

## **React Lab 2 - Instruction**

### **Objective:**

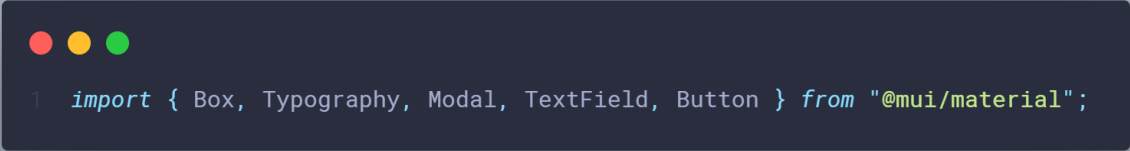
1. Be able to use and style components from MUI Framework
2. Be able to validate the text form
3. Be able to understand how to pass and render props from parent component

### **Step 1:** Install MUI Framework via package manager (npm)

```
npm install @mui/material @emotion/react @emotion/styled
```

### **Step 2:** Inside src/components/ToDo.jsx

Import Modal, Box, Button, TextField and Typography from MUI Library



```
import { Box, Typography, Modal, TextField, Button } from "@mui/material";
```

### Step 3: Create style objects

Create these objects outside the Todo function

```
1  const addNewTaskBarStyle = {
2    padding: "20px",
3    background: "#feffff",
4    width: { xs: "300px", md: "500px" },
5    borderRadius: "10px",
6    margin: "20px 0px",
7    fontSize: "25px",
8    color: "#7b7b7b",
9    display: "flex",
10   alignItems: "center",
11   justifyContent: "center",
12   boxShadow: "0 4px 8px 0 rgba(0,0,0,0.2)",
13 };
14
15 const wrapperTodoListStyle = {
16   display: "flex",
17   flexDirection: "column",
18   alignItems: "center",
19   marginTop: "20px",
20 };
21
22 const wrapperHeaderStyle = {
23   display: "flex",
24   justifyContent: "space-between",
25   alignItems: "center",
26 };
27
```



```
1  const headerTextStyle = {
2    color: "#feffff",
3    fontSize: { xs: "50px", md: "60px" },
4  };
5
6  const headerTodoListLengthStyle = {
7    padding: "20px 30px",
8    backgroundColor: "#b0a3f5",
9    boxShadow: "0 4px 8px 0 rgba(0,0,0,0.2)",
10   fontSize: "50px",
11   borderRadius: "10px",
12   color: "white",
13   fontWeight: "bold",
14 };
15
16 const modalStyle = {
17   display: "flex",
18   flexDirection: "column",
19   alignItems: "center",
20   justifyContent: "center",
21   position: "absolute",
22   top: "50%",
23   left: "50%",
24   transform: "translate(-50%, -50%)",
25   width: 400,
26   bgcolor: "background.paper",
27   border: "solid 3px #b0a3f5",
28   boxShadow: 24,
29   p: 4,
30   borderRadius: "10px",
31 };
32
33 const horizontalStyle = {
34   width: "90%",
35   backgroundColor: "white",
36   boxShadow: "0 4px 8px 0 rgba(0,0,0,0.2)",
37 };
38
```

## Step 4: Remove all style object from the JSX

```
1  return (  
2    <div>  
3      <div  
4        style={}  
5      >  
6        <div>  
7          <h1 style={}>Incoming</h1>  
8        </div>  
9        <div  
10         style={}  
11       >  
12         {todo.length}  
13       </div>  
14     </div>  
15     <div  
16       style={}  
17       onClick={handleAdd}  
18     >  
19       + Add New Tasks  
20     </div>  
21     {todo.map((item)=>{  
22       return <CardList task={item}/>  
23     })}  
24   </div>  
25 );  
26 }
```

**Step 5:** Change the first div to be Box component and add style objects back to the JSX as shown in the picture. Change the h1 tag to be Typography component with variant of "h1"

