

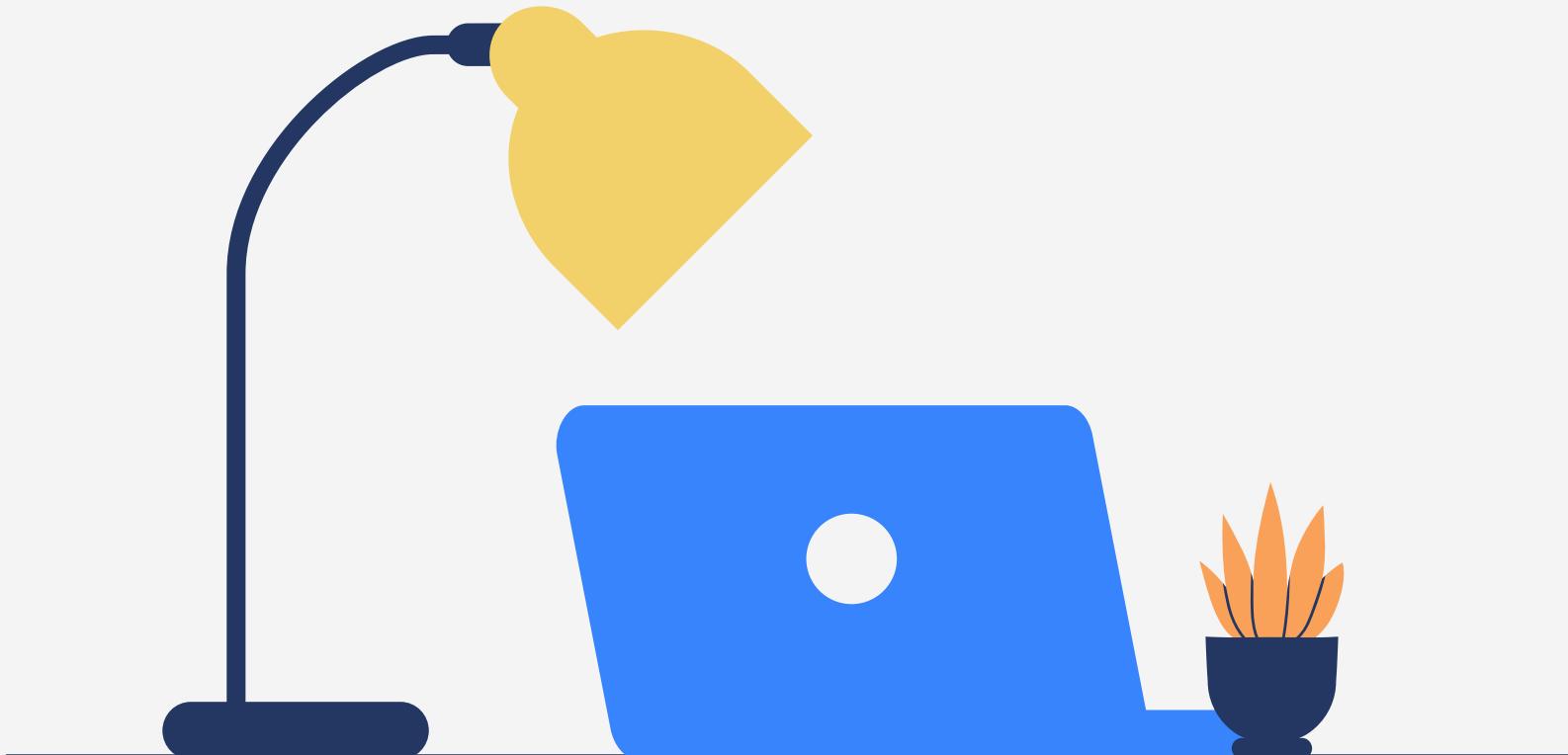


A note-taking app from
developers for developers.



Problem

Taking **good notes** is an important skill that reinforces what you've learnt by **writing it down in words that makes sense to you**. This can be difficult when there isn't an app that lets you easily write and manage your notes.



Unorganised

Notes that are messy and unorganised can quickly become **cluttered and hard to read**. This can discourage you from making further progress.



Difficult to share

Sharing notes and collaborating with others can be hard especially when working or studying remotely, making it hard to build connections and study together in groups.



Inefficient

Sometimes there's a **line of code that you would like to reuse in the future**. How can we save this piece of information that doesn't involve making numerous bookmarks or individual text files?

