Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

Experiment Title: JavaScript OOPs Concepts and react Hooks: Implement react hooks by using student grade system

Aim/Objective

To implement a Student Grade System using JavaScript Object-Oriented Programming (OOPs) concepts and React Hooks, demonstrating state management, component-based architecture, and reusability in a modern frontend development workflow.

Description:

This project involves building a Student Grade System that utilizes JavaScript Object-Oriented Programming (OOPs) concepts for data modeling and logic, along with React Hooks (useState, useEffect, etc.) for managing component state and lifecycle in a functional way. The system allows users to add, update, and display student grades dynamically with an interactive UI.

Prerequisites:

- Understanding of HTML, CSS, and JavaScript (ES6+).
- Basic knowledge of JavaScript OOP concepts such as classes, constructors, methods, and inheritance.
- Familiarity with React.js fundamentals – components, props, and JSX.
- Hands-on experience with React Hooks like useState and useEffect.
- Knowledge of array operations (e.g., map, filter, reduce) for data manipulation.
- Working setup of Node.js and React development environment (e.g., Create React App).

Pre-Lab: Before starting the lab activity, students should:

- Review JavaScript OOP concepts such as classes, objects, constructors, and methods for structuring student data.
- Revise React fundamentals, including creating functional components and using props.
- Understand how React Hooks work, especially useState for managing component state and useEffect for side effects.
- Set up a React project using Create React App or another preferred setup.
- Plan the component structure, such as components for student form input, grade list display, and overall layout.
- Prepare sample student data for testing and practice how to add, update, and display it using both OOP logic and React Hooks.

In-Lab: During the lab session, students will:

- Create a React application using Create React App or Vite.
- Design a class in JavaScript (e.g., Student) to model student data like name, ID, and marks, applying OOP concepts.

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

- Build functional React components for:
    - Adding new student data
    - Displaying a list of students with their grades
    - Calculating grades based on marks
- Use useState Hook to manage form inputs and student list dynamically.
- Use useEffect Hook to trigger calculations or update the grade view when student data changes.
- Implement user interactions such as adding, editing, and clearing student records.
- Style the components with basic CSS to enhance user experience.
- Test the functionality thoroughly to ensure all features work as expected.

Procedure/Program:

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

Data and Results:

Analysis and Inferences:

Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	<TO BE FILLED BY STUDENT>

Sample viva voce questions:

1. **What is the difference between a class and an object in JavaScript OOP?**

2. **How does useState work in React, and why is it used in this project?**

3. **Explain how you used the useEffect hook in the Student Grade System.**

4. **How do React components re-render when state is updated?**

5. **What are the advantages of using functional components and hooks over class-based components in React?**

Evaluator Remark (if Any):	**Marks Secured:_____out of 50**
	Signature of the Evaluator with Date