Blink - The Social Media Platform

Introduction: -

Blink is a social media application designed to foster a community where users can share their thoughts, experiences, and connect with like-minded individuals. The app enables users to upload pictures, like and comment on posts, manage their profiles, and build connections with others.

Features: -

- **User Authentication and Login:** Secure user registration and login functionality.
- **Picture Upload and Sharing:** Users can upload and share their pictures with the community.
- **Like and Comment:** Users can interact with posts by liking and commenting.
- **User Profile Management:** Users can add their location and occupation details and manage their profiles.
- **Friend Management:** Users can add and remove friends to build their network.

Technologies Used: -

- **Frontend:** React.js
- **Backend:** Node.js, Express.js
- **Database:** MongoDB
- **Authentication:** JWT (JSON Web Token)
- **Styling:** CSS, Material-UI (or any other relevant styling framework)

Installation: -

Follow these steps to set up and run the application locally:

1. **Clone the Repository**

   ```
   git clone https://github.com/Kaysanshaikh/FutureIntern_FSD_05.git
   ```

2. **Setting Up the Client**

   Navigate to the client directory and install the necessary dependencies:

   ```
   cd client
   npm install
   npm start
   ```

3. **Setting Up the Server**

   Navigate to the server directory and install the necessary dependencies:

   ```
   cd server
   npm install
   npm start
   ```

API Endpoints: -

- **POST /api/auth/register:** Register a new user.
- **POST /api/auth/login:** Authenticate and log in a user.
- **GET /api/posts:** Retrieve all posts.
- **POST /api/posts:** Create a new post.
- **GET /api/posts/:id:** Retrieve a specific post by ID.
- **PUT /api/posts/:id:** Update an existing post by ID.
- **DELETE /api/posts/:id:** Delete a post by ID.
- **GET /api/users/:id:** Retrieve a specific user profile by ID.
- **PUT /api/users/:id:** Update a user profile by ID.
- **POST /api/friends/add:** Add a friend.
- **POST /api/friends/remove:** Remove a friend.

Usage: -

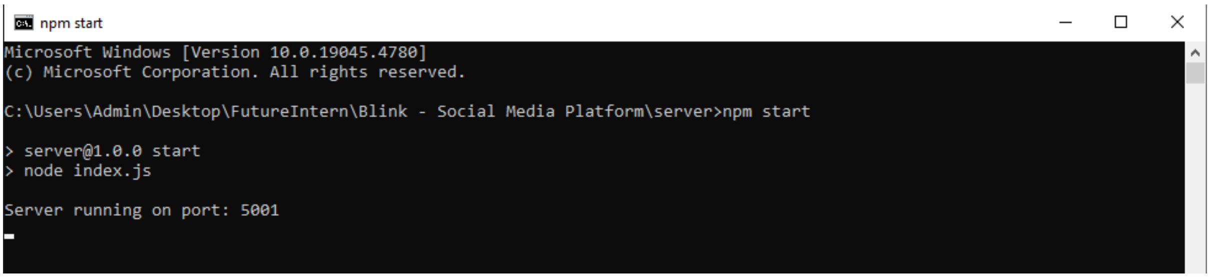
1. **Register and Login:** Use the authentication endpoints to register and log in to the application.
2. **Create and Share Posts:** Navigate to the "Create Post" page to upload and share pictures.
3. **Interact with Posts:** Like and comment on posts from the feed.
4. **Manage Profile:** Update your profile details and manage your account settings.
5. **Manage Friends:** Add or remove friends to build your network.

Security Considerations: -

- **Authentication:** Secure user registration and login using JWT.
- **Authorization:** Implemented role-based access control for protected routes and actions.
- **Error Handling:** Proper error handling to ensure a smooth user experience and avoid exposing sensitive information.

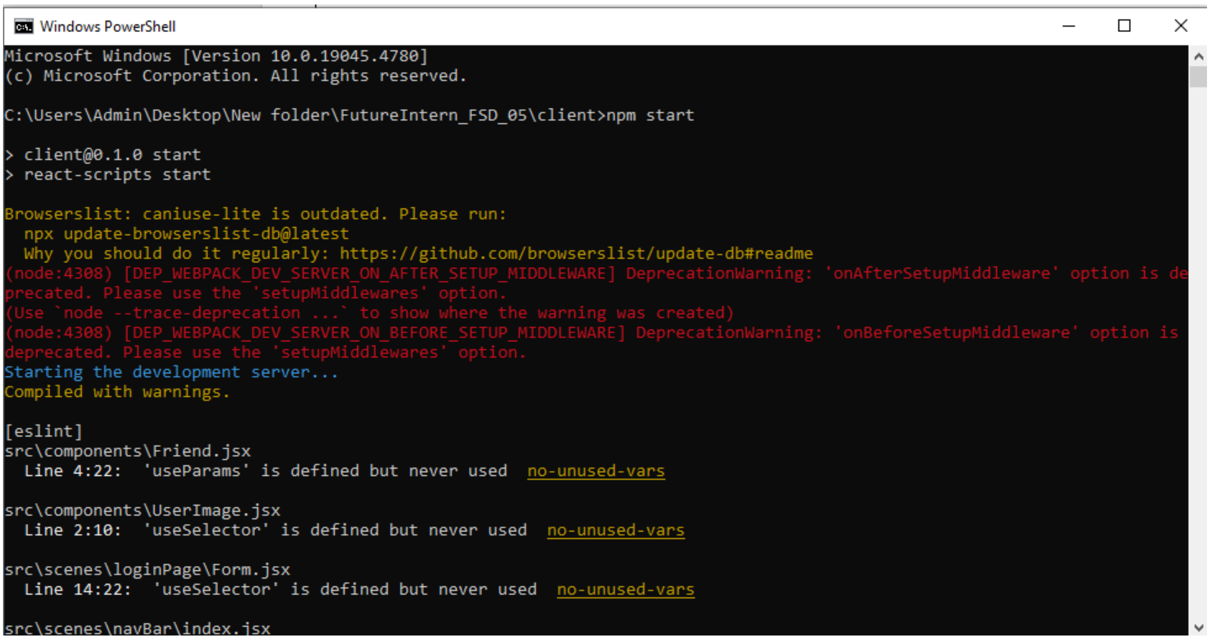
Screenshots: -

server:



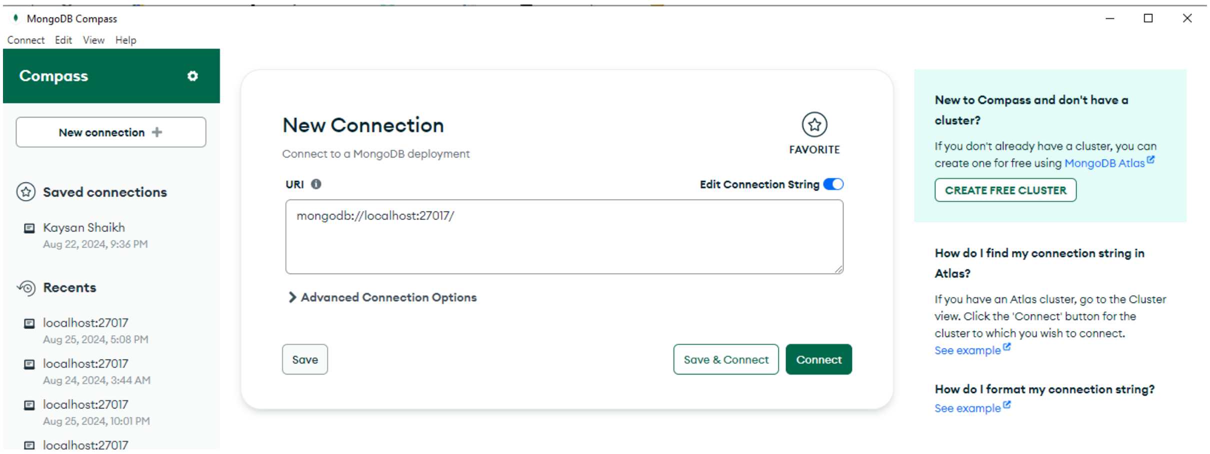
```
Microsoft Windows [Version 10.0.19045.4780]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Desktop\FutureIntern\Blink - Social Media Platform\server>npm start

> server@1.0.0 start
> node index.js

Server running on port: 5001
```

client:



```
Microsoft Windows [Version 10.0.19045.4780]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Desktop\New folder\FutureIntern_FSD_05\client>npm start

> client@0.1.0 start
> react-scripts start

Browserslist: caniuse-lite is outdated. Please run:
  npx update-browserslist-db@latest
  Why you should do it regularly: https://github.com/browserslist/update-db#readme
(node:4308) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is de
precated. Please use the 'setupMiddlewares' option.
(Use `node --trace-deprecation ...` to show where the warning was created)
(node:4308) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is
deprecated. Please use the 'setupMiddlewares' option.
Starting the development server...
Compiled with warnings.

[eslint]
src\components\Friend.jsx
  Line 4:22:  'useParams' is defined but never used  no-unused-vars

src\components\UserImage.jsx
  Line 2:10:  'useSelector' is defined but never used  no-unused-vars

src\scenes\loginPage\Form.jsx
  Line 14:22:  'useSelector' is defined but never used  no-unused-vars

src\scenes\navBar\index.jsx
```

MongoDB:



Kaysan Shaikh
Aug 22, 2024, 9:36 PM

localhost:27017
Aug 25, 2024, 5:08 PM

localhost:27017
Aug 24, 2024, 3:44 AM

localhost:27017
Aug 25, 2024, 10:01 PM

localhost:27017

New Connection

Connect to a MongoDB deployment

URI: mongodb://localhost:27017/

> Advanced Connection Options

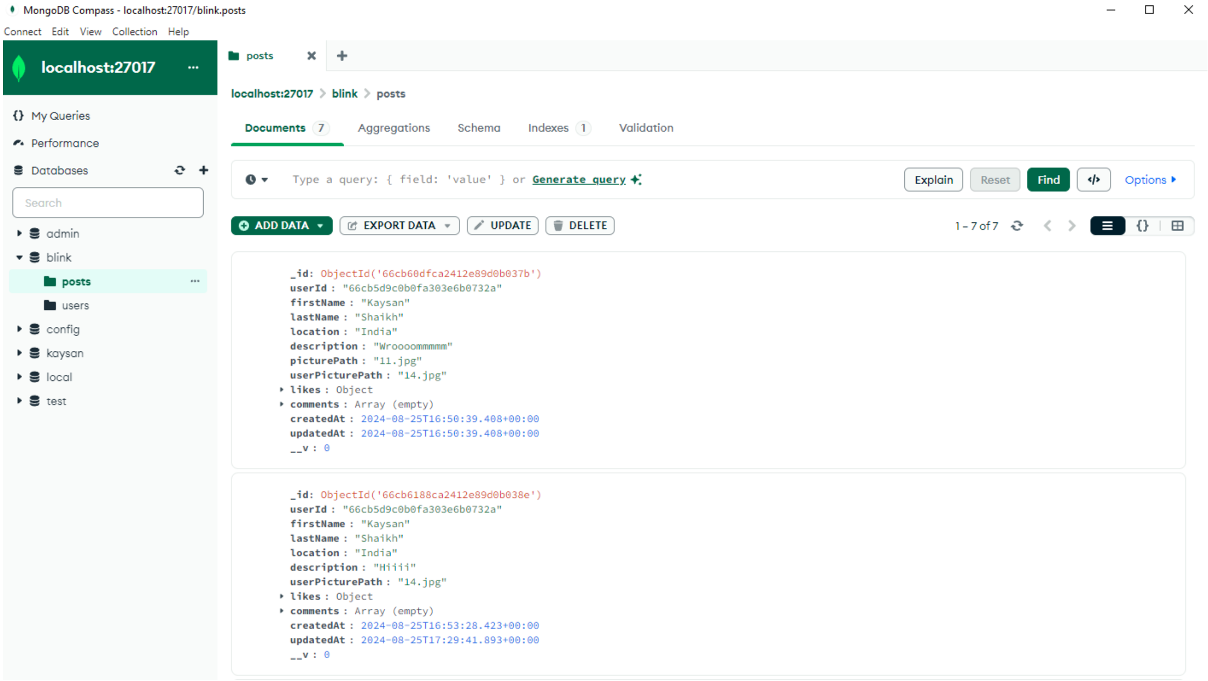
New to Compass and don't have a cluster?

