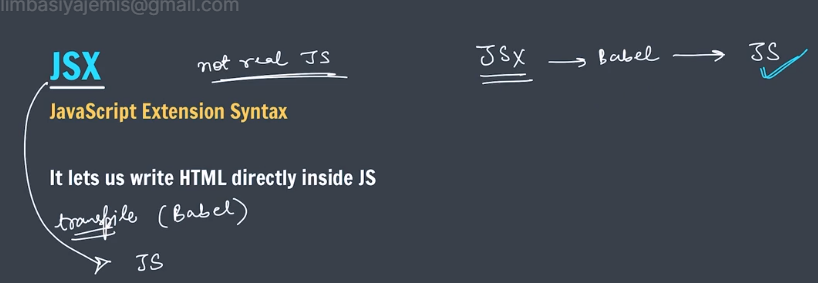
* React.JS Part 1

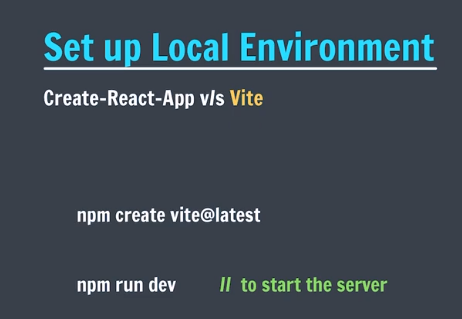
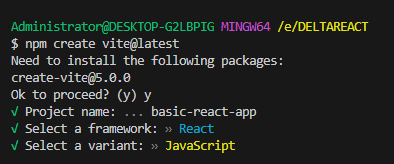
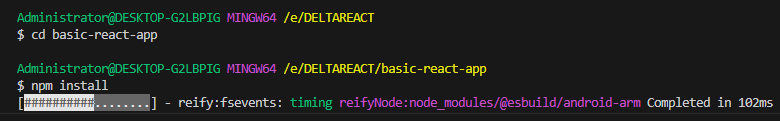
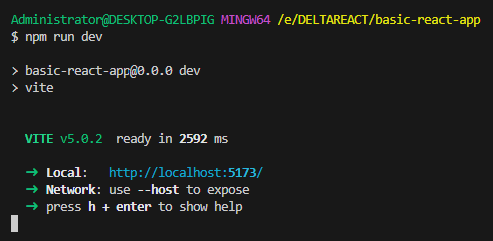
# What is React?

* JS Library for creating UI.
* It’s written in special syntax called JSX.

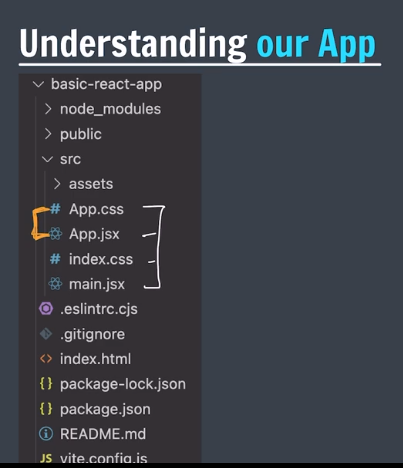
# JSX

* 
* First we write in JSX because JSX is easier than JavaScript.
* And then Babel convert JSX to JavaScript.

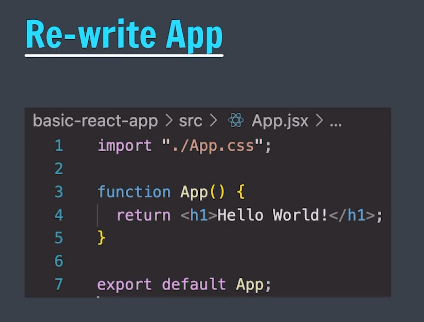
# Set up Local Environment.

* 
* We will use Vite because it is recommended by facebook.
* Create-react-app is a older version.
* First step to setup
* 
* Step 2 to install node-modules
* 
* Step 3 to start the server
* 

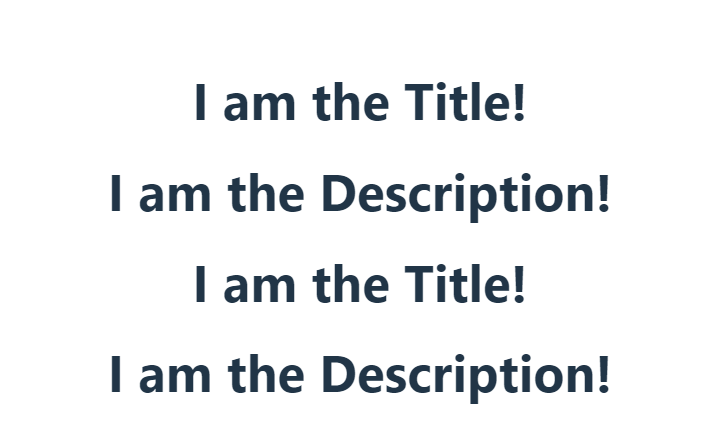
# Understanding our app

* 
* 
* Index.html – root element create and connect with main.jsx.
* Main.jsx – render app component in main

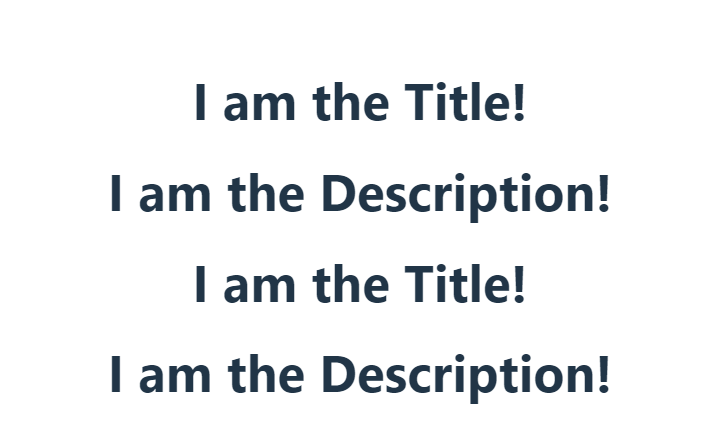
# Re-Write App

* 
* The app is return it is the app component.

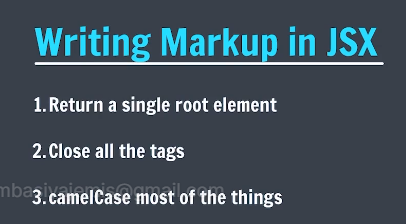
# Our 1st Component

* 
* Ex:1
* import './App.css'
* function Title() {
* return <h1>I am the Title!</h1>
* }
* function App() {
* return (
* <div>
* <h1>This is my app component</h1>
* <p>inside app component we have :</p>
* <Title />
* </div>
* );
* }
* export default App;
* Ex:2
* import './App.css'
* function Title() {
* return <h1>I am the Title!</h1>
* }
* function Description() {
* return <h1>I am the Description!</h1>
* }
* function App() {
* return (
* <div>
* <Title />
* <Description />
* <Title />
* <Description />
* </div>
* );
* }
* export default App;
* 

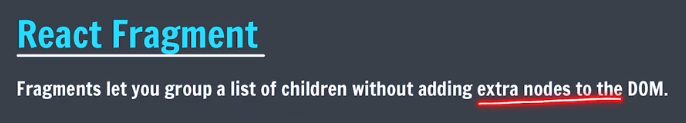
# Import-Export

* 
* App.jsx
* import './App.css'
* // import Title from "./Title.jsx" -- FOR SINGLE IMPORT
* import { Title } from './Title.jsx'; // for multiple imports.
* function Description() {
* return <h1>I am the Description!</h1>
* }
* function App() {
* return (
* <div>
* <Title />
* <Description />
* <Title />
* <Description />
* </div>
* );
* }
* export default App;
* Title.jsx
* function Title() {
* return <h1>I am the Title!</h1>
* }
* // export default Title; -- method 1 for export
* export {Title}; // FOR MULTIPLE EXPORT
* O/P:-
* 

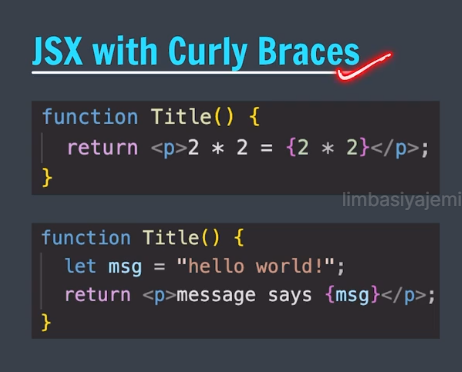
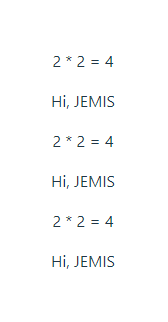
# Writing Markup in JSX

* 
* 1. Return should be single. If we want to send multiple things then we can use div to return whole div.
* 2. All tag should be close state.
* 3. camelCase should be used . If we want to give Class then we used className.

# React Fragment

* 
* It is used to stop create extra node like div so instead of div we used <> content </>.
* Ex:
* function App() {
* return (
* <>
* <Title />
* <Title />
* <Title />
* </>
* );
* }
* Result is the same.

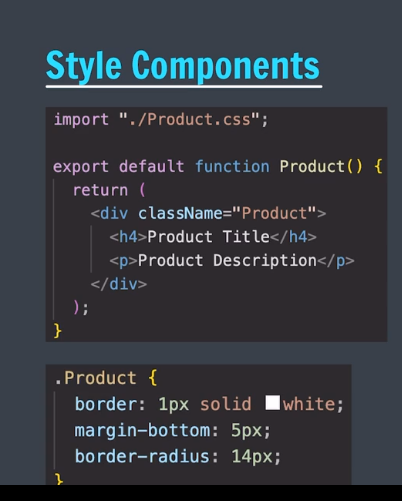
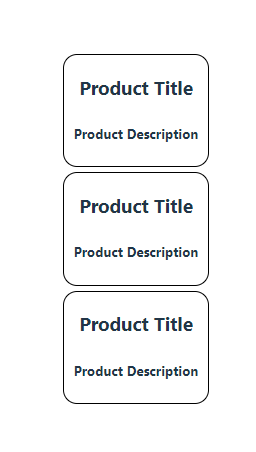
# JSX with curly braces.

