Guide to integrating with our sockets server

Example with React js

```
import { io } from 'socket.io-client';
import {
  useState,
  useEffect
} from 'react';

const socketUrl = 'wss://friends-si-exam.azurewebsites.net'; // cloud

const socket = io(socketUrl, {
  extraHeaders: {
    Authorization: "Bearer
eyJhbGciOiJSUzI1NiIsImtpZCI6ImE5NmFkY2U5OTk5YmJmNWNkMzBmMjlmNDljZDM3ZjRjNW
U2NDI3NDAiLCJ0eXAiOiJKV1QifQ.eyJpc3MiOiJodHRwczovL3NlY3VyZXRva2VuLmdvb2dsZ
S5jb20vYXV0aC1zZXJ2aWNlLXNpIiwiYXVkIjoiYXV0aC1zZXJ2aWNlLXNpIiwiYXV0aF90aW1
lIjoxNjY5NzQ3MDA3LCJ1c2VyX2lkIjoib0tCVk9RYUo3YlhKaWdTcFRiSXV5akYzUE9LMiIsI
nN1YiI6Im9LQlZPUWFKN2JYSmlnU3BUYkl1eWpGM1BPSzIiLCJpYXQiOjE2Njk3NDcwMDcsImV
4cCI6MTY2OTc1MDYwNywiZW1haWwiOiJ0ZXN0NkBnbWFpbC5jb20iLCJlbWFpbF92ZXJpZmllZ
CI6ZmFsc2UsImZpcmViYXNlIjp7ImlkZW50aXRpZXMiOnsiZW1haWwiOlsidGVzdDZAZ21haWw
uY29tIl19LCJzaWduX2luX3Byb3ZpZGVyIjoicGFzc3dvcmQifX0.CsVy1qjjfJqigrtgfSS95
8zG8eowk2BZ8uqkDkLR5AhjTKczsY7TLfSrX5tN9cIZUvk0czZHXVJNi7UFEel5Jj-J1xEM-tF
JHCskCD99-h4hsID5slQI2v4-JDp1PiKZy4PfuZTREfP1obePGTujc2nNy4b6fPagrcnVMUz7n
g7beaoAyHpBHY7sfOs-OzGvnr9gAeDs3wK-5Xt_i4xyO6nsCzjSE1_8UicFY0rwixTCiQhYHkv
NszshGE7EjDMRzHnAbKSinEg7DyZ1N1xLpblDExiAIOFT4kFC64qMYq_Ifpj68ouTUAtd88FTx
3PYw31wSGYpC5HGJki6FdHMTA"
  }
});

const App = () => {
  const [isConnected, setIsConnected] = useState(false);
  const [users, setUsers] = useState([]);

  useEffect(() => {
```

```
    // this event is fired automatically when the client connects to the
server
    socket.on('connect', async () => {
      setIsConnected(true);
    });

    // this event is emitted if the Authorization header is missing
    socket.on('authFailed', (error) => {
      console.log(error);
      socket.disconnect();
    });

    // the server emits this event when the client connects/disconnects
    // fetches the users who are online and share wishlists with the
logged in user.
    socket.on('onlineUsers', (data) => {
      setUsers(data.onlineUsersUniqueValues);
    });

    return () => {
      socket.off('connect');
      socket.off('authFailed');
      socket.off('disconnect');
      socket.off('onlineUsers');
    };
  }, [users]);

  return (
    <div>
      <p> Is connected: {'' + isConnected} </p>
      <p>Who is online:</p>
        {users.map((user, idx) => {
          return (
            <div key={idx}>
              <p> {`${user}`}</p>
            </div>)
        })}
    </div>
  );
}
```

```
export default App;
```

Full list of events

Listens	connection, disconnect		These events are fired automatically when a socket connects and disconnects.
Emits	onlineUsers	Sends emails of the users who you share wishlists with.	When a user joins/leaves the website, the server sends the emails of the users who share wishlists with a given user.
Emits	authFailed		When a socket connects without Authorization header. Initial request must have Authorization header: “Authorization”: “Bearer <<access token>>”
Emits	inviteSucceeded		When a request to to invite someone via email succeeded.
Emits	inviteFailed		When a request to invite someone via email failed.
Listens	createWishlist	`socket.emit('createWishlist', ({ wishlistName: 'Christmas 2022' }))`	Request to create a new wishlist.
Emits	createWishlistSucceeded		When a request to create a new wishlist succeeded.
Emits	createWishlistFailed		When a request to create a new wishlist failed.
Listens	addProductToWishlist	`socket.emit('acceptInvite', ({ wishlistId: 69, productId: 15 }))`	Request to add a new product to a wishlist.
Emits	addProductToWishlistSucceeded		When a request to add new product succeeded.
Emits	addProductToWishlistFailed		When a request to add new

			product failed.
Listens	invite	`socket.emit('invite', ({ wishlistId: 69, wishlistName: 'Christmas 2022', emailTo: 'argyro12@gmail.com'}));`	Request to invite someone to join a wishlist.