

# UI Reviews Summary Deep-dive:

## Development Standards:

### Naming Convention:

File Name: PascaleCase

Variable Name: camelCase

Directory: kebab-case

CSS & Style: snake\_case

Naming Format: [component].[element]-[function-description].[extension]

### Examples:

Application level -

epax.app-xyz.js

epax.app-api-proxy.js

Component Level -

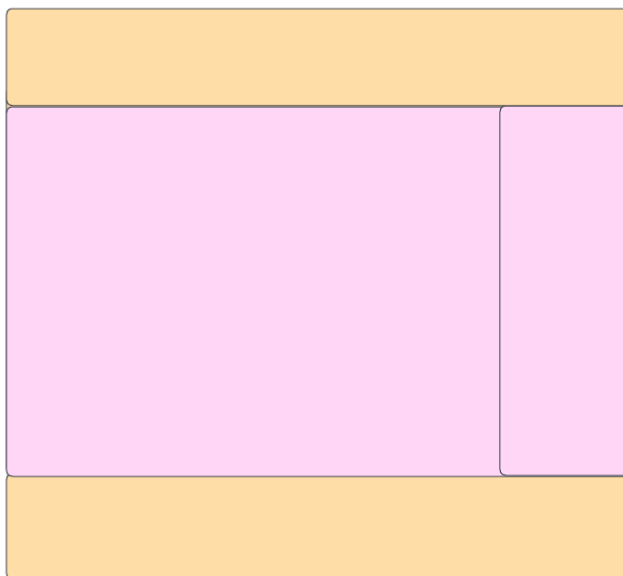
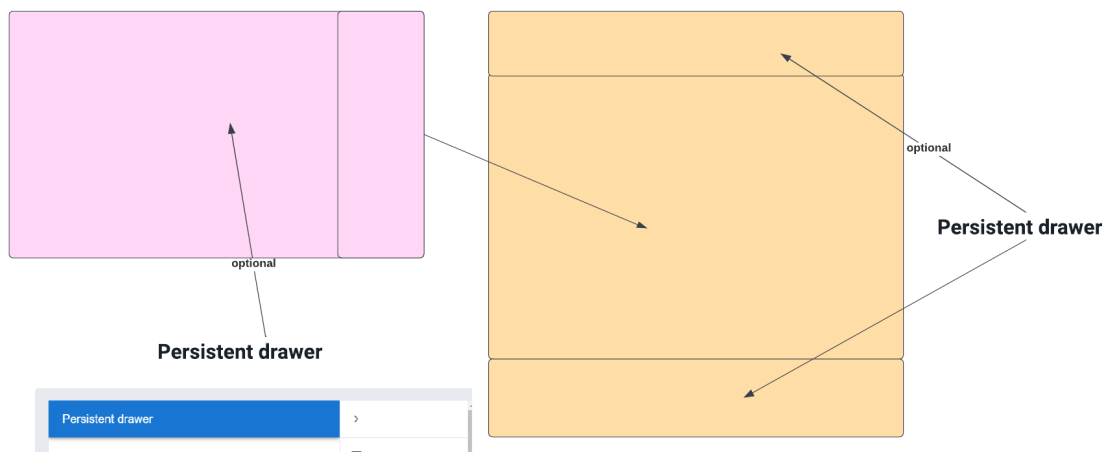
search.widget-context-search.js

## Directory Structure:

```
▼ application
  ▶ api
  ▼ assets
    ▼ i18n
      ▼ lang
        ▼ country
          ▶ variant
      ▼ images
    ▼ themes
      ▼ common
        ▶ css
        ▶ scss
      ▼ theme_name
        ▼ css
        ▶ scss
    ▶ common
  ▼ config
    epax.app-config.js
  ▶ const
  ▶ lib
  ▶ redux
  ▶ route
▼ componets
  ▶ common
  ▶ routing
▼ wiidgets
  ▼ control
    control.js
  ▼ navigation
  ▶ operation
  ▼ search
```