* 
* In JSX if we want to write pure javascript then we can write it in curly braces.
* Ex. Title.jsx
* function Title() {
* let name = "Jemis";
* return (
* <div>
* <p>2 \* 2 = {2 \* 2}</p>
* <p>Hi, {name.toUpperCase()}</p>
* </div>
* )
* }
* // export default Title; -- method 1 for export
* export {Title}; // FOR MULTIPLE EXPORT
* O/P:-
* 

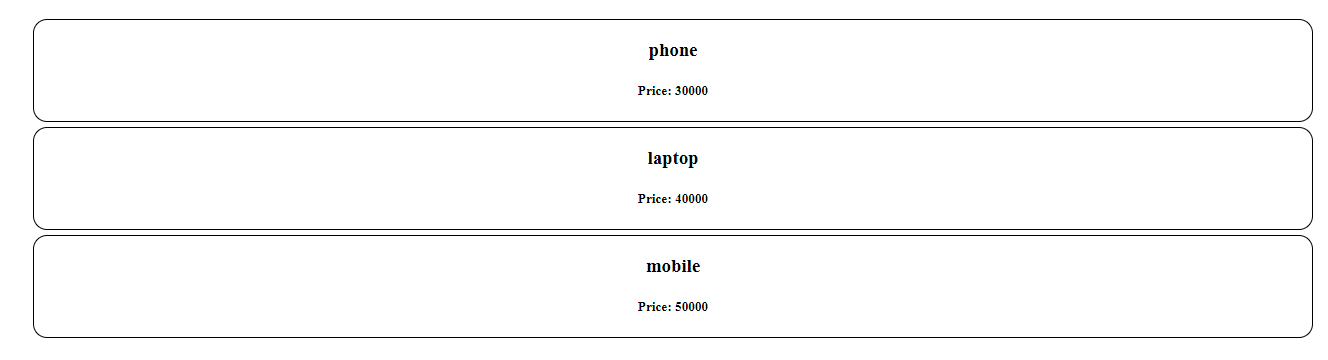
# Structuring Components

* Product.jsx
* function Product() {
* return(
* <div>
* <h3>Product Title</h3>
* <h5>Product Description</h5>
* </div>
* );
* }
* export default Product;
* ProductTab.jsx
* import Product from "./Product.jsx";
* function ProductTab(){
* return(
* <>
* <Product />
* <Product />
* <Product />
* </>
* )
* }
* export default ProductTab;
* App.jsx
* import './App.css'
* // import Title from "./Title.jsx" -- FOR SINGLE IMPORT
* import { Title } from './Title.jsx'; // for multiple imports.
* import ProductTab from './Producttab.jsx';
* function App() {
* return <ProductTab />
* }
* export default App;

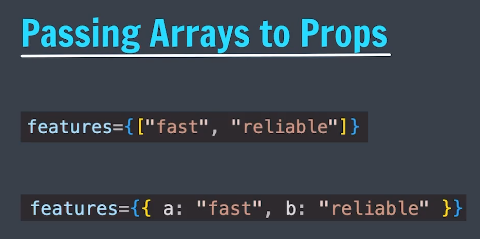
# Style Components

* 
* How we apply css to all components
* 
* Product.jsx
* import "./Product.css";
* function Product() {
* return(
* <div className="Product">
* <h3>Product Title</h3>
* <h5>Product Description</h5>
* </div>
* );
* }
* export default Product;
* product.css
* .Product{
* border: 1px solid black;
* margin-bottom: 5px;
* border-radius: 14px;
* padding: 0 10px 0 10px;
* }
* O/P:
* 
* React.JS Part 2

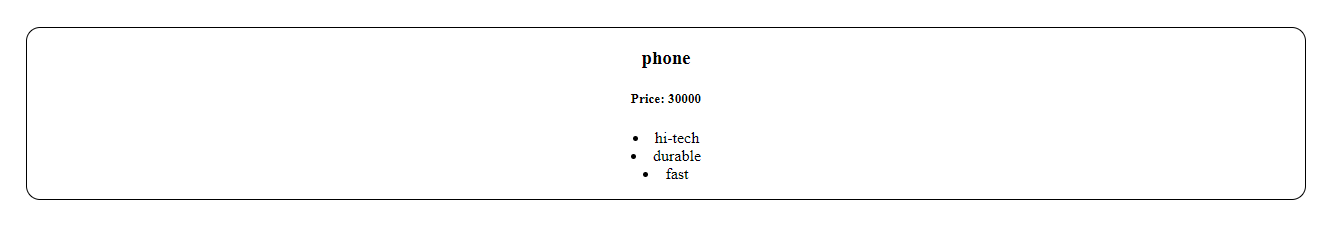
# React Props

* 
* Props means information in components.
* If we want to pass value of price bydefault then we can define while declaring.
* Product.jsx
* import "./Product.css";
* function Product({title, price}) {
* return(
* <div className="Product">
* <h3>{title}</h3>
* <h5>Price: {price}</h5>
* </div>
* );
* }
* export default Product;
* producttab.jsx
* import Product from "./Product.jsx";
* function ProductTab(){
* return(
* <>
* <Product title="phone" price={30000}/>
* <Product title="laptop" price={40000}/>
* <Product title="mobile" price={50000}/>
* </>
* )
* }
* export default ProductTab;
* O/P:-
* 

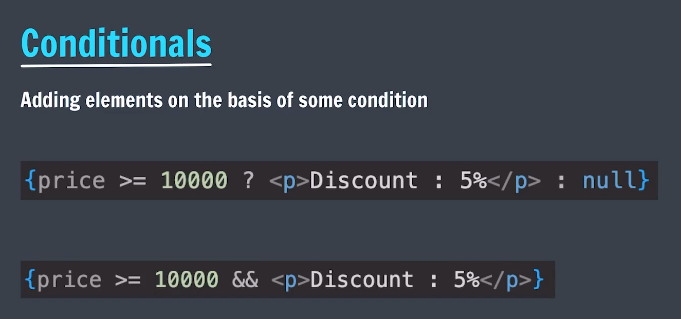
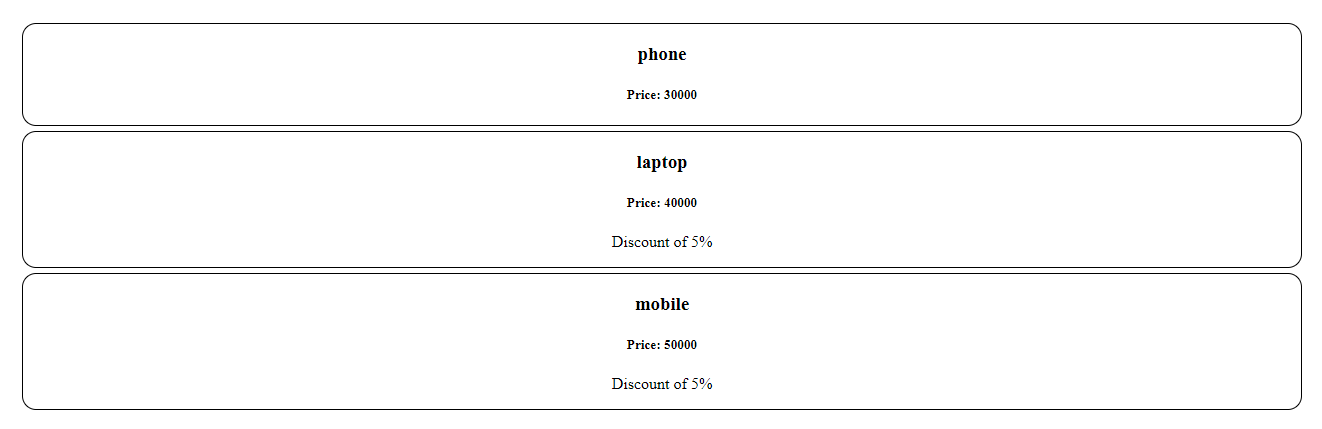
# Passing Arrays to Props

* 
* Product.jsx
* import "./Product.css";
* function Product({title, price, features, features2}) {
* return(
* <div className="Product">
* <h3>{title}</h3>
* <h5>Price: {price}</h5>
* <p>{features}</p>
* <p>{features2.b}</p>
* </div>
* );
* }
* export default Product;
* producttab.jsx
* import Product from "./Product.jsx";
* function ProductTab(){
* let options = ["hi-tech", "durable", "fast"];
* let options2 = {a: "hi-tech", b: "durable", c: "fast"};
* return(
* <>
* <Product title="phone" price={30000} features = {options} features2 = {options2}/>
* {/\* <Product title="laptop" price={40000}/>
* <Product title="mobile" price={50000}/> \*/}
* </>
* )
* }
* export default ProductTab;
* O/P:-
* 