Solution



Create

Easily create notes, code snippets and pastes and store them all in one place.

Organise

Documentation can be hard to read, but your notes shouldn't be. Organise your notes by storing everything in one place and make your notes easier to find by grouping notes together or assigning keywords.

Collaborate

Work together with your peers and help each other succeed. Share your notes and your knowledge with each other to foster a collaborative learning environment.

Organise

Filter, search, sort and organize your notes with tags

#array #backend
#docs #quiz #node #github
#snippet #code
#homework #css
#async #html #assignment
#terminal #cs
#ruby #rubyonrails
#frontend #javascript #object
#function

Syntax Highlighting

Save code with the correct syntax highlighted according to the language

```
class HelloMessage extends React.Component {  
  handlePress = () => {  
    alert('Hello')  
  }  
  render() {  
    return (  
      <div>  
        <p>Hello {this.props.name}</p>  
        <button onClick={this.handlePress}>Say Hello</button>  
      </div>  
    );  
  }  
}
```

User Stories

Personas



Johnny

+ Student
+ Aspiring Dev



Wallace

+ Teaching
Assistant



Stacy

+ Professional
Developer

As **Johnny**, I want to...

- Create notes
- Filter notes by programming language
- Add images to my note

As **Wallace**, I want to...

- Create a group for collaborate note taking
- Add other users to the group
- Remove users from the group

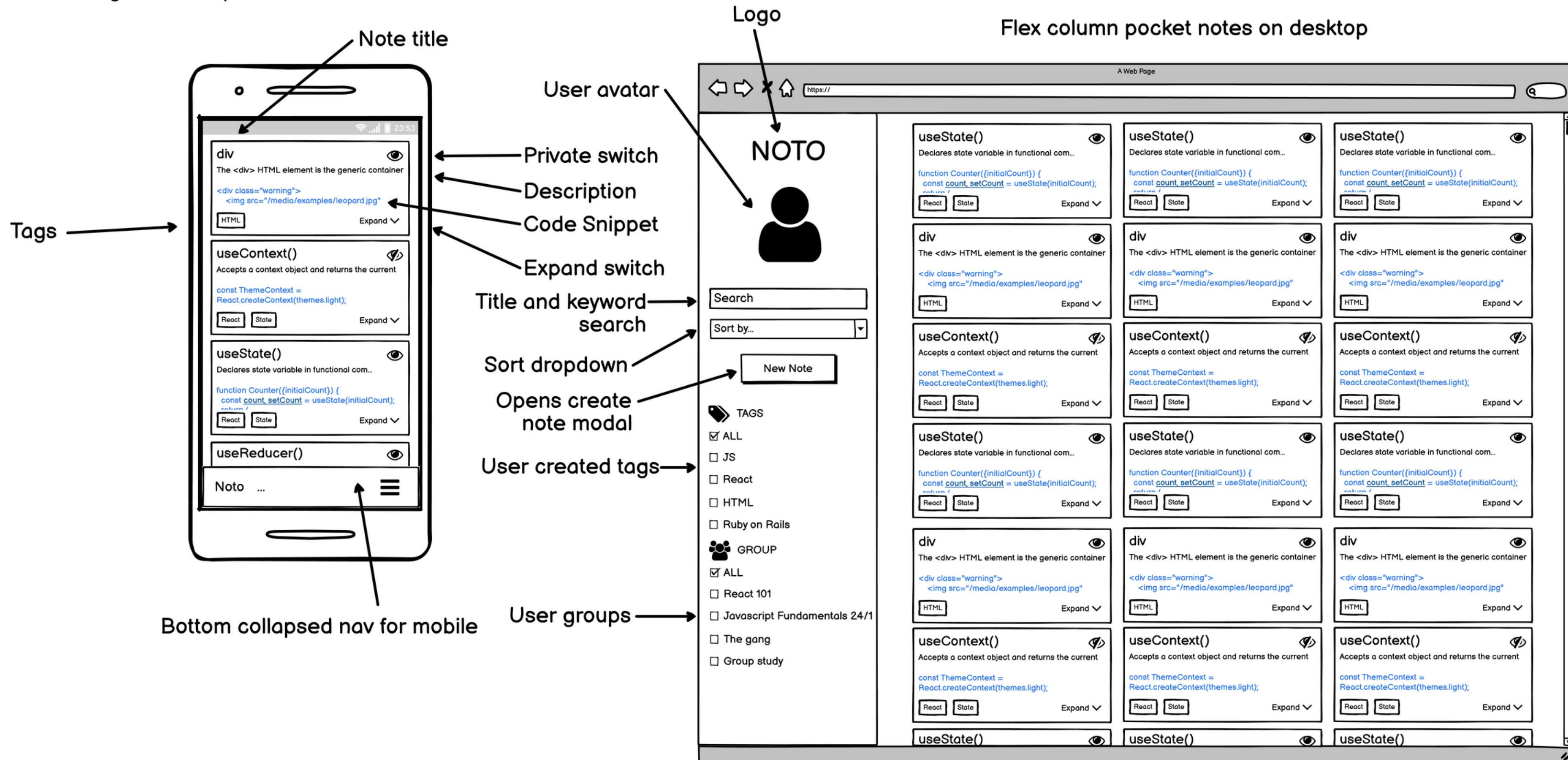
As **Stacy**, I want to...