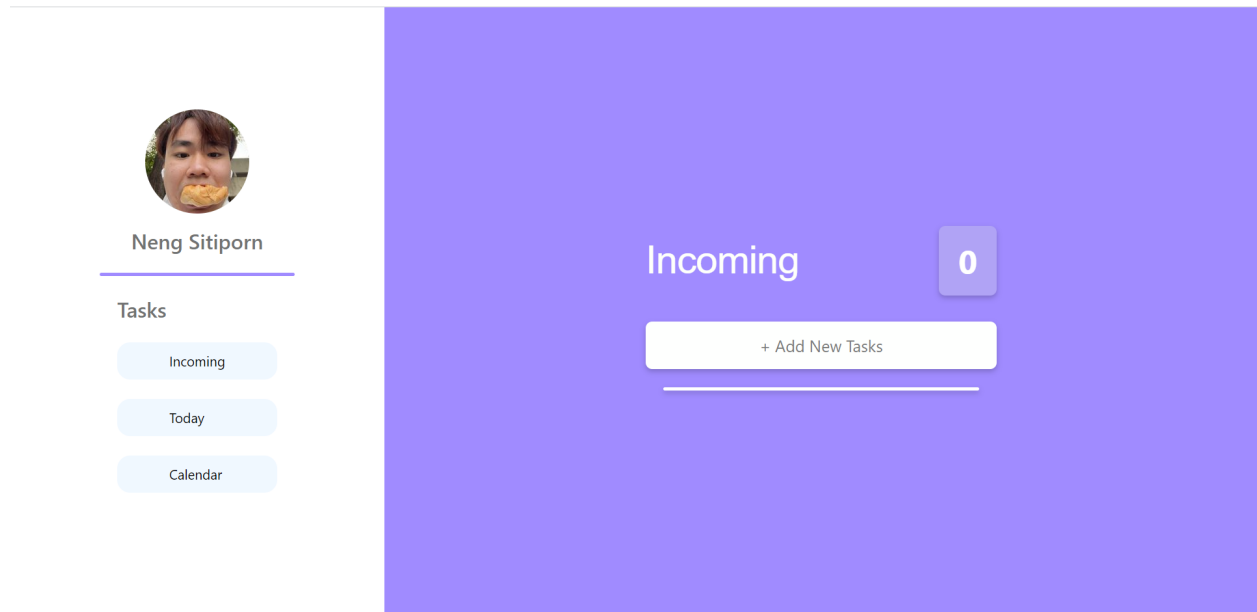
```
1 <Box sx={{ margin: "20px" }}>
2   <div style={wrapperHeaderStyle}>
3     <div>
4       <Typography variant="h1" sx={headerTextStyle}>
5         Incoming
6       </Typography>
7     </div>
8     <div style={headerTodoListLengthStyle}>{todo.length}</div>
9   </div>
10  <div style={addNewTaskBarStyle} onClick={handleAdd}>
11    + Add New Tasks
12  </div>
13  {todo.map((item) => {
14    return <CardList task={item} />;
15  })}
16 </Box>
```

**Step 6:** Wrap the div that has onClick with Box and change the div itself to be Box and Add hr tag at the end

```
1 <Box sx={{ margin: "20px" }}>
2   <div style={wrapperHeaderStyle}>
3     <div>
4       <Typography variant="h1" sx={headerTextStyle}>
5         Incoming
6       </Typography>
7     </div>
8     <div style={headerTodoListLengthStyle}>{todo.length}</div>
9   </div>
10  <Box sx={wrapperTodoListStyle}>
11    <Box sx={addNewTaskBarStyle} onClick={handleAdd}>
12      + Add New Tasks
13    </Box>
14    {todo.map((item) => {
15      return <CardList task={item} />;
16    })}
17  </Box>
18  <hr style={horizontalStyle} />
19 </Box>
```

**!!! NOTE !!!:** Change onClick={handleAdd} to be  
onClick={handleOpen}

## Checkpoint 1: Your App should look like this picture



**Step 7:** In Todo.jsx at the bottom of component outside the first Box, create Modal with TextField inside

```
1 <Modal
2     open={open}
3     onClose={handleClose}
4     aria-labelledby="modal-modal-title"
5     aria-describedby="modal-modal-description"
6   >
7     <Box sx={modalStyle}>
8       <TextField
9         error={todoError}
10        id="todo"
11        label="What to you want to do ?"
12        variant="outlined"
13        onChange={(e) => handleInput(e)}
14      />
15      <Button
16        variant="contained"
17        onClick={handleAdd}
18        sx={{ margin: "20px 0px 0px 0px" }}
19        disabled={todoError}
20      >
21        Add
22      </Button>
23    </Box>
24  </Modal>
```



**Step 8:** Then wrap all JSX with React.Fragment `<></>`

```
1  return (  
2    <>  
3  
4      {your previous code}  
5  
6    </>  
7  );
```

**Step 9:** Create a state for the Modal at the top of your function component where the rest of the state components are.

```
1  function Todo() {  
2    //state of modal  
3    const [open, setOpen] = useState(false);  
4  
5    ...  
6  }
```

