Passing the Props

Integration of the user and activeUserID into the Main as props so that we can access them later for display.

Let's move on.

In App.js, retrieve the user and activeUserId from the store, like this:

```
const { contacts, user, activeUserId  } = store.getState();
```

What we had previously was this:

```
const { contacts } = store.getState();
```

Now, pass on these values as props to the <Main /> component.

```
<Main user={user} activeUserId={activeUserId} />
```

What we had previously was this:

```
<Main />
```

Now, let's have the render logic fleshed out in <Main />

Before:

```
import React from "react";
import "./Main.css";
const Main = () => {
  return <main className="Main">Main Stuff</main>;
};
export default Main;
```

Now:

```
import React from "react";
import "./Main.css";
import Empty from "../components/Empty";
```

```
import Empty from "../components/Empty";
import ChatWindow from "../components/ChatWindow";
const Main = ({ user, activeUserId }) => {
  const renderMainContent = () => {
    if (!activeUserId) {
      return <Empty user={user} activeUserId={activeUserId} />;
    } else {
        return <ChatWindow activeUserId={activeUserId} />;
       }
  };
  return <main className="Main">{renderMainContent()}</main>;
};
export default Main;
```

Main.js

What has changed isn't difficult to grasp. user and activeUserId are received as props. The return statement within the component has the function renderMainContent invoked.

All renderMainContent does is check if activeUserId doesn't exist. If it doesn't, it renders the empty screen. If it does exist, then the ChatWindow is rendered.

Great!

We don't have the Empty and ChatWindow components built out yet.