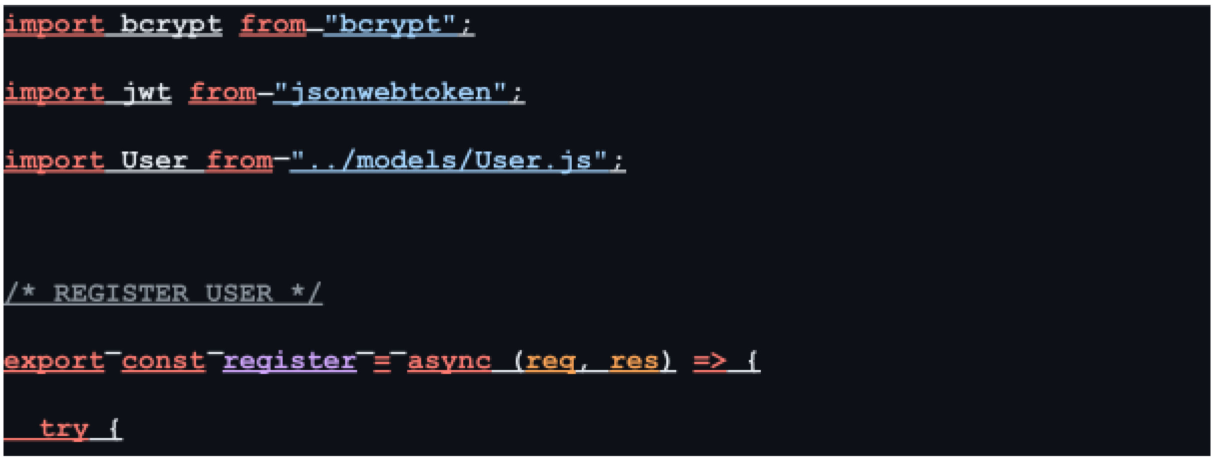
```
app.use(helmet.crossOriginResourcePolicy({ policy: "cross-origin"
}));

app.use(morgan("common"));

app.use(bodyParser.json({ limit: "30mb", extended: true }));

app.use(bodyParser.urlencoded({ limit: "30mb", extended: true

})); app.use(cors());

app.use("/assets", express.static(path.join(_dirname,
"public/assets")));


/* FILE STORAGE */

const storage = multer.diskStorage({

  destination: function (req, file, cb) {

    cb(null, "public/assets");

  },

  filename: function (req, file, cb) {

    cb(null, file.originalname);

  },

});

const upload = multer({ storage });


/* ROUTES WITH FILES */

app.post("/auth/register", upload.single("picture"), register);

app.post("/posts", verifyToken, upload.single("picture"),
createPost);


/* ROUTES */

app.use("/auth", authRoutes);

app.use("/users", userRoutes);

app.use("/posts", postRoutes);
```



```
/* MONGOOSE SETUP */

const PORT = process.env.PORT || 6001;

mongoose

  .connect(process.env.MONGO_URL, {

    useNewUrlParser: true,

    useUnifiedTopology: true,

  })

  .then(() => {

    app.listen(PORT, () => console.log(`Server Port: ${PORT}`));


    /* ADD DATA ONE TIME */

   // User.insertMany(users);

    //Post.insertMany(posts);

  })

  .catch((error) => console.log(`${error} did not connect`));
```

Controllers
:

Auth.js:

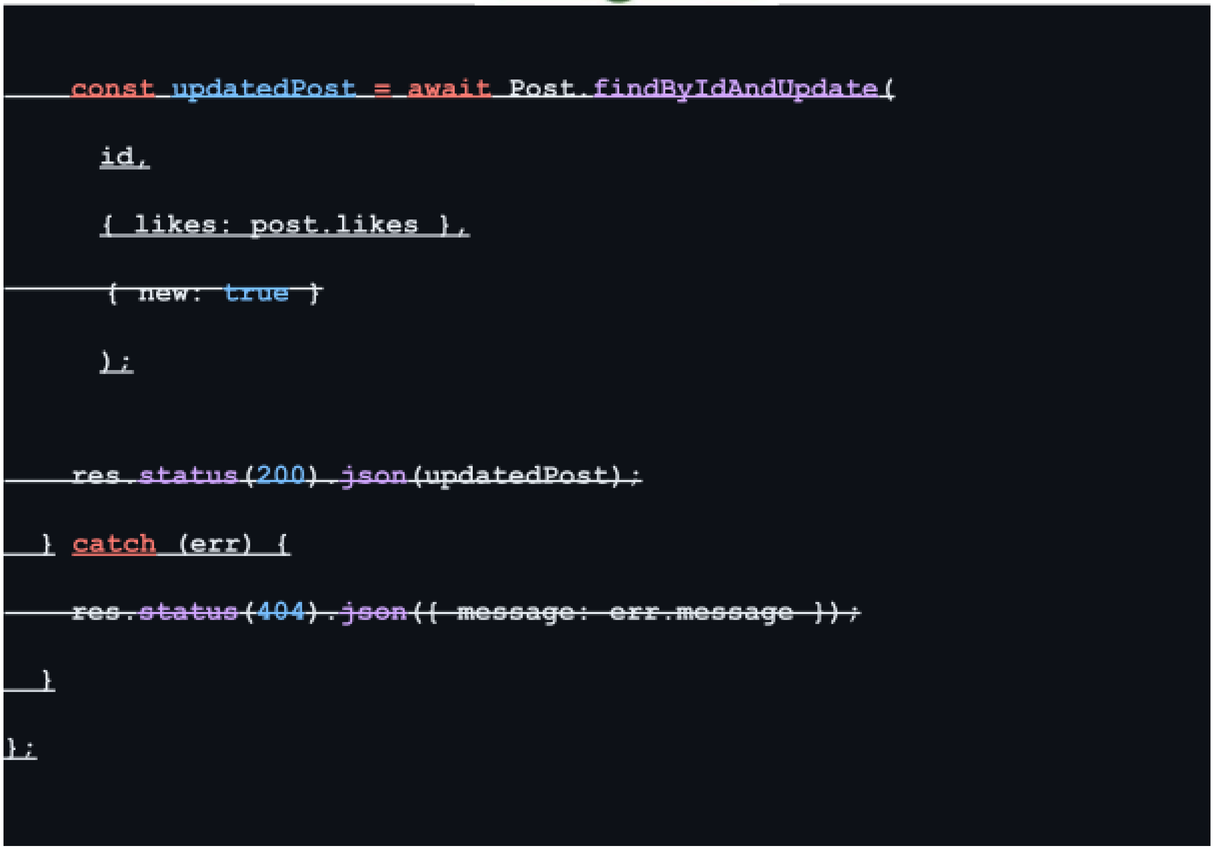


```
import bcrypt from "bcrypt";

import jwt from "jsonwebtoken";

import User from "../models/User.js";


/* REGISTER USER */

export const register = async (req, res) => {

  try {
    const {
      firstName,
      lastName,
      email,
      password,
      picturePath,
      friends,
      location,
      occupation,
    } = req.body;

    const salt = await bcrypt.genSalt();
    const passwordHash = await bcrypt.hash(password, salt);

    const newUser = new User({
      firstName,
      lastName,
      email,
      password: passwordHash,
      picturePath,
      friends,
      location,
      occupation,
      viewedProfile: Math.floor(Math.random() * 10000),
      impressions: Math.floor(Math.random() * 10000),
    });
    const savedUser = await newUser.save();
    res.status(201).json(savedUser);
  } catch (err) {
    res.status(500).json({ error: err.message });

  }

};


/* LOGGING IN */

export const login = async (req, res) => {

  try {

    const { email, password } = req.body;

    const user = await User.findOne({ email: email });

    if (!user) return res.status(400).json({ msg: "User does not exist. 
" });


    const isMatch = await bcrypt.compare(password, user.password);
    if (!isMatch) return res.status(400).json({ msg: "Invalid 
credentials. " });


    const token = jwt.sign({ id: user._id }, process.env.JWT_SECRET);

    delete user.password;

    res.status(200).json({ token, user });

  } catch (err) {

    res.status(500).json({ error: err.message });

  }

};
```

