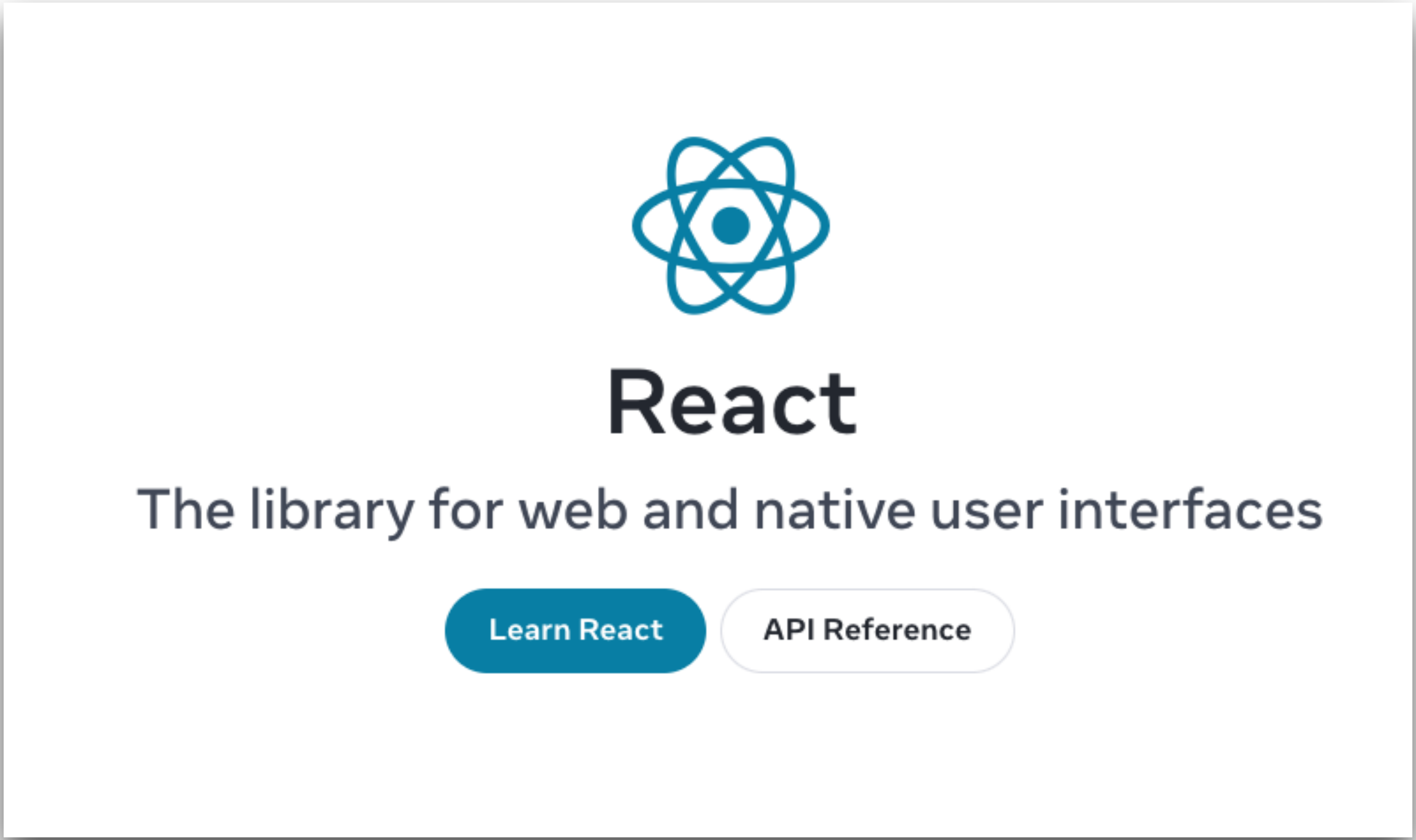
React Hooks Workshop

สยามชำนาญกิจ
SIAM CHAMNAN KIT

SHU HA RI
SOFTWARE DEVELOPMENT SCHOOL

React Hooks Workshop

React Hooks


React

The library for web and native user interfaces

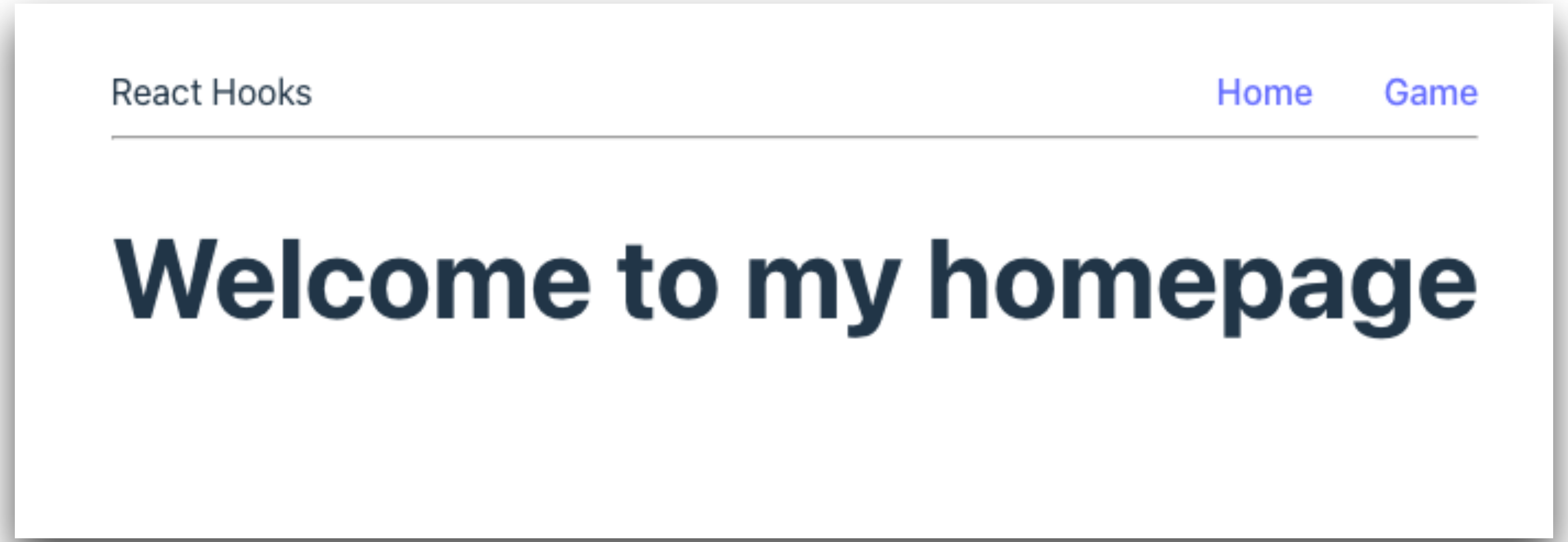
https://react.dev/reference/react/hooks

Start workshop

```
$git clone https://github.com/up1/workshop-react-hook.git
$cd workshop-react-hook
$npm install
$npm run dev
```