# Rendering Arrays

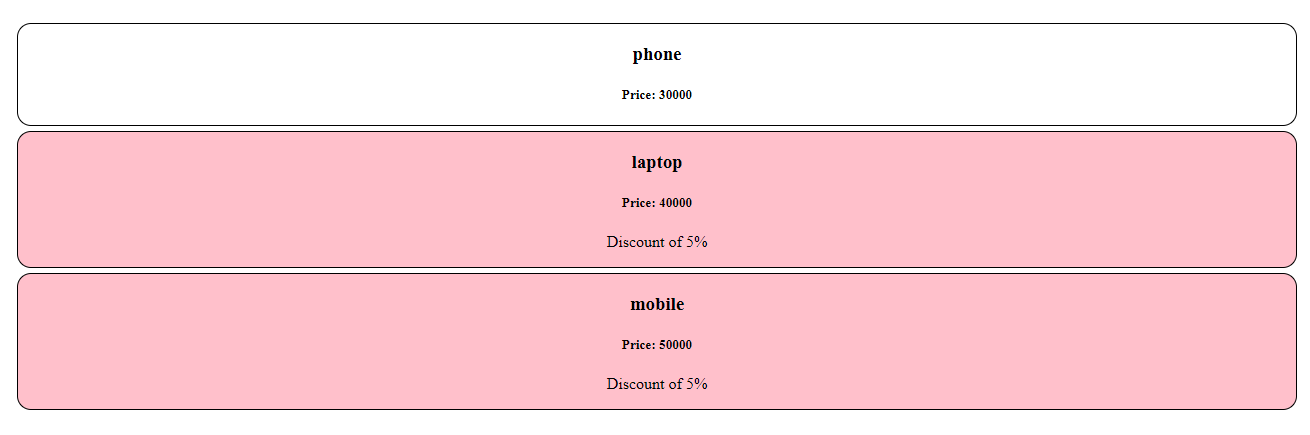
* Product.jsx
* import "./Product.css";
* function Product({title, price, features}) {
* return(
* <div className="Product">
* <h3>{title}</h3>
* <h5>Price: {price}</h5>
* <p>{features.map((feature) => <li>{feature}</li>)}</p>
* </div>
* );
* }
* export default Product;
* producttab.jsx
* import Product from "./Product.jsx";
* function ProductTab(){
* let options = ["hi-tech", "durable", "fast"];
* return(
* <>
* <Product title="phone" price={30000} features = {options}/>
* {/\* <Product title="laptop" price={40000}/>
* <Product title="mobile" price={50000}/> \*/}
* </>
* )
* }
* export default ProductTab;
* for rendering we always use map method.
* O/P:-
* 

# Conditionals

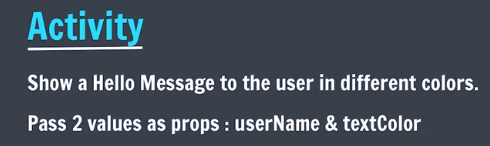
* 
* First method for conditions
* import "./Product.css";
* function Product({title, price, features}) {
* if(price>30000){
* return(
* <div className="Product">
* <h3>{title}</h3>
* <h5>Price: {price}</h5>
* <p>Discount of 5%</p>
* </div>
* );
* } else {
* return(
* <div className="Product">
* <h3>{title}</h3>
* <h5>Price: {price}</h5>
* </div>
* );
* }
* }
* export default Product;
* second method for condition
* import "./Product.css";
* function Product({title, price, features}) {
* return(
* <div className="Product">
* <h3>{title}</h3>
* <h5>Price: {price}</h5>
* <p>{price > 30000 ? "Discount of 5%" : ""}</p>
* </div>
* );
* }
* export default Product;
* O/P:- in this code p is generated every time so we use below method.
* 
* Third method for condition which we used everytime.
* import "./Product.css";
* function Product({title, price, features}) {
* return(
* <div className="Product">
* <h3>{title}</h3>
* <h5>Price: {price}</h5>
* {Price > 30000? <p>"Discount of 5%"</p> : null}
* </div>
* );
* }
* export default Product;
* Output will be the same.
* Fourth method
* import "./Product.css";
* function Product({title, price, features}) {
* return(
* <div className="Product">
* <h3>{title}</h3>
* <h5>Price: {price}</h5>
* {price > 30000 && <p>Discount of 5%</p>}
* </div>
* );
* }
* export default Product;
* if first con satisfied then p tag will automatically print the text otherwise it will not print.

# Dynamic component styling

1. import "./Product.css";
2. function Product({title, price, features}) {
3. let isDiscount = price > 30000;
4. let styles = { backgroundColor: isDiscount ? "pink" : ""};
5. return(
6. <div className="Product" style={styles}>
7. <h3>{title}</h3>
8. <h5>Price: {price}</h5>
9. {isDiscount && <p>Discount of 5%</p>}
10. </div>
11. );
12. }
13. export default Product;

* o/p:-
* 

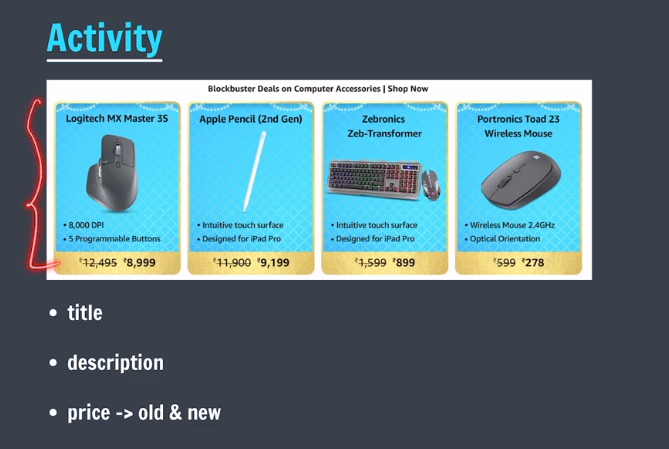
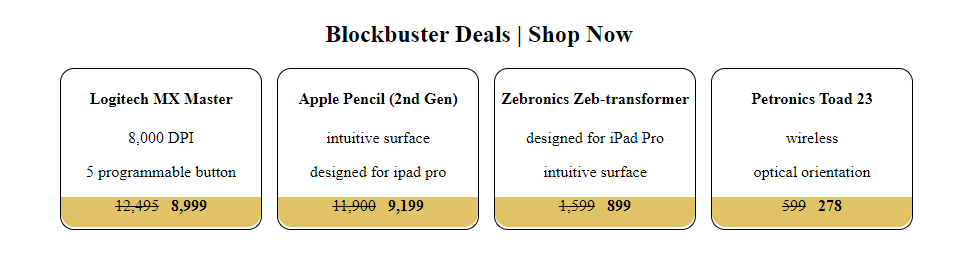
# Activity

* 
* MsgBox.jsx
* export default function MsgBox({userName, textColor}){
* return <h1 style={{color: textColor}}>Hello, {userName}</h1>
* }
* App.jsx
* import './App.css'
* // import Title from "./Title.jsx" -- FOR SINGLE IMPORT
* import { Title } from './Title.jsx'; // for multiple imports.
* import ProductTab from './Producttab.jsx';
* import MsgBox from './MsgBox.jsx';
* function App() {
* return (
* <>
* <MsgBox userName="Jemis" textColor="yellow"/>
* <MsgBox userName="apnacollege" textColor="green"/>
* <MsgBox userName="krishna" textColor="blue"/>
* <ProductTab />
* </>
* );
* }
* export default App;
* o/p:
* 

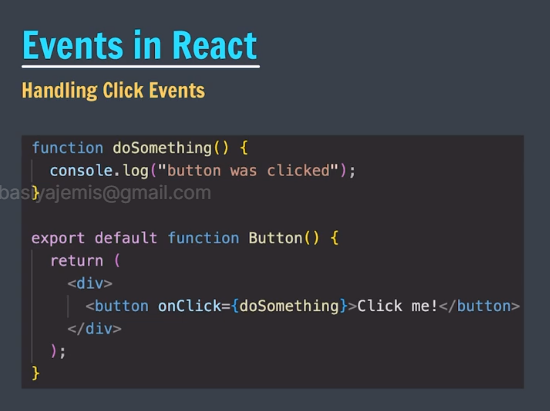
# Install React Developer Tools

* <https://react.dev/learn/react-developer-tools>
* Extention add to chorme for react .

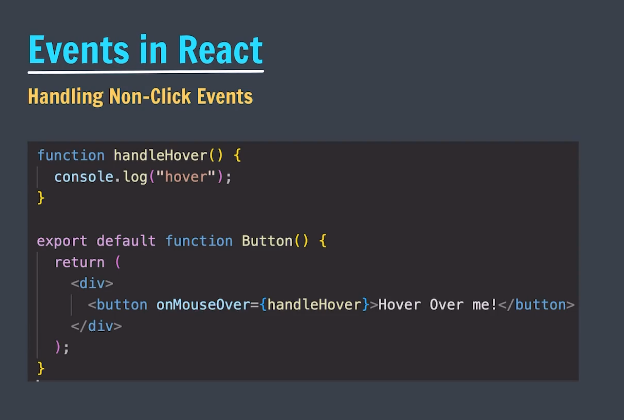
# Activity : Amazon Cards

* 
* Price.jsx
* export default function Price({oldPrice, newPrice}) {
* let oldStyle = {
* textDecorationLine: "line-through",
* }
* let newStyle = {
* fontWeight: "bold",
* }
* let styles = {
* backgroundColor: "#e0c367",
* height: "30px",
* width: "200px",
* borderBottomLeftRadius: "14px",
* borderBottomRightRadius: "14px"
* }
* return (
* <div style={styles}>
* <span style={oldStyle}>{oldPrice}</span>
* &nbsp;&nbsp;&nbsp;
* <span style={newStyle}>{newPrice}</span>
* </div>
* );
* }
* Product.jsx
* import "./Product.css";
* import Price from "./Price.jsx";
* function Product({title, idx}) {
* let oldPrices = ["12,495", "11,900", "1,599", "599"];
* let newPrices = ["8,999", "9,199", "899", "278"];
* let description = [["8,000 DPI", "5 programmable button"], ["intuitive surface", "designed for ipad pro"], ["designed for iPad Pro", "intuitive surface"], ["wireless", "optical orientation"]];
* return(
* <div className="Product">
* <h4>{title}</h4>
* <p>{description[idx][0]}</p>
* <p>{description[idx][1]}</p>
* <Price oldPrice={oldPrices[idx]} newPrice={newPrices[idx]}/>
* </div>
* );
* }
* export default Product;
* app.jsx
* import './App.css'
* import ProductTab from './Producttab.jsx';
* function App() {
* return (
* <>
* <h2>Blockbuster Deals | Shop Now</h2>
* <ProductTab />
* </>
* );
* }
* export default App;
* Product.css
* .Product{
* border: 1px solid black;
* margin-bottom: 5px;
* border-radius: 14px;
* /\* padding: 0 10px 0 10px; \*/
* margin-left: 15px;
* width: 200px;
* height: 160px;
* }
* 
* React.JS Part 3