If you don't already have a cluster, you can create one for free using MongoDB Atlas

How do I find my connection string in Atlas?

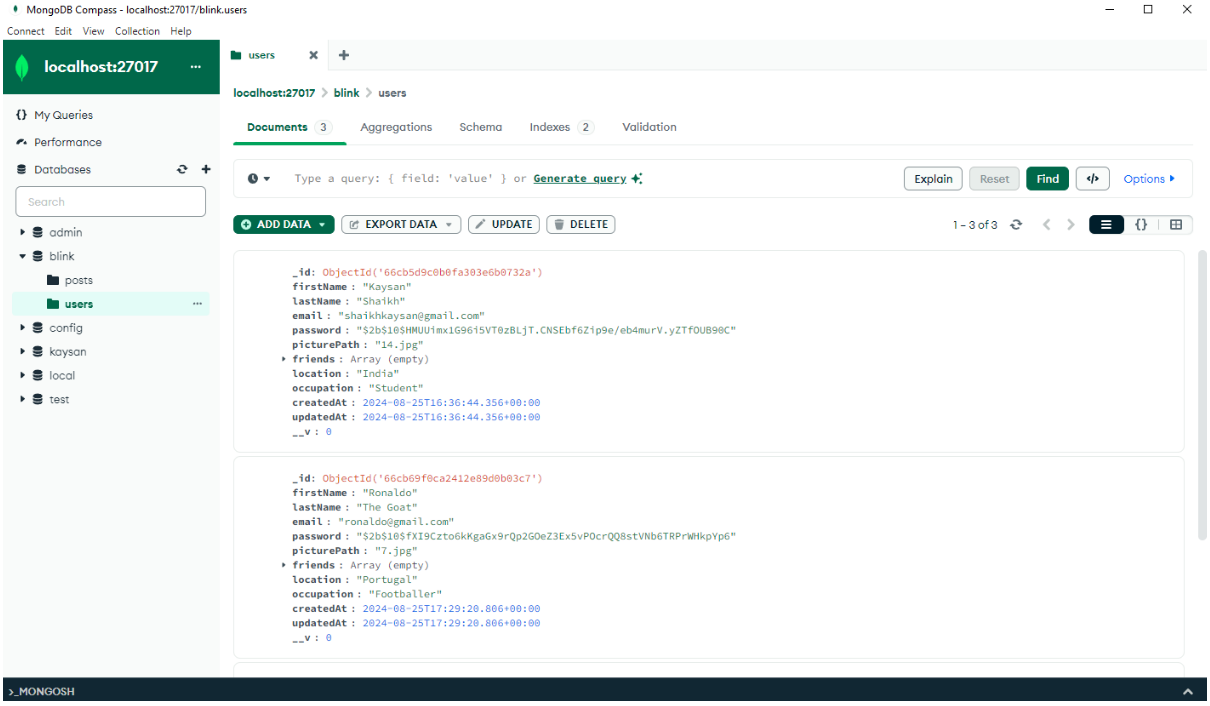
If you have an Atlas cluster, go to the Cluster view. Click the 'Connect' button for the cluster to which you wish to connect.

How do I format my connection string?



```
_id: ObjectId('66cb60dfca2412e89d0b037b')
userId : "66cb5d9c0b0fa303e6b0732a"
firstName : "Kaysan"
lastName : "Shaikh"
location : "India"
description : "Wroooommmmm"
picturePath : "11.jpg"
userPicturePath : "14.jpg"
likes : Object
comments : Array (empty)
createdAt : 2024-08-25T16:50:39.408+00:00
updatedAt : 2024-08-25T16:50:39.408+00:00
__v : 0
```

```
_id: ObjectId('66cb6188ca2412e89d0b038e')
userId : "66cb5d9c0b0fa303e6b0732a"
firstName : "Kaysan"
lastName : "Shaikh"
location : "India"
description : "Hiiii"
userPicturePath : "14.jpg"
likes : Object
comments : Array (empty)
createdAt : 2024-08-25T16:53:28.423+00:00
updatedAt : 2024-08-25T17:29:41.893+00:00
__v : 0
```

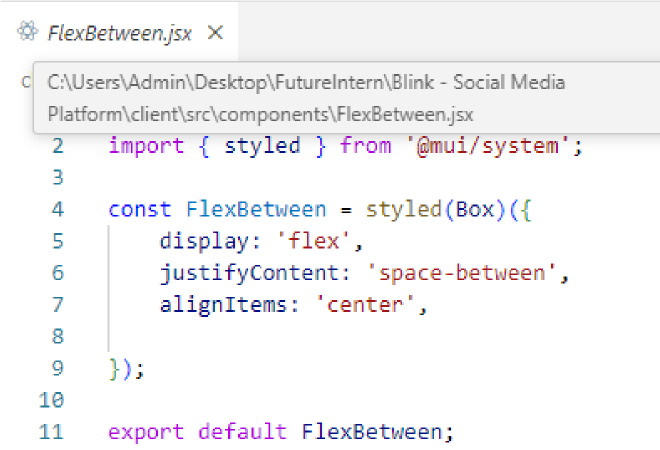


```
_id: ObjectId('66cb5d9c0b0fa303e6b0732a')
firstName : "Kaysan"
lastName : "Shaikh"
email : "shaikhkaysan@gmail.com"
password : "$2b$10$HMUUimx1G96i5VT0zBLjT.CNSEbf6Zip9e/eb4murV.yZTfOUB90C"
picturePath : "14.jpg"
friends : Array (empty)
location : "India"
occupation : "Student"
createdAt : 2024-08-25T16:36:44.356+00:00
updatedAt : 2024-08-25T16:36:44.356+00:00
__v : 0
```

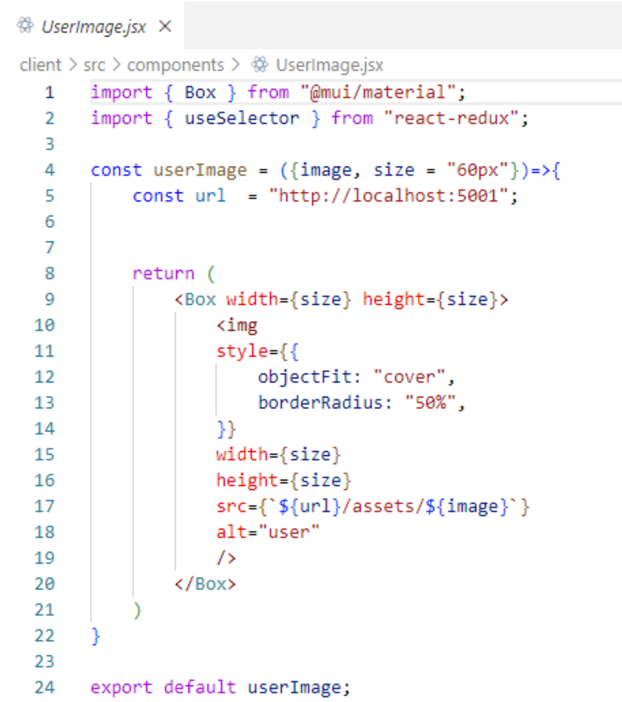
```
_id: ObjectId('66cb69f0ca2412e89d0b03c7')
firstName : "Ronaldo"
lastName : "The Goat"
email : "ronaldo@gmail.com"
password : "$2b$10$fXI9Czto6kKgaGx9rQp2GOeZ3Ex5vPOcrQQ8stVNb6TRPrWHkpYp6"
picturePath : "7.jpg"
friends : Array (empty)
location : "Portugal"
occupation : "Footballer"
createdAt : 2024-08-25T17:29:20.806+00:00
updatedAt : 2024-08-25T17:29:20.806+00:00
__v : 0
```

Code: -

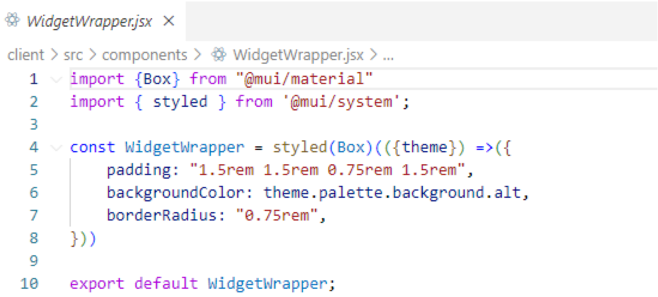
client:


FlexBetween.jsx ×
C:\Users\Admin\Desktop\FutureIntern\Blink - Social Media Platform\client\src\components\FlexBetween.jsx

```jsx
import { styled } from '@mui/system';

const FlexBetween = styled(Box)({
    display: 'flex',
    justifyContent: 'space-between',
    alignItems: 'center',

});

export default FlexBetween;
```

