deeecode

5 CUSTOM REACT HOOKS

YOU WOULD LOVE TO TRY OUT

swipe left

deeecode

5 REACT HOOKS

1. REACT-HOOK-FORM

This hook provides an easy and declarative way to manage your forms (the inputs and errors).

```
import React from 'react';
import { useForm } from 'react-hook-form';

function App() {
  const {
    register,
    handleSubmit,
    formState: { errors },
  } = useForm();
  const onSubmit = (data) => console.log(data);

  return (
    <form onSubmit={handleSubmit(onSubmit)}>
      <input {...register('firstName')} />
      <input {...register('lastName', { required: true })} />
      <input {...register('age', { pattern: /\d+/ })} />

      {/* display errors */}
      {errors.lastName && <p>Last name is required.</p>}
      {errors.age && <p>Please enter number for age.</p>}

      <input type="submit" />
    </form>
  );
}
```

swipe left

5 REACT HOOKS

2. USE-CLIPPY

This hook makes it easy for you to manage data in the clipboard (for copying and pasting).

```
import useClippy from "use-clippy"

function Example() {
  const [ clipboard, setClipboard ] = useClipy()

  return (
    <div>
      <p>Data in clipboard is {clipboard}</p>

      <button onClick={() => setClipboard('deeecode')}>
        Update clipboard
      </button>
    </div>
  )
}
```

5 REACT HOOKS

3. USE-MEDIA

This hook tracks CSS media queries and you can use the state of media queries to conditionally render components.

```
import useMedia from 'use-media';

function Example() {
  const isWide = useMedia({ minWidth: 1000 });
  // or
  const isWide = useMedia('(min-width: 1000px)')

  return (
    <div>
      Screen is wide: {isWide ? '😃' : '😢'}
    </div>
  );
};
```

5 REACT HOOKS

4. USE-ONCLICKOUTSIDE

This hook listens for clicks outside an element and you can hide elements afterward.

```
import useOnClickOutside from 'use-onclickoutside'
import { useRef } from 'react'

export default function Modal({ close }) {
  const ref = useRef(null)

  useOnClickOutside(ref, close)

  return <div ref={ref}>{'Modal content'}</div>
}
```

swipe left

deeecode

5 REACT HOOKS

5. USE-DEBOUNCE

This hook is used for value and callback debouncing. Debouncing is a concept of reducing the rate a which functions are called.

```
import { useDebouncedCallback } from 'use-debounce';

function Input({ defaultValue }) {
  const [value, setValue] = useState(defaultValue);

  // Debounce callback
  const debounced = useDebouncedCallback(
    // function
    (value) => {
      setValue(value);
    },
    // delay in ms
    1000
  );

  return (
    <div>
      <input
        defaultValue={defaultValue}
        onChange={(e) => debounced(e.target.value)}
      />
      <p>Debounced value: {value}</p>
    </div>
  );
}
```