# MSc CA Semester 1 – React Practical Solutions (Q2–Q5)

Q2: React Component – StudentList ---------- Create a React component StudentList that: - Accepts an array of student objects {id, name, marks} - Displays them in a table - Highlights rows where marks < 35 using red background

```
import React from "react";

const StudentList = ({ students }) => {
  return (
    <table border="1" cellPadding="10" style={{ width: "60%" }}>
      <thead>
        <tr>
          <th>ID</th>
          <th>Name</th>
          <th>Marks</th>
        </tr>
      </thead>

      <tbody>
        {students.map((s) => (
          <tr
            key={s.id}
            style={{
              backgroundColor: s.marks < 35 ? "red" : "white",
              color: s.marks < 35 ? "white" : "black",
            }}
          >
            <td>{s.id}</td>
            <td>{s.name}</td>
            <td>{s.marks}</td>
          </tr>
        ))}
      </tbody>
    </table>
  );
};

export default StudentList;
```

Q3: React Hooks – Counter Application ---------- A React app that: - Uses useState to store a counter - Buttons: Increment, Decrement, Reset - Prevents count from going below 0

```
import React, { useState } from "react";

const CounterApp = () => {
  const [count, setCount] = useState(0);

  const increment = () => setCount(count + 1);

  const decrement = () => {
    if (count > 0) setCount(count - 1);
  };

  const reset = () => setCount(0);

  return (
    <div style={{textAlign:"center"}}>
      <h2>Counter: {count}</h2>
      <button onClick={increment}>Increment</button>
      <button onClick={decrement} style={{margin:"0 10px"}}>Decrement</button>
      <button onClick={reset}>Reset</button>
    </div>
  );
};

export default CounterApp;
```

Q4: API Integration – Users Component ---------- Fetches user data from API and displays: - Name - Email - Phone Shows "Loading..." while fetching.

```
import React, { useEffect, useState } from "react";

const Users = () => {
  const [users, setUsers] = useState([]);
  const [loading, setLoading] = useState(true);

  useEffect(() => {
    fetch("https://jsonplaceholder.typicode.com/users")
      .then((res) => res.json())
      .then((data) => {
        setUsers(data);
        setLoading(false);
      });
  }, []);

  if (loading) {
    return <h3>Loading...</h3>;
  }

  return (
    <div>
      <h2>User List</h2>
      <table border="1" cellPadding="10" style={{ width: "70%" }}>
        <thead>
          <tr>
            <th>Name</th>
            <th>Email</th>
            <th>Phone</th>
          </tr>
        </thead>

        <tbody>
          {users.map((u) => (
            <tr key={u.id}>
              <td>{u.name}</td>
              <td>{u.email}</td>
              <td>{u.phone}</td>
            </tr>
          ))}
        </tbody>
      </table>
    </div>
  );
};

export default Users;
```

Q5: Redux Toolkit – Todo Slice and Component ---------- Operations: - addTodo - deleteTodo - toggleComplete

```
import { createSlice } from "@reduxjs/toolkit";

const todoSlice = createSlice({
  name: "todos",
  initialState: [],
  reducers: {
    addTodo: (state, action) => {
      state.push({
        id: Date.now(),
        text: action.payload,
        completed: false,
      });
    },
    deleteTodo: (state, action) => {
      return state.filter((todo) => todo.id !== action.payload);
    },
    toggleComplete: (state, action) => {
      const todo = state.find((t) => t.id === action.payload);
```

```
      if (todo) {
        todo.completed = !todo.completed;
      }
    },
  },
});

export const { addTodo, deleteTodo, toggleComplete } = todoSlice.actions;
export default todoSlice.reducer;
import React, { useState } from "react";
import { useSelector, useDispatch } from "react-redux";
import { addTodo, deleteTodo, toggleComplete } from "./todoSlice";

const TodoApp = () => {
  const [text, setText] = useState("");
  const todos = useSelector((state) => state.todos);
  const dispatch = useDispatch();

  return (
    <div>
      <h2>Todo List</h2>

      <input
        type="text"
        value={text}
        placeholder="Enter todo"
        onChange={(e) => setText(e.target.value)}
      />
      <button onClick={() => dispatch(addTodo(text))}>Add</button>

      <ul>
        {todos.map((t) => (
          <li
            key={t.id}
            onClick={() => dispatch(toggleComplete(t.id))}
            style={{
              textDecoration: t.completed ? "line-through" : "none",
              cursor: "pointer",
            }}
          >
            {t.text}

            <button
              onClick={() => dispatch(deleteTodo(t.id))}
              style={{ marginLeft: "10px" }}
            >
              Delete
            </button>
          </li>
        ))}
      </ul>
    </div>
  );
};

export default TodoApp;
```