Start workshop

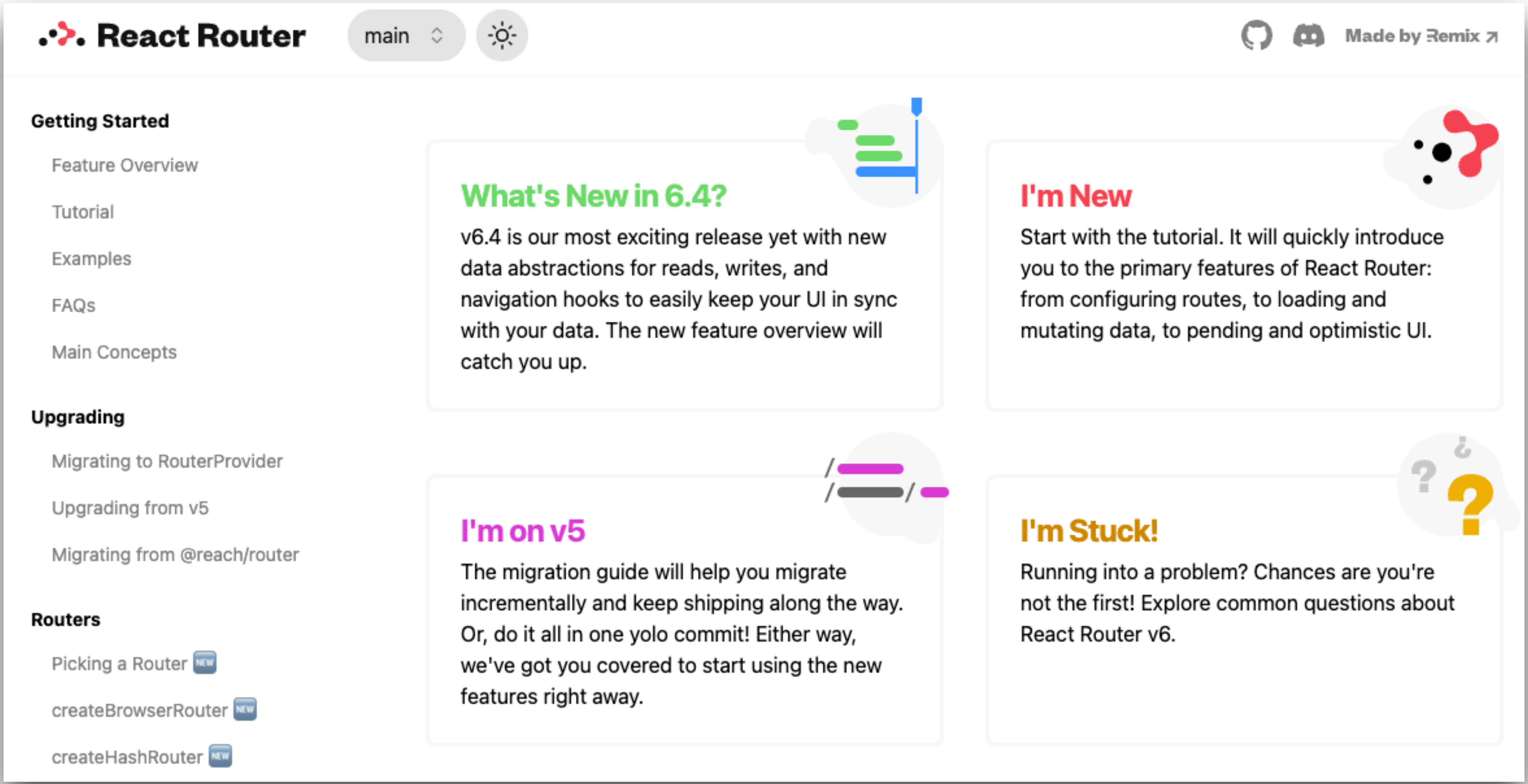
http://localhost:5173/



Welcome to my homepage

Client-side router

Working with React Router



React Router main

Made by Remix

Getting Started

- Feature Overview
- Tutorial
- Examples
- FAQs
- Main Concepts

Upgrading

- Migrating to RouterProvider
- Upgrading from v5
- Migrating from @reach/router

Routers

- Picking a Router NEW
- createBrowserRouter NEW
- createHashRouter NEW

What's New in 6.4?

v6.4 is our most exciting release yet with new data abstractions for reads, writes, and navigation hooks to easily keep your UI in sync with your data. The new feature overview will catch you up.

I'm New

Start with the tutorial. It will quickly introduce you to the primary features of React Router: from configuring routes, to loading and mutating data, to pending and optimistic UI.

I'm on v5

The migration guide will help you migrate incrementally and keep shipping along the way. Or, do it all in one yolo commit! Either way, we've got you covered to start using the new features right away.

I'm Stuck!

Running into a problem? Chances are you're not the first! Explore common questions about React Router v6.

https://reactrouter.com/en/main

Step 1 :: Create router

```
import {createBrowserRouter, RouterProvider} from "react-router-dom";


const router = createBrowserRouter([
  {
    path: "/",
    element: <HomePage/>,
  },
  {
    path: "/game",
    element: <GuessNumberPage/>,
  },
  {
    path: "/challenge",
    element: <ChallengePage/>,
  },
]);
```

Step 2 :: Router provider

```
function App() {
  return (
    <RouterProvider router={router}/>
  )
}
```

Step 3 :: Add navigation bar

```
import { NavLink } from "react-router-dom";

export const Navbar = () => {
  return (
    <>
      <div className={'nav-container'}>
        <div>React Hooks</div>
        <nav id="sidebar" className={'nav-item-container'}>
          <NavLink to="/" className={"nav-item"}>Home</NavLink>
          <NavLink to="/game">Game</NavLink>
        </nav>
      </div>
      <hr />
    </>
  )
}
```

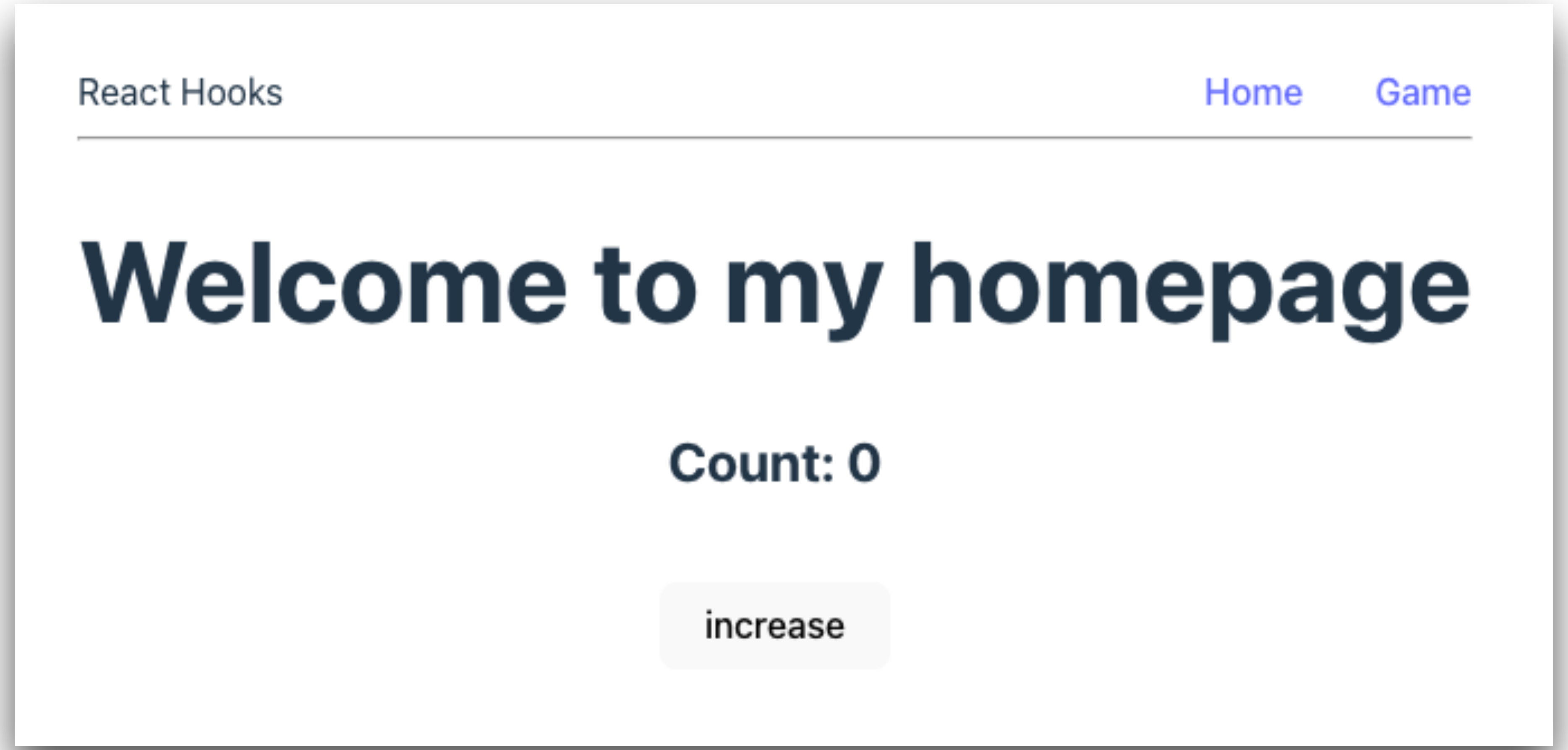
Working with React Hooks

HomePage

Home page

Working with counter (useState)
Retrieve data from APIs (useEffect)