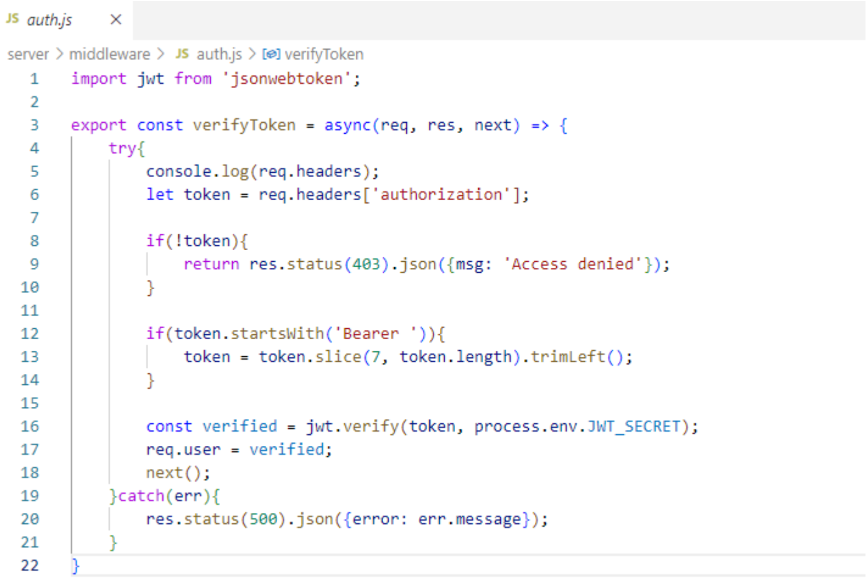


```jsx
import { Box } from "@mui/material";
import { useSelector } from "react-redux";

const userImage = ({image, size = "60px"})=>{
    const url  = "http://localhost:5001";


    return (
        <Box width={size} height={size}>
            <img
            style={{
                objectFit: "cover",
                borderRadius: "50%",
            }}
            width={size}
            height={size}
            src={`${url}/assets/${image}`}
            alt="user"
            />
        </Box>
    )
}

export default userImage;
```

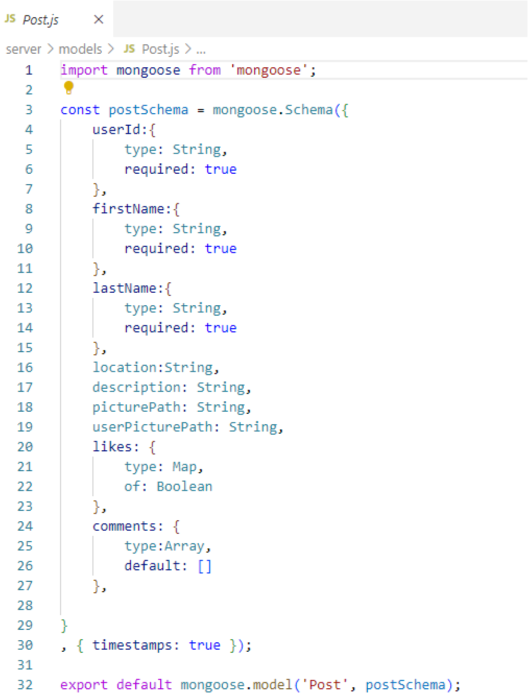


```jsx
import {Box} from "@mui/material"
import { styled } from '@mui/system';

const WidgetWrapper = styled(Box)(({theme}) =>({
    padding: "1.5rem 1.5rem 0.75rem 1.5rem",
    backgroundColor: theme.palette.background.alt,
    borderRadius: "0.75rem",
}))

export default WidgetWrapper;
```

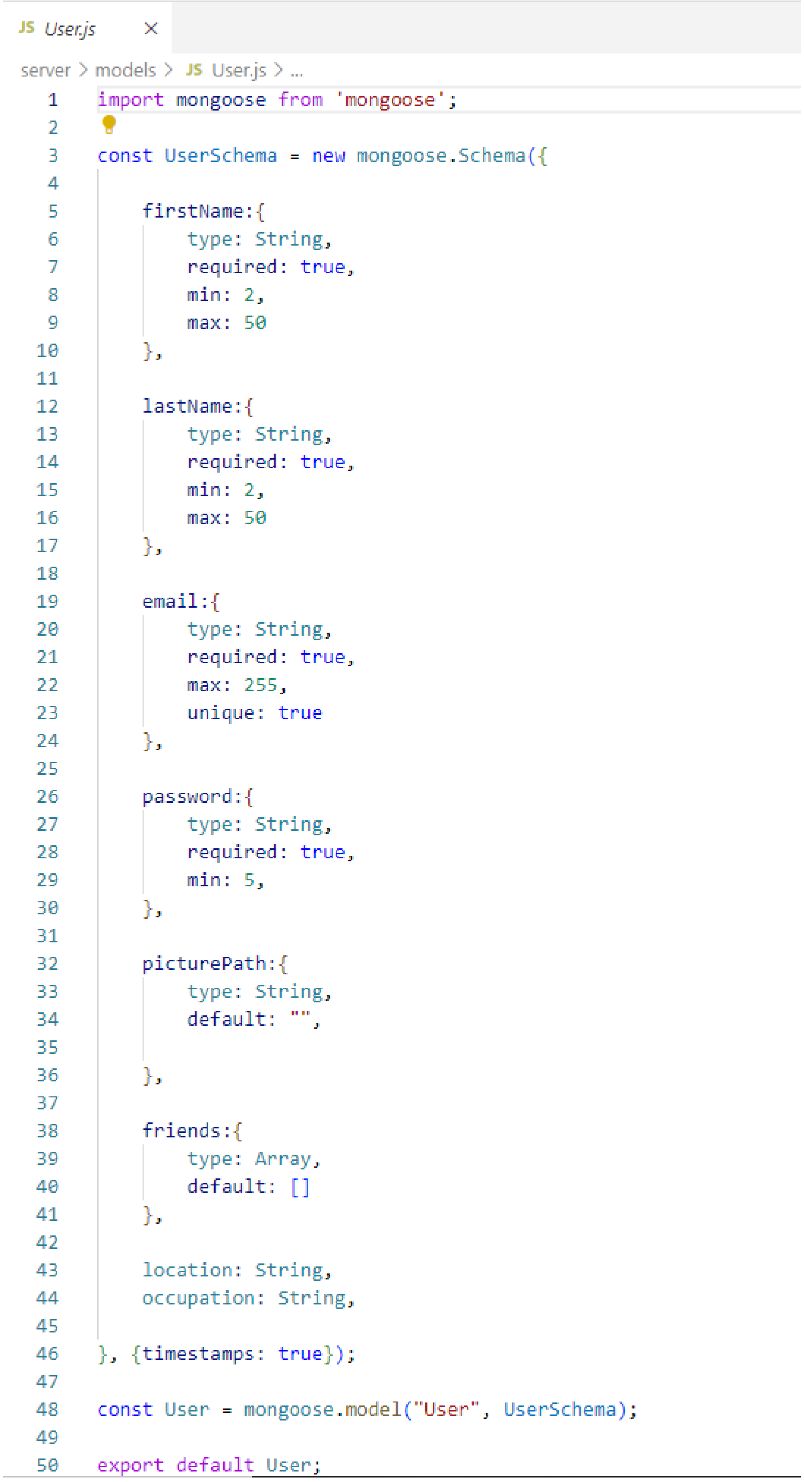
server: (Backend)



```js
import jwt from 'jsonwebtoken';

export const verifyToken = async(req, res, next) => {
    try{
        console.log(req.headers);
        let token = req.headers['authorization'];

        if(!token){
            return res.status(403).json({msg: 'Access denied'});
        }

        if(token.startsWith('Bearer ')){
            token = token.slice(7, token.length).trimLeft();
        }

        const verified = jwt.verify(token, process.env.JWT_SECRET);
        req.user = verified;
        next();
    }catch(err){
        res.status(500).json({error: err.message});
    }
}
```

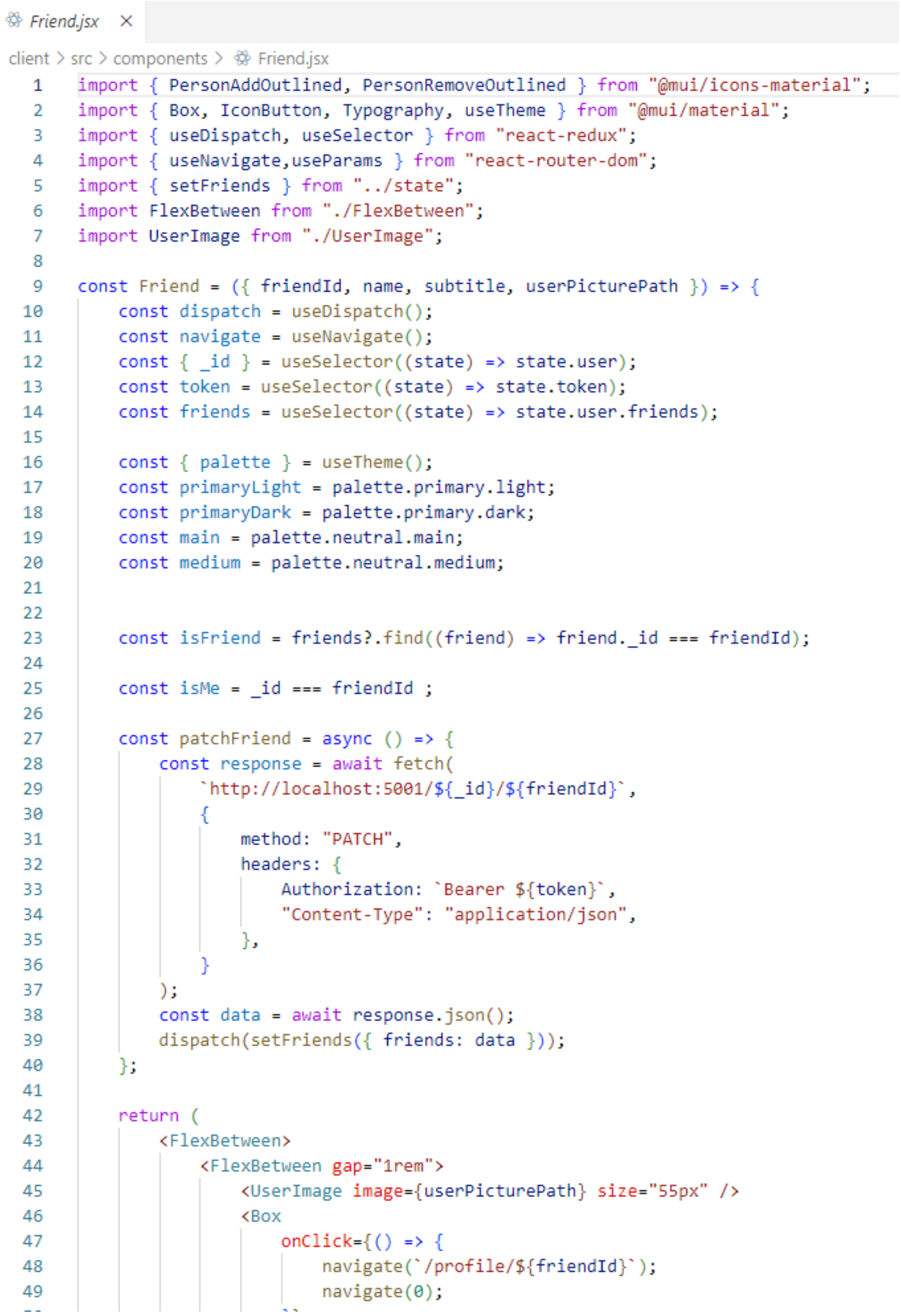


```js
import mongoose from 'mongoose';

const postSchema = mongoose.Schema({
    userId:{
        type: String,
        required: true
    },
    firstName:{
        type: String,
        required: true
    },
    lastName:{
        type: String,
        required: true
    },
    location:String,
    description: String,
    picturePath: String,
    userPicturePath: String,
    likes: {
        type: Map,
        of: Boolean
    },
    comments: {
        type:Array,
        default: []
    },

}
, { timestamps: true });

export default mongoose.model('Post', postSchema);
```