Posts.js:

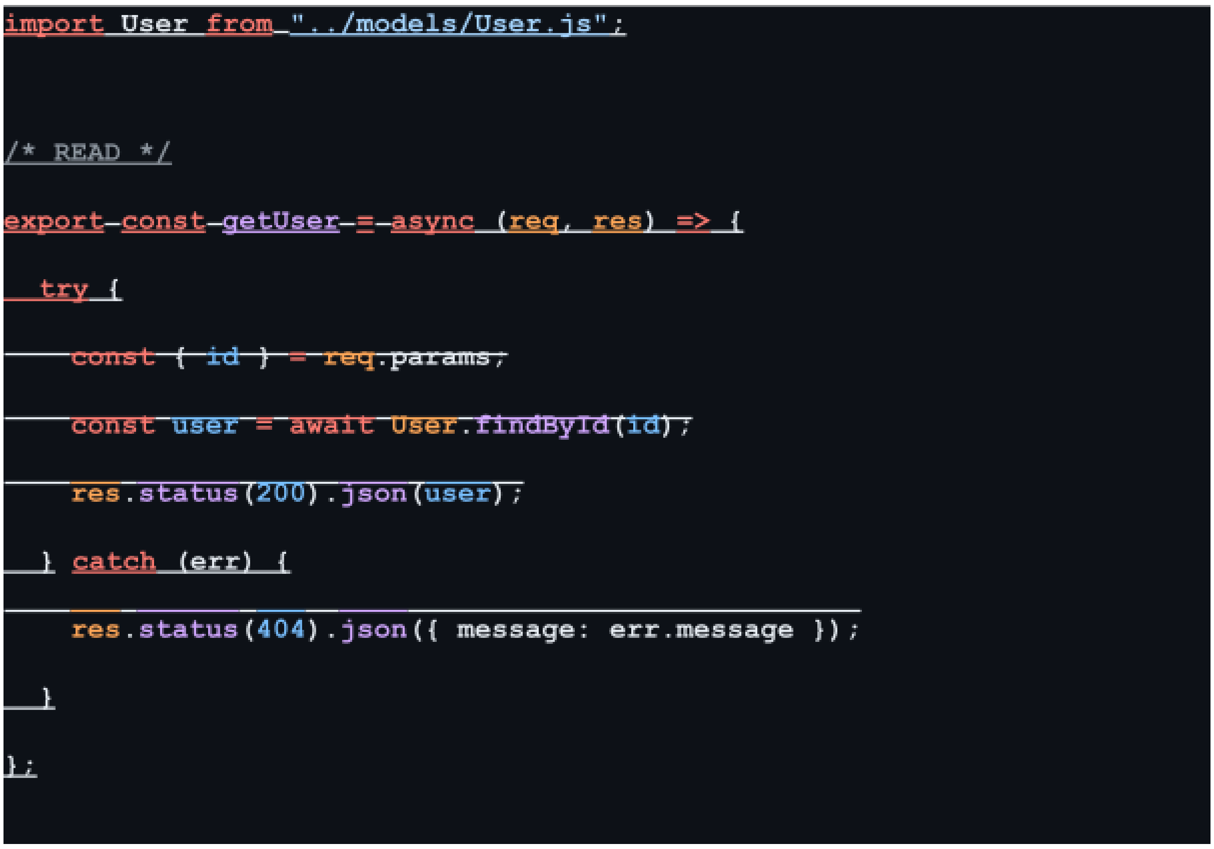
```
import Post from "../models/Post.js";

import User from "../models/User.js";
```

```
/* CREATE */
export const createPost = async (req, res) => {
  try {
    const { userId, description, picturePath } = req.body;
    const user = await User.findById(userId);
    const newPost = new Post({
      userId,
      firstName: user.firstName,
      lastName: user.lastName,
      location: user.location,
      description,
      userPicturePath: user.picturePath,
      picturePath,
      likes: {},
      comments: [],
    });
    await newPost.save();

    const post = await Post.find();
    res.status(201).json(post);
  } catch (err) {
    res.status(409).json({ message: err.message });
  }
};


/* READ */
export const getFeedPosts = async (req, res) => {
  try {
    const post = await Post.find().sort({"updatedAt":-1});
    res.status(200).json(post);
  } catch (err) {
    res.status(404).json({ message: err.message });
  }
};

export const getUserPosts = async (req, res) => {
  try {
    const { userId } = req.params;
    const post = await Post.find({ userId });
    res.status(200).json(post);
  } catch (err) {
    res.status(404).json({ message: err.message });
  }
};

/* UPDATE */
export const likePost = async (req, res) => {
  try {
    const { id } = req.params;
    const { userId } = req.body;
    const post = await Post.findById(id);
    const isLiked = post.likes.get(userId);

    if (isLiked) {
      post.likes.delete(userId);
    } else {
      post.likes.set(userId, true);
    }
```