Working with counter


React Hooks

Welcome to my homepage

Count: 0

https://react.dev/reference/react/useState

useState

```
const [count, setCount] = useState(0)
setCount((prevCount) => prevCount+1)
```

https://react.dev/reference/react/useState

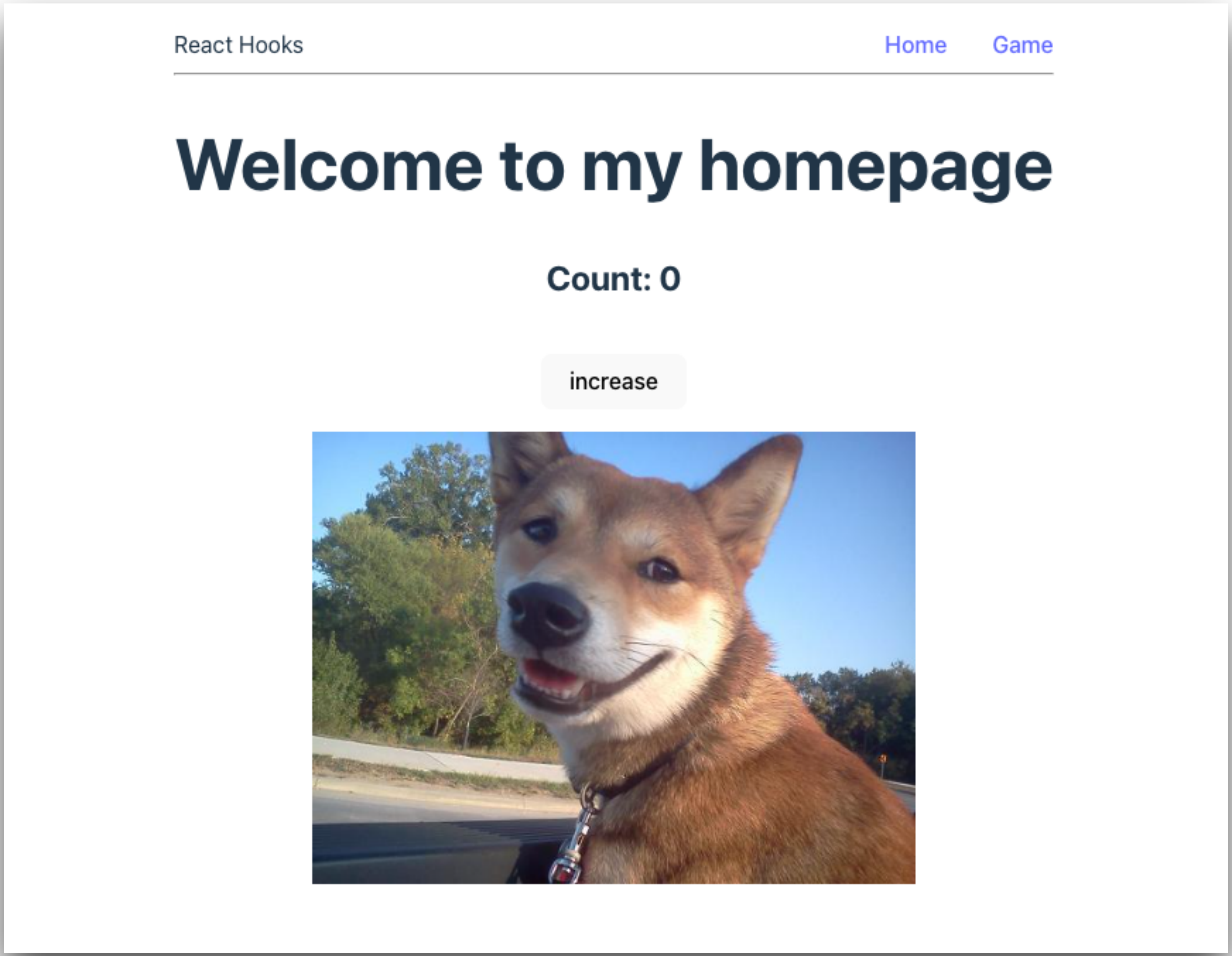
useState

```
import { useState } from "react";

export const HomePage = () => {
  const [count, setCount] = useState(0)
  return (
    <>
      <Navbar />
      <h1>Welcome to my homepage</h1>

      <h2>Count: {count}</h2>
      <button onClick={() => {
        setCount(count + 1)
      }}>increase
      </button>
  )
}
```

Retrieve data from APIs

Get list of image from APIs
Show image (automatic with intervals)

Retrieve data from APIs

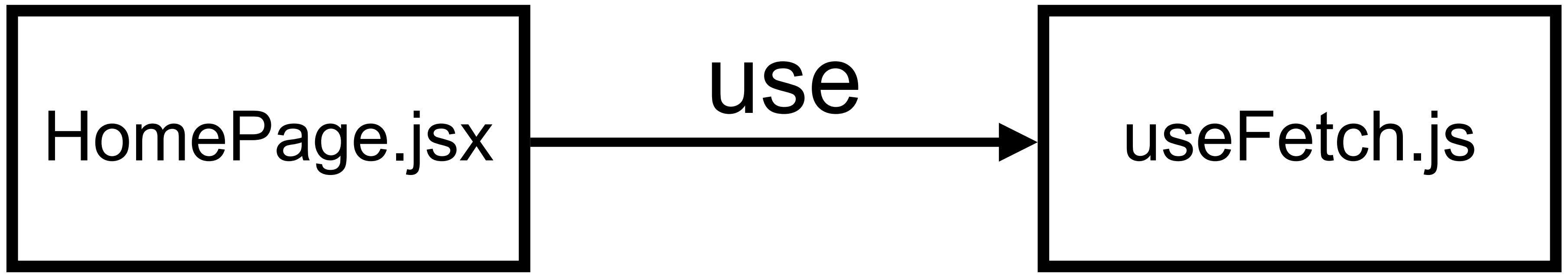


Welcome to my homepage

Count: 0

https://shibe.online/api/shibes?count=10

Retrieve data from APIs



useFetch with useEffect()

```
import { useEffect, useState } from "react";

export const useFetch = (url, initialValue = null) => {
    const [result, setResult] = useState(initialValue)

    useEffect(() => {
        fetch(url).then(async res => {
            const json = await res.json()
            setResult(json)
        })

    }, [url]);

    return result
}
```

