Local Storage

The local storage is supported by most of the browsers. You can check the browser compatibility and read even more about the topic in [the official documentation](https://developer.mozilla.org/en/docs/Web/API/Window/localStorage). The usage of the local storage is fairly straight forward. In your JavaScript code, running in the browser, you should have access to the localStorageobject. The object has a setter and getter to store and retrieve data from the object.

// setter

localStorage.setItem('myData', data);

// getter

localStorage.getItem('myData');

Once you close the browser and open the app again, you will find the data still in the object.

import React from 'react';

class App extends React.Component {

constructor(props) {

super(props);

this.state = { hits: null };

}

onSearch = (e) => {

e.preventDefault();

const { value } = this.input;

if (value === '') {

return;

}

const cachedHits = localStorage.getItem(value);

if (cachedHits) {

this.setState({ hits: JSON.parse(cachedHits) });

return;

}

fetch('<https://hn.algolia.com/api/v1/search?query=>' + value)

.then(response => response.json())

.then(result => this.onSetResult(result, value));

}

onSetResult = (result, key) => {

localStorage.setItem(key, JSON.stringify(result.hits));

this.setState({ hits: result.hits });

}

render() {

return (

<div>

<h1>Search Hacker News with Local Storage</h1>

<p>

There shouldn't be a second network request,

when you search for something twice.

</p>

<form type="submit" onSubmit={this.onSearch}>

<input type="text" ref={node => this.input = node} />

<button type="button">Search</button>

</form>

{

this.state.hits &&

this.state.hits.map(item => <div key={item.objectID}>{item.title}</div>)

}

</div>

);

}

}

export default App;