在管理和偵測登入狀態這個問題上，使用Redux（及其與userReducer結合）與在應用程式的某個上下文（例如AuthContext）中直接管理登入狀態，這兩種方法各有優勢，你可以根據你的 應用需求和個人偏好來選擇其中一種，或將兩者結合使用。 以下是每種方法的一些考慮因素：

使用Redux和userReducer

集中管理狀態：Redux允許你在一個集中的地方管理應用程式的所有狀態，這使得狀態的讀取和更新變得更加一致和可預測。

可預測的狀態更新：透過使用純函數（reducers）來處理所有狀態更新，Redux確保了應用程式狀態的變化是可預測且可追蹤的。

中間件和工具支援：Redux支援各種中間件和工具，如Redux DevTools，這些工具可以幫助你更好地開發和除錯應用程式。

適用於大型應用：如果你的應用程式很大，或者預計會變得很大，並且包含許多相互依賴的狀態，使用Redux可能更有優勢。

使用Context API（如AuthContext）

簡化狀態管理：對於簡單的應用程式或當僅需要跨元件共享少數幾個狀態時，React的Context API提供了一種更簡單直接的方式來共享狀態。

避免額外的依賴：如果你的應用不需要Redux提供的那些額外功能和能力，使用Context API可以避免引入額外的庫和學習成本。

更容易使用Hooks：React Context與Hooks結合使用可以編寫更少、更簡潔的程式碼，使得狀態管理和元件間的通訊更加簡單。

結合使用Redux和Context API

在某些情況下，結合使用Redux和Context API可能是個好主意。 例如，你可以使用Redux來管理全域狀態（如使用者登入狀態、應用程式的主題等），同時使用Context API來管理更具體的、不需要全域管理的狀態（如UI元件的顯示/隱藏狀態）。

如果你已經在使用AuthContext來偵測登入狀態，而且它滿足你的需求，你可能不需要切換到Redux。 但是，如果你發現狀態管理變得複雜，或者需要更強大的開發工具支持，遷移到Redux可能是一個好選擇。

最終決定取決於你的應用需求、預期的應用程式規模以及你個人對這些工具的熟悉程度。 可以從專案的實際需求出發，選擇最適合你狀況的狀態管理方案。