# Handling Click Events

* 
* Button.jsx
* function printHello() {
* console.log("Hello!");
* }
* export default function Button() {
* return(
* <div>
* <button onClick={printHello}>Click me!</button>
* </div>
* );
* }
* App.jsx
* import './App.css'
* import Button from './Button';
* import ProductTab from './Producttab.jsx';
* function App() {
* return (
* <>
* <Button />
* </>
* );
* }
* export default App;
* o/p:
* when you click hello will be print in console.log

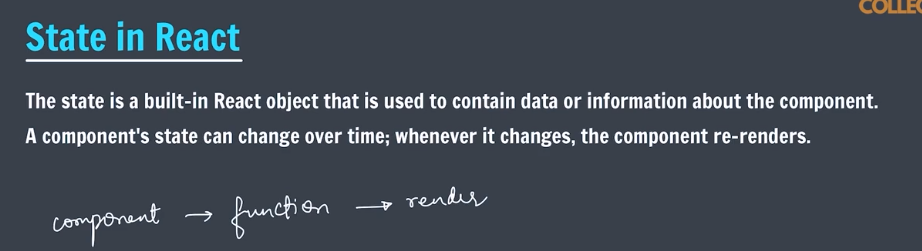
# Events in React

* 
* Button.jsx
* function handleClick() {
* console.log("Hello!");
* }
* function handleMouseOver() {
* console.log("Bye!")
* }
* function handleDblClick() {
* console.log("you doubled click")
* }
* export default function Button() {
* return(
* <div>
* <button onClick={handleClick}>Click me!</button>
* <p onMouseOver={handleMouseOver}>Lorem ipsum dolor, sit amet consectetur adipisicing elit. Quibusdam excepturi sed, facilis laboriosam sapiente quas placeat harum totam quis ipsam sequi consectetur tempore aliquid dolorum beatae? Ipsam illum laborum dolore!</p>
* <button onDoubleClick={handleDblClick}>double click me</button>
* </div>
* );
* }
* o/p
* when you took aerrow over paragraph bye will be print same for double click.

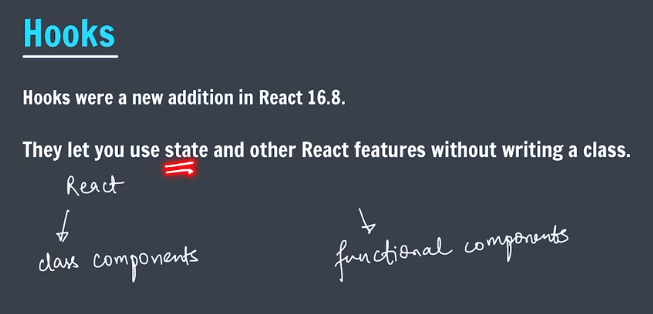
# Event Object

* 
* Form.jsx
* function handleFormSubmit(event) {
* event.preventDefault();
* console.log("form was submitted");
* }
* export default function Form() {
* return(
* <form onSubmit={handleFormSubmit}>
* <input placeholder="Write something" />
* <button>Submit</button>
* </form>
* );
* }
* App.jsx
* import './App.css'
* import Button from './Button';
* import ProductTab from './Producttab.jsx';
* import Form from './Form.jsx';
* function App() {
* return (
* <>
* <Form />
* </>
* );
* }
* export default App;
* Event bachale rakhega output ko.

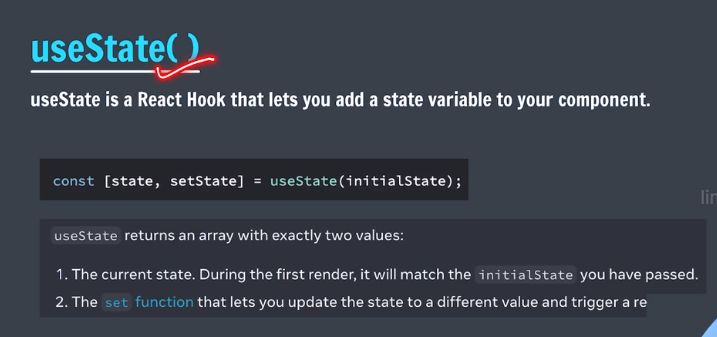
# State in React

* State means once the change is done in component then state render component again so counter example works.
* 

# Hooks

* 

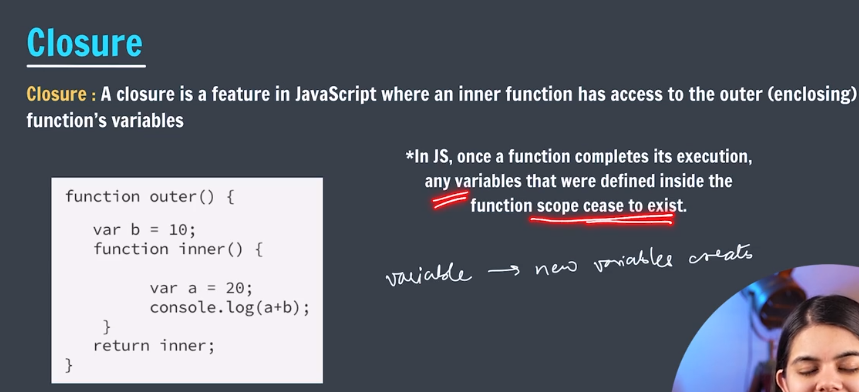
# useState()

* 
* In this ss state is count and it’s value is 0 and setstate is rerender the component so counter will work . and initial state is 0 .
* First useState we have to import from react.
* useState() is a hook

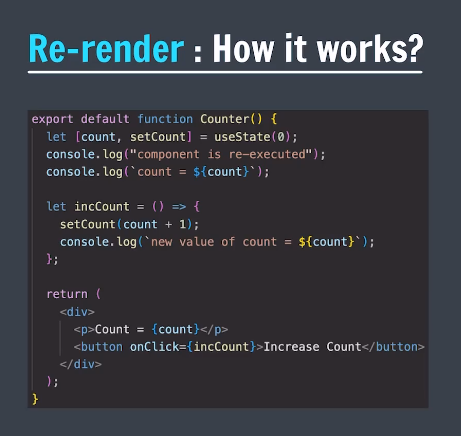
# Activity : Create Like Button

* LikeButton.jsx
* import { useState } from "react";
* export default function LikeButton () {
* let [isLiked, setIsLiked] = useState(false);
* let toggleLike = ()=>{
* setIsLiked(!isLiked);
* };
* let likeStyle = {color: "red"}
* return(
* <div>
* <p onClick={toggleLike}>
* {isLiked ? (<i className="fa-solid fa-heart" style={likeStyle}></i>) : (<i className="fa-regular fa-heart"></i>) }
* </p>
* </div>
* );
* }
* Like button in output .
* We have used cdn links in index.html for heat symbol .

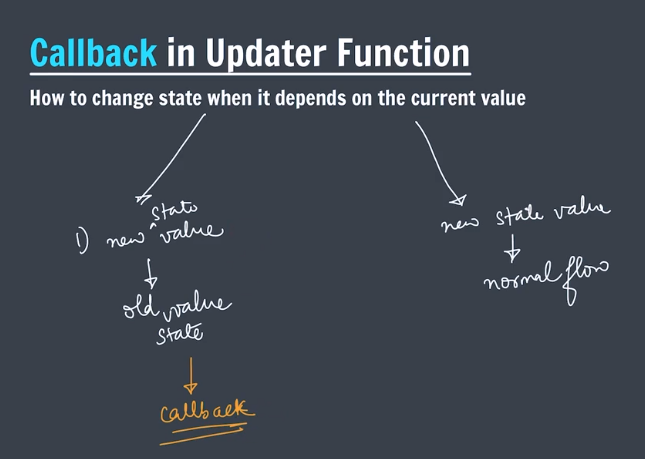
# Closure in JS

* 

# Re-render : How does it work ?

* 

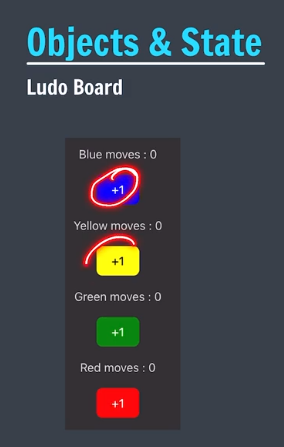
# Callback in Set State Function

* 
* When updation in values is needed then callback is required. If we want to use fixed value then we write like setCount(25).
* Counter.jsx
* import { useState } from "react";
* export default function Counter() {
* let [count, setCount] = useState(0); //initialization
* let incCount = () => {
* setCount((currCount) => {   //this is callback
* return currCount + 1;
* });
* setCount((currCount) => {   //this is callback
* return currCount + 1;
* });
* };
* return (
* <div>
* <h3>count = {count}</h3>
* <button onClick={incCount}>Increase Count</button>
* </div>
* );
* }

# More about State

* If we use callback then and then only state will re-render the components. If we use fixed value in setCount then state will not re-render.
* Re-render only happen when updation happens.
* Re-render happens == state changed happens.
* When we use some function in set then never write like below
* useState(init());
* always write like below
* useState(init);
* Counter.jsx
* import { useState } from "react";
* function init() {
* console.log("init initialized") //this is will execute only once
* return Math.random();
* }
* export default function Counter() {
* let [count, setCount] = useState(init); //initialization
* let incCount = () => {
* console.log("setCount rendered")  //this will execute every time.
* setCount((currCount) => {   //this is callback
* return currCount + 1;
* });
* };
* return (
* <div>
* <h3>count = {count}</h3>
* <button onClick={incCount}>Increase Count</button>
* </div>
* );
* }
* React.JS Part 4