- Add code snippets to my notes
- Sort and filter my notes
- Organize my notes by tags

Wireframes

Notes index

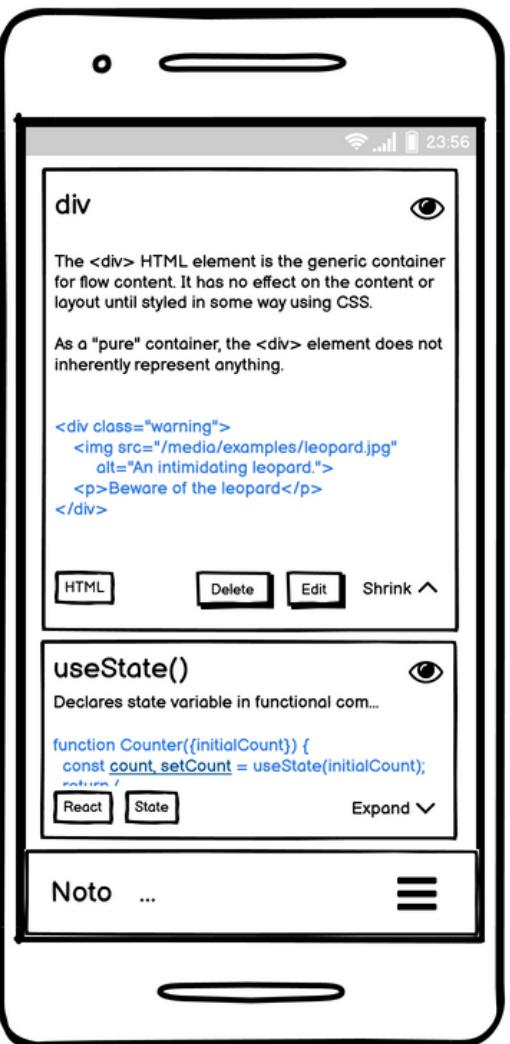
Single column pocket notes on tablet and mobile



