**MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI**

**DEPARTMENT OF AI&DS**

**19AD602 – FULL STACK DEVELOPMENT**

**Exercise 9**

**Even Batch**

1. Write a Type script to calculate the Pay of the Employees in an organization using classes. The program makes an invocation for getting the calculated pay. Display appropriate results.

2. Create a generic class called Stack that implements a stack data structure. The class should have a generic type parameter that specifies the type of values that can be stored in the stack. The class should have the following methods:

• push(value: T): void - adds a value to the top of the stack

• pop(): T | undefined - removes and returns the value at the top of the stack, or returns undefined if the stack is empty

• isEmpty(): boolean - returns true if the stack is empty, or false if it contains one or more values

3. Write a Typescript code to Create an interface called Book with properties for title, author, and publisher. Create an array of Book objects and display their information in a list, sorted by author name in ascending order.

**ODD Batch**

1. Write a Type script to handle the Bank Account details of different account Holders. Provide functions for withdraw, credit and checkbalance operations. Use suitable menu and invoke the functions. Display appropriate results.

2. Write a Typescript code that define an interface called Product. Then define an array of Product objects called products, which contains the data for all of our products. Export a function called getProductById that takes a product ID as a parameter and returns the Product object with that ID from the products array, or undefined if no such product exists.

Use the function displayProductDetails for displaying the product details by invoking getProductById with a product ID as a parameter.

3. Create a generic class called Queue that implements a Queue data structure. The class should have a generic type parameter that specifies the type of values that can be stored in the Queue. The class should have the following methods:

• Enqueue(value: T): void - adds a value at the rear of the Queue

• Dequeue(): T | undefined - removes and returns the value the front of the Queue, or returns undefined if the Queue is empty

• isEmpty(): boolean - returns true if the Queue is empty, or false if it contains one or more values