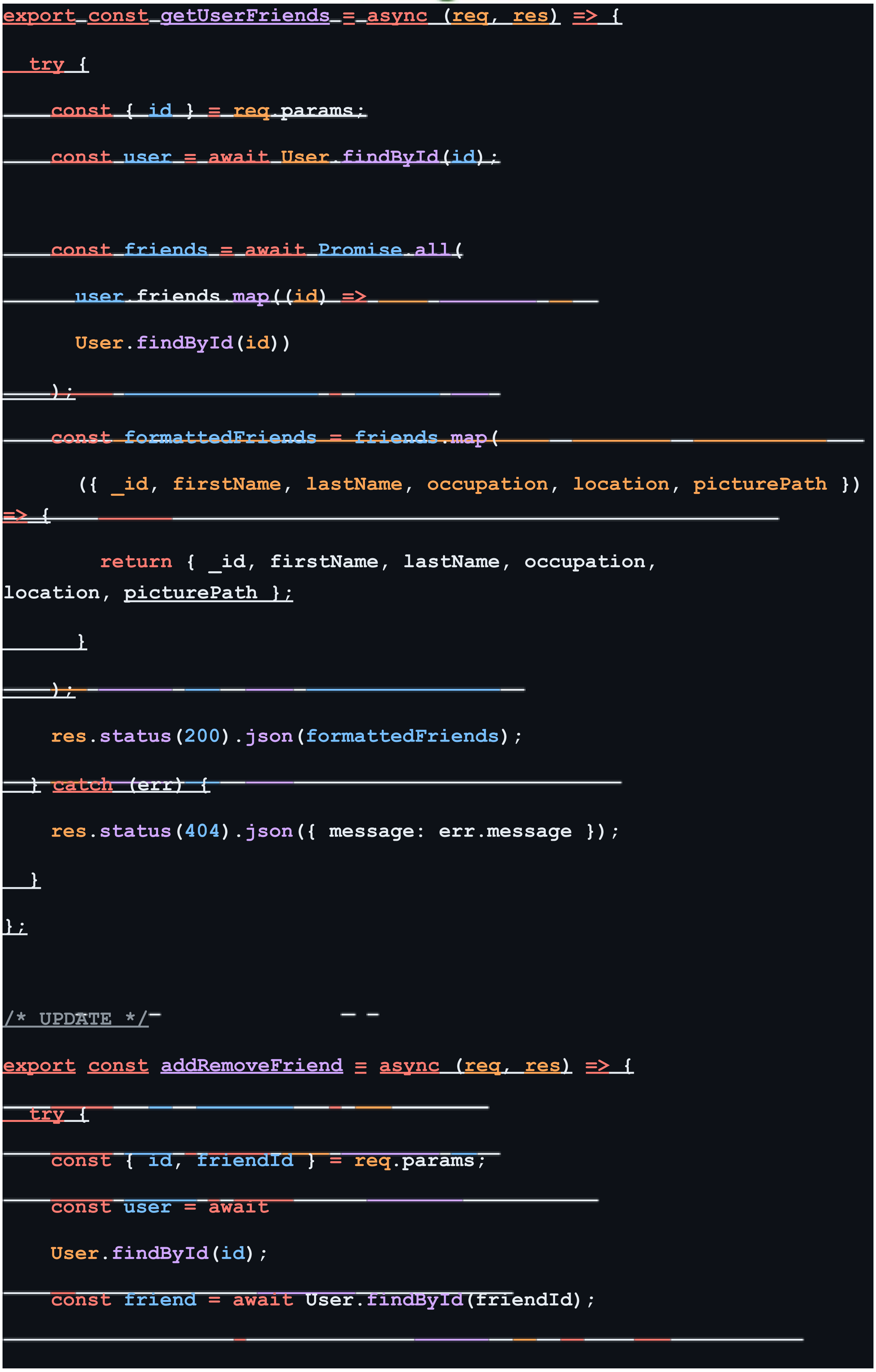


```
    const updatedPost = await Post.findByIdAndUpdate(
      id,
      { likes: post.likes },
      { new: true }
    );

    res.status(200).json(updatedPost);
  } catch (err) {
    res.status(404).json({ message: err.message });
  }
};
```

