## MODUL REACT NATIVE CRUD LOCAL

- Buatlah sebuah projek baru dengan nama TodoList
- Buat kode pada file App.js dengan kode berikut

- Buat folder src dalam direktori utama, kemudian dalam folder src buat folder screens (TodoList/src/screens)
- Dalam folder screens, buatlah file bernama TodoList.js dengan kode sebagai berikut

```
// components/TodoList.js
import React, { useState } from "react";
import { View, TextInput, Button } from "react-native";
import TodoItem from "./TodoItem";
const TodoList = () => {
  // State Hooks
  const [tasks, setTasks] = useState([
    { id: 1, text: "Doctor Appointment", completed: true }, { id: 2, text: "Meeting at School", completed: false },
  const [text, setText] = useState("");
// Function to Add Task
  function addTask() {
    const newTask = { id: Date.now(), text, completed: false };
    setTasks([...tasks, newTask]);
setText("");
  // Function to Delete Task
  function deleteTask(id)
    setTasks(tasks.filter((task) => task.id !== id));
  // Function to Toggle Task Completion
  function toggleCompleted(id) {
    setTasks(
      tasks.map((task) =>
        task.id === id ? { ...task, completed: !task.completed } : task
 );
      {tasks.map((task) => (
         <TodoItem
           key={task.id}
           task={task}
           deleteTask={deleteTask}
           toggleCompleted={toggleCompleted}
      ))]
      <TextInput value={text} onChangeText={setText} placeholder="New Task" />
       <Button title="Add" onPress={addTask} />
    </View>
 );
};
export default TodoList;
```

## Penjelasan:

• Import Required Modules: Impor modul penting untuk Komponen TodoList Project

- **Define TodoList Component**: Kita deklarasikan TodoList functional component, komponen inti untuk memanajemen komponen lainnya.
- State Hooks: Menggunakan useState hook, kita inisialisasi 2 state variabel: tasks untuk
  memanajemen jumlah tasks, and text mengatasi input text untuk tasks baru. Variabel tasks
  state diinisialisasi dengan 2 contoh data.
- Function to Add Task: Fungsi add Task function membuat task baru dengan ID Unique, input text, dan completion status set false.
- Function to Delete Task: Fungsi delete Task menghapus sebuah task berdasar ID dari task yang dipilih.
- Function to Toggle Task Completion: Fungsi toggleCompleted untuk merubah status berdasar ID yang diklik.
- Render TodoList Component: TodoList component renders menampilkan list tasks menggunakan metode.
- Dalam folder screens, buatkan file bernama Todoltem.js dengan kode sebagai berikut

```
import React from 'react';
import { View, Text, StyleSheet, Button, TouchableOpacity } from 'react-native';
const TodoItem = ({ task, deleteTask, toggleCompleted }) => {
   return(
        <View>
        <Text>
          {task.text}
        </Text>
        <TouchableOpacity>
          <Text style={{ color: '#fff' }}>Delete</Text>
        </TouchableOpacity>
        <TouchableOpacity
         onPress={() => toggleCompleted(task.id)}
          <Text style={{ color: '#fff' }}>{task.completed && 'completed'}{!task.completed &&
'uncompleted'}</Text>
        </TouchableOpacity>
      </View>
  export default TodoItem;
```

Y" II I O P

C V B N M

L

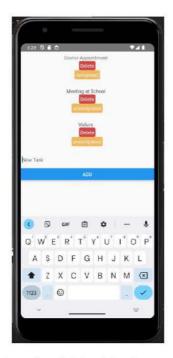
GHJ

W E R

0

• Tambahkan styling pada file Todoltem.js (tambahkan/sesuaikan yang ditebalkan saja)

```
// components/TodoItem.js
import React from 'react'
import { View, Text, StyleSheet, Button, TouchableOpacity } from 'react-native';
const styles = StyleSheet.create({
    todoitem: {
         display: 'flex',
justifiContent: 'space-between',
         alignItems: 'center', /* Align items vertically in the center */
         marginBottom: 8,
         padding: 8,
         border: 1,
         borderColor: '#ddd'.
        borderRadius: 4,
       todoitemtext: {
         // flex: 1, /* Allow the text to take up remaining space */ marginRight: 8,
         color: '#333'.
       completed: {
         textDecorationLine: 'line-through',
         color: '#888',
       deletebutton: {
         backgroundColor: '#D04848', /* Tomato color */
         color: '#fff',
         padding: 4,
         border: 'none'
         borderRadius: 4.
         cursor: 'pointer',
       completebutton: {
         backgroundColor: '#F3B95F',
         color: '#0000'.
         padding: 4,
         border: 'none',
        borderRadius: 4,
         cursor: 'pointer',
});
const TodoItem = ({ task, deleteTask, toggleCompleted }) => {
    <View style={styles.todoitem}>
       <Text style={[styles.todoitemtext, task.completed && styles.completed]}>
         {task.text}
       </Text>
       <TouchableOpacity
         style={styles.deletebutton}
        onPress={() => deleteTask(task.id)}
        <Text style={{ color: "#fff" }}>Delete</Text>
      </TouchableOpacity>
      <TouchableOpacity
        style={styles.completebutton}
         onPress={() => toggleCompleted(task.id)}
        <Text style={{ color: "#fff" }}> {task.completed && "completed"
           {!task.completed && "uncompleted"}
         </Text>
       </TouchableOpacity>
);
};
     </View>
export default TodoItem;
```



• Silakan dicoba, harusnya semua fungsi sudah berjalan!

## **TUGAS**

- Lakukan styling atau ubah user interface atau tampilan dari aplikasi yang sudah jadi!
- Kumpulkan dalam bentuk pdf isinya kode program dan UI before UI After!