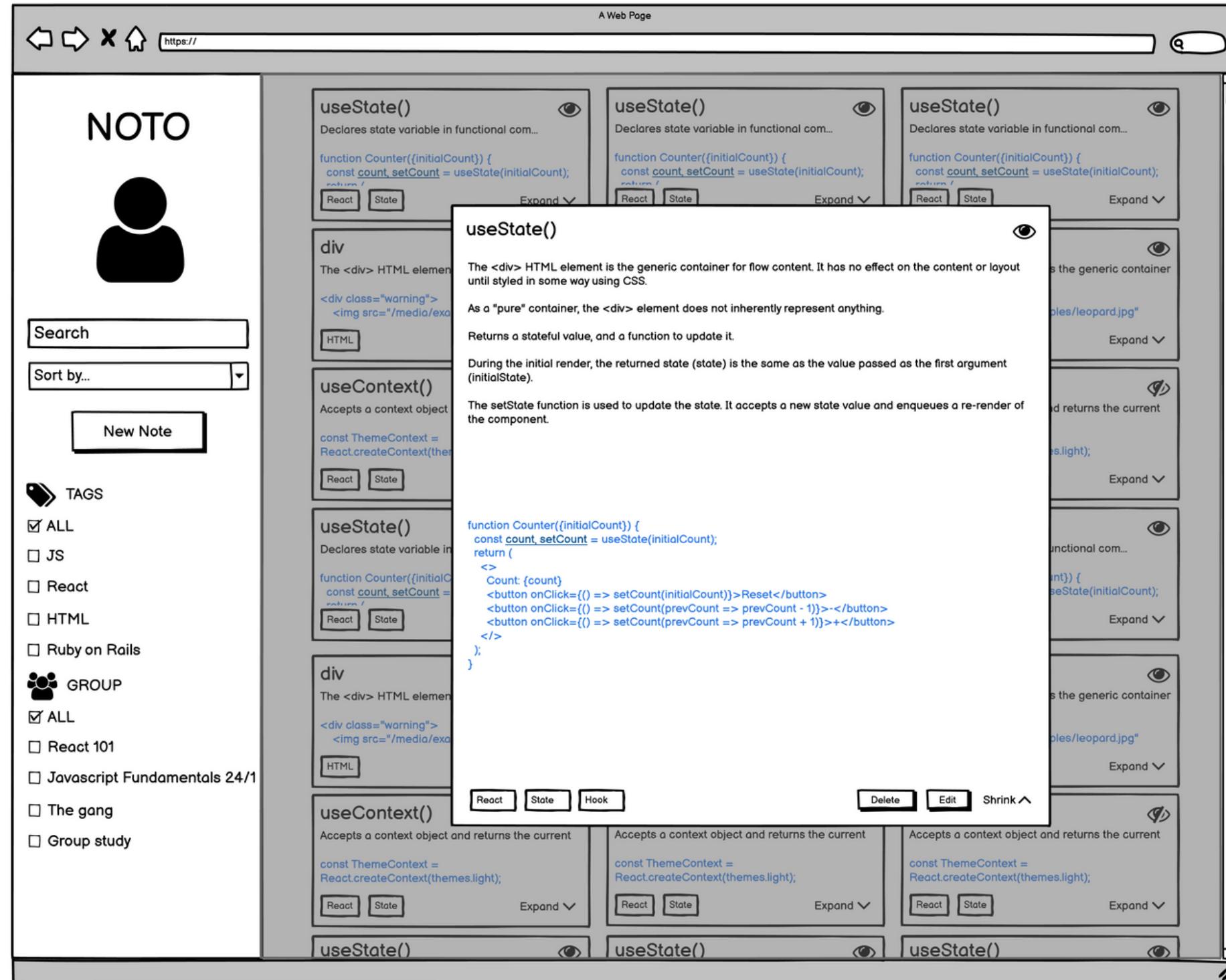
Wireframes

Expanded Note

Note expands to detailed content size on tablet and mobile



Detailed note expands to modal on desktop

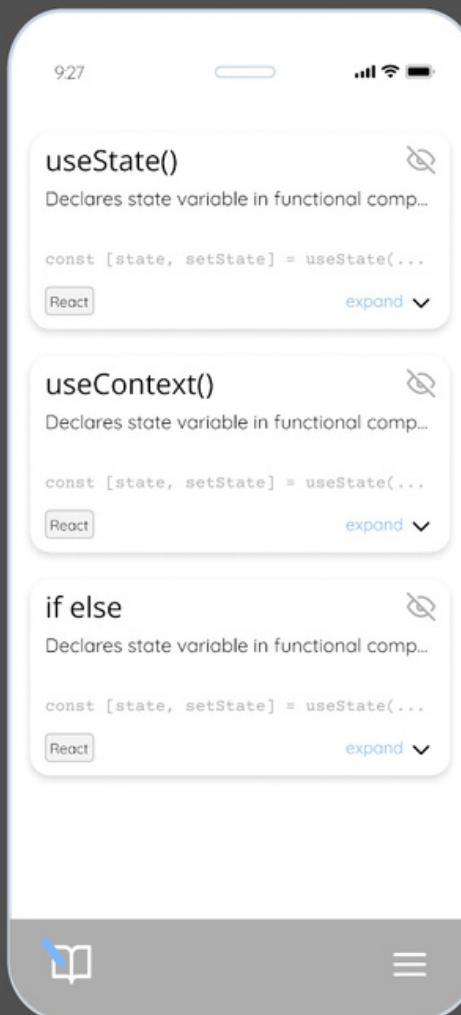


Prototype

MVP - Notes index

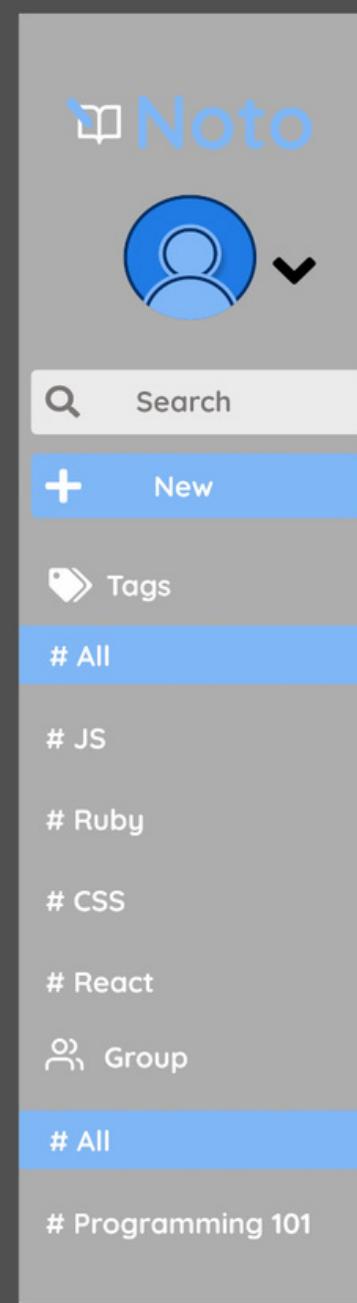
Pocket
notes

Nav

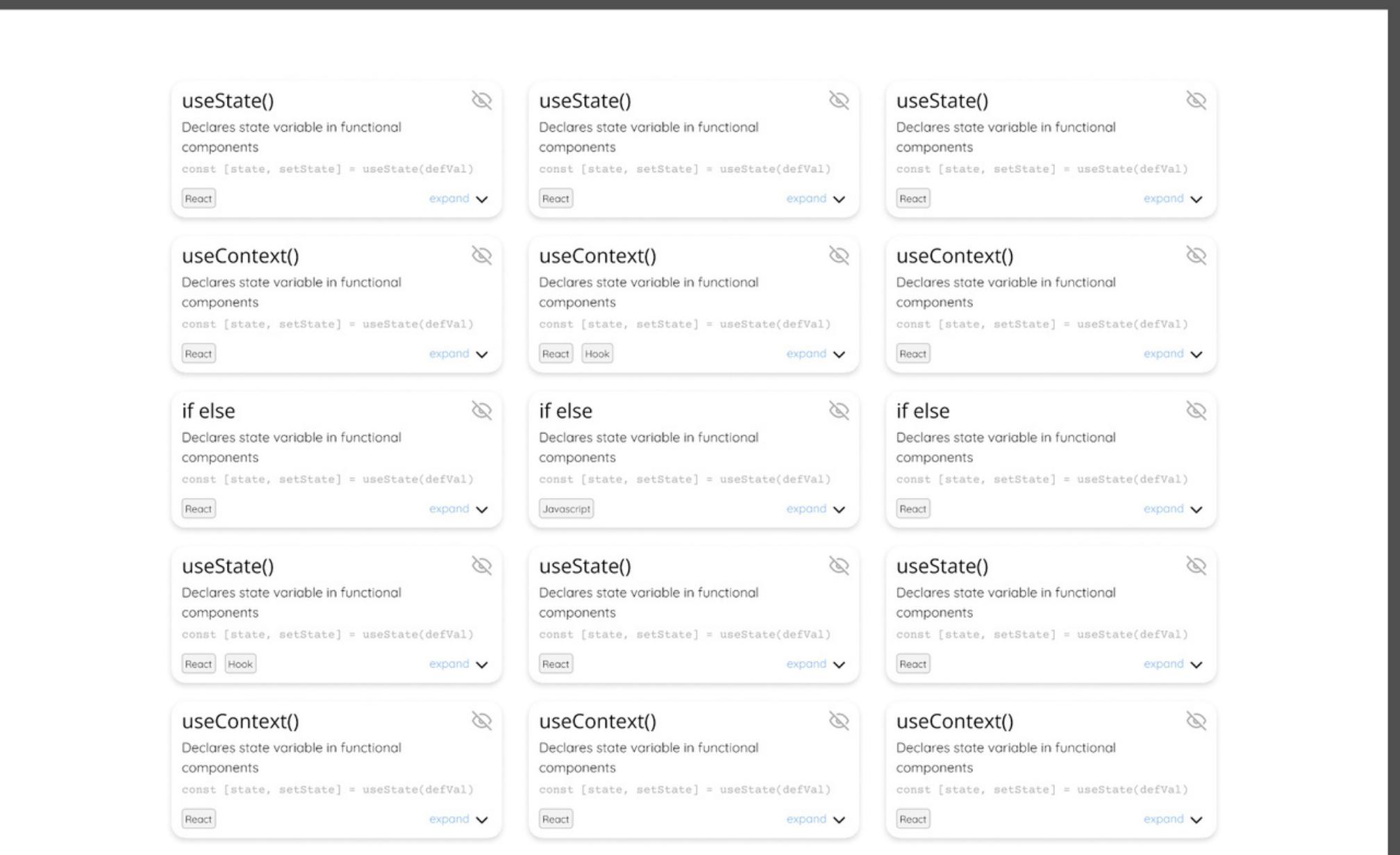


iPhone 13 - 390 x 844px

Nav



Pocket notes

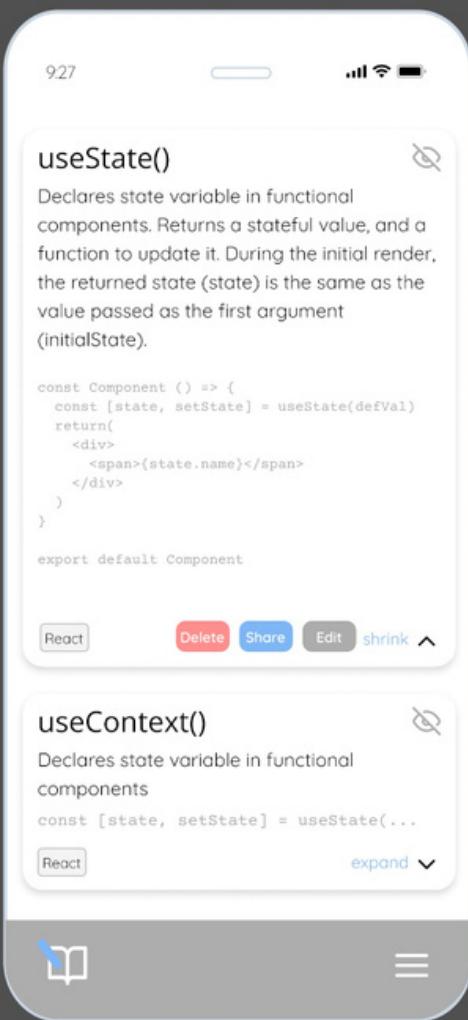


Desktop - 1920 x 1080px

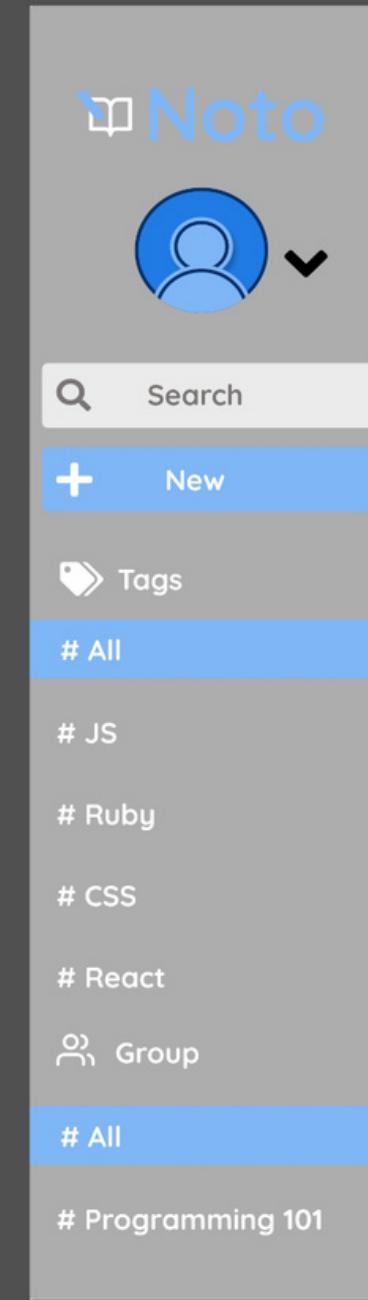
Prototype

MVP - Detailed note

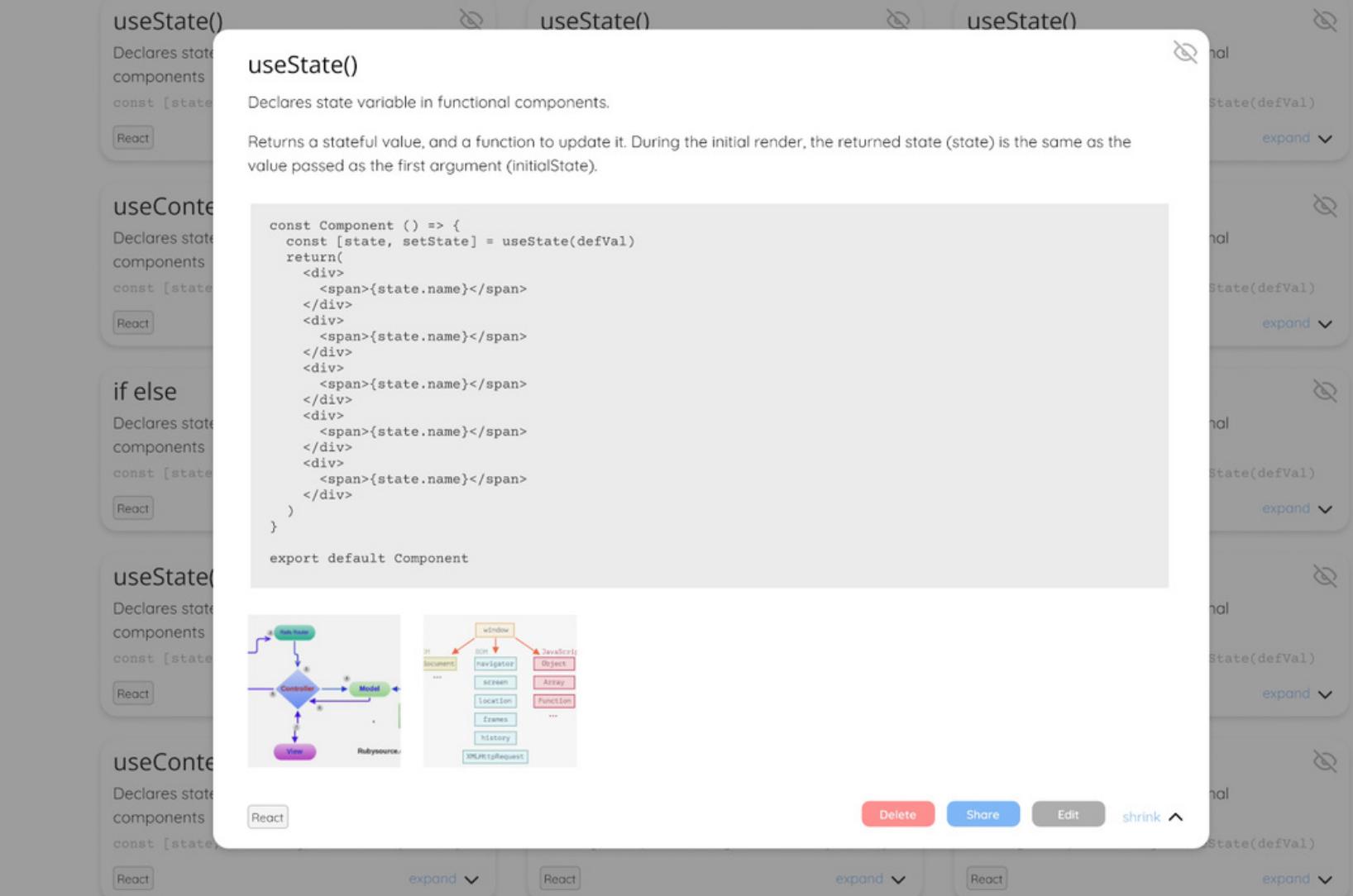
Expanded
Note



iPhone 13 - 390 x 844px



Expanded Note Modal



Desktop - 1920 x 1080px

Prototype

Version 2 - Live editing

Pocket notes

The screenshot shows the Noto app interface. On the left is a sidebar with a user icon, a search bar, a 'New' button, and a 'Tags' section containing '# All', '# JS', '# Ruby', '# CSS', '# React', and '# Group'. Below these are '# All' and '# Programming 101'. The main area displays a grid of notes. Each note card has a title, a brief description, and a 'React' tag. Some notes also have an 'expand' button. The notes are:

- useState()**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- useState()**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- useContext()**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- useContext()**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- if else**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- if else**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- useState()**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- useState()**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- useContext()**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`
- useContext()**
Declares state variable in functional components
`const [state, setState] = useState(defVal)`

Pre-filled form

The screenshot shows a 'Live Note' editor. At the top, there are 'Private note' and 'Public note' buttons. Below that is a 'React 101' section with the following content:

Declares state variable in functional components.
Returns a stateful value, and a function to update it. During the initial render, the returned state (state) is the same as the value passed as the first argument (initialState).