Users.js:



```
import User from "../models/User.js";


/* READ */
export const getUser = async (req, res) => {
  try {
    const { id } = req.params;
    const user = await User.findById(id);
    res.status(200).json(user);
  } catch (err) {
    res.status(404).json({ message: err.message });
  }
};
```

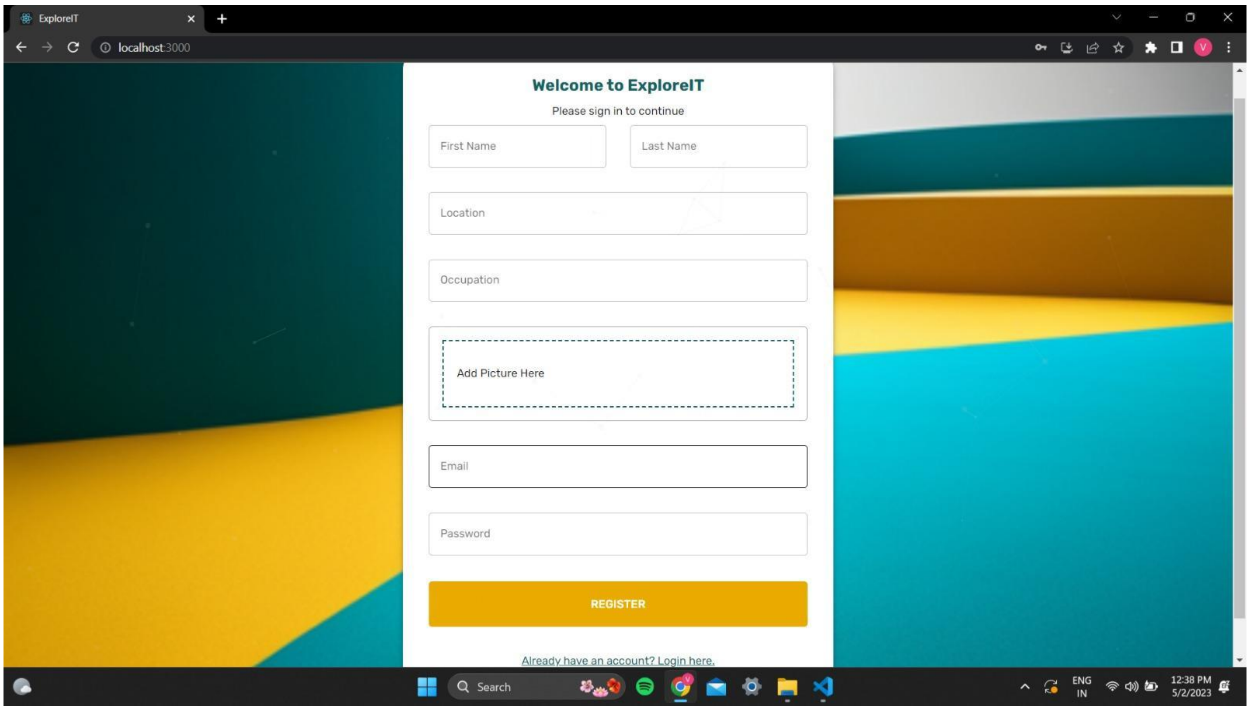


```
export const getUserFriends = async (req, res) => {
  try {
    const { id } = req.params;
    const user = await User.findById(id);

    const friends = await Promise.all(
      user.friends.map((id) =>
      User.findById(id))
    );
    const formattedFriends = friends.map(
      ({ _id, firstName, lastName, occupation, location, picturePath })
=> {
        return { _id, firstName, lastName, occupation,
location, picturePath };
      }
    );
    res.status(200).json(formattedFriends);
  } catch (err) {
    res.status(404).json({ message: err.message });
  }
};

/* UPDATE */
export const addRemoveFriend = async (req, res) => {
  try {
    const { id, friendId } = req.params;
    const user = await
    User.findById(id);
    const friend = await User.findById(friendId);

    if (user.friends.includes(friendId)) {
      user.friends = user.friends.filter((id) => id !== friendId);
      friend.friends = friend.friends.filter((id) => id !== id);
    } else {
      user.friends.push(friendId);
      friend.friends.push(id);
    }
    await user.save();
    await
    friend.save();

    const friends = await Promise.all(
      user.friends.map((id) =>
      User.findById(id))
    );
    const formattedFriends = friends.map(
      ({ _id, firstName, lastName, occupation, location, picturePath })
=> {
        return { _id, firstName, lastName, occupation,
location, picturePath };
      }
    );

    res.status(200).json(formattedFriends);
  } catch (err) {
    res.status(404).json({ message: err.message });
  }
};
```

WORKING

1. User will open the site and will see the Login Page.
2. On the Login page, we have two options: signup and login.
3. Here a new user can sign up and can create an account, Existing Users can login by entering the desired details.
4. After login, the customer will be directed to the main landing page of the website, where we have all our services.

Sign Up Page:



Welcome to ExploreIT

Please sign in to continue

First Name

Last Name

Location

Occupation

Add Picture Here

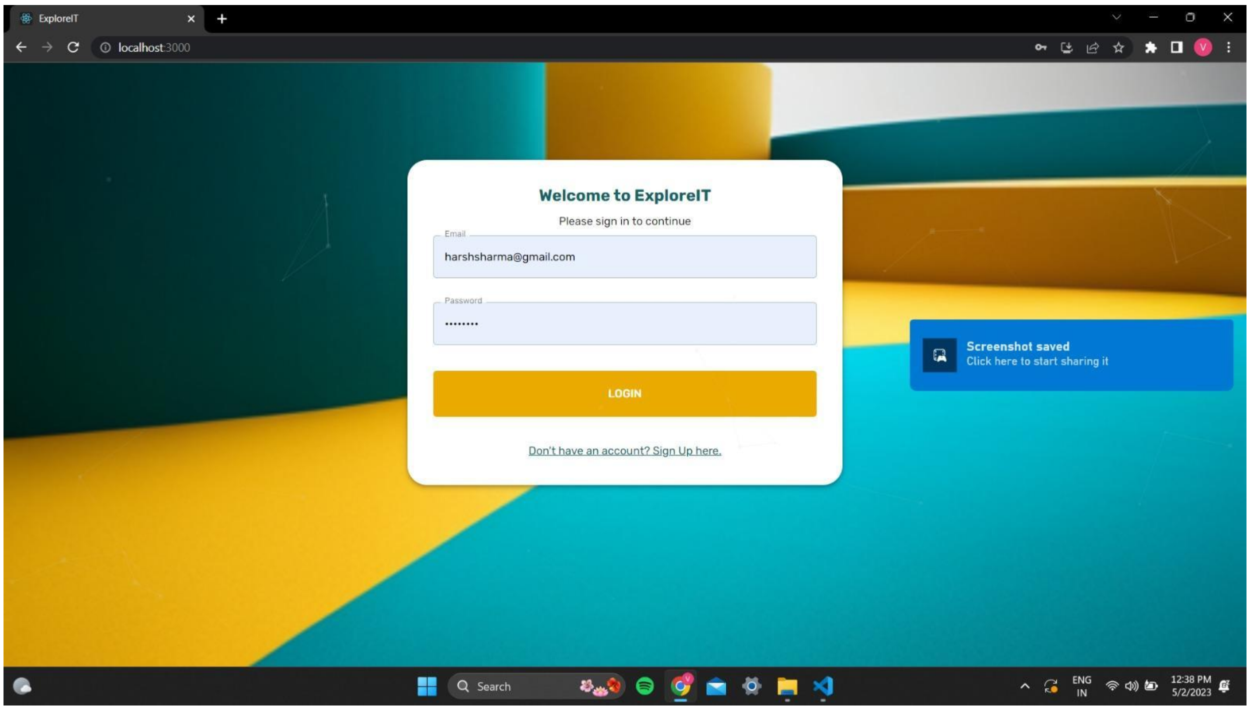
Email

Password

REGISTER

Already have an account? Login here.

Sign In Page:



Welcome to ExploreIT

Please sign in to continue

Email: harshsharma@gmail.com

Password: ••••••••

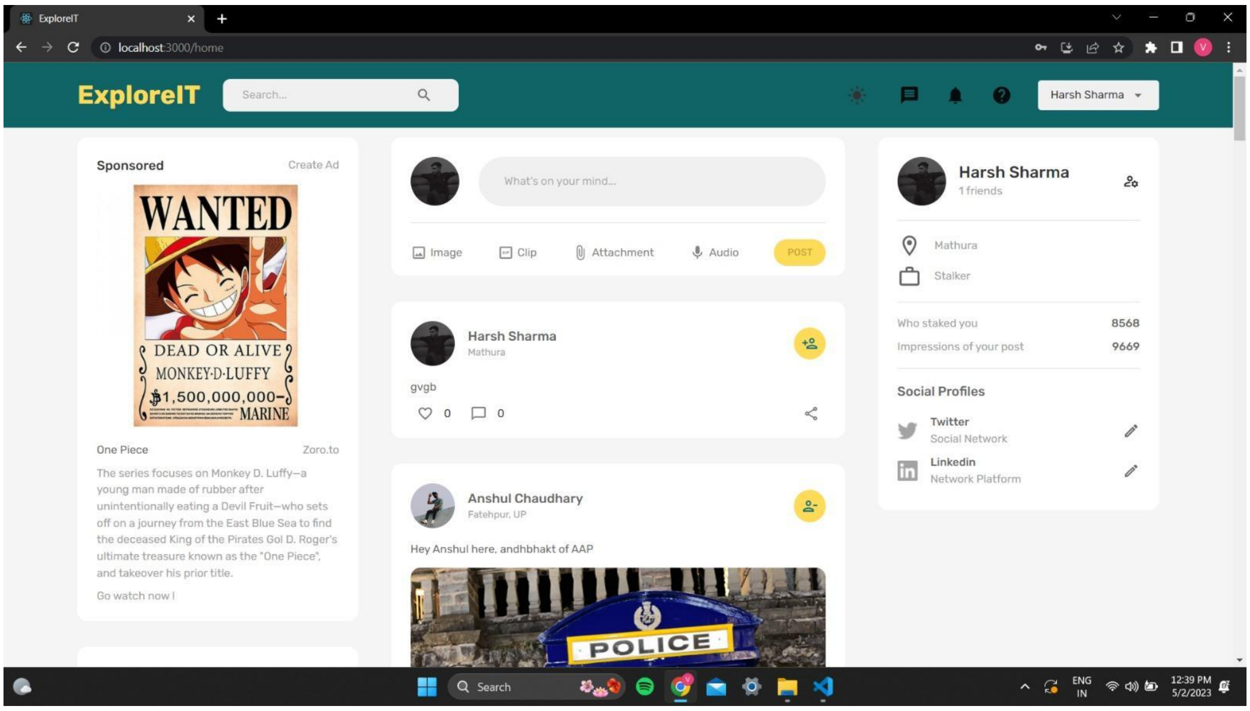
LOGIN

Don't have an account? Sign Up here.

Screenshot saved

Click here to start sharing it

Home Page:-


ExploreIT

localhost:3000/home

ExploreIT Search... Harsh Sharma

Sponsored Create Ad

One Piece Zoro.to

The series focuses on Monkey D. Luffy—a young man made of rubber after unintentionally eating a Devil Fruit—who sets off on a journey from the East Blue Sea to find the deceased King of the Pirates Gol D. Roger's ultimate treasure known as the "One Piece", and takeover his prior title.

Go watch now !

What's on your mind...