## Code Refactoring:

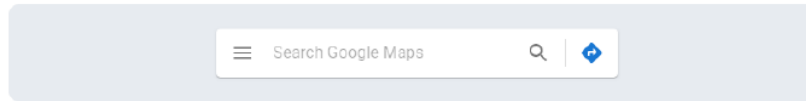
### Application Container Implementation - Use MUI Persistent Drawer



# Application Identity, Search Widget Implementation - Use App Bar and Custom Text Field

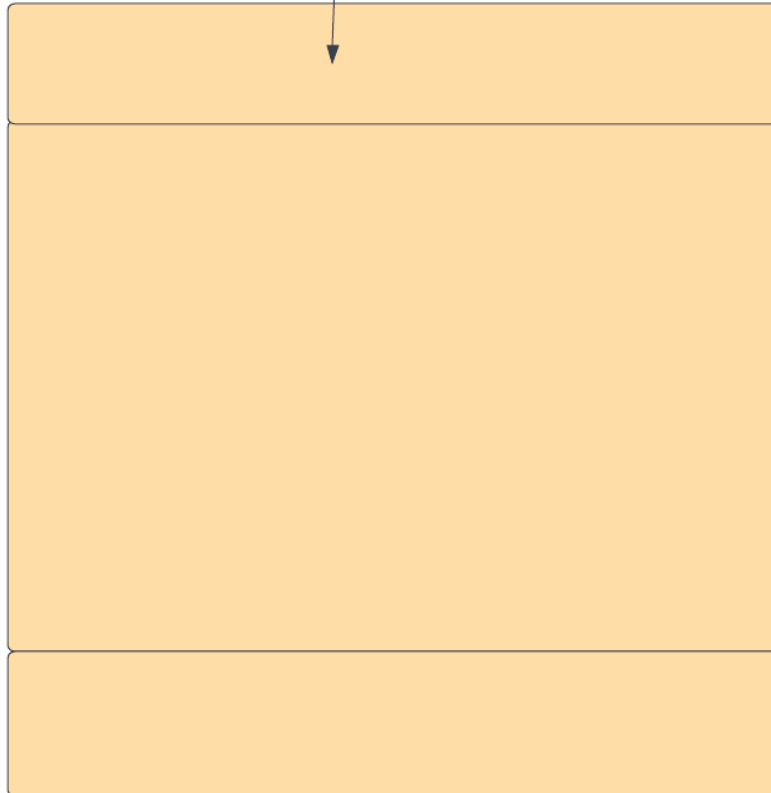
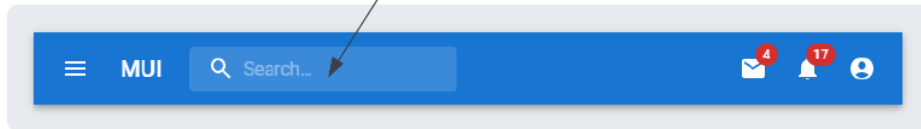
## Use Text Field

Customization does not stop at CSS. You can use composition to build custom components and give your app a unique feel. Below is an example using the `InputBase` component, inspired by Google Maps.