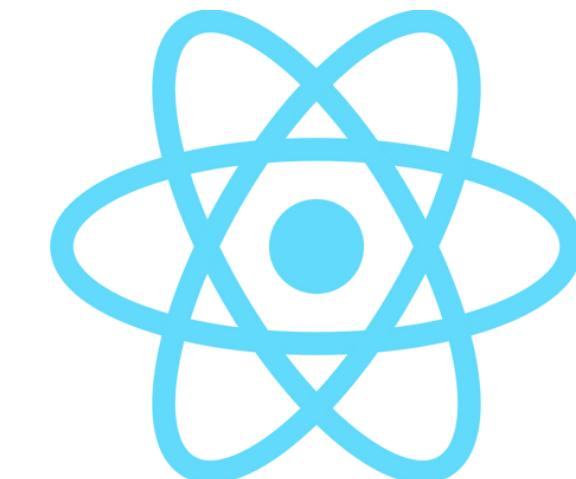
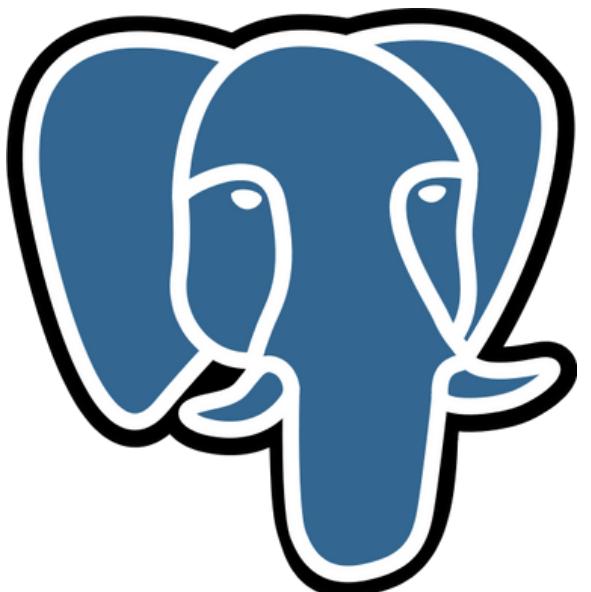
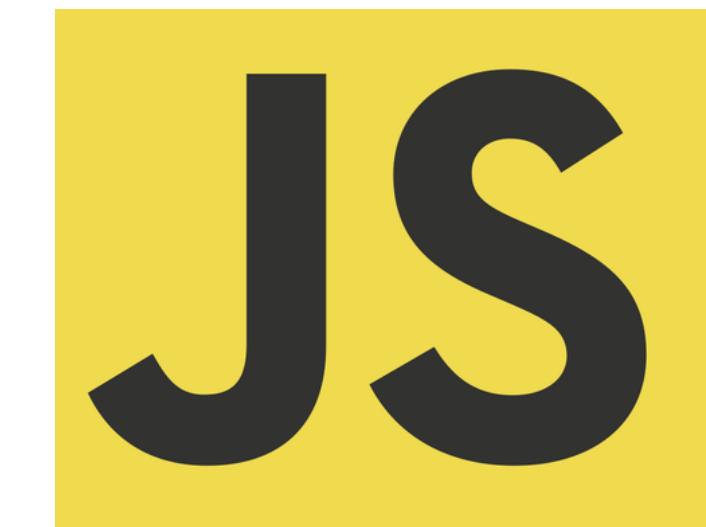
Below this is a code editor containing a component definition:

```
const Component () => {  
  const [state, setState] = useState(defVal)  
  return(  
    <div>  
      <span>{state.name}</span>  
    </div>  
    <div>  
      <span>{state.name}</span>  
    </div>
```

At the bottom are 'Add Image', 'Discard', and 'Add Note' buttons.

Utilizing state, edit notes live without separate edit & update forms

Tech Stack



create,
share,
collaborate

 **Noto**