Image Clip Attachment Audio POST

Harsh Sharma
Mathura

gvgb

0 0

Anshul Chaudhary
Fatehpur, UP

Hey Anshul here, andhbhakt of AAP

Harsh Sharma
1 friends

Mathura

Stalker

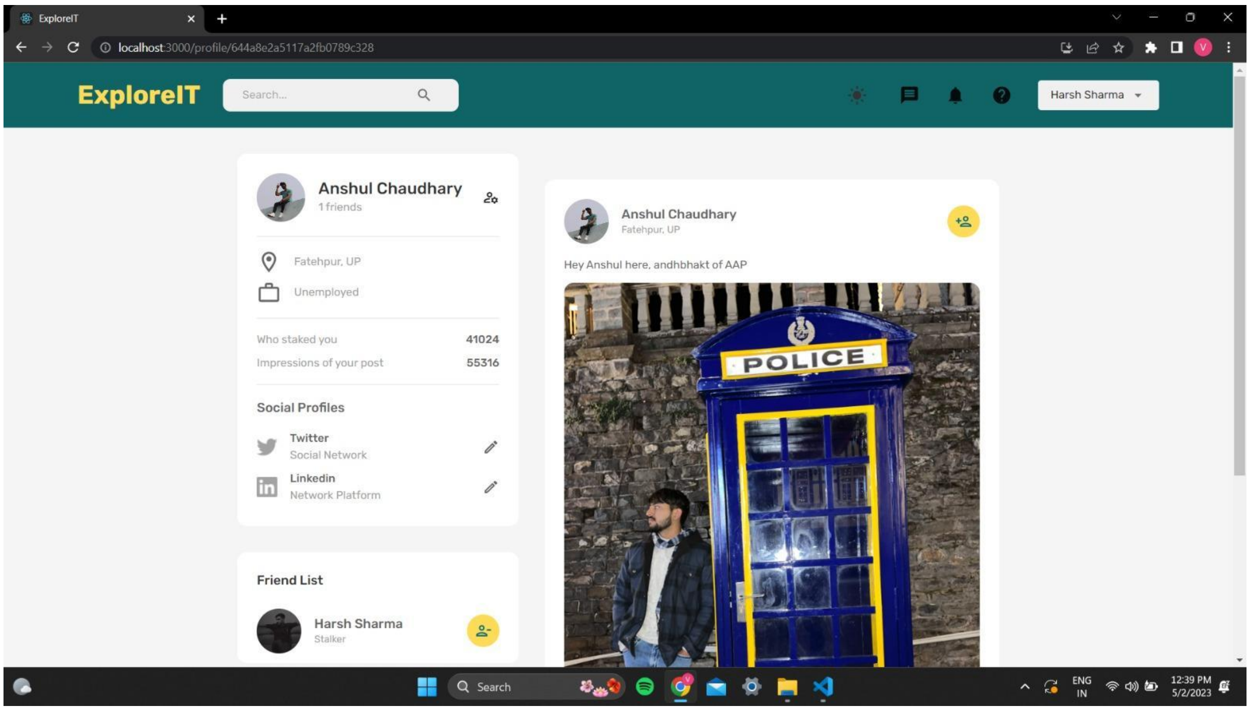
Who staked you	8568
Impressions of your post	9669

Social Profiles

Twitter
Social Network

Linkedin
Network Platform

Profile Page:-



ExploreIT

localhost:3000/profile/644a8e2a5117a2fb0789c328

ExploreIT

Search...

Harsh Sharma

Anshul Chaudhary
1 friends

Fatehpur, UP

Unemployed

Who staked you	41024
Impressions of your post	55316

Social Profiles

Twitter
Social Network

Linkedin
Network Platform

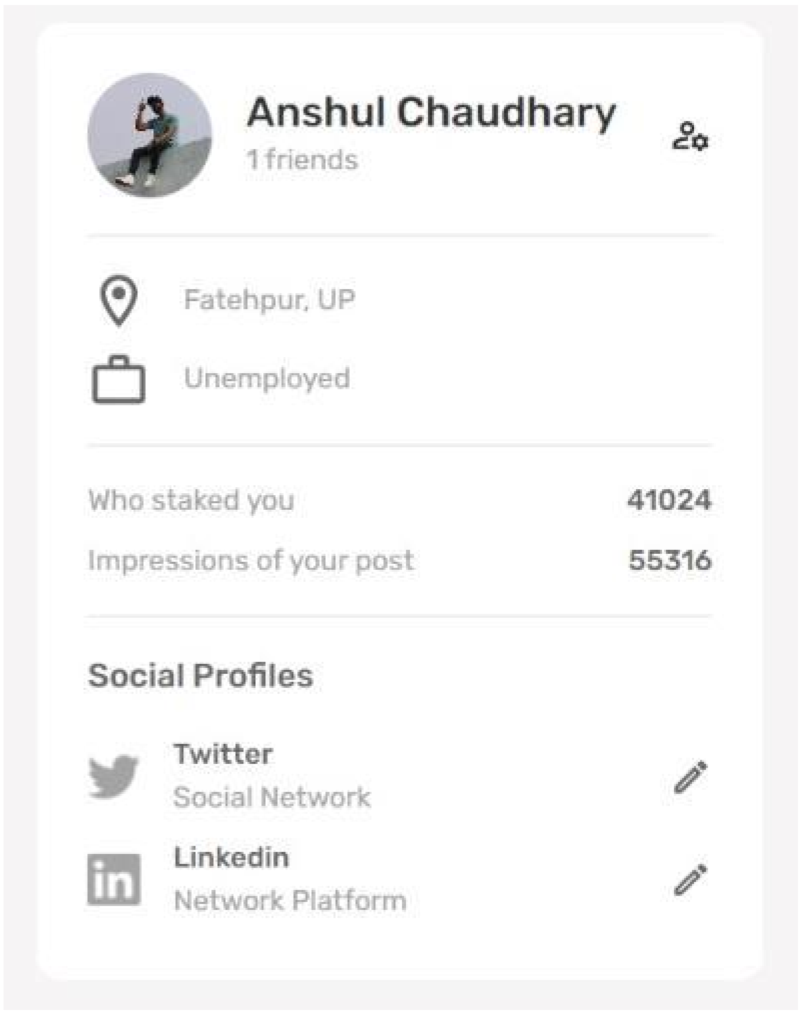
Friend List

Harsh Sharma
Stalker

Anshul Chaudhary
Fatehpur, UP

Hey Anshul here, andhbhakt of AAP

UserWidget:-



Anshul Chaudhary
1 friends

Fatehpur, UP
Unemployed

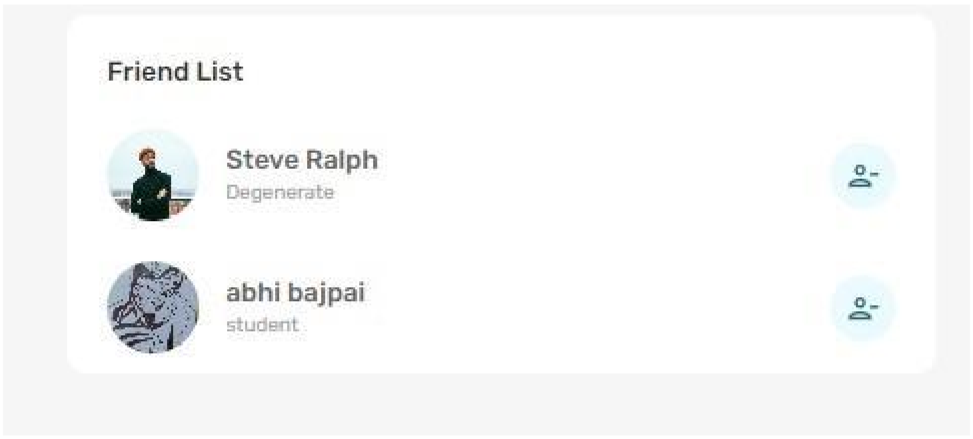
Who staked you 41024
Impressions of your post 55316

Social Profiles

Twitter
Social Network

Linkedin
Network Platform

Friends List:-

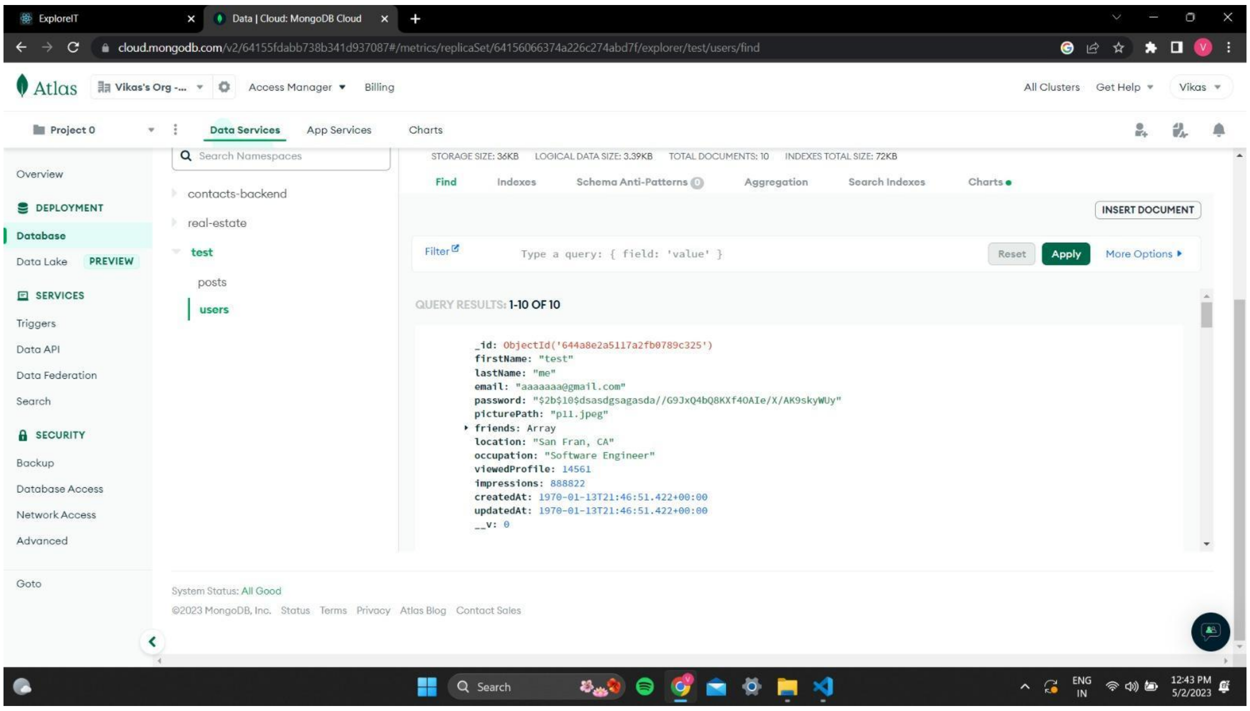


Friend List

Steve Ralph
Degenerate

abhi bajpai
student

Users Database:-



cloud.mongodb.com/v2/64155fdabb738b341d937087#/metrics/replicaSet/64156066374a226c274abd7f/explorer/test/users/find

contacts-backend
real-estate
test
posts
users

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 3.39KB TOTAL DOCUMENTS: 10 INDEXES TOTAL SIZE: 72KB

Filter Type a query: { field: 'value' }

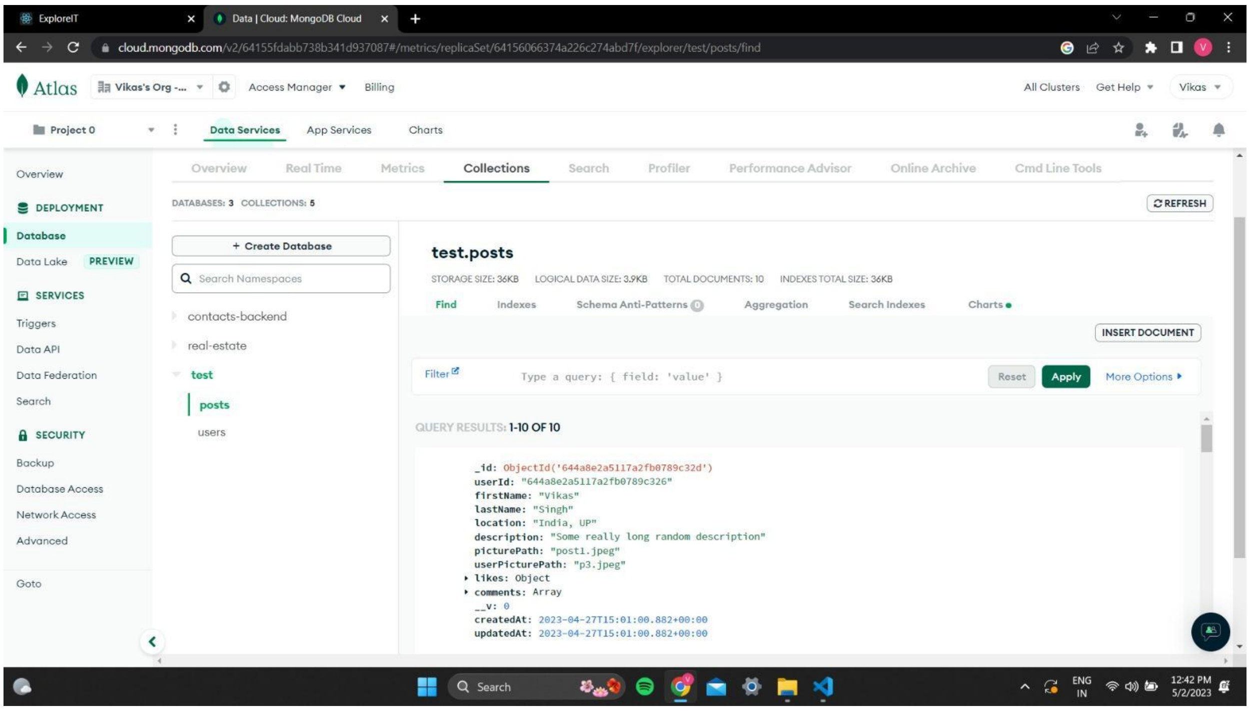
QUERY RESULTS: 1-10 OF 10

```
_id: ObjectId('644a8e2a5117a2fb0789c325')
firstName: "test"
lastName: "me"
email: "aaaaaaa@gmail.com"
password: "$2b$10$dsasdgsagasda//G9JxQ4bQ8KXf4OAIe/X/AK9skyWUy"
picturePath: "p11.jpeg"
▸ friends: Array
location: "San Fran, CA"
occupation: "Software Engineer"
viewedProfile: 14561
impressions: 888822
createdAt: 1970-01-13T21:46:51.422+00:00
updatedAt: 1970-01-13T21:46:51.422+00:00
__v: 0
```

System Status: All Good
©2023 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

Search ENG IN 12:43 PM 5/2/2023

Posts Database:-



cloud.mongodb.com/v2/64155fdabb738b341d937087#/metrics/replicaSet/64156066374a226c274abd7f/explorer/test/posts/find

DATABASES: 3 COLLECTIONS: 5

contacts-backend
real-estate
test
posts
users

test.posts

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 3.9KB TOTAL DOCUMENTS: 10 INDEXES TOTAL SIZE: 36KB

Filter Type a query: { field: 'value' }

QUERY RESULTS: 1-10 OF 10

```
_id: ObjectId('644a8e2a5117a2fb0789c32d')
userId: "644a8e2a5117a2fb0789c326"
firstName: "Vikas"
lastName: "Singh"
location: "India, UP"
description: "Some really long random description"
picturePath: "post1.jpeg"
userPicturePath: "p3.jpeg"
▸ likes: Object
▸ comments: Array
__v: 0
createdAt: 2023-04-27T15:01:00.882+00:00
updatedAt: 2023-04-27T15:01:00.882+00:00
```

Search ENG IN 12:42 PM 5/2/2023

CONCLUSION AND FUTURE WORK

Conclusion:

As per the attempt of the project is to create a safe and fun platform for the artists to create, upload and interact with other artists of all forms. The artists, photographers, creators and collaborators can interact with, get along with and promote their works on our platform. They can create all kinds of content and get amazing deals while working with the other creators and artists.

The data of the users and the posts are saved in the database. We used MongoDB database for saving the data of the users and the posts they create, upload and promote.

The users can register using our sign up page and once registered, the data is saved in the backend. The data will be fetched and used again when log in using the saved credentials from the backend.

Once landed on the home page, they can create a post by uploading a picture.
We have also created a theme changing option: which included light and dark themes.

Future work: -

The scope of the project is to help more and more creators promote their works and get ahead in their career by increasing their reach by collaborating with the other artists and creators and providing the audience with the original exclusive content for entertainment.

Their data will be used by using multiple algorithms to create a better environment and a better platform for the collaborators to engage with the audience.

The plan is to add more features for the users to chat with others on the network, create an even easier interface to make better and more effective content.

Github Link:

https://github.com/Vikas-Singh-10/FullStack-Project-Social-Media