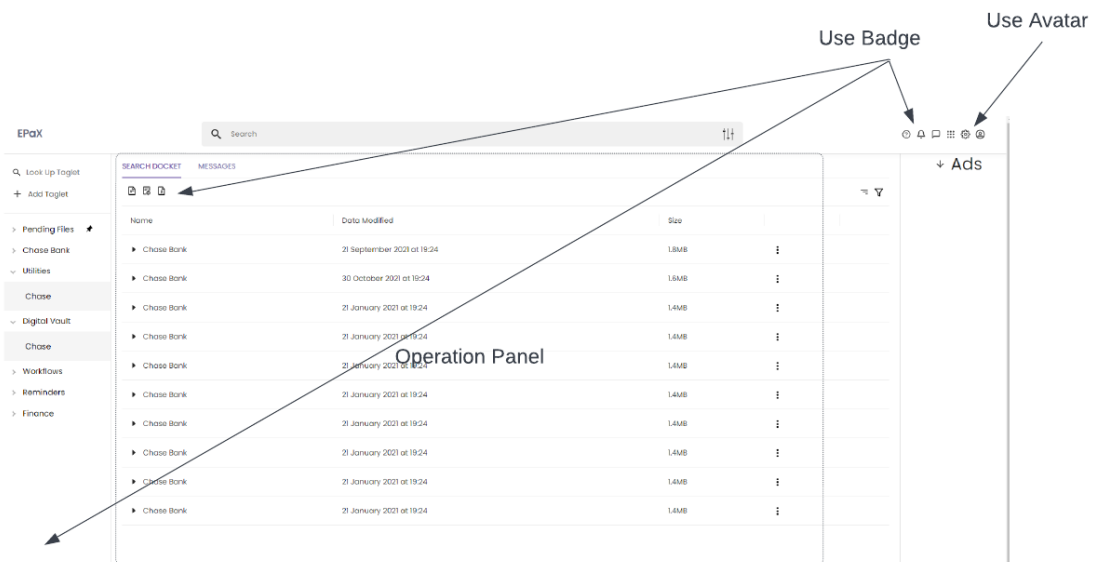


## App bar with a primary search field

A primary searchbar.