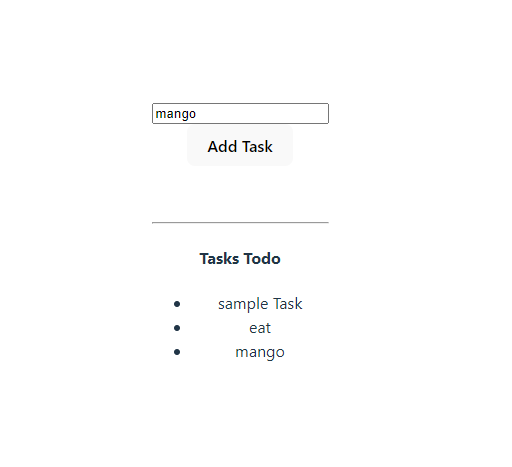
# Objects & State

* 
* In case of object the counter will not work because in object value will not change we have to copy the object and create new object to use it .
* Because until and unless state doesn’t detect change component will not re-render.
* In this we have used object as state variable.
* import { useState } from "react";
* export default function LudoBoard() {
* let [moves, setmoves] = useState({blue: 0, red: 0, yellow: 0, green: 0});
* let updateBlue = () => {
* setmoves((prevMoves) => {
* return {...prevMoves, blue: prevMoves.blue + 1}
* }); //we use this spread for creating new object of state. ...moves means copy full object and then update if wish.
* }
* let updateRed = () => {
* setmoves((prevMoves) => {
* return {...prevMoves, red: prevMoves.red + 1}
* }); //we use this spread for creating new object of state. ...moves means copy full object and then update if wish.
* }
* let updateYellow = () => {
* setmoves((prevMoves) => {
* return {...prevMoves, yellow: prevMoves.yellow + 1}
* }); //we use this spread for creating new object of state. ...moves means copy full object and then update if wish.
* }
* let updateGreen = () => {
* setmoves((prevMoves) => {
* return {...prevMoves, green: prevMoves.green + 1}
* }); //we use this spread for creating new object of state. ...moves means copy full object and then update if wish.
* }
* return (
* <div>
* <p>Game Begins!</p>
* <div className="board">
* <p>Blue moves = {moves.blue}</p>
* <button style={{backgroundColor: "blue"}} onClick={updateBlue}>+1</button>
* <p>Yellow moves = {moves.yellow}</p>
* <button style={{backgroundColor: "yellow"}} onClick={updateYellow}>+1</button>
* <p>Green moves = {moves.green}</p>
* <button style={{backgroundColor: "green"}} onClick={updateGreen}>+1</button>
* <p>Red moves = {moves.red}</p>
* <button style={{backgroundColor: "red"}} onClick={updateRed}>+1</button>
* </div>
* </div>
* );
* }

# Arrays & State

* In this we have used Arrays as state variable.
* Same concept as above concept.
* export default function LudoBoard() {
* let [moves, setmoves] = useState({blue: 0, red: 0, yellow: 0, green: 0});
* let [arr, SetArr] = useState(["no moves"]);
* let updateBlue = () => {
* setmoves((prevMoves) => {
* return {...prevMoves, blue: prevMoves.blue + 1};
* }); //we use this spread for creating new object of state. ...moves means copy full object and then update if wish.
* SetArr((prevArr) => {
* return [...prevArr, "blue moves"];
* })
* };

# Activity: create a todo

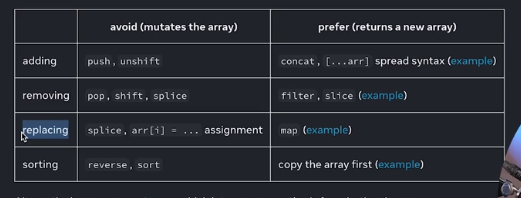
* TodoList.jsx
* import { useState } from "react";
* export default function TodoList() {
* let [todos, setTodos] = useState(["sample Task"]);
* let [newTodo, setNewTodo] = useState("");
* let addNewTask = () => {
* setTodos([...todos, newTodo]);
* }
* let updateTodoValue = (event) => {
* setNewTodo(event.target.value);
* }
* return (
* <div>
* <input placeholder="add a task" value={newTodo} onChange={updateTodoValue}></input>
* <br></br>
* <button onClick={addNewTask}>Add Task</button>
* <br></br>
* <br></br>
* <br></br>
* <hr></hr>
* <h4>Tasks Todo</h4>
* <ul>
* {todos.map((todo) => (
* <li>{todo}</li>
* ))}
* </ul>
* </div>
* );
* }
* O/P:-
* 

# Unique key for List items

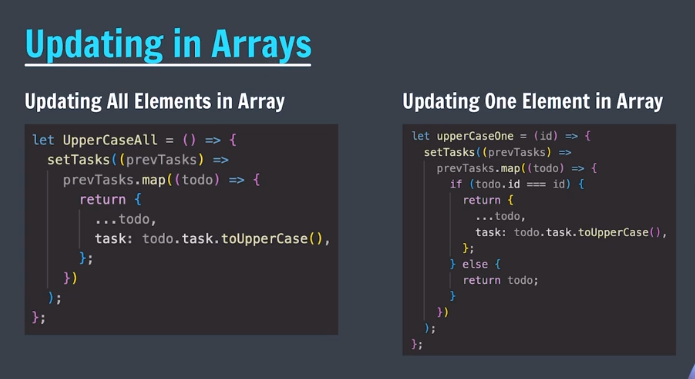
* We will use UUID npm package.
* Npm install uuid
* import { useState } from "react";
* import { v4 as uuidv4 } from 'uuid';
* export default function TodoList() {
* let [todos, setTodos] = useState([{task: "sample-task", id: uuidv4()}]); //array of object
* let [newTodo, setNewTodo] = useState("");
* let addNewTask = () => {
* setTodos([...todos, {task: newTodo, id: uuidv4()}]);
* }
* let updateTodoValue = (event) => {
* setNewTodo(event.target.value);
* }
* return (
* <div>
* <input placeholder="add a task" value={newTodo} onChange={updateTodoValue}></input>
* <br></br>
* <button onClick={addNewTask}>Add Task</button>
* <br></br>
* <br></br>
* <br></br>
* <hr></hr>
* <h4>Tasks Todo</h4>
* <ul>
* {todos.map((todo) => (
* <li key={todo.id}>{todo.task}</li>
* ))}
* </ul>
* </div>
* );
* }
* Unique id is added in this lecture.

# Deleting from array

1. import { useState } from "react";
2. import { v4 as uuidv4 } from 'uuid';
3. export default function TodoList() {
4. let [todos, setTodos] = useState([{task: "sample-task", id: uuidv4()}]); //array of object
5. let [newTodo, setNewTodo] = useState("");
6. let addNewTask = () => {
7. setTodos((prevTodos) => {
8. return [...prevTodos, {task: newTodo, id: uuidv4()}]
9. });
10. setNewTodo("");
11. }
12. let updateTodoValue = (event) => {
13. setNewTodo(event.target.value);
14. }
15. let deleteTodo = (id) => {
16. setTodos((prevTodos) => todos.filter((prevTodos) => prevTodos.id != id));
17. }
18. return (
19. <div>
20. <input placeholder="add a task" value={newTodo} onChange={updateTodoValue}></input>
21. <br></br>
22. <button onClick={addNewTask}>Add Task</button>
23. <br></br>
24. <br></br>
25. <br></br>
26. <hr></hr>
27. <h4>Tasks Todo</h4>
28. <ul>
29. {todos.map((todo) => (
30. <li key={todo.id}>
31. <span>{todo.task}</span>
32. &nbsp;&nbsp;&nbsp;&nbsp;
33. <button onClick={() => deleteTodo(todo.id)}>delete</button>
34. </li>
35. ))}
36. </ul>
37. </div>
38. );
39. }

* Important notes from react document
* 

# Update All in Array

* 
* import { useState } from "react";
* import { v4 as uuidv4 } from 'uuid';
* export default function TodoList() {
* let [todos, setTodos] = useState([{task: "sample-task", id: uuidv4()}]); //array of object
* let [newTodo, setNewTodo] = useState("");
* let addNewTask = () => {
* setTodos((prevTodos) => {
* return [...prevTodos, {task: newTodo, id: uuidv4()}]
* });
* setNewTodo("");
* }
* let updateTodoValue = (event) => {
* setNewTodo(event.target.value);
* }
* let deleteTodo = (id) => {
* setTodos((prevTodos) => todos.filter((prevTodos) => prevTodos.id != id));
* }
* let upperCaseAll = () => {
* setTodos( (prevTodos) => (
* prevTodos.map((todo) => {
* return {
* ...todo,
* task: todo.task.toUpperCase(),
* };
* })));
* };
* return (
* <div>
* <input placeholder="add a task" value={newTodo} onChange={updateTodoValue}></input>
* <br></br>
* <button onClick={addNewTask}>Add Task</button>
* <br></br>
* <br></br>
* <br></br>
* <hr></hr>
* <h4>Tasks Todo</h4>
* <ul>
* {todos.map((todo) => (
* <li key={todo.id}>
* <span>{todo.task}</span>
* &nbsp;&nbsp;&nbsp;&nbsp;
* <button onClick={() => deleteTodo(todo.id)}>delete</button>
* </li>
* ))}
* </ul>
* <br></br>
* <button onClick={upperCaseAll}>UpperCase All</button>
* </div>
* );
* }

