Word to Smiley Convertor

**What Does Application Does ?**

I have designed an App that takes word and returns the smiley pertaining to word on Click of Button using React Hooks for State management and Form Created using React Form .



**Prerequisites knowledge**

A)-**HTML, CSS, and JavaScript** (Fundamentals to get started)

Some Materials you can look - W3SCHOOLS, FREECODECAMP

B)-Some ES6 concepts of JavaScript like -

**Arrow Functions**

//The old way

function greet ()

{

console.log ('Hello');

}

var greet1 = function(){

console.log ('hello');

}

//The ES6 way

var greet2 = () => {

console.log ('hello');

}

C) **Let and Const**

Let — is used to define a local variable limiting their scope to the block .

Const — constant variable whose values cannot be changed.

D) **Code Editors**

1. Visual Studio Code — VSCODE

2. Sublime Text

3. Atom

**Steps To create React JS Project**

1) Install npm and Node

2) npx create-react-app my-app

cd my-app

npm start

**React Life Cycle**

React has a lifecycle which you can monitor and manipulate during its three main phases.

React has a lifecycle which you can monitor and manipulate during its three main phases.

The three phases are: **Mounting**, **Updating**, and **Unmounting**

React has four built-in methods that gets called, in this order, when mounting a component:

**Mounting**

These methods are called in the following order when an instance of a component is being created and inserted into the DOM:

constructor()  
static getDerivedStateFromProps()  
render()  
componentDidMount()

**render()**

The render() method is required, and is the method that actually outputs the HTML to the DOM. The render() method is required and will always be called, the others are optional and will be called if you define them.

**componentDidMount()**

The componentDidMount() method is called after the component is rendered

**What is a Hook?** A Hook is a special function that lets you “hook into” React features. For example, useState is a Hook that lets you add React state to function components.

Components might perform some data fetching in componentDidMount and componentDidUpdate. However, the same componentDidMount method might also contain some unrelated logic that sets up event listeners, with clean-up performed in componentWillUnmount.

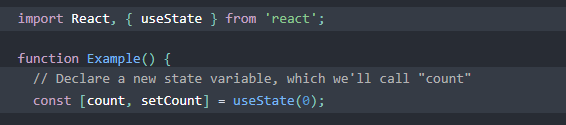
This often introduces too much abstraction, requires you to jump between different files, and makes reusing components more difficult.

**Hooks let you split one component into smaller functions based on what pieces are related (such as setting up a subscription or fetching data)**, rather than forcing a split based on lifecycle methods.

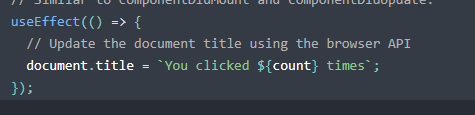
**When would I use a Hook?** We use Hook for life cycle method.

**Types**

**-Usestate**



**-UseEffect**



**-Custom Hooks**

**Building Your Own Hooks**

Custom Hook is a JavaScript function whose name starts with”use” and that may call other Hooks



**Forms in React**

Just like in HTML, React uses forms to allow users to interact with the web page.

Snippet:

class MyForm extends React.Component {

render() {

return (

<form>

<h1>Hello</h1>

<p>Enter your name:</p>

<input

type="text"

/>

</form>

);

}

}

ReactDOM.render(<MyForm />, document.getElementById('root'));

Source w3school

In HTML, form data is usually handled by the DOM.

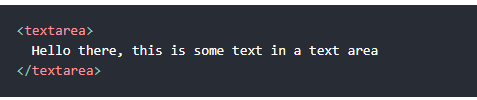
In React, form data is usually handled by the components.

When the data is handled by the components, all the data is stored in the component state.

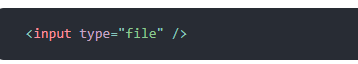
You can control changes by adding event handlers in the on Change attribute.

Form elements such as <input>, <textarea>, and <select> typically maintain their own state and update it based on user input. In React, mutable state is typically kept in the state property of components, and only updated with [setState()](https://reactjs.org/docs/react-component.html" \l "setstate).

**Text Area Tag**



**Input Tag**



Conditional Rendering

Conditional rendering in React works the same way conditions work in JavaScript. Use JavaScript operators like [if](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/if...else) or the [conditional operator](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Conditional_Operator) to create elements representing the current state, and let React update the UI to match them

If you do not want to display the h1 element until the user has done any input, you can add if statement.

{ val ?? null ?

     <> <h4 id ="emoji">Smiley Detected :  </h4> <h1>{smiley}</h1> </>: <h4> Please Enter a Word </h4>}

Val?? Null is check for null value ES2020 introduced the nullish coalescing operator denoted by the double question marks (??)

On condition true it will return a statement and on being False it return other statement.

**Inline If with Logical && Operator**

You may [embed expressions in JSX](https://reactjs.org/docs/introducing-jsx.html#embedding-expressions-in-jsx) by wrapping them in curly brace

**Inline If-Else with Conditional Operator**

[condition? true: false](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Operators/Conditional_Operator).

The conditional (ternary) operator is the only JavaScript operator that takes three operands: a condition followed by a question mark (?), then an expression to execute if the condition is [truthy](https://developer.mozilla.org/en-US/docs/Glossary/truthy) followed by a colon (:), and finally the expression to execute if the condition is [falsy](https://developer.mozilla.org/en-US/docs/Glossary/falsy). This operator is frequently used as a shortcut for the [if](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/if...else) statement.

Entry point to Application is APP.js