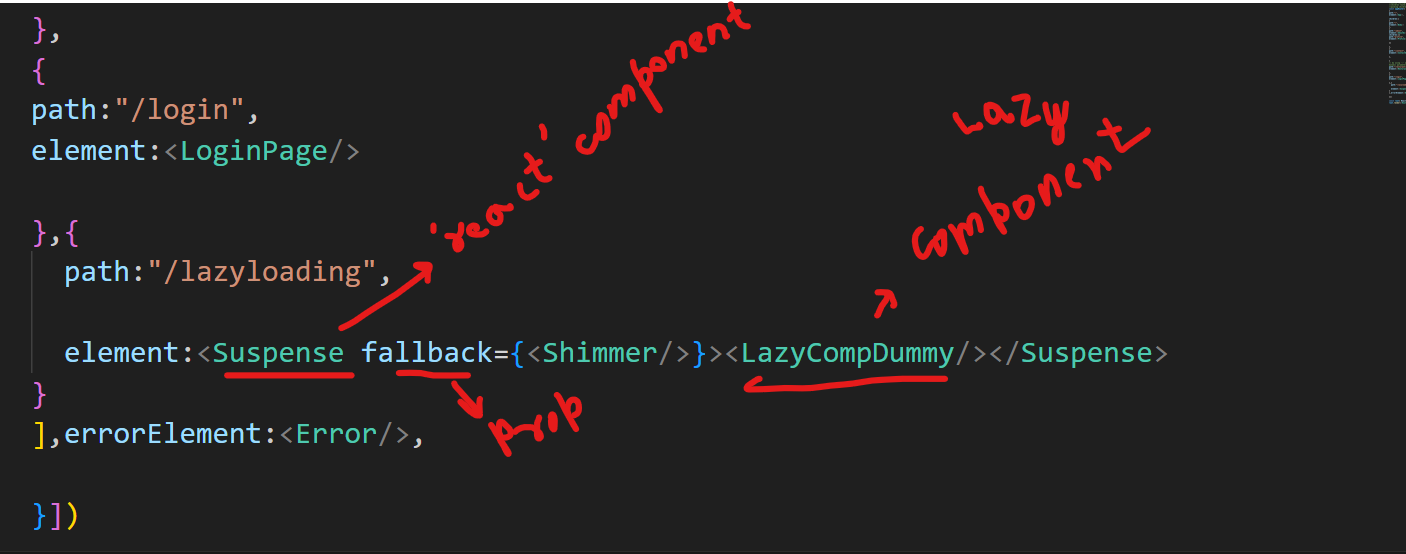
Ep:09  
  
For Optimizing app :  
1>we create custom hooks so that logical part related to a component move into these hooks and only task of components is to display itself on browser, it enhance readability, , ,maintainability , testable, modularity of code etc .  
  
2>we also distribute app into smaller-2 chunk/bundle so that when a particular component load of some application then only its related component code load by which bundled all the related component/feature into single file i.e component.js file .  
>As we know there is only one index.js file in the whole application after bundling , and sometimes this index.js file become larger nd larger in size which slow down the performance of an application as it takes time to load bcoz it contains codes of all the components/.js files inside the application, which is not required for rendering a particular feature of a web page, no need to load whole code of the application for any single feature is required by web page .  
>index.js/main.js file consist all code io js of a application into a single file.  
>when we build any smaller bundler than that code removed from index.js and and a new chunk/bundle file use use to render a particular component which consist code of all related component.  
  
custom hooks:  
>It's a utility/helper function, also it’s a normal js function.  
>It make code modular and optimize./

//A component should have single responsibility function like display the component.  
>and all related logic keep inside hooks.

> Create Separate files for separate hooks.  
>For creating hook try to use ‘use’ keyword.  
  
>Parcel or any bundler bundle all files into single files, like single for CSS, html, Javascript as well.  
>to create more chunks/smaller bundle file we use lazy () method, however bundle should have enough code for a particular feature.  
>when we do lazy loading code only comes on browser ,when we request for that code by clicking on that feature/component.

//whenever we add a eventListener we’ve to clear it as well when we go out of component i.e during unmounting phase.  
  
\*\*\*\*chunking/ code splitting / Dynamic Bundling / Lazy loading / on demand loading / dynamic Import :  
>It create logical bundling.  
  
  
>when we load component on demand it got suspended by react ,as react try to quickly load the component which is not their bcoz it’s under request and it takes time to load component.js file in browser, so this component get suspended by react.  
>to resolve this issue we use <suspense> component it comes from ‘react’ core library, it also takes fallback as a props where we can pass any piece JSX , also shimmer so that it display on as long as component.js files not get loaded.  
   
>don’t write lazy() method inside a component.  
  
const LazyCompDummy= lazy(()=>{return import('./components/LazyComp')})  
  


//when app bundle size increase do chunking.

//For image heavy websites , optimize images keep on CDN   
  
//There is only one javascript files unless we do chunking .  
  
//Q. why do we use hooks?  
Ans. For Reusability, Readability , modularity, testable, separation of concern, maintainable, wrap some logic inside a function and use as number of times  
>modular=> code into smaller different component.

//Q.Difference b/w functional component, hooks, normal js function.  
  
functional component=>It return JSX  
  
hooks>it contains logic  
  
Normal function=>We can’t use state or useEffect hooks or neither reconciliation work here so react can’t keep track of state and hooks.