# Update one in Array

1. import { useState } from "react";
2. import { v4 as uuidv4 } from 'uuid';
3. export default function TodoList() {
4. let [todos, setTodos] = useState([{task: "sample-task", id: uuidv4()}]); //array of object
5. let [newTodo, setNewTodo] = useState("");
6. let addNewTask = () => {
7. setTodos((prevTodos) => {
8. return [...prevTodos, {task: newTodo, id: uuidv4()}]
9. });
10. setNewTodo("");
11. }
12. let updateTodoValue = (event) => {
13. setNewTodo(event.target.value);
14. }
15. let deleteTodo = (id) => {
16. setTodos((prevTodos) => todos.filter((prevTodos) => prevTodos.id != id));
17. }
18. let upperCaseAll = () => {
19. setTodos( (prevTodos) => (
20. prevTodos.map((todo) => {
21. return {
22. ...todo,
23. task: todo.task.toUpperCase(),
24. };
25. })));
26. };
27. let UpperCaseOne = (id) => {
28. setTodos( (prevTodos) => (
29. prevTodos.map((todo) => {
30. if(todo.id == id) {
31. return {
32. ...todo,
33. task: todo.task.toUpperCase(),
34. };
35. } else {
36. return todo;
37. }
39. })));
40. }
41. return (
42. <div>
43. <input placeholder="add a task" value={newTodo} onChange={updateTodoValue}></input>
44. <br></br>
45. <button onClick={addNewTask}>Add Task</button>
46. <br></br>
47. <br></br>
48. <br></br>
49. <hr></hr>
50. <h4>Tasks Todo</h4>
51. <ul>
52. {todos.map((todo) => (
53. <li key={todo.id}>
54. <span>{todo.task}</span>
55. &nbsp;&nbsp;&nbsp;&nbsp;
56. <button onClick={() => deleteTodo(todo.id)}>delete</button>
57. <button onClick={() => UpperCaseOne(todo.id)}>UpperCase One</button>
58. </li>
59. ))}
60. </ul>
61. <br></br>
62. <button onClick={upperCaseAll}>UpperCase All</button>
63. </div>
64. );
65. }

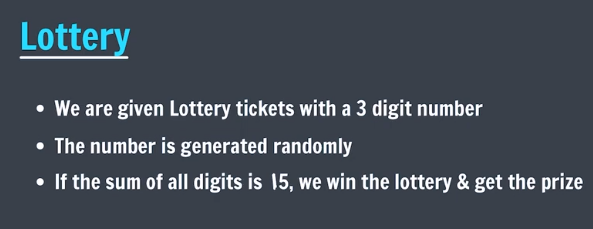
# Add a “Mark as done” Feature

* Home work
* React.JS Part 5

# Todo solution – mark as done

* React has mainly basic four concepts as below
* 
* Solution for todo list
* import { useState } from "react";
* import { v4 as uuidv4 } from 'uuid';
* export default function TodoList() {
* let [todos, setTodos] = useState([{task: "sample-task", id: uuidv4(), isDone:false}]); //array of object
* let [newTodo, setNewTodo] = useState("");
* let addNewTask = () => {
* setTodos((prevTodos) => {
* return [...prevTodos, {task: newTodo, id: uuidv4(), isDone:false}]
* });
* setNewTodo("");
* }
* let updateTodoValue = (event) => {
* setNewTodo(event.target.value);
* }
* let deleteTodo = (id) => {
* setTodos((prevTodos) => todos.filter((prevTodos) => prevTodos.id != id));
* }
* let markAllDone = () => {
* setTodos( (prevTodos) => (
* prevTodos.map((todo) => {
* return {
* ...todo,
* isDone: true,
* };
* })));
* };
* let markAsDone = (id) => {
* setTodos( (prevTodos) => (
* prevTodos.map((todo) => {
* if(todo.id == id) {
* return {
* ...todo,
* isDone: true,
* };
* } else {
* return todo;
* }
* })));
* }
* return (
* <div>
* <input placeholder="add a task" value={newTodo} onChange={updateTodoValue}></input>
* <br></br>
* <button onClick={addNewTask}>Add Task</button>
* <br></br>
* <br></br>
* <br></br>
* <hr></hr>
* <h4>Tasks Todo</h4>
* <ul>
* {todos.map((todo) => (
* <li key={todo.id}>
* <span style={todo.isDone ? {textDecorationLine: "line-through"} : {}}>{todo.task}</span>
* &nbsp;&nbsp;&nbsp;&nbsp;
* <button onClick={() => deleteTodo(todo.id)}>delete</button>
* <button onClick={() => markAsDone(todo.id)}>markAsDone</button>
* </li>
* ))}
* </ul>
* <br></br>
* <button onClick={markAllDone}>Mark All as done</button>
* </div>
* );
* }

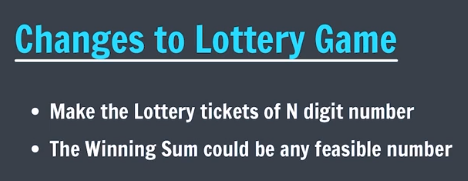
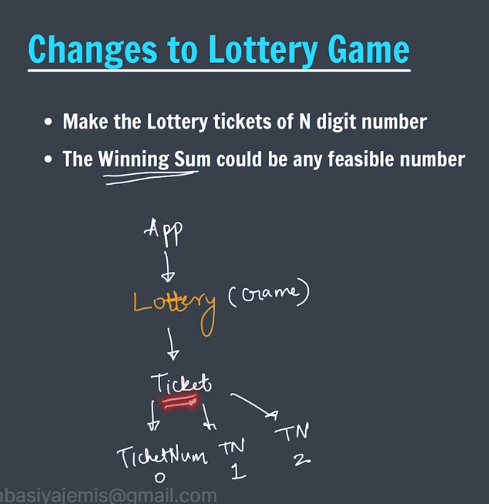
# Lottery Game (Part a)

* 

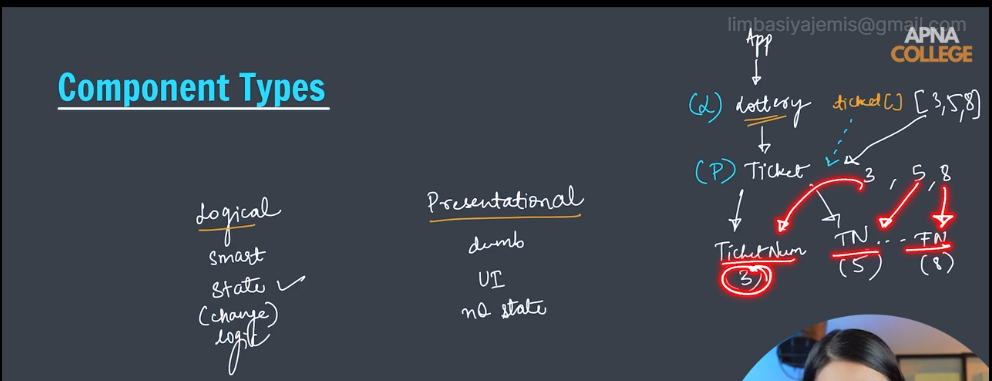
# Lottery Game (part b)

* Lottery.jsx
* import { useState } from "react";
* import "./Lottery.css";
* import { genTicket, sum } from "./helper";
* export default function Lottery() {
* let [ticket, setTicket] = useState(genTicket(3));
* let isWinning = sum(ticket) === 15;
* let buyTicket = () => {
* setTicket(genTicket(3));
* };
* return (
* <div>
* <h1>Lottery Game!</h1>
* <div className="ticket">
* <span>{ticket[0]}</span>
* <span>{ticket[1]}</span>
* <span>{ticket[2]}</span>
* </div>
* <br></br>
* <button onClick={buyTicket}>Buy New Ticket</button>
* <h3>{isWinning && "Congratulations, you won!"}</h3>
* </div>
* );
* }
* Helper.js
* function genTicket(n) {
* let arr = new Array(n);
* for(let i = 0; i < n; i++) {
* arr[i] = Math.floor(Math.random() \* 10);
* }
* return arr;
* }
* function sum(arr) {
* return arr.reduce((sum, curr) => sum + curr, 0);
* }
* export { genTicket, sum };

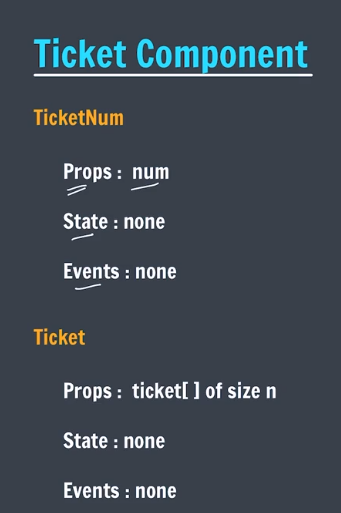
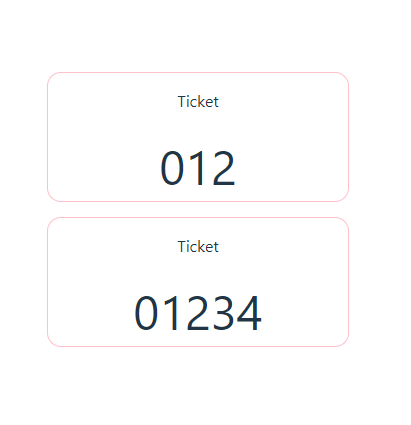
# Changes to lottery game

* 
* 
* 

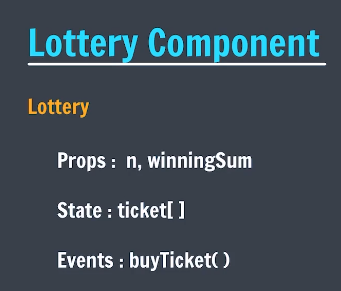
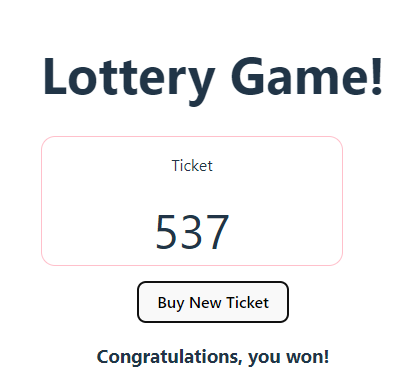
# Component types