## Step 10: Create functions for handling Modal

```
1 function Todo() {  
2   //state of modal  
3   const [open, setOpen] = useState(false);  
4  
5   //Modal Function  
6   const handleOpen = () => setOpen(true);  
7   const handleClose = () => setOpen(false);  
8  
9   ...  
10 }
```

**Step 11:** Create another 2 states for storing todo text and state of validation



```
1 function Todo() {  
2  
3   ...  
4   //state of modal  
5   const [open, setOpen] = useState(false);  
6  
7   //state for validation  
8   const [todoInput, setTodoInput] = useState("");  
9   const [todoError, settodoError] = useState(true);  
10  
11   ...  
12 }
```

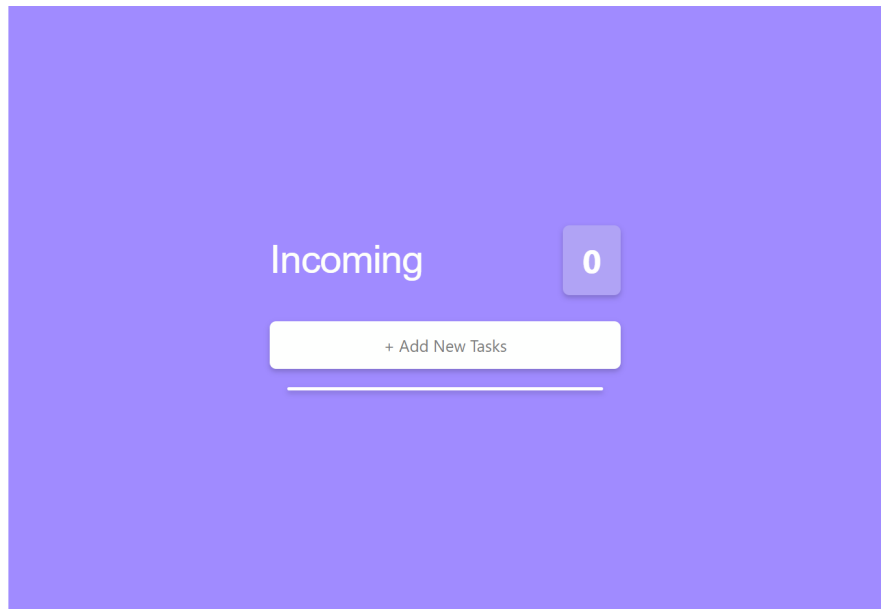
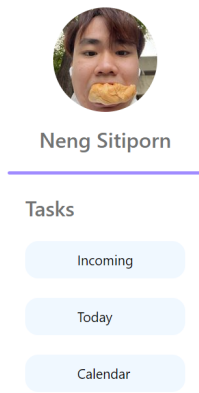
## Step 12: Create function for handling handle input of TextField

```
1 function Todo() {  
2  
3   //Modal Function  
4   const handleOpen = () => setOpen(true);  
5   const handleClose = () => setOpen(false);  
6  
7   //Handle Input  
8   function handleInput(e) {  
9     setTodoInput(e.target.value);  
10    if (e.target.value === null || e.target.value === "") {  
11      settodoError(true);  
12    } else {  
13      settodoError(false);  
14    }  
15  }  
16  
17  ...  
18 }
```