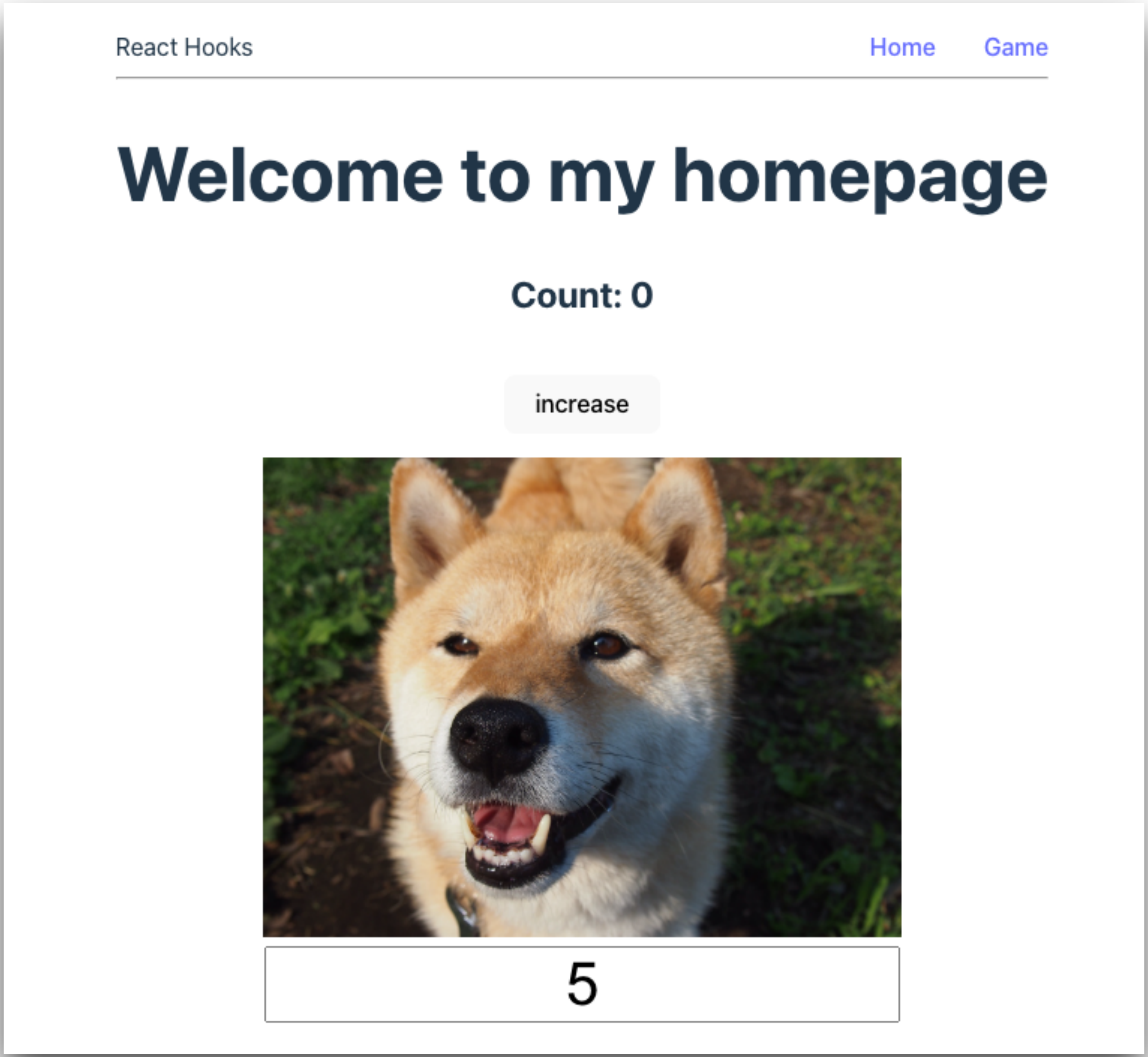
https://react.dev/reference/react/useEffect

HomePage

Get list of image from APIs

```
import { useFetch } from "../hooks/useFetch.js";

export const HomePage = () => {
  const [count, setCount] = useState(0)
  const coverImages = useFetch(`https://shibe.online/api/shibes?count=10`, [])
  return (
    <>
      <Navbar />
      <div>
        <img height={320} src={coverImages[0]}/>
      </div>
    </>
  )
}
```

Show image with interval time



Welcome to my homepage

Count: 0

5

HomePage

Show image with interval time

```
const COVER_COUNT = 10
const MIN_DURATION = 1

const [currentIdx, setCurrentIdx] = useState(0)
  const [transitionDuration, setTransitionDuration] = useState(1)

  useEffect(() => {
    const duration = transitionDuration < MIN_DURATION
            ? MIN_DURATION : transitionDuration
    const interval = setInterval(() => {
      setCurrentIdx(prevCurrentIdx => (prevCurrentIdx + 1) % 10)
    }, duration * 1000)

    return () => {
      console.log('cleanup')
      clearInterval(interval)
    }
  }, [transitionDuration]);
```

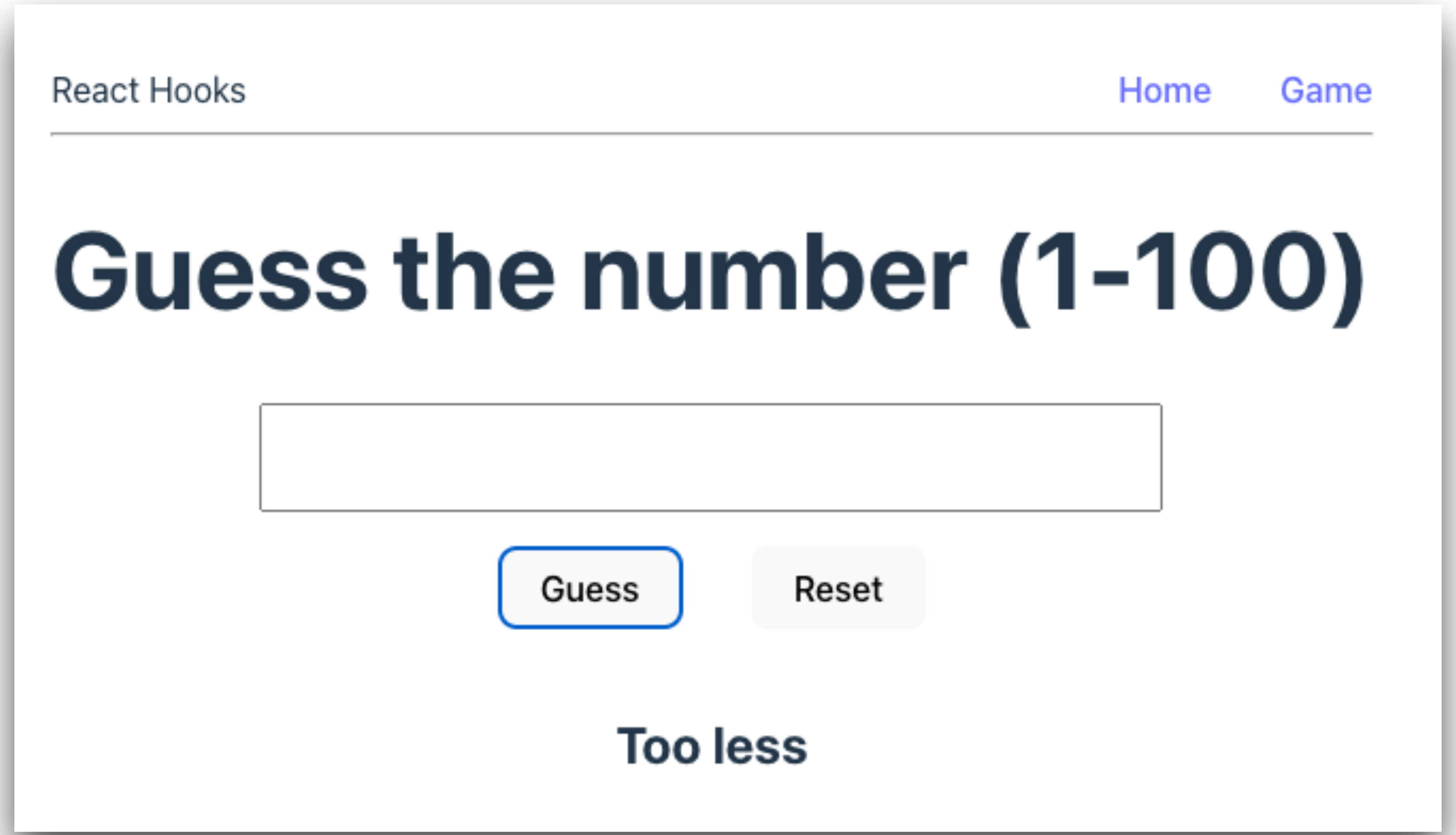
HomePage

Show image and update duration time

```
return (
    <>
      <Navbar />

      <div>
        <img height={320} src={coverImages[currentIdx]}/>
      </div>
      <input value={transitionDuration} onChange={(e) => {
        setTransitionDuration(e.target.value)
      }}/>
    </>
  )
```

Guess Number

Guess Number

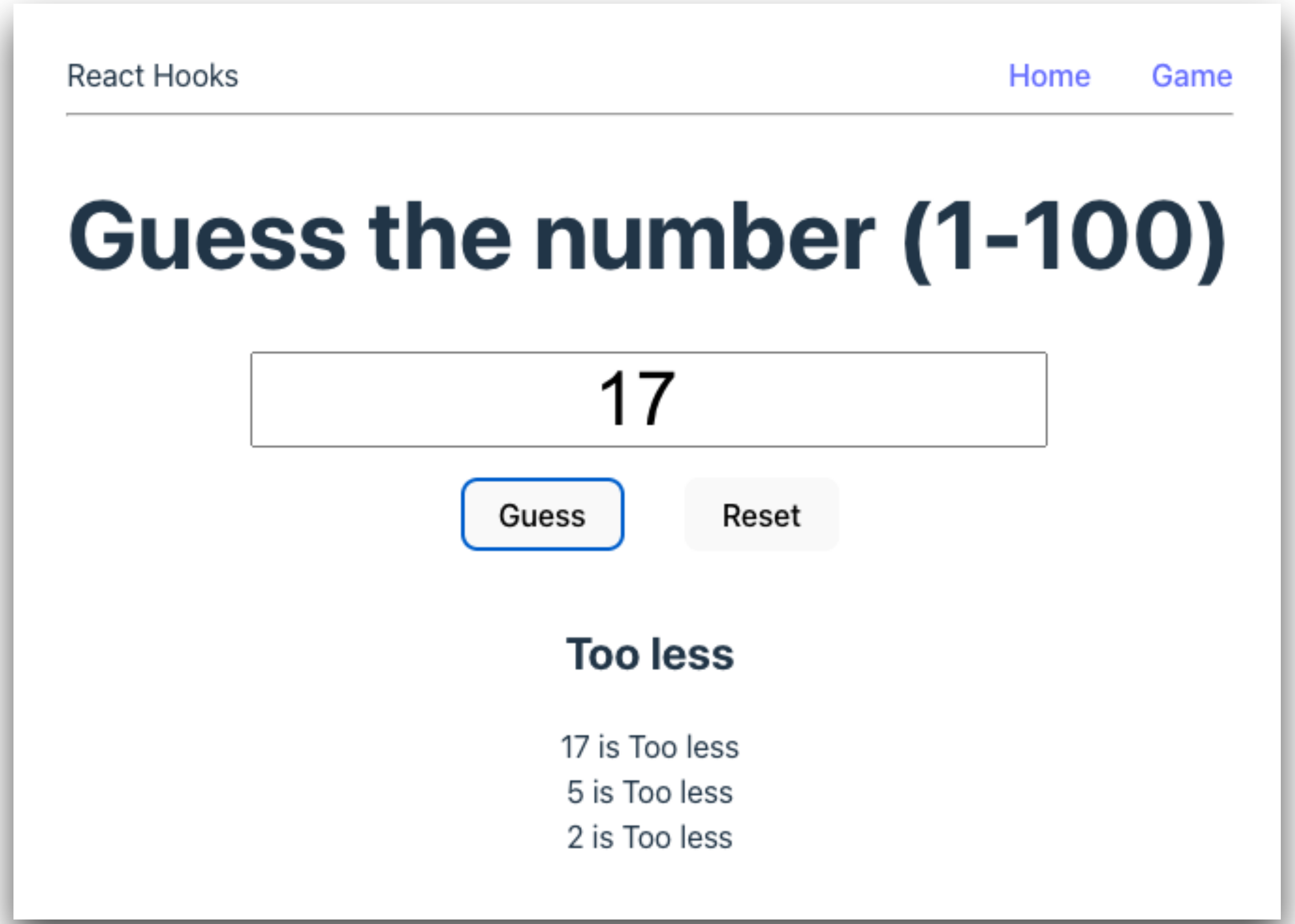


Guess the number (1-100)

Too less

Guess Number

Show list of guess numbers



Guess the number (1-100)

Too less

17 is Too less
5 is Too less
2 is Too less

GuessNumberPage

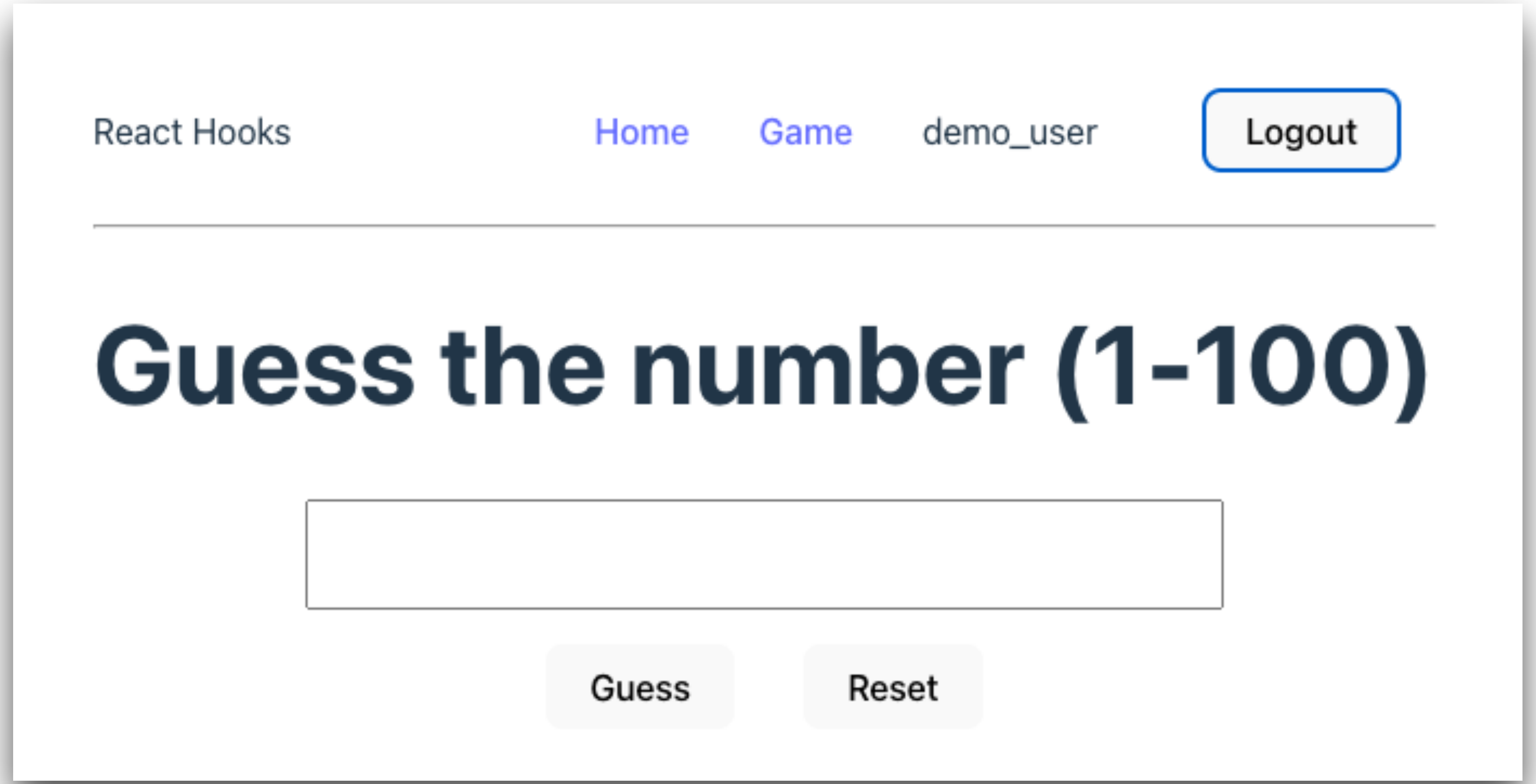
Create a new state to store list of guess numbers

```
import {v4 as uuidv4} from 'uuid';


const [guessedNumbers, setGuessedNumbers] = useState([])

// Guess
setGuessedNumbers([{ id: uuidv4(), number: guessingNumber,
hint }, ...guessedNumbers])

// Reset
setGuessedNumbers([])

// Return
{guessedNumbers.map((item) => (<div key={item.id}>
        {item.number} is {item.hint}
      </div>))}
```

Working with useContext()

https://react.dev/reference/react/useContext

Share data with useContext



Authenticate with useContext

AuthContext.js	Create context
useAuth.js	Create context provider to manage authen process
App.jsx	Config context provider to AuthContext
Navbar.jsx	useContext(AuthContext)

AuthenContext.js

Create a new context

```
import { createContext } from "react";

export const AuthContext = createContext()
```

useAuth.js

Custom hooks for auth process (login, logout)

```
import { useContext, useState } from "react";
import { AuthContext } from "../contexts/AuthContext.js";

export const useAuth = () => useContext(AuthContext)

export const useAuthContext = () => {
    const [user, setUser] = useState('')
    const [isLoggedIn, setIsLoggedIn] = useState(false)
    const login = (user) => {
        setUser(user)
        setIsLoggedIn(true)
    }
    const logout = () => {
        setUser('')
        setIsLoggedIn(false)
    }

    return { user, isLoggedIn, login, logout }
}
```

App.jsx

Config provider context

```
function App() {

  const authContextValue = useAuthContext()

  return (
    <AuthContext.Provider value={authContextValue}>
      <RouterProvider router={router} />
    </AuthContext.Provider>
  )

}
```

Navbar.jsx

Use data from AuthContext

```
import { useContext } from 'react';
import { AuthContext } from "../contexts/AuthContext";

export const Navbar = () => {

  const authContext = useContext(AuthContext);

  return (
    {authContext.isLoggedIn && <div>{authContext.user}</div>}

    {authContext.isLoggedIn
      ? <button onClick={() => authContext.logout()}>Logout</button>
      : <button onClick={() => authContext.login(‘demo_user')}>
          Login
        </button>
      }
   )
}
```

Problem !!

Try to refresh your browser

Keep data in local storage !!

AuthContext.js	Create context
useAuth.js	Create context provider to manage authen process
App.jsx	Config context provider to AuthContext
Navbar.jsx	useContext(AuthContext)

useAuth.js

Store and remove data in local storage

```
export const useAuthContext = () => {


    const login = (user) => {
        setUser(user)
        setIsLoggedIn(true)
        localStorage.setItem("user_data", JSON.stringify(
                                  { user: user, isLoggedIn: true }))
    }

    const logout = () => {
        setUser('')
        setIsLoggedIn(false)
        localStorage.removeItem("user_data")
    }
}
```

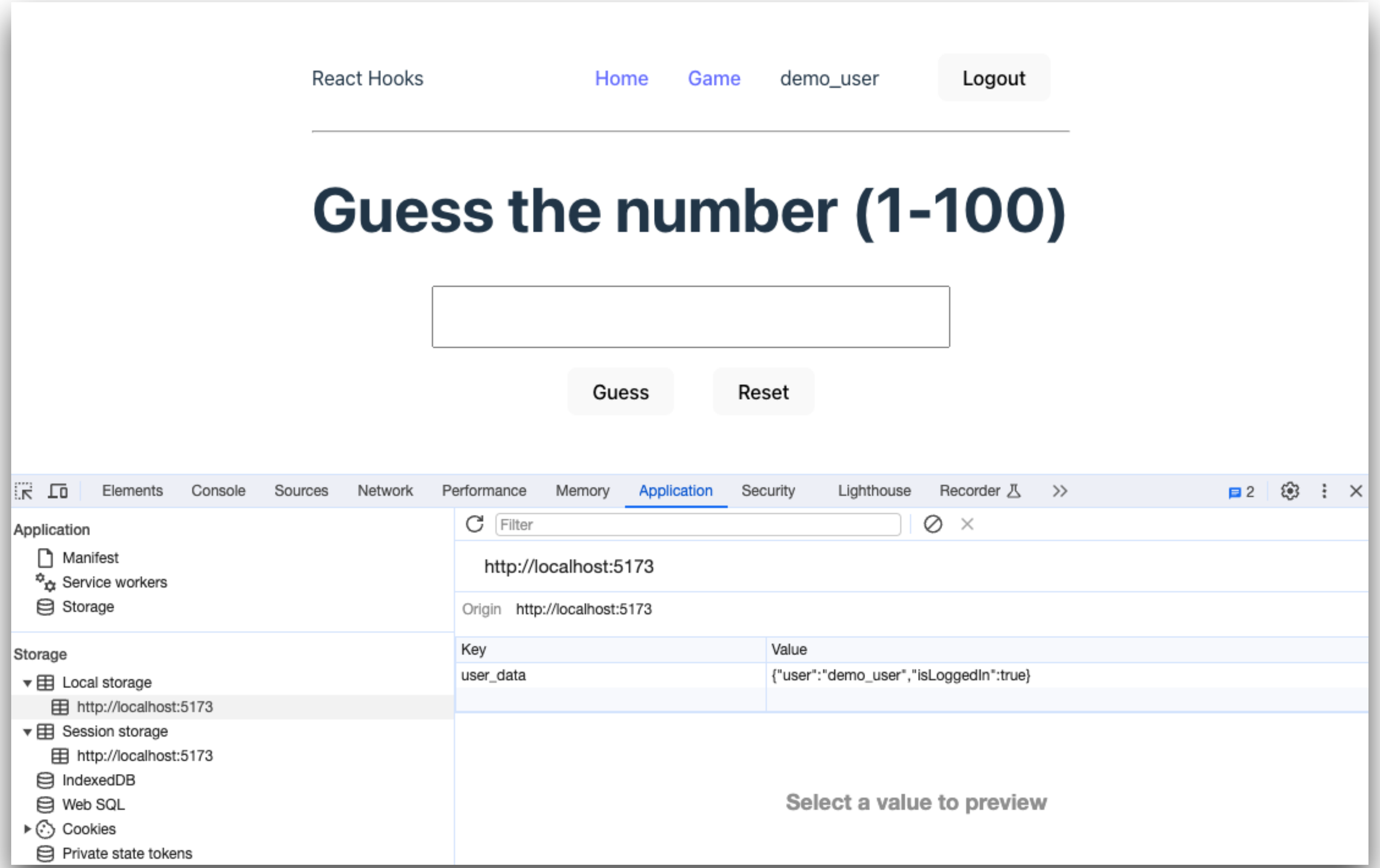
useAuth.js

Get data from local storage with useEffect

```
export const useAuthContext = () => {

    useEffect(() => {
        const userData = JSON.parse(localStorage.getItem('user_data'))

        if (userData) {
            setUser(userData.user)
            setIsLoggedIn(userData.isLoggedIn)
        }
    }, [])

}
```

Try to refresh your browser again !!

Working with local storage



http://localhost:5173

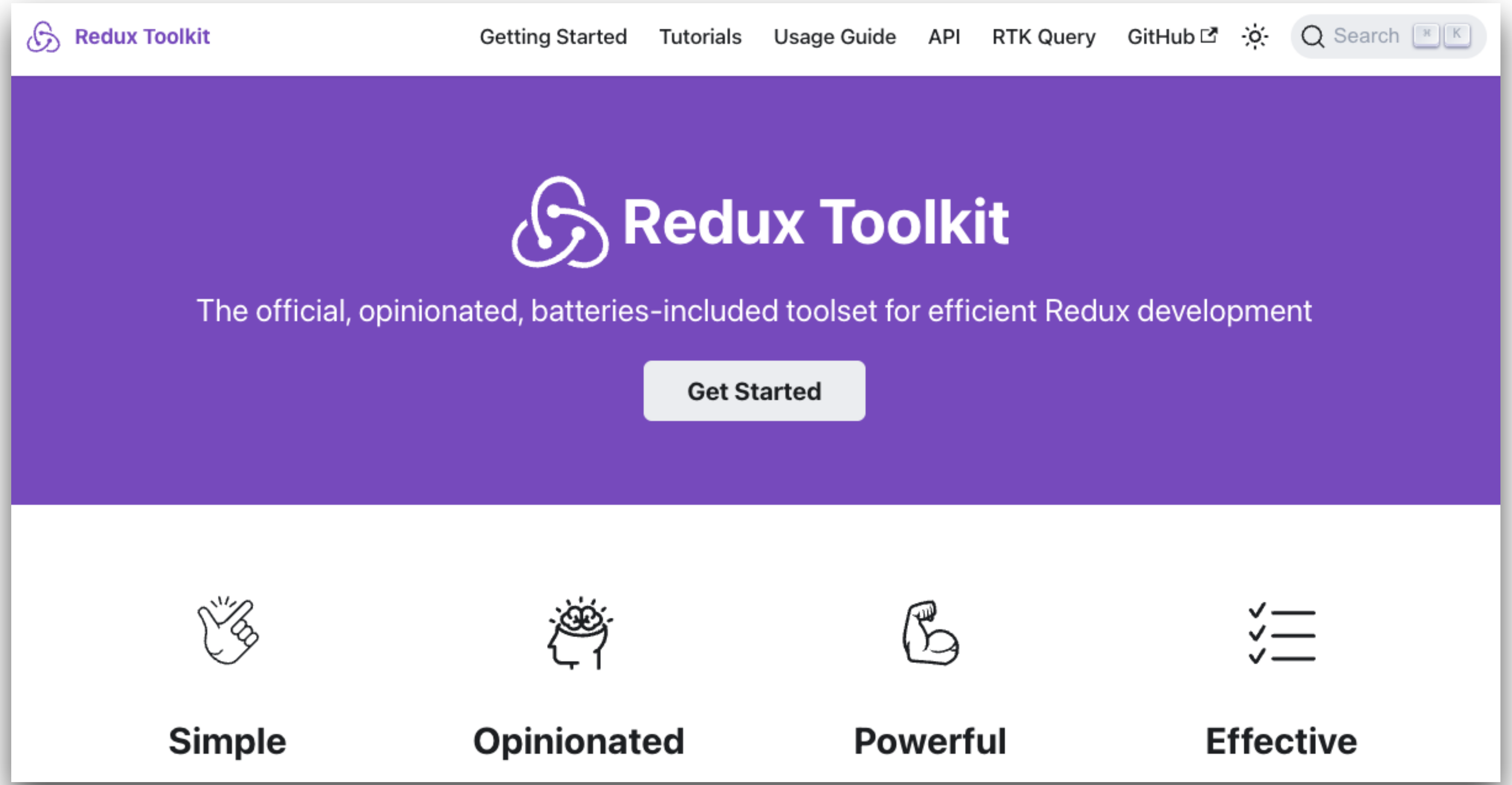
Origin http://localhost:5173

Key	Value
user_data	{"user":"demo_user","isLoggedIn":true}

Manage state with Redux

https://redux-toolkit.js.org/

Manage state with Redux

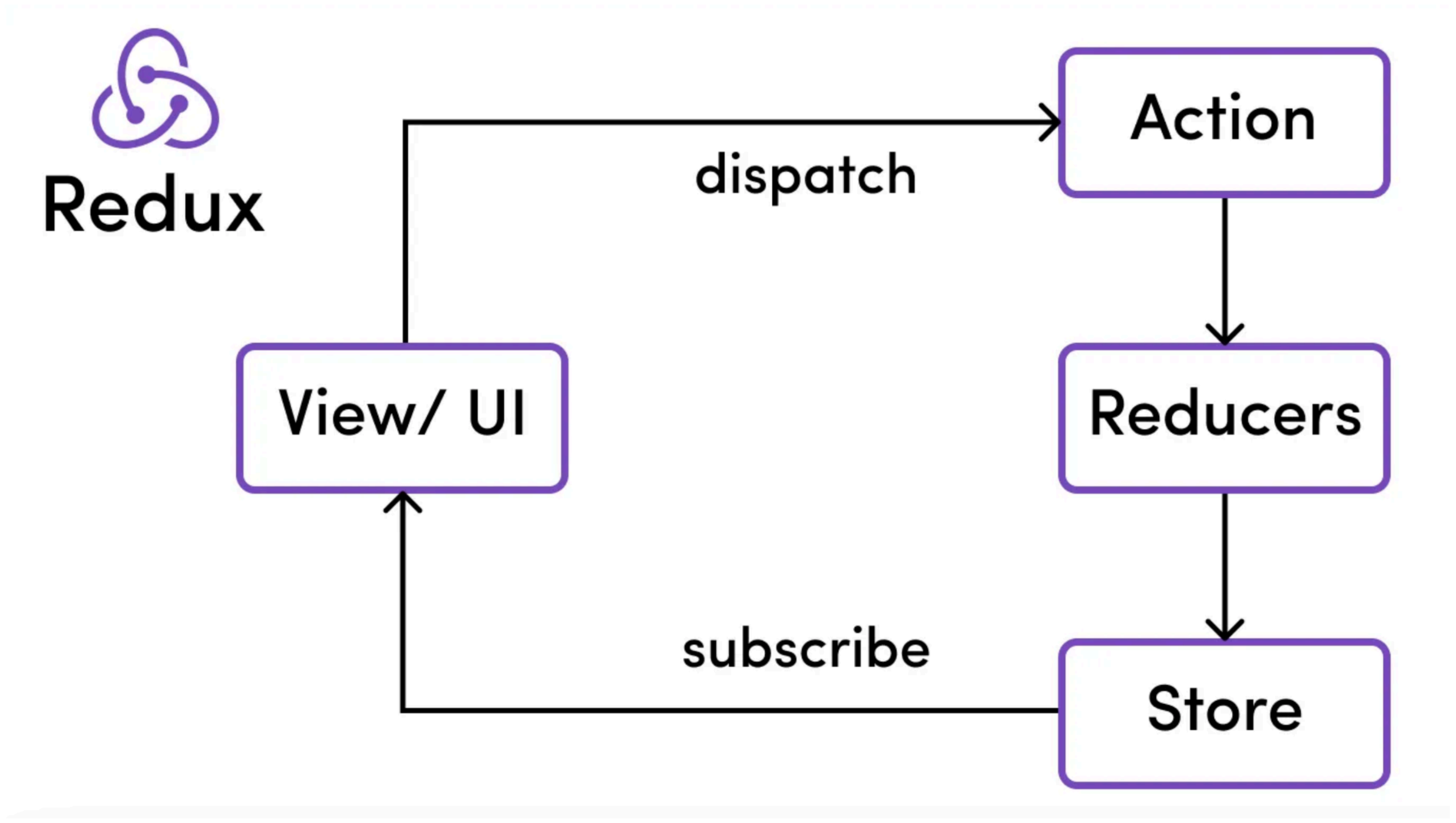


Redux Toolkit

The official, opinionated, batteries-included toolset for efficient Redux development

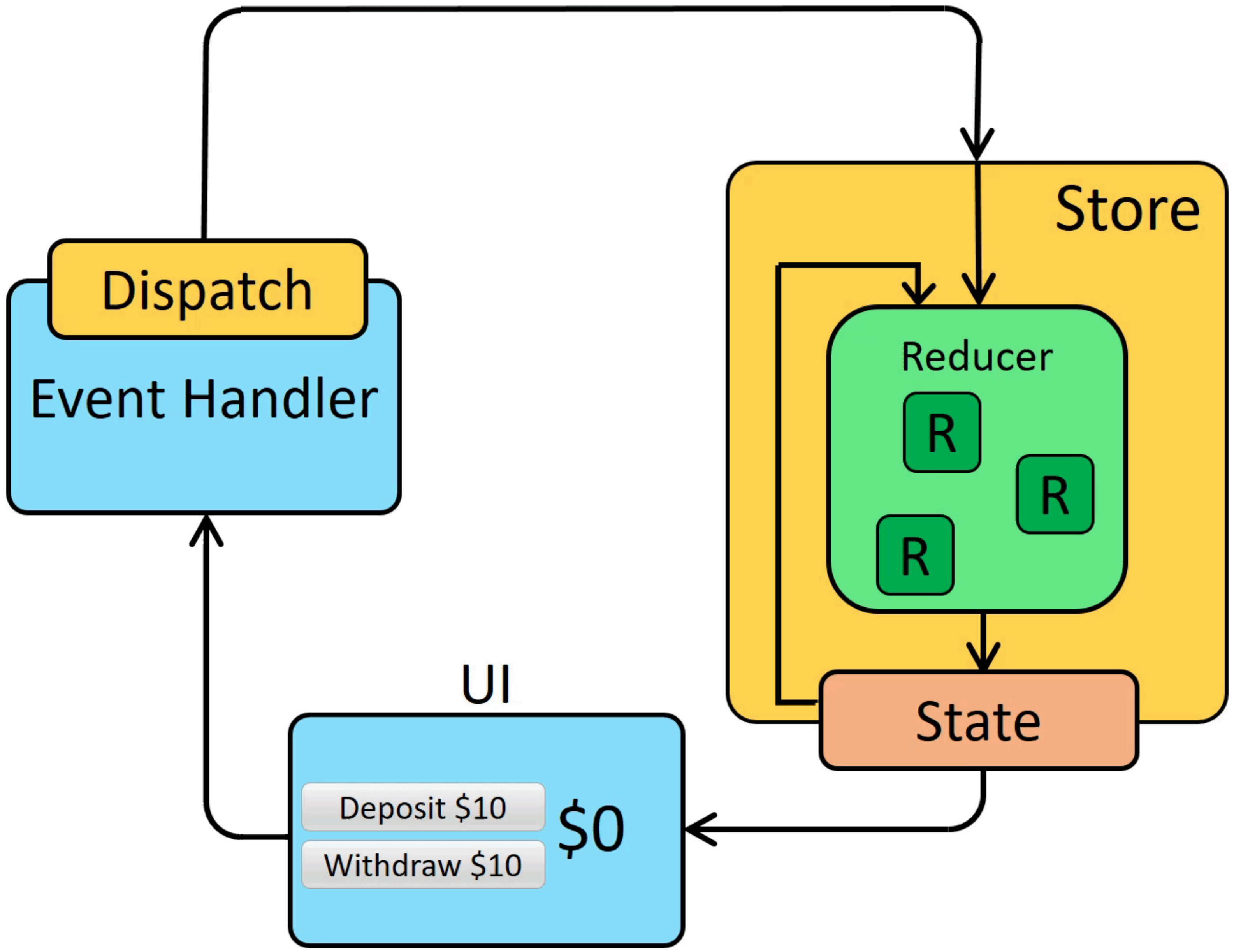
https://redux-toolkit.js.org/

Redux



https://redux.js.org/tutorials/fundamentals/part-1-overview

Redux flow



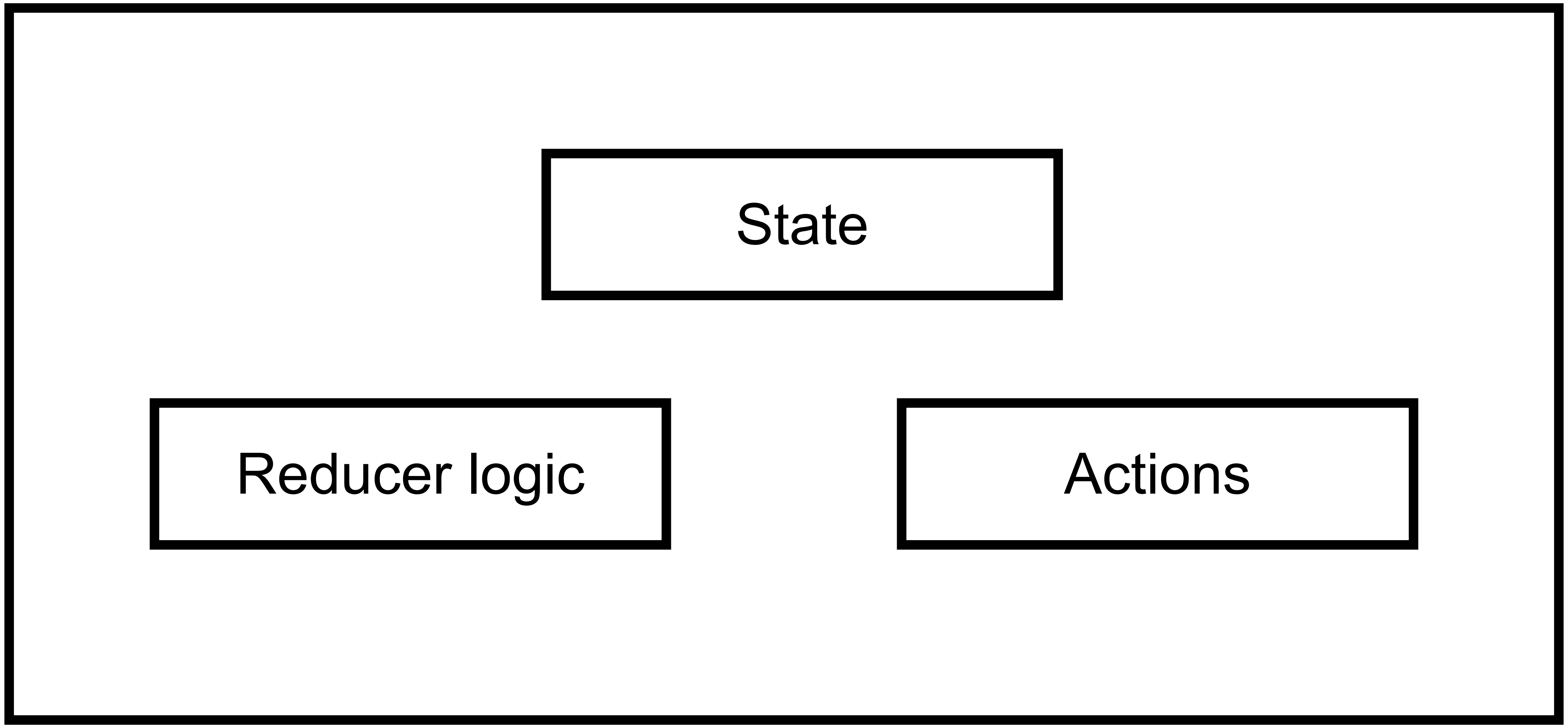
https://redux.js.org/tutorials/fundamentals/part-1-overview

Install Redux toolkit

```
$npm i -S @reduxjs/toolkit react-redux
```

https://redux.js.org/tutorials/quick-start#install-redux-toolkit-and-react-redux

Slice in Redux Toolkit



useSlice.js

Create slice for authen state (initial state, reducers, action)

```
import { createSlice } from '@reduxjs/toolkit'

const initialState = {
  user: '',
  isLoggedIn: false
}

const authSlice = createSlice({
  name: 'auth',
  initialState,
  reducers: {
    login: (state, action) => {
      state.user = action.payload
      state.isLoggedIn = true
    },
    logout: (state) => {
      state.user = ''
      state.isLoggedIn = false
    },
    setUser: (state, action) => {
      state.user = action.payload.user
      state.isLoggedIn = action.payload.isLoggedIn
    }
  }
})

export const { login, logout, setUser } = authSlice.actions

export default authSlice.reducer
```

store.js

Create store to hold the state

```
import { configureStore } from '@reduxjs/toolkit'
import authReducer from './authSlice'

export default configureStore({
    reducer: {
        auth: authReducer
    }
})
```

Navbar.jsx

Use the useSelector and useDispatcher from redux

```
import {login, logout} from '../redux/authSlice.js'
import { useDispatch, useSelector } from 'react-redux'

export const Navbar = () => {

  const dispatch = useDispatch()
  const user = useSelector(state => state.auth.user)
  const isLoggedIn = useSelector(state => state.auth.isLoggedIn)

  return (
    <>
       {user && <div>{user}</div>}
       {isLoggedIn ? <button onClick={() => dispatch(logout())}>Logout</
button> :
            <button onClick={() => dispatch(login('demo_user'))}>Login</
button>}

    </>
  )
}
```

Problem !!

Try to refresh your browser

useSlice.js

Store data in local storage

```
// Try to load the auth state from localStorage
let savedState = localStorage.getItem('auth')
savedState = savedState ? JSON.parse(savedState)
     : { user: '', isLoggedIn: false }
const initialState = savedState

const authSlice = createSlice({
  initialState,
  reducers: {
    login: (state, action) => {
      localStorage.setItem('auth',
         JSON.stringify({ user: action.payload, isLoggedIn: true }))
    },
    logout: (state) => {
      localStorage.removeItem('auth')
    },
    setUser: (state, action) => {
      localStorage.setItem('auth',
         JSON.stringify({ user: action.payload.user,
                          isLoggedIn: action.payload.isLoggedIn }))
    }
  }
})
```

Try to refresh your browser again !!

Q/A