## Control Widget Implementation - Use Badge & Avatar



## Sign-in Implementation -

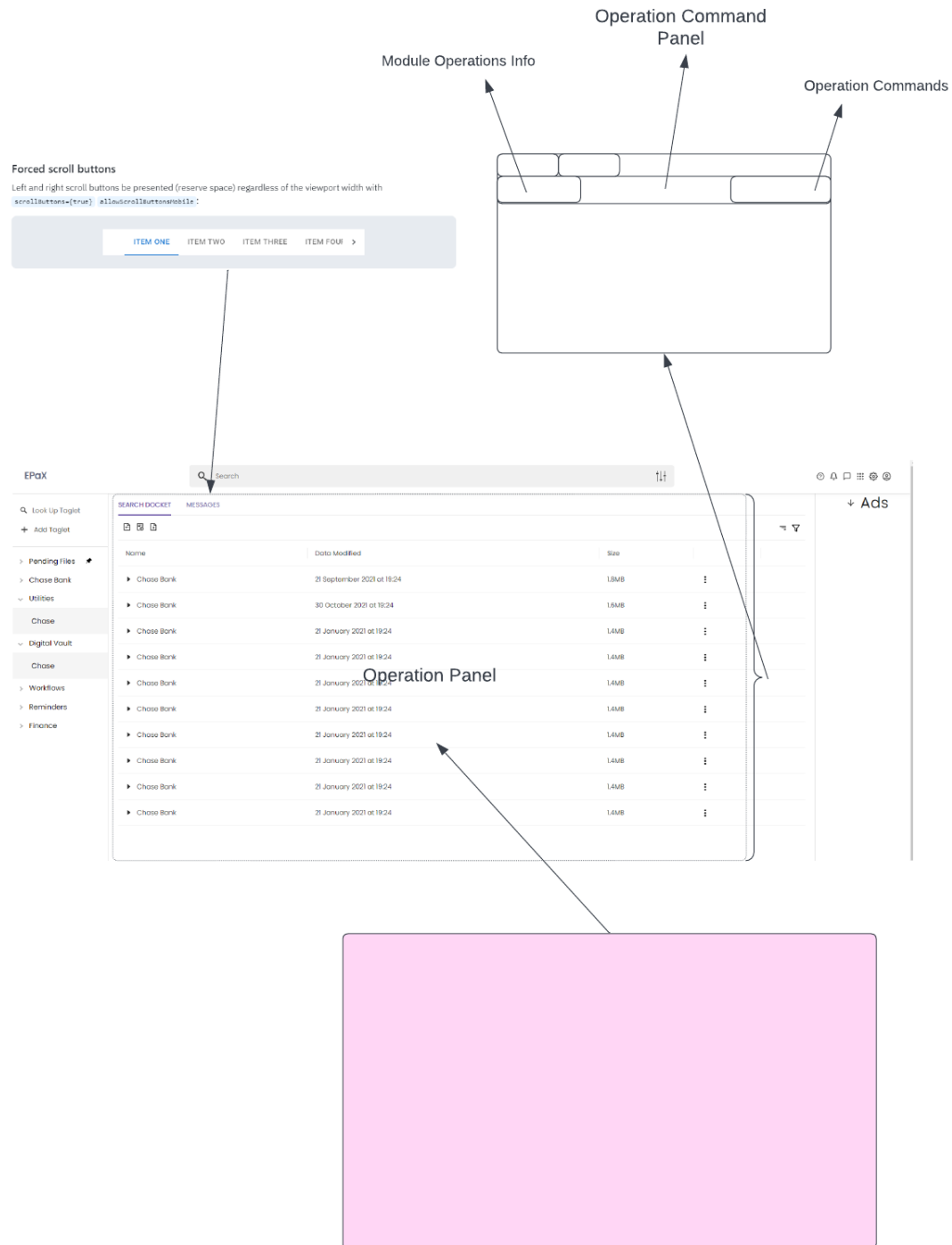
### App bar with a primary search field

A primary searchbar.

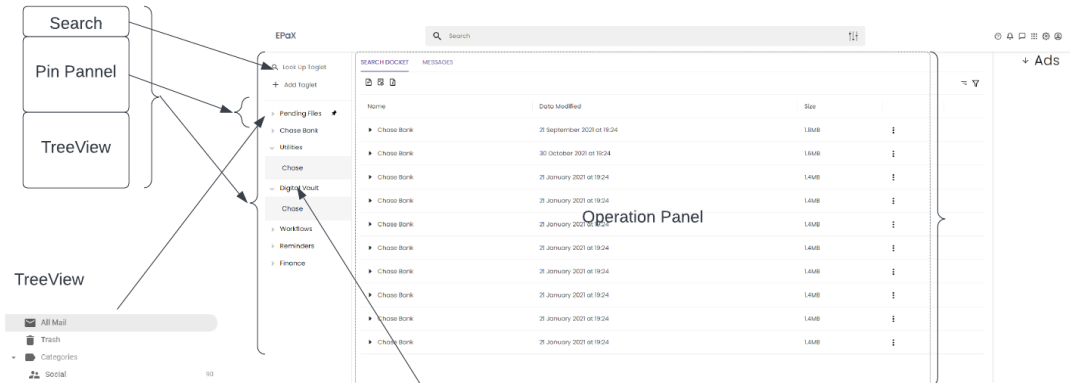
The image displays a UI design for a sign-in implementation. At the top is a blue app bar with a hamburger menu icon, the text 'MUI', a search bar with a magnifying glass icon and the placeholder text 'Search...', and three notification icons (envelope, bell with '17', and profile) with red badge counts. Below the app bar is a large orange rectangular area containing a white 'User Login' form. The form has three input fields: 'Account Name', 'User Name', and 'Password'. Below these is a 'Remember me' checkbox (checked) and a 'Forgot your Password' link. A 'Log In' button is at the bottom of the form. Below the button are two links: 'Can't Log In ?' and 'Sign up for account'. To the right of the orange area is a 'Sign Up For Your Account' form with four input fields: 'Account Name', 'Enter Email', 'User Name', and 'Password', followed by a 'Password Again' field. A 'Sign Up' button is at the bottom of this form. Below the button is a link: 'Already have an iPhoto account? [Log In](#)'.

## Operation Widget Implementation - Forced Scroll Buttons Tab

Dynamic creation of tab <https://gist.github.com/Rahul-RB/273dbb24faf411fa6cc37488e1af2415>



## Navigation Widget Implementation - Use Containers with three Search Widget, Treeview



You can use the `ContentComponent` prop and the `useTreeView` hook to further customize the behavior of the `TreeView`.

## Setting Theme using Global Reset - Dynamically reset the theme

## Coordinate Components -

Use react-click-away-listener

<https://mui.com/material-ui/react-click-away-listener/>

## Adapting Mediator Pattern Library for component communication

<https://www.dofactory.com/javascript/design-patterns/mediator>

<https://www.npmjs.com/search?q=mediator%20design%20pattern>

## Create Central API Proxy Script

## Create Constant Text Script

All Code should be driven off stubbed data - > Use API Proxy