* 

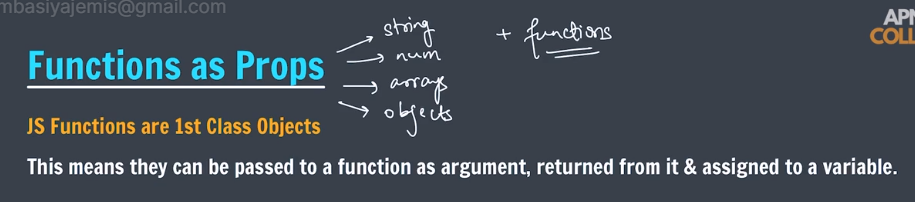
# Ticket component

* 
* TicketNum.jsx
* import "./TicketNum.css";
* export default function TicketNum({num}) {
* return <span className="TicketNum">{num}</span>
* }
* TicketNum.css
* .TicketNum {
* font-size: 3rem;
* }
* Ticket.jsx
* import TicketNum from "./TicketNum";
* import "./Ticket.css";
* export default function Ticket({ticket}) {
* return (
* <div className="Ticket">
* <p>Ticket</p>
* {ticket.map((num, idx) => (
* <TicketNum num={num} key={idx}/>
* ))}
* </div>
* );
* }
* Ticket.css
* .Ticket {
* border: 1px solid pink;
* border-radius: 14px;
* width: 300px;
* margin-bottom: 15px;
* }
* App.js
* import './App.css'
* import Lottery from './Lottery'
* import LudoBoard from './LudoBoard'
* import Ticket from './Ticket.jsx'
* import TodoList from './TodoList'
* function App() {
* return (
* <>
* <Ticket ticket={[0,1,2]}/>
* <Ticket ticket={[0,1,2,3,4]}/>
* </>
* );
* }
* export default App
* O/P:
* 

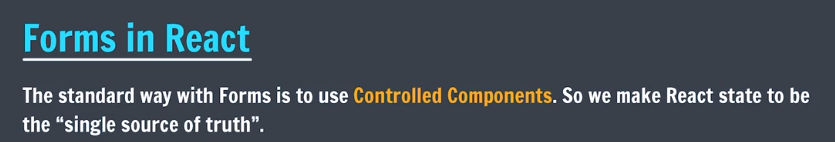
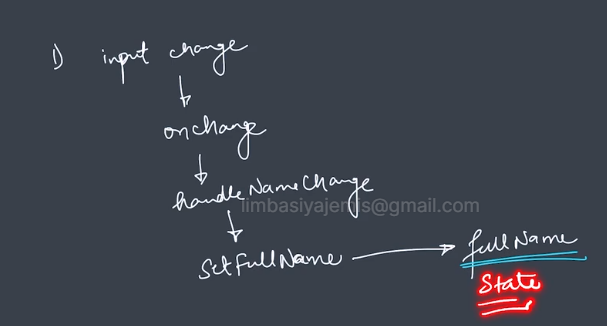
# Lottery Component

* 
* App.jsx
* import './App.css'
* import Lottery from './Lottery'
* function App() {
* return (
* <>
* <Lottery n={3} winningSum={15}/>
* </>
* );
* }
* export default App
* Lottery.jsx
* import { useState } from "react";
* import { genTicket, sum } from "./helper";
* import Ticket from "./Ticket";
* export default function Lottery({n=3, winningSum=15}) {
* let [ticket, setTicket] = useState(genTicket(n));
* let isWinning = sum(ticket) === winningSum;
* let buyTicket = () => {
* setTicket(genTicket(n));
* };
* return (
* <div>
* <h1>Lottery Game!</h1>
* <Ticket ticket={ticket}/>
* <button onClick={buyTicket}>Buy New Ticket</button>
* <h3>{isWinning && "Congratulations, you won!"}</h3>
* </div>
* );
* }
* Ticket.jsx
* import TicketNum from "./TicketNum";
* import "./Ticket.css";
* export default function Ticket({ticket}) {
* return (
* <div className="Ticket">
* <p>Ticket</p>
* {ticket.map((num, idx) => (
* <TicketNum num={num} key={idx}/>
* ))}
* </div>
* );
* }
* TicketNum.jsx
* import "./TicketNum.css";
* export default function TicketNum({num}) {
* return <span className="TicketNum">{num}</span>
* }
* 

# Functions as props

* 
* App.jsx
* import './App.css'
* import Lottery from './Lottery'
* import { sum } from './helper';
* function App() {
* let winCondition = (ticket) => {
* return sum(ticket) === 15;
* }
* return (
* <>
* <Lottery n={3} winCondition={winCondition}/>
* </>
* );
* }
* export default App
* Lottery.jsx
* import { useState } from "react";
* import { genTicket, sum } from "./helper";
* import Ticket from "./Ticket";
* export default function Lottery({n=3, winCondition}) {
* let [ticket, setTicket] = useState(genTicket(n));
* let isWinning = winCondition(ticket);
* let buyTicket = () => {
* setTicket(genTicket(n));
* };
* return (
* <div>
* <h1>Lottery Game!</h1>
* <Ticket ticket={ticket}/>
* <button onClick={buyTicket}>Buy New Ticket</button>
* <h3>{isWinning && "Congratulations, you won!"}</h3>
* </div>
* );
* }
* React.JS Part 6

# Forms in React

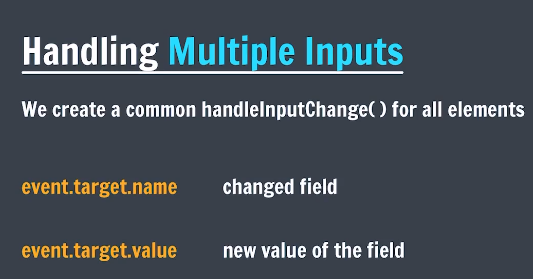
* 
* 
* Form.jsx
* import { useState } from "react";
* export default function Form() {
* let [fullName, setFullName] = useState("Jemis");
* let handleNameChange = (event) => {
* setFullName(event.target.value);
* }
* return(
* <form>
* <input placeholder="enter full name" type="text" value={fullName} onChange={handleNameChange}/>
* <button>Submit</button>
* </form>
* );
* }
* App.jsx
* import './App.css'
* import Form from './Form';
* function App() {
* return (
* <>
* <Form/>
* </>
* );
* }
* export default App

# Labels in React

1. import { useState } from "react";
2. export default function Form() {
3. let [fullName, setFullName] = useState("");
4. let handleNameChange = (event) => {
5. setFullName(event.target.value);
6. }
7. return(
8. <form>
9. <label htmlFor="username">Full Name</label>
10. <input placeholder="enter full name" type="text" value={fullName} onChange={handleNameChange} id="username"/>
11. <button>Submit</button>
12. </form>
13. );
14. }

* For lables in React we use htmlFor because in react for is forloop.

# Handling Multiple Input

* 
* Form.jsx
* import { useState } from "react";
* export default function Form() {
* let [formData, setFormData] = useState({
* fullName: "",
* username: "",
* password: "",
* });
* let handleInputChange = (event) => {
* setFormData((currData) => {
* return { ...currData, [event.target.name]: event.target.value };
* });
* }
* let handleSubmit = (event) => {
* event.preventDefault();
* setFormData({
* fullName: "",
* username: "",
* password: "",
* });
* };
* return(
* <form onSubmit={handleSubmit}>
* <label htmlFor="fullName">Full Name</label>
* <input placeholder="enter full name" type="text" value={formData.fullName} name="fullName" onChange={handleInputChange} id="fullName"/>
* <br></br>
* <br></br>
* <label htmlFor="username">UserName</label>
* <input placeholder="enter full name" type="text" value={formData.username} name="username" onChange={handleInputChange} id="username"/>
* <button>Submit</button>
* <br></br>
* <br></br>
* <label htmlFor="password">Password</label>
* <input placeholder="enter password" type="password" value={formData.password} name="password" onChange={handleInputChange} id="password"/>
* <button>Submit</button>
* </form>
* );
* }

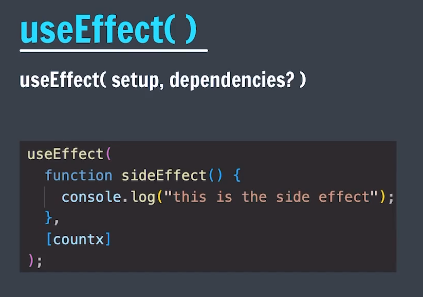
# Comment Form

1. import { useState } from "react";
2. export default function CommentsForm() {
3. let [formData, setFormData] = useState({
4. username: "",
5. remarks: "",
6. rating: 5,
7. });
8. let handleInputChange = (event) => {
9. setFormData((currData) => {
10. return { ...currData, [event.target.name]: event.target.value };
11. });
12. };
13. let handleSubmit = (event) => {
14. console.log(formData);
15. event.preventDefault();
16. setFormData({
17. username: "",
18. remarks: "",
19. rating: 5,
20. });
21. };
22. return(
23. <div>
24. <h4>Give a comment!</h4>
25. <form onSubmit={handleSubmit}>
26. <label htmlFor="username">Username</label>
27. <input placeholder="username" id="username" type="text" name="username" value={formData.username} onChange={handleInputChange}/>
28. <br></br><br></br>
29. <label htmlFor="remarks">Remarks</label>
30. <textarea  id="remarks" name="remarks" placeholder="add few remarks" value={formData.remarks} onChange={handleInputChange}></textarea>
31. <br></br><br></br>
32. <label htmlFor="rating">Rating</label>
33. <input placeholder="rating" type="number" id="rating" name="rating" min={1} max={5} value={formData.rating} onChange={handleInputChange}/>
34. <br></br><br></br>
35. <button>Add Comment</button>
36. </form>
37. </div>
38. );
39. }

# useEffect()

* 
* Counter.jsx
* import { useState, useEffect } from "react";
* export default function Counter() {
* let [count, setCount] = useState(0);
* let incCount = () => {
* setCount((currCount) => currCount + 1);
* }
* useEffect(function printSomething() {
* console.log("this is a side-effect")
* })
* return(
* <div>
* <h3>count = {count}</h3>
* <button onClick={incCount}>+1</button>
* </div>
* );
* }

# Dependencies in useEffect()

* 
* Counter.jsx
* import { useState, useEffect } from "react";
* export default function Counter() {
* let [countx, setCountx] = useState(0);
* let [county, setCounty] = useState(0);
* let incCountx = () => {
* setCountx((currCount) => currCount + 1);
* }
* let incCounty = () => {
* setCounty((currCount) => currCount + 1);
* }
* useEffect(
* function printSomething() {
* console.log("this is a side-effect");
* },
* []   // or [countx] or [county] or [countx, county] empty arr means this will not print.
* )
* return(
* <div>
* <h3>countx = {countx}</h3>
* <button onClick={incCountx}>+1</button>
* <h3>county = {county}</h3>
* <button onClick={incCounty}>+1</button>
* </div>
* );
* }

# Use cases

* Joker.jsx
* import { useState, useEffect } from "react";
* export default function Joker() {
* let [joke, setJoke] = useState({});
* const  URL = "https://official-joke-api.appspot.com/random\_joke";
* const getNewJoke = async () => {
* let response = await fetch(URL);
* let jsonResponse = await response.json();
* setJoke({setup: jsonResponse.setup, punchline: jsonResponse.punchline})
* }
* useEffect(()=>{async function getFirstJoke() {
* let response = await fetch(URL);
* let jsonResponse = await response.json();
* console.log(jsonResponse);
* setJoke({setup: jsonResponse.setup, punchline: jsonResponse.punchline})
* }
* getFirstJoke();
* }, [])
* return (
* <div>
* <h3>Joker!</h3>
* <h2>{joke.setup}</h2>
* <h2>{joke.punchline}</h2>
* <button onClick={getNewJoke}>New Joke</button>
* </div>
* );
* }
* React (Mini-Project)

# Material UI

* Npm create vite
* mini-project-react
* cd mini-project-react
* npm install
* npm run dev
* Material UI is similer to bootstrap so we can directly reed from docs and use it.
* <https://mui.com/material-ui/react-button/>

# Building Search box

* Searchbox.jsx
* import TextField from '@mui/material/TextField';
* import Button from '@mui/material/Button';
* import './SearchBox.css';
* import { useState } from 'react';
* export default function SearchBox() {
* let [city, setCity] = useState("");
* let handleChange = (evt) => {
* setCity(evt.target.value)
* }
* let handleSubmit = (evt) => {
* evt.preventDefault();
* console.log(city);
* setCity("");
* }
* return(
* <div className='SearchBox'>
* <h3>Search for the weather</h3>
* <form onSubmit={handleSubmit}>
* <TextField id="city" label="City Name" variant="outlined" required value={city} onChange={handleChange}/>
* <br></br>
* <br></br>
* <Button variant="contained" type="submit">Search</Button>
* </form>
* </div>
* );
* }
* Searchbox.css
* .SearchBox {
* text-align: center;
* }

# Using Weather API

* Searchbox.jsx
* import TextField from '@mui/material/TextField';
* import Button from '@mui/material/Button';
* import './SearchBox.css';
* import { useState } from 'react';
* export default function SearchBox() {
* let [city, setCity] = useState("");
* const API\_URL = "https://api.openweathermap.org/data/2.5/weather";
* const API\_KEY = "92febcbb08fa594e0686f80954ad16e5"
* let getWeatherInfo = async () => {
* let response = await fetch(`${API\_URL}?q=${city}&appid=${API\_KEY}&units=metric`);
* let jsonResponse = await response.json();
* console.log(jsonResponse);
* let result = {
* temp: jsonResponse.main.temp,
* tempMin: jsonResponse.main.temp\_min,
* tempMax: jsonResponse.main.temp\_max,
* humidity: jsonResponse.main.humidity,
* feelsLike: jsonResponse.main.feels\_like,
* weather: jsonResponse.weather[0].description,
* };
* console.log(result);
* };
* let handleChange = (evt) => {
* setCity(evt.target.value)
* }
* let handleSubmit = (evt) => {
* evt.preventDefault();
* console.log(city);
* setCity("");
* getWeatherInfo();
* }
* return(
* <div className='SearchBox'>
* <h3>Search for the weather</h3>
* <form onSubmit={handleSubmit}>
* <TextField id="city" label="City Name" variant="outlined" required value={city} onChange={handleChange}/>
* <br></br>
* <br></br>
* <Button variant="contained" type="submit">Search</Button>
* </form>
* </div>
* );
* }

# Building Info Box

* Infobox.jsx
* import Card from '@mui/material/Card';
* import CardContent from '@mui/material/CardContent';
* import CardMedia from '@mui/material/CardMedia';
* import Typography from '@mui/material/Typography';
* import "./InfoBox.css";
* export default function InfoBox() {
* const INIT\_URL = "https://images.unsplash.com/photo-1693711836001-99859bb7185a?w=500&auto=format&fit=crop&q=60&ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxzZWFyY2h8Nnx8ZHVzdHklMjB3ZWF0aGVyfGVufDB8fDB8fHww"
* let info = {
* city: "Delhi",
* feelslike: 24.84,
* temp: 25.05,
* tempMin: 25.05,
* tempMax: 25.05,
* humidity: 47,
* weather: "haze"
* };
* return (
* <div className="InfoBox">
* <h1>WeatherInfo - {info.weather}</h1>
* <Card sx={{ maxWidth: 345 }}>
* <CardMedia
* sx={{ height: 140 }}
* image={INIT\_URL}
* title="green iguana"
* />
* <CardContent>
* <Typography gutterBottom variant="h5" component="div">
* {info.city}
* </Typography>
* <Typography variant="body2" color="text.secondary" component={"span"}>
* <p>Temperature = {info.temp}&deg;C</p>
* <p>Humidity = {info.humidity}</p>
* <p>Min Temp = {info.tempMin}&deg;C</p>
* <p>Max Temp = {info.tempMax}&deg;C</p>
* <p>The weather can be described as <i>{info.weather}</i> and feels like {info.feelslike}&deg;C</p>
* </Typography>
* </CardContent>
* </Card>
* </div>
* )
* }
* Infobox.css
* .InfoBox {
* text-align: center;
* }

# Weather App Component

* Weatherapp.jsx
* import { useState } from "react";
* import InfoBox from "./InfoBox";
* import SearchBox from "./SearchBox";
* export default function WeatherApp() {
* const [weatherInfo, setWeatherInfo] = useState({
* city: "Wonderland",
* feelslike: 24.84,
* temp: 25.05,
* tempMin: 25.05,
* tempMax: 25.05,
* humidity: 47,
* weather: "haze"
* })
* let updateInfo = (newInfo) => {
* setWeatherInfo(newInfo);
* }
* return(
* <div style={{textAlign: "center"}}>
* <h2>Weather App by Delta</h2>
* <SearchBox updateInfo={updateInfo}/>
* <InfoBox info={weatherInfo}/>
* </div>
* );
* }
* Searchbox.jsx
* import TextField from '@mui/material/TextField';
* import Button from '@mui/material/Button';
* import './SearchBox.css';
* import { useState } from 'react';
* export default function SearchBox({updateInfo}) {
* let [city, setCity] = useState("");
* let [error, setError] = useState(false);
* const API\_URL = "https://api.openweathermap.org/data/2.5/weather";
* const API\_KEY = "92febcbb08fa594e0686f80954ad16e5"
* let getWeatherInfo = async () => {
* try{
* let response = await fetch(`${API\_URL}?q=${city}&appid=${API\_KEY}&units=metric`);
* let jsonResponse = await response.json();
* let result = {
* city: city,
* temp: jsonResponse.main.temp,
* tempMin: jsonResponse.main.temp\_min,
* tempMax: jsonResponse.main.temp\_max,
* humidity: jsonResponse.main.humidity,
* feelsLike: jsonResponse.main.feels\_like,
* weather: jsonResponse.weather[0].description,
* };
* console.log(result);
* return result;
* }catch(err){
* throw err;
* }
* };
* let handleChange = (evt) => {
* setCity(evt.target.value)
* }
* let handleSubmit = async (evt) => {
* try{
* evt.preventDefault();
* console.log(city);
* setCity("");
* let info = await getWeatherInfo();
* updateInfo(info);
* }catch(err){
* setError(true)
* }
* }
* return(
* <div className='SearchBox'>
* <form onSubmit={handleSubmit}>
* <TextField id="city" label="City Name" variant="outlined" required value={city} onChange={handleChange}/>
* <br></br>
* <br></br>
* <Button variant="contained" type="submit">Search</Button>
* {error && <p style={{color: "red"}}>No such place exists!</p>}
* </form>
* </div>
* );
* }
* Infobox.jsx
* import Card from '@mui/material/Card';
* import CardContent from '@mui/material/CardContent';
* import CardMedia from '@mui/material/CardMedia';
* import Typography from '@mui/material/Typography';
* import "./InfoBox.css";
* export default function InfoBox({info}) {
* const INIT\_URL = "https://images.unsplash.com/photo-1693711836001-99859bb7185a?w=500&auto=format&fit=crop&q=60&ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxzZWFyY2h8Nnx8ZHVzdHklMjB3ZWF0aGVyfGVufDB8fDB8fHww"
* return (
* <div className="InfoBox">
* <div className='cardContainer'>
* <Card sx={{ maxWidth: 345 }}>
* <CardMedia
* sx={{ height: 140 }}
* image={INIT\_URL}
* title="green iguana"
* />
* <CardContent>
* <Typography gutterBottom variant="h5" component="div">
* {info.city}
* </Typography>
* <Typography variant="body2" color="text.secondary" component={"span"}>
* <p>Temperature = {info.temp}&deg;C</p>
* <p>Humidity = {info.humidity}</p>
* <p>Min Temp = {info.tempMin}&deg;C</p>
* <p>Max Temp = {info.tempMax}&deg;C</p>
* <p>The weather can be described as <i>{info.weather}</i> and feels like {info.feelslike}&deg;C</p>
* </Typography>
* </CardContent>
* </Card>
* </div>
* </div>
* )
* }

# Added feature

* Icon feature added