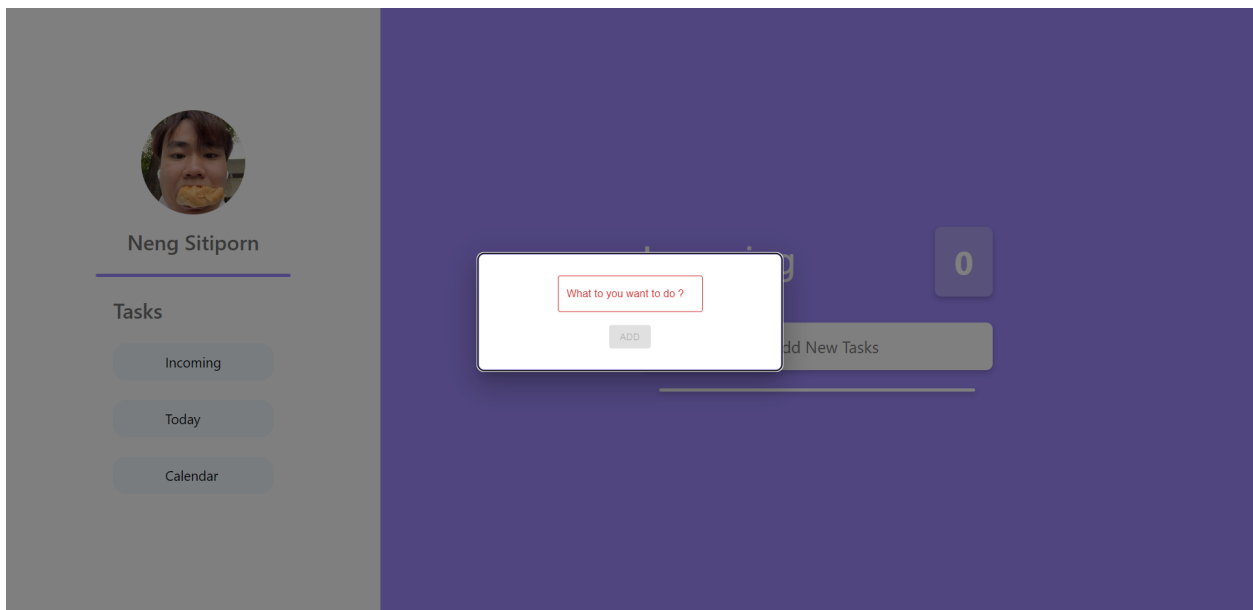
### Step 13: Create function for handling add button

```
1 function Todo() {  
2  
3   ...  
4  
5   //Handle Input Change  
6   function handleInput(e) {  
7     setTodoInput(e.target.value);  
8     if (e.target.value === null || e.target.value === "") {  
9       settodoError(true);  
10    } else {  
11      settodoError(false);  
12    }  
13  }  
14  
15  //Handle Add button  
16  function handleAdd() {  
17    setTodo([...todo, todoInput]);  
18    setTodoInput("");  
19    settodoError(true);  
20    handleClose();  
21  }  
22  
23  ...  
24 }
```

## Checkpoint 2:



When click the +Add new Tasks:




## Step 14: Pass the prop to render the CardList

```
1  {/* Render Todo(s) into CardList */}
2      {todo.map((item, index) => {
3          return (
4              <CardList
5                  todo={item}
6                  key={index}
7                  setState={setTodo}
8                  state={todo}
9              />
10         );
11     })}
```

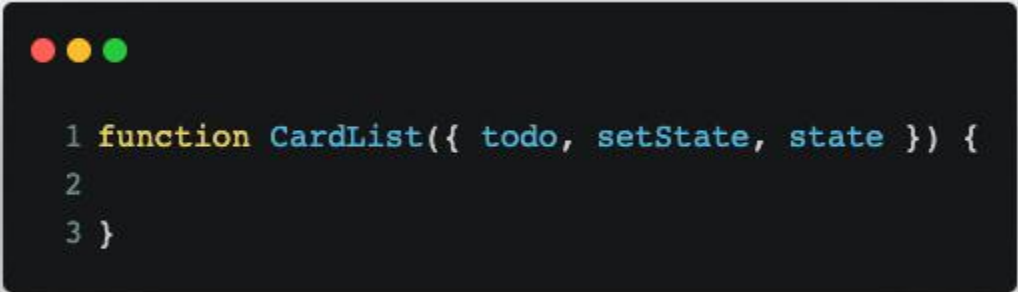
**Step 15:** In CardList.jsx file

Import Box and Checkbox component



```
1 import { Box, Checkbox } from "@mui/material";
```

**Step 16:** Add props to the CardList



```
1 function CardList({ todo, setState, state }) {  
2  
3 }
```



## Step 17: Create style object outside the CardList function

```
1  const wrapperStyle = {
2    padding: "10px 20px",
3    background: "#feffff",
4    width: { xs: "300px", md: "500px" },
5    borderRadius: "10px",
6    margin: "20px 0px",
7    fontSize: "25px",
8    color: "#7b7b7b",
9    display: "flex",
10   alignItems: "center",
11   justifyContent: "space-between",
12   boxShadow: "0 4px 8px 0 rgba(0,0,0,0.2)",
13 };
```

**Step 18:** Remove all your JSX and Create Box that has Checkbox. Your CardList component should have JSX like shown in the picture

```
1  <Box sx={wrapperStyle}>
2    <Checkbox onChange={(e) => handleCheck(e)} />
3    <div>{todo}</div>
4  </Box>
```

## Step 19: Create a handleCheck function

```
1 function CardList({ todo, setState, state }) {  
2  
3   function handleCheck(e) {  
4     if (e.target.checked) {  
5       const result = state.filter((item) => item !== todo);  
6       setState(result);  
7     }  
8   }  
9  
10  ...  
11  
12  
13 }
```

## Checkpoint 3:

