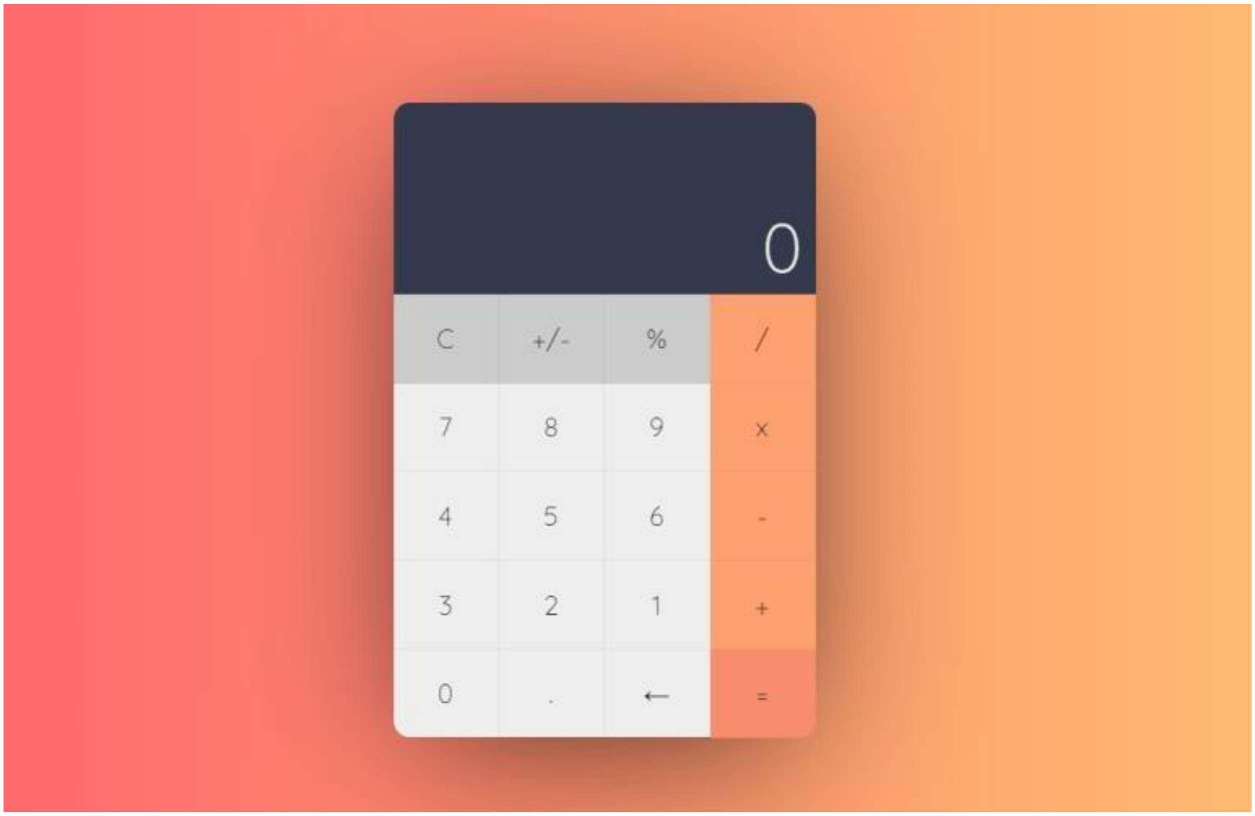
Week-4. Write a program to create a simple calculator Application using React JS

```
class App extends Component {
constructor() {
super()
this.state = { operations: []
} }
.....
}
render()
{
return (
   <div className="App">
    <Display data={this.state.operations} />
    <Buttons>
     <Button onClick={this.handleClick} label="C" value="clear" />
     <Button onClick={this.handleClick} label="7" value="7" />
     <Button onClick={this.handleClick} label="4" value="4" />
     <Button onClick={this.handleClick} label="1" value="1" />
     <Button onClick={this.handleClick} label="0" value="0" />     <Button
onClick={this.handleClick} label="/" value="/" />
     <Button onClick={this.handleClick} label="8" value="8" />
     <Button onClick={this.handleClick} label="5" value="5" />
     <Button onClick={this.handleClick} label="2" value="2" />
     <Button onClick={this.handleClick} label="." value="." />     <Button
onClick={this.handleClick} label="x" value="*" />
     <Button onClick={this.handleClick} label="9" value="9" />
     <Button onClick={this.handleClick} label="6" value="6" />
     <Button onClick={this.handleClick} label="3" value="3" />
     <Button label="" value="null" />     <Button onClick={this.handleClick}
label="-" value="-" />
     <Button onClick={this.handleClick} label="+" size="2" value="+" />
     <Button onClick={this.handleClick} label="=" size="2" value="equal" />
</Buttons>
   </div>
  )
class Buttons extends Component {
```

```
render() {
return <div className="Buttons"> {this.props.children} </div>
}
} class Button extends Component {
 render() {
return (
<div
    onClick={this.props.onClick}
className="Button"     data-
size={this.props.size}
    data-value={this.props.value}
   >
    {this.props.label}
   </div>
  )
 }
}
class Display extends Component {
render() {
  const string = this.props.data.join('')
  return <div className="Display"> {string} </div>
 } } handleClick =
e => {
  const value = e.target.getAttribute('data-
value')   switch (value) {    case 'clear':
this.setState({      operations: [],
    })     break    case
'equal':
this.calculateOperations()
    break
default:
    const newOperations = update(this.state.operations, {
     $push: [value],
    })
this.setState({
     operations: newOperations,
    })
    break
  }


calculateOperations = () => {     let
result = this.state.operations.join('')
if (result) {
   result = math.eval(result)
   result = math.format(result, { precision: 14 })
   result = String(result)
this.setState({
operations: [result],
   })
  }
 }
```

OUTPUT:


0
C
+/-
%
/
7
8
9
x
4
5
6
-
3
2
1
+
0
.
←
=