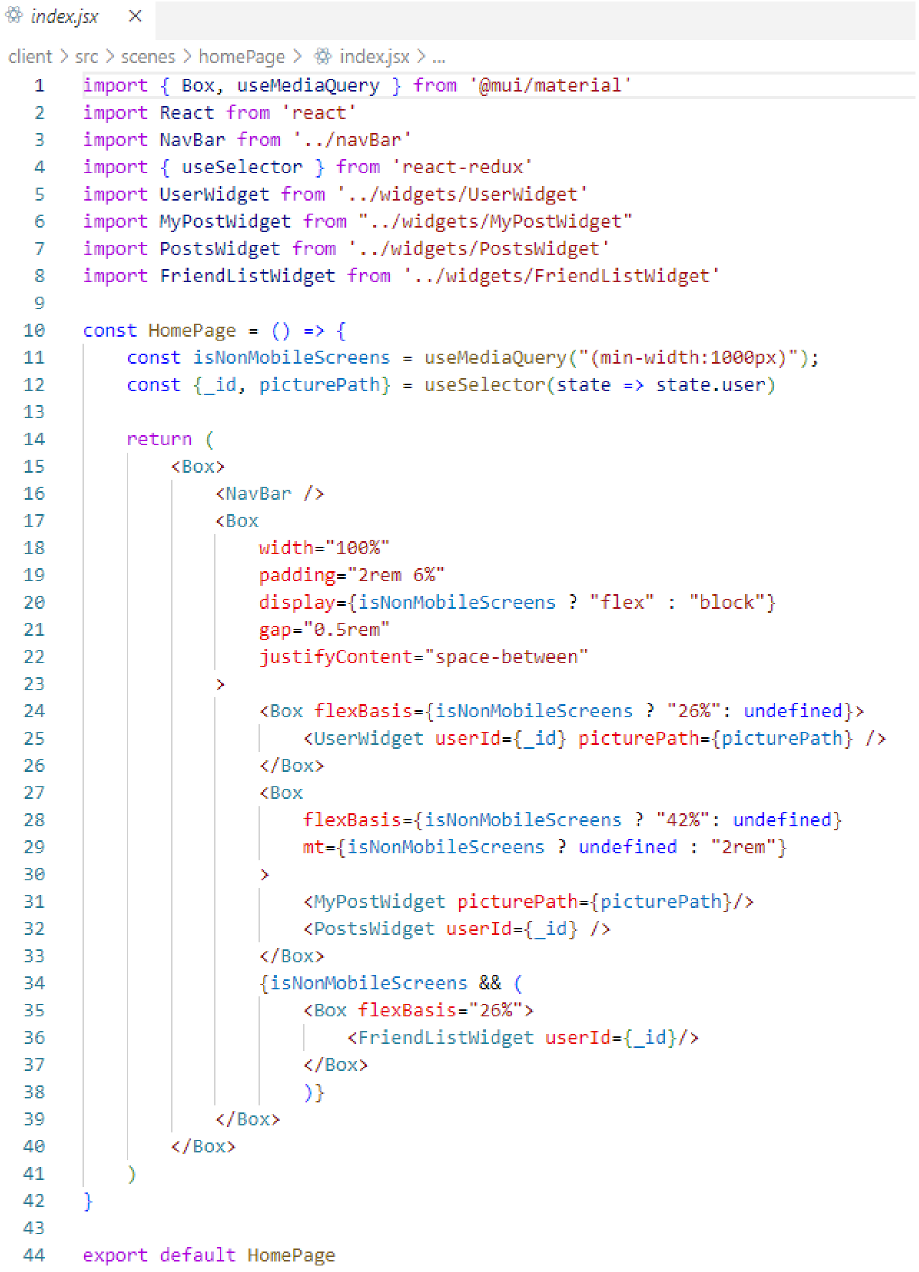


```
import mongoose from 'mongoose';

const UserSchema = new mongoose.Schema({

    firstName:{
        type: String,
        required: true,
        min: 2,
        max: 50
    },

    lastName:{
        type: String,
        required: true,
        min: 2,
        max: 50
    },

    email:{
        type: String,
        required: true,
        max: 255,
        unique: true
    },

    password:{
        type: String,
        required: true,
        min: 5,
    },

    picturePath:{
        type: String,
        default: "",

    },

    friends:{
        type: Array,
        default: []
    },

    location: String,
    occupation: String,

}, {timestamps: true});

const User = mongoose.model("User", UserSchema);

export default User;
```

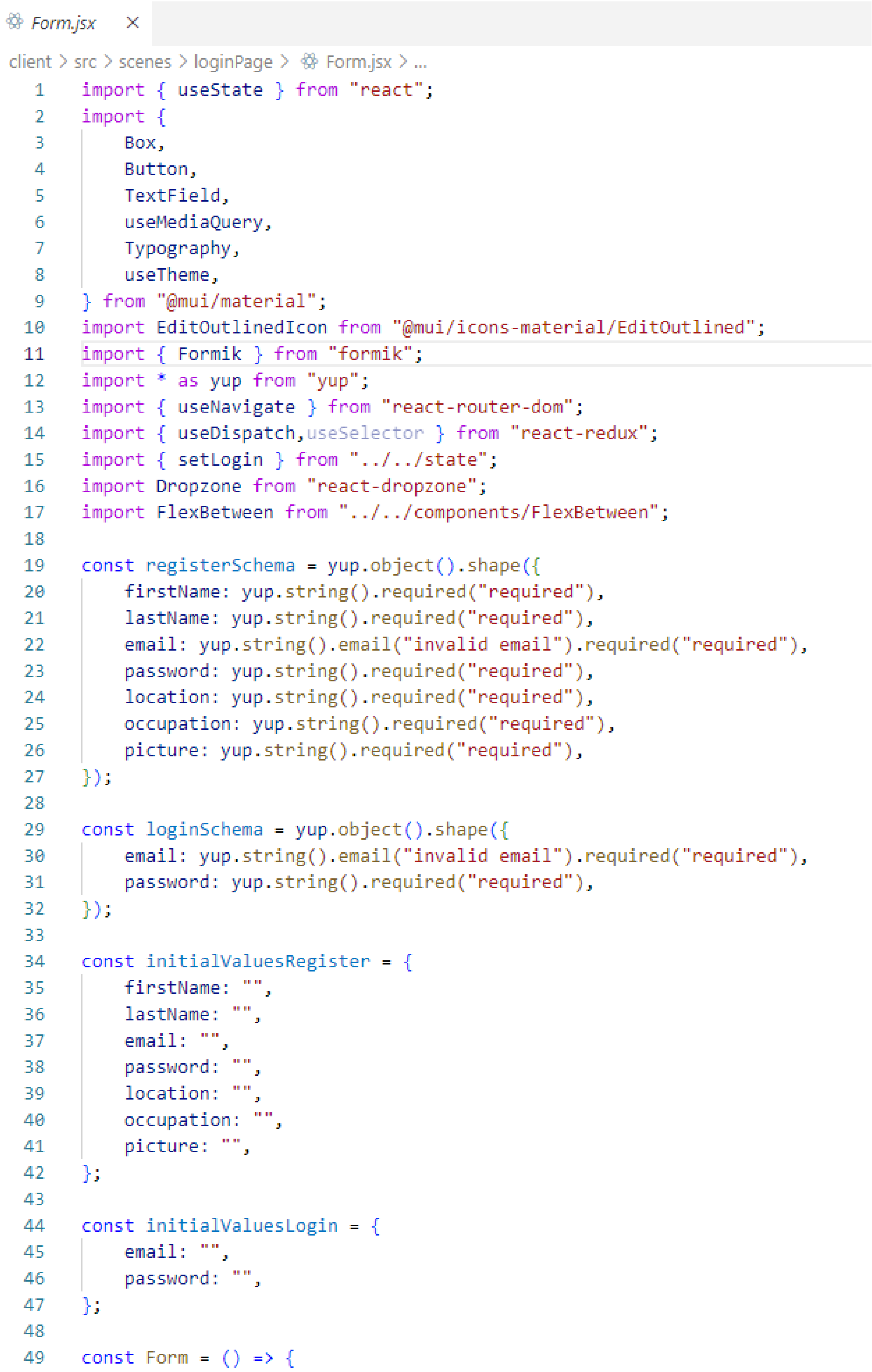
client: (Frontend)



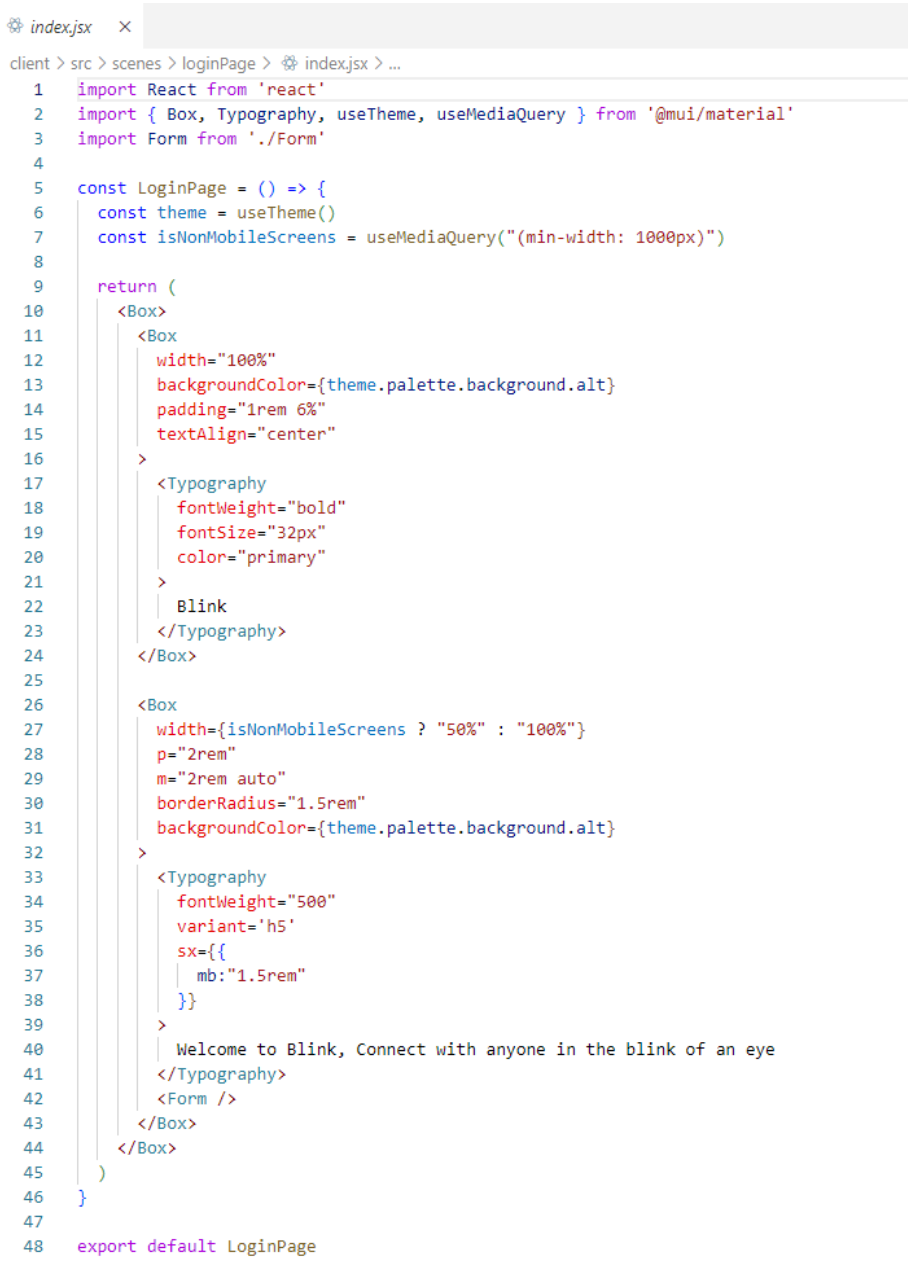
```jsx
import { PersonAddOutlined, PersonRemoveOutlined } from "@mui/icons-material";
import { Box, IconButton, Typography, useTheme } from "@mui/material";
import { useDispatch, useSelector } from "react-redux";
import { useNavigate,useParams } from "react-router-dom";
import { setFriends } from "../state";
import FlexBetween from "./FlexBetween";
import UserImage from "./UserImage";

const Friend = ({ friendId, name, subtitle, userPicturePath }) => {
    const dispatch = useDispatch();
    const navigate = useNavigate();
    const { _id } = useSelector((state) => state.user);
    const token = useSelector((state) => state.token);
    const friends = useSelector((state) => state.user.friends);

    const { palette } = useTheme();
    const primaryLight = palette.primary.light;
    const primaryDark = palette.primary.dark;
    const main = palette.neutral.main;
    const medium = palette.neutral.medium;


    const isFriend = friends?.find((friend) => friend._id === friendId);

    const isMe = _id === friendId ;

    const patchFriend = async () => {
        const response = await fetch(
            `http://localhost:5001/${_id}/${friendId}`,
            {
                method: "PATCH",
                headers: {
                    Authorization: `Bearer ${token}`,
                    "Content-Type": "application/json",
                },
            }
        );
        const data = await response.json();
        dispatch(setFriends({ friends: data }));
    };

    return (
        <FlexBetween>
            <FlexBetween gap="1rem">
                <UserImage image={userPicturePath} size="55px" />
                <Box
                    onClick={() => {
                        navigate(`/profile/${friendId}`);
                        navigate(0);
```



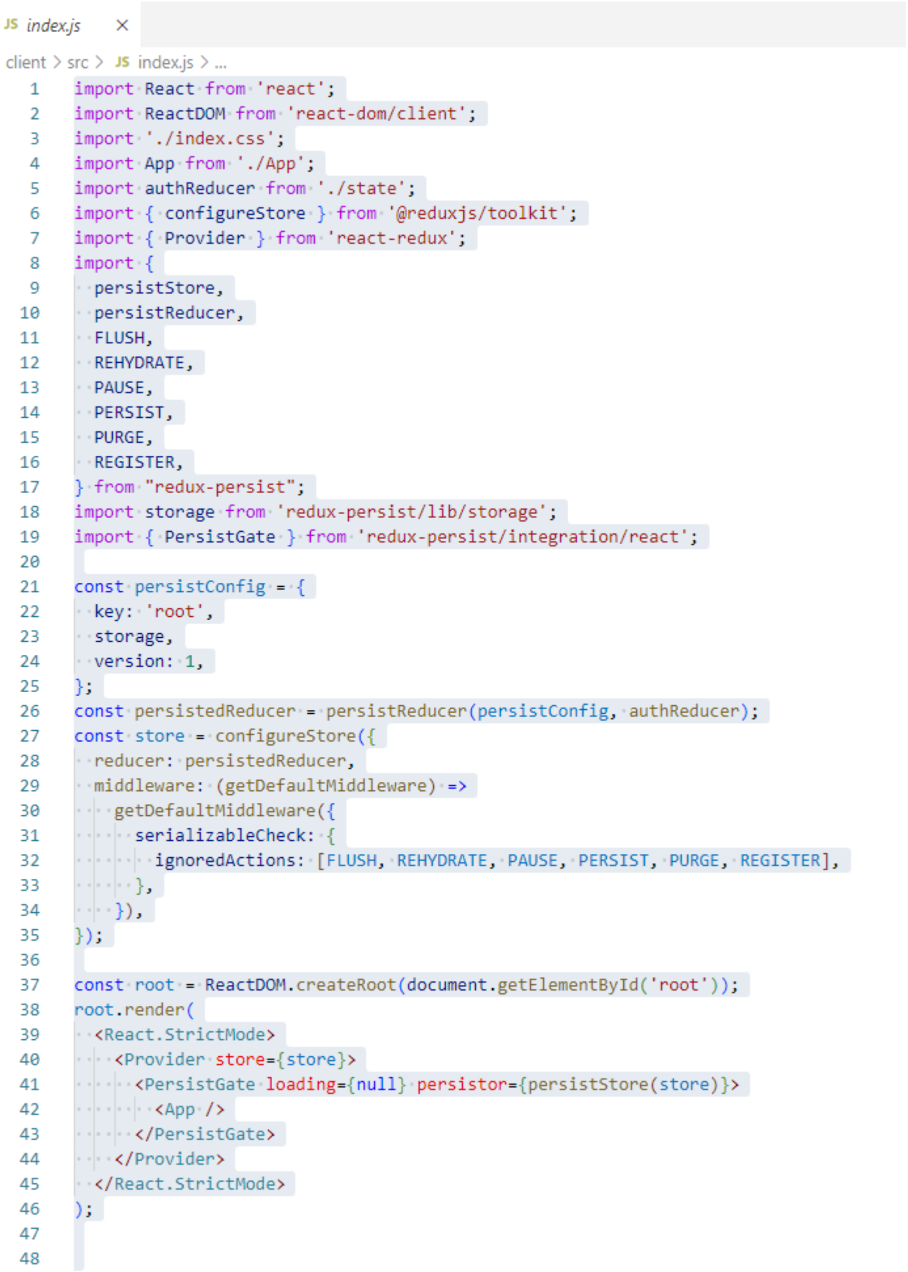
```jsx
import { Box, useMediaQuery } from '@mui/material'
import React from 'react'
import NavBar from '../navBar'
import { useSelector } from 'react-redux'
import UserWidget from '../widgets/UserWidget'
import MyPostWidget from "../widgets/MyPostWidget"
import PostsWidget from '../widgets/PostsWidget'
import FriendListWidget from '../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>
                <Box
                    flexBasis={isNonMobileScreens ? "42%": undefined}
                    mt={isNonMobileScreens ? undefined : "2rem"}
                >
                    <MyPostWidget picturePath={picturePath}/>
                    <PostsWidget userId={_id} />
                </Box>
                {isNonMobileScreens && (
                    <Box flexBasis="26%">
                        <FriendListWidget userId={_id}/>
                    </Box>
                )}
            </Box>
        </Box>
    )
}

export default HomePage
```



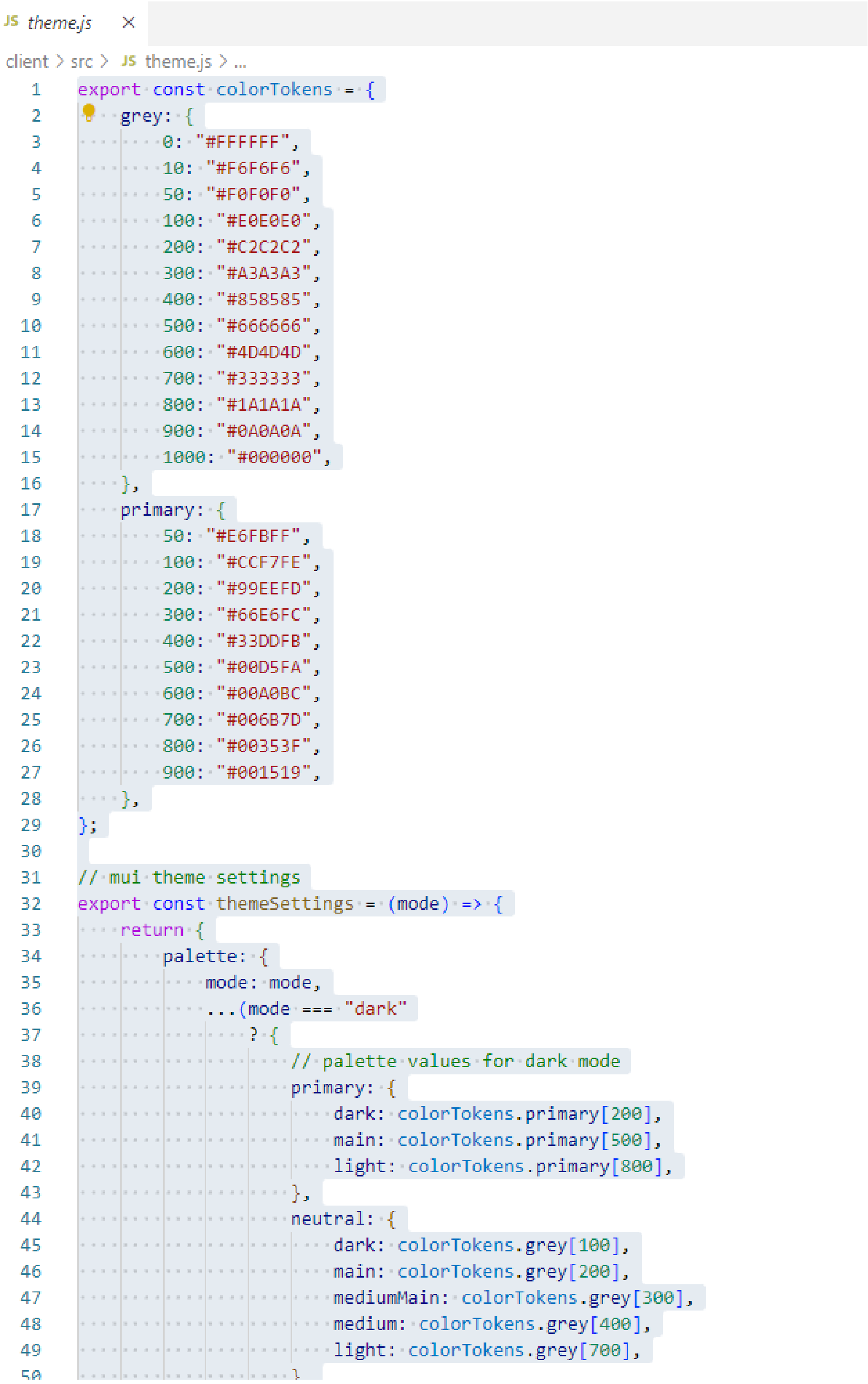
```jsx
import { useState } from "react";
import {
    Box,
    Button,
    TextField,
    useMediaQuery,
    Typography,
    useTheme,
} from "@mui/material";
import EditOutlinedIcon from "@mui/icons-material/EditOutlined";
import { Formik } from "formik";
import * as yup from "yup";
import { useNavigate } from "react-router-dom";
import { useDispatch,useSelector } 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 = () => {
```



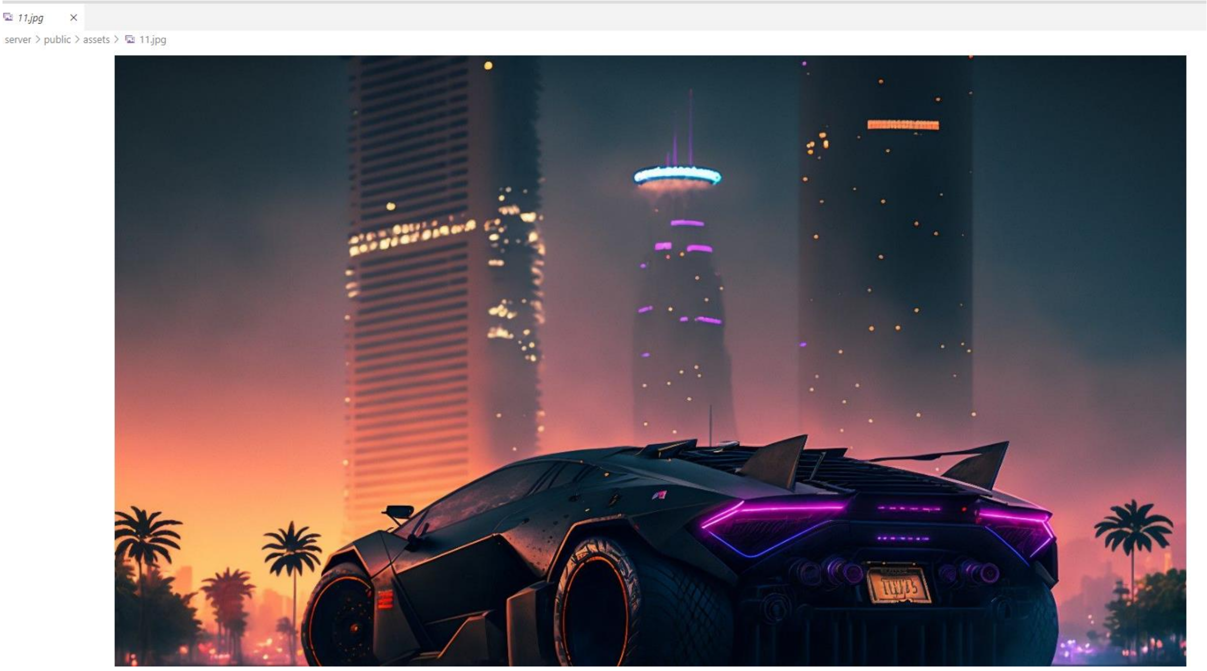
```jsx
import React from 'react'
import { Box, Typography, useTheme, useMediaQuery } from '@mui/material'
import Form from './Form'

const LoginPage = () => {
  const theme = useTheme()
  const isNonMobileScreens = useMediaQuery("(min-width: 1000px)")

  return (
    <Box>
      <Box
        width="100%"
        backgroundColor={theme.palette.background.alt}
        padding="1rem 6%"
        textAlign="center"
      >
        <Typography
          fontWeight="bold"
          fontSize="32px"
          color="primary"
        >
          Blink
        </Typography>
      </Box>

      <Box
        width={isNonMobileScreens ? "50%" : "100%"}
        p="2rem"
        m="2rem auto"
        borderRadius="1.5rem"
        backgroundColor={theme.palette.background.alt}
      >
        <Typography
          fontWeight="500"
          variant='h5'
          sx={{
            mb:"1.5rem"
          }}
        >
          Welcome to Blink, Connect with anyone in the blink of an eye
        </Typography>
        <Form />
      </Box>
    </Box>
  )
}

export default LoginPage
```



```javascript
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import authReducer from './state';
import { configureStore } from '@reduxjs/toolkit';
import { Provider } from 'react-redux';
import {
  persistStore,
  persistReducer,
  FLUSH,
  REHYDRATE,
  PAUSE,
  PERSIST,
  PURGE,
  REGISTER,
} from "redux-persist";
import storage from 'redux-persist/lib/storage';
import { PersistGate } from 'redux-persist/integration/react';

const persistConfig = {
  key: 'root',
  storage,
  version: 1,
};
const persistedReducer = persistReducer(persistConfig, authReducer);
const store = configureStore({
  reducer: persistedReducer,
  middleware: (getDefaultMiddleware) =>
    getDefaultMiddleware({
      serializableCheck: {
        ignoredActions: [FLUSH, REHYDRATE, PAUSE, PERSIST, PURGE, REGISTER],
      },
    }),
});

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <Provider store={store}>
      <PersistGate loading={null} persistor={persistStore(store)}>
        <App />
      </PersistGate>
    </Provider>
  </React.StrictMode>
);

```

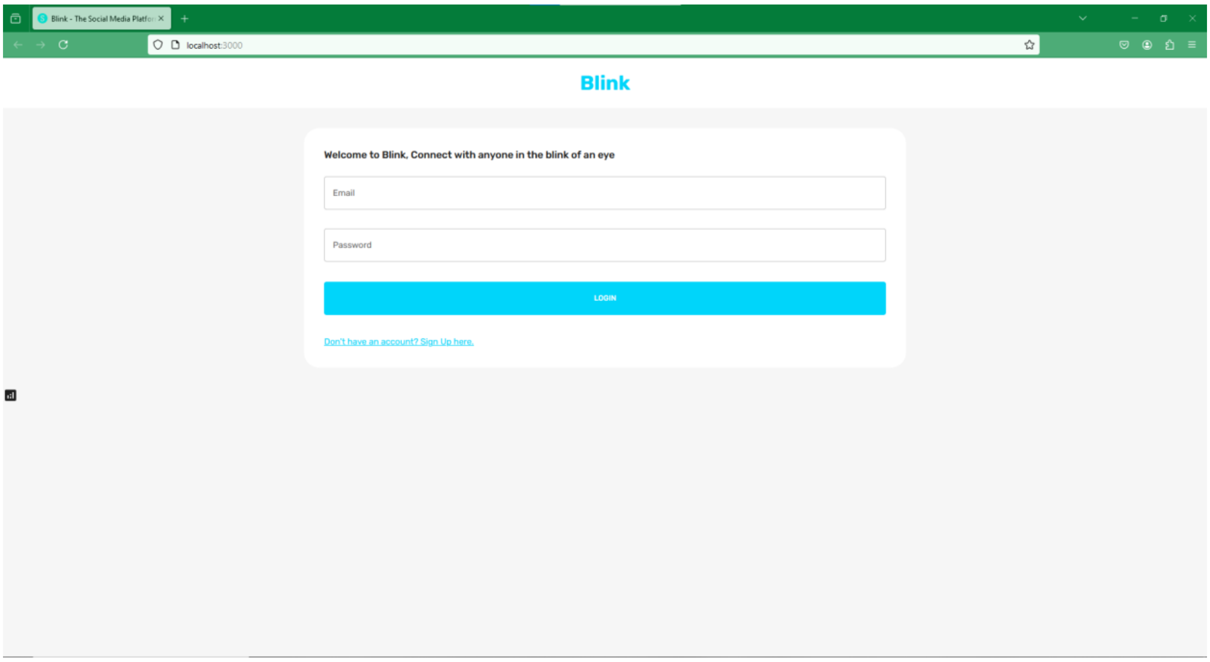


```javascript
export const colorTokens = {
    grey: {
        0: "#FFFFFF",
        10: "#F6F6F6",
        50: "#F0F0F0",
        100: "#E0E0E0",
        200: "#C2C2C2",
        300: "#A3A3A3",
        400: "#858585",
        500: "#666666",
        600: "#4D4D4D",
        700: "#333333",
        800: "#1A1A1A",
        900: "#0A0A0A",
        1000: "#000000",
    },
    primary: {
        50: "#E6FBFF",
        100: "#CCF7FE",
        200: "#99EEFD",
        300: "#66E6FC",
        400: "#33DDFB",
        500: "#00D5FA",
        600: "#00A0BC",
        700: "#006B7D",
        800: "#00353F",
        900: "#001519",
    },
};

// mui theme settings
export const themeSettings = (mode) => {
    return {
        palette: {
            mode: mode,
            ...(mode === "dark"
                ? {
                    // palette values for dark mode
                    primary: {
                        dark: colorTokens.primary[200],
                        main: colorTokens.primary[500],
                        light: colorTokens.primary[800],
                    },
                    neutral: {
                        dark: colorTokens.grey[100],
                        main: colorTokens.grey[200],
                        mediumMain: colorTokens.grey[300],
                        medium: colorTokens.grey[400],
                        light: colorTokens.grey[700],
                    }
```



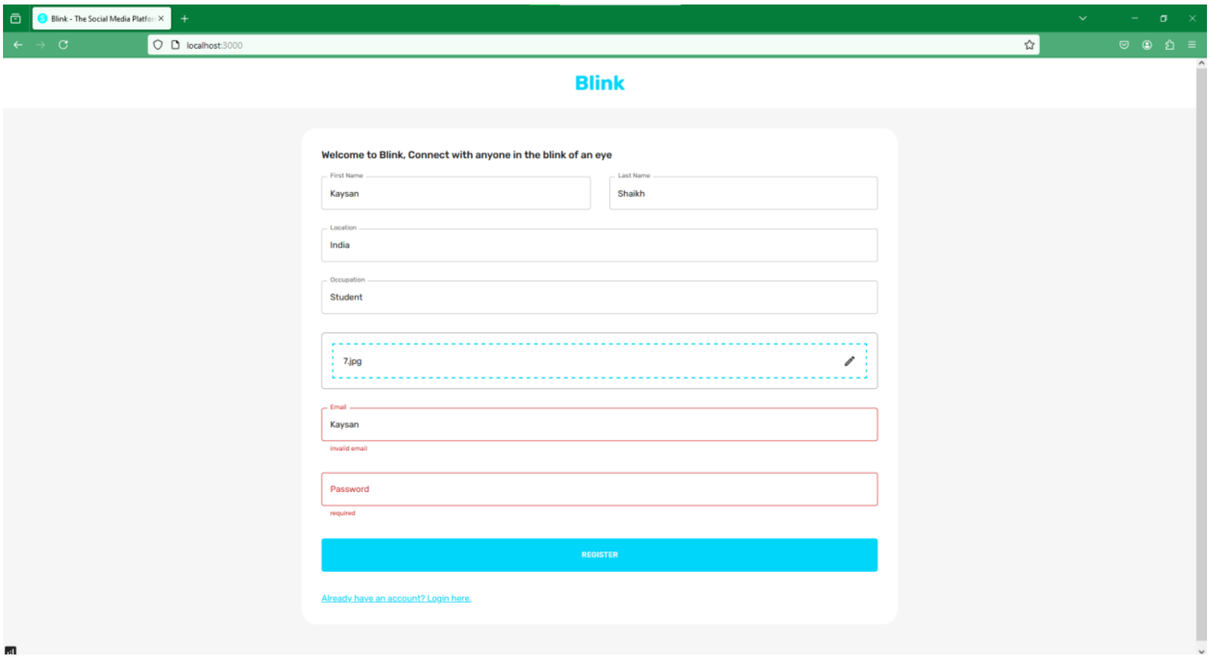
Output: -

Sign in



Welcome to Blink, Connect with anyone in the blink of an eye

Don't have an account? Sign Up here.



Blink - The Social Media Platfo
localhost:3000
Blink
REGISTER
Already have an account? Login here.

Welcome to Blink, Connect with anyone in the blink of an eye

First Name: Kaysan

Last Name: Shaikh

Location: India

Occupation: Student

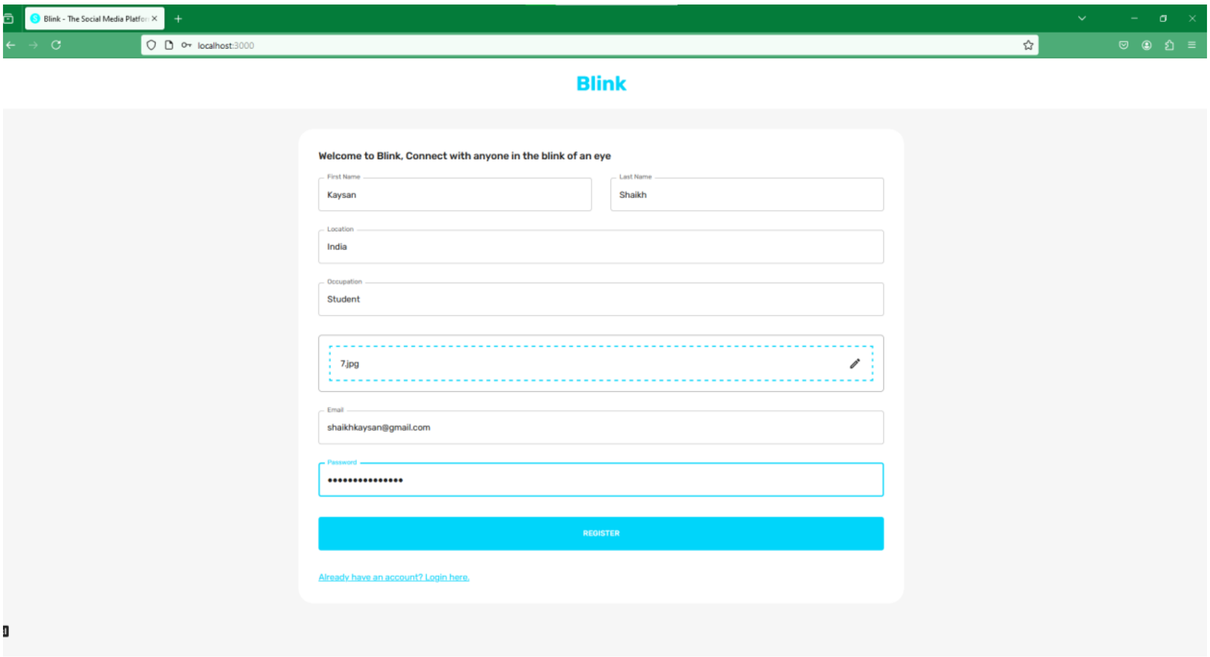
7.jpg

Email: Kaysan

invalid email

Password

required



Blink - The Social Media Platfo
localhost:3000
Blink
REGISTER
Already have an account? Login here.

Welcome to Blink, Connect with anyone in the blink of an eye

First Name: Kaysan

Last Name: Shaikh

Location: India

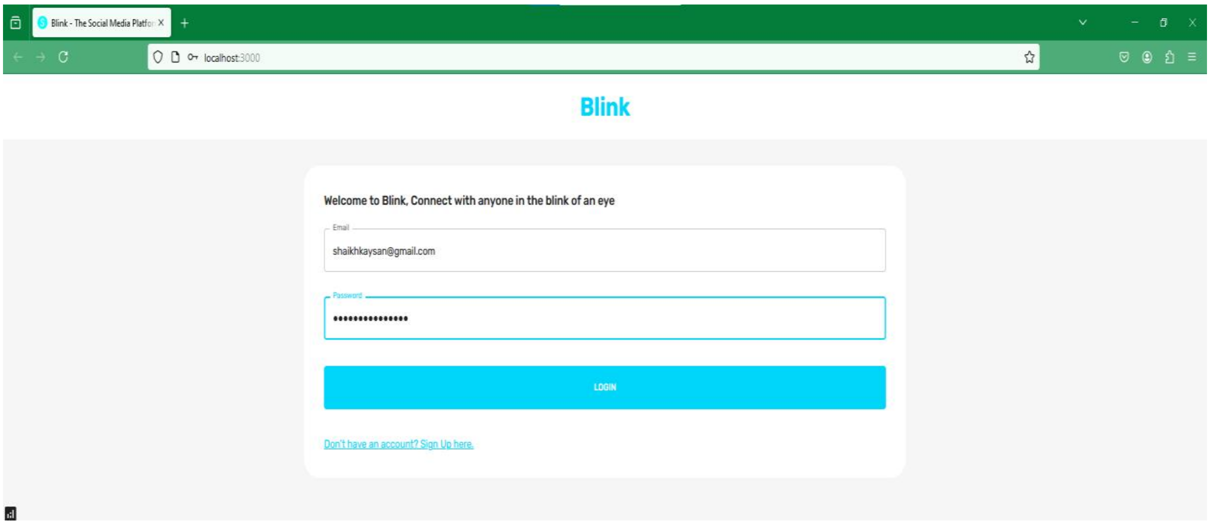
Occupation: Student

7.jpg

Email: shaikhkaysan@gmail.com

Password: •••••••••••••••

Log in



localhost:3000

Blink

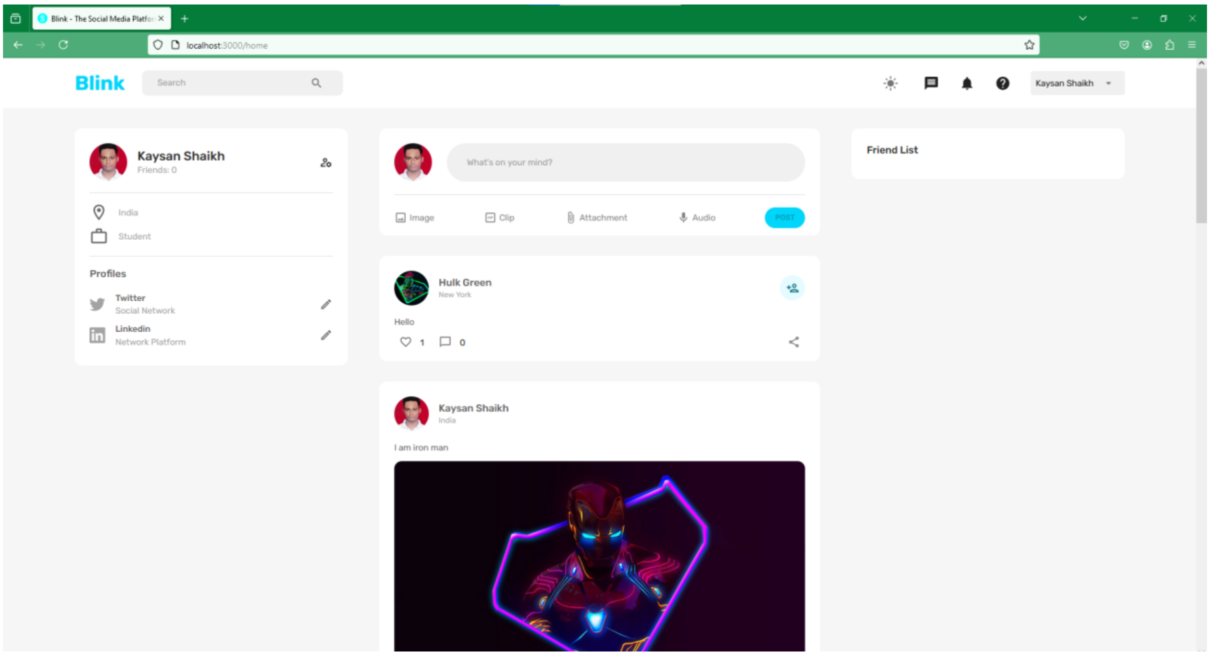
Welcome to Blink, Connect with anyone in the blink of an eye

Email: shaikhkaysan@gmail.com

Password: •••••••••••••••

Blink - The Social Media Platfor
LOGIN
Don't have an account? Sign Up here.

After Log in:



localhost:3000/home

Kaysan Shaikh

Friends: 0

India

Student

Profiles

Twitter
Social Network

Linkedin
Network Platform

What's on your mind?

Hulk Green
New York

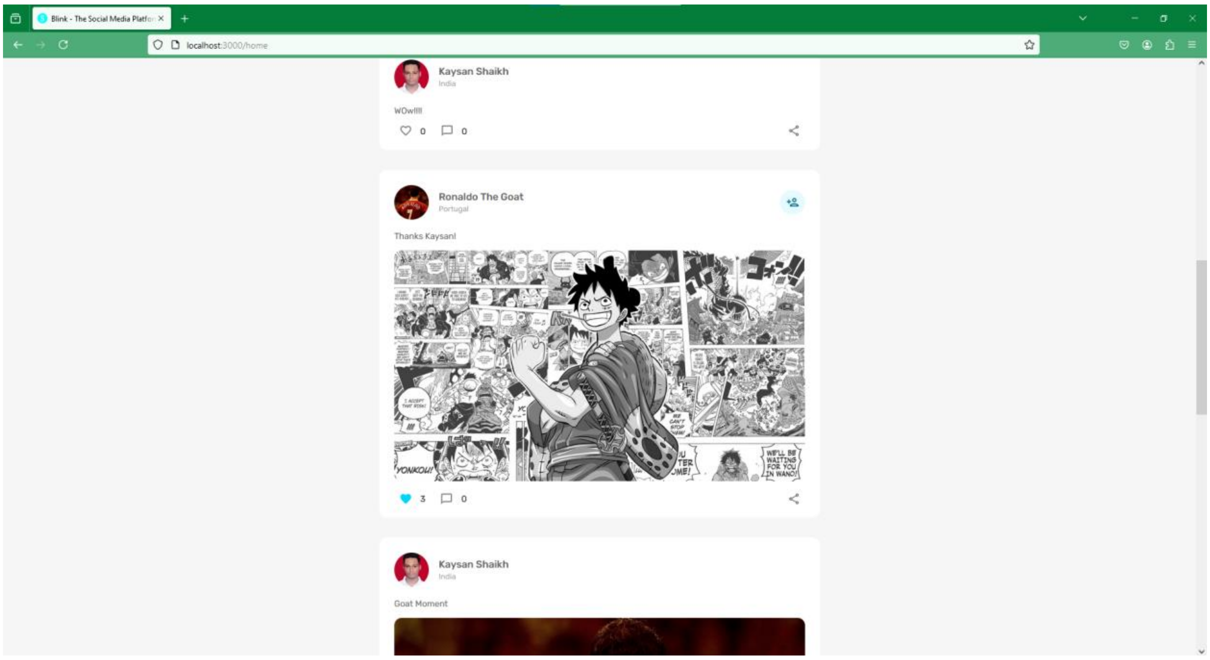
Hello

1 0

Kaysan Shaikh
India

I am iron man

Friend List



Kaysan Shaikh
India

WOw!!!!

0 0

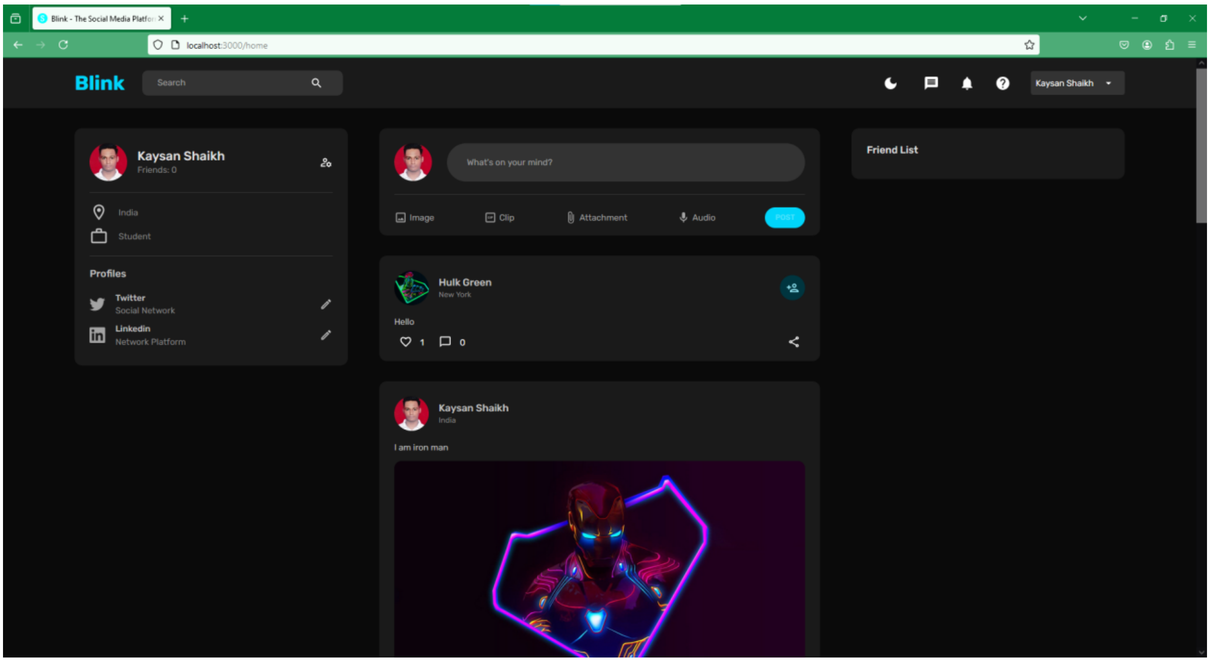
Ronaldo The Goat
Portugal

Thanks Kaysan!

3 0

Kaysan Shaikh
India

Goat Moment



Kaysan Shaikh
Friends: 0

India

Student

Profiles

Twitter
Social Network

Linkedin
Network Platform

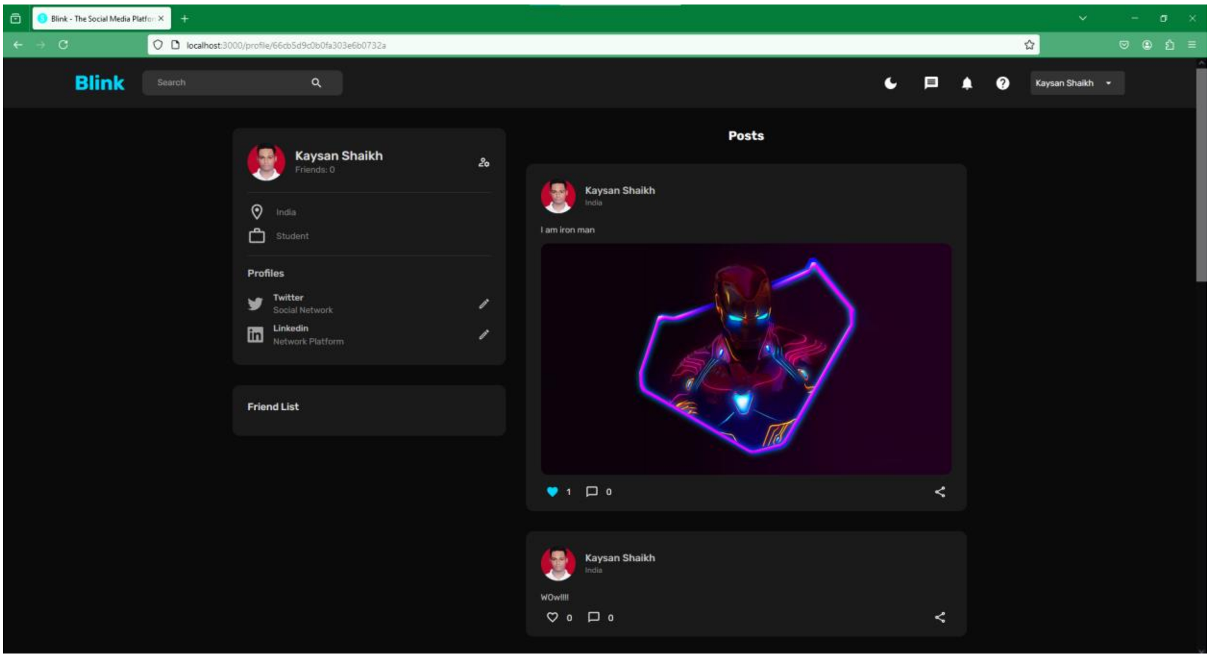
Hulk Green
New York

Hello

1 0

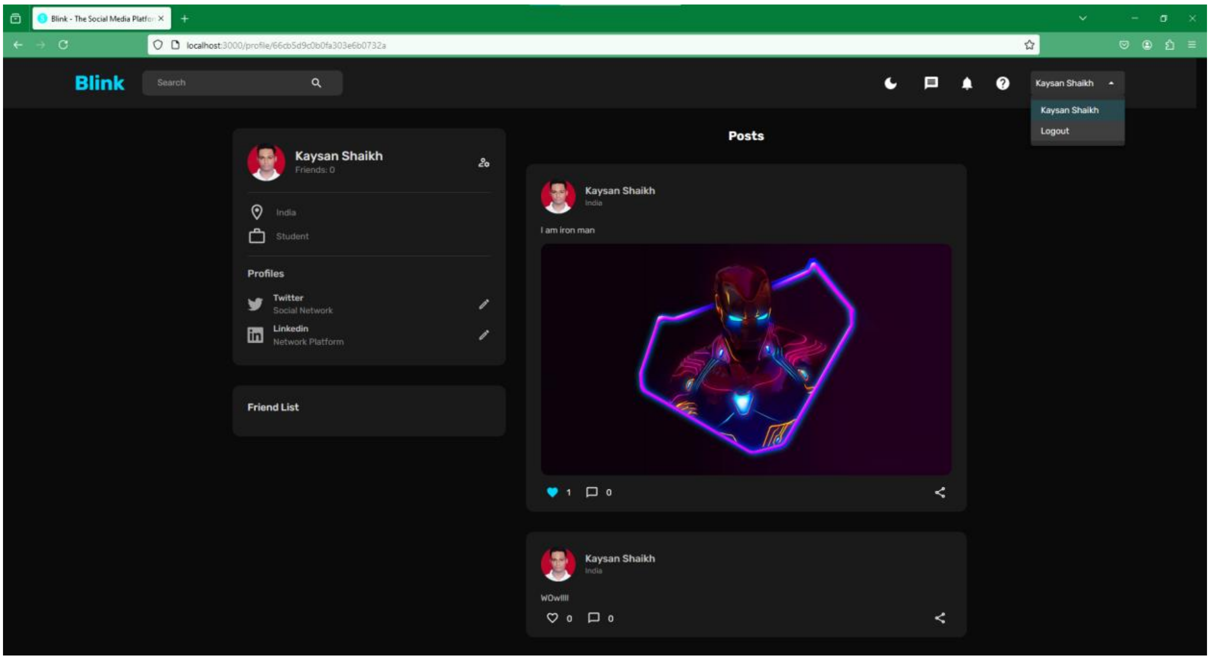
Kaysan Shaikh
India

I am iron man



I am iron man

WOwlll



I am iron man

WOwlll

Future Enhancements: -

- **Real-time Notifications:** Implement real-time notifications for post interactions and friend requests.
- **Search Functionality:** Add search capabilities for posts and user profiles.
- **Enhanced User Analytics:** Provide insights and analytics for user engagement and activity

Contributions: -

Contributions are welcome! Please fork the repository and submit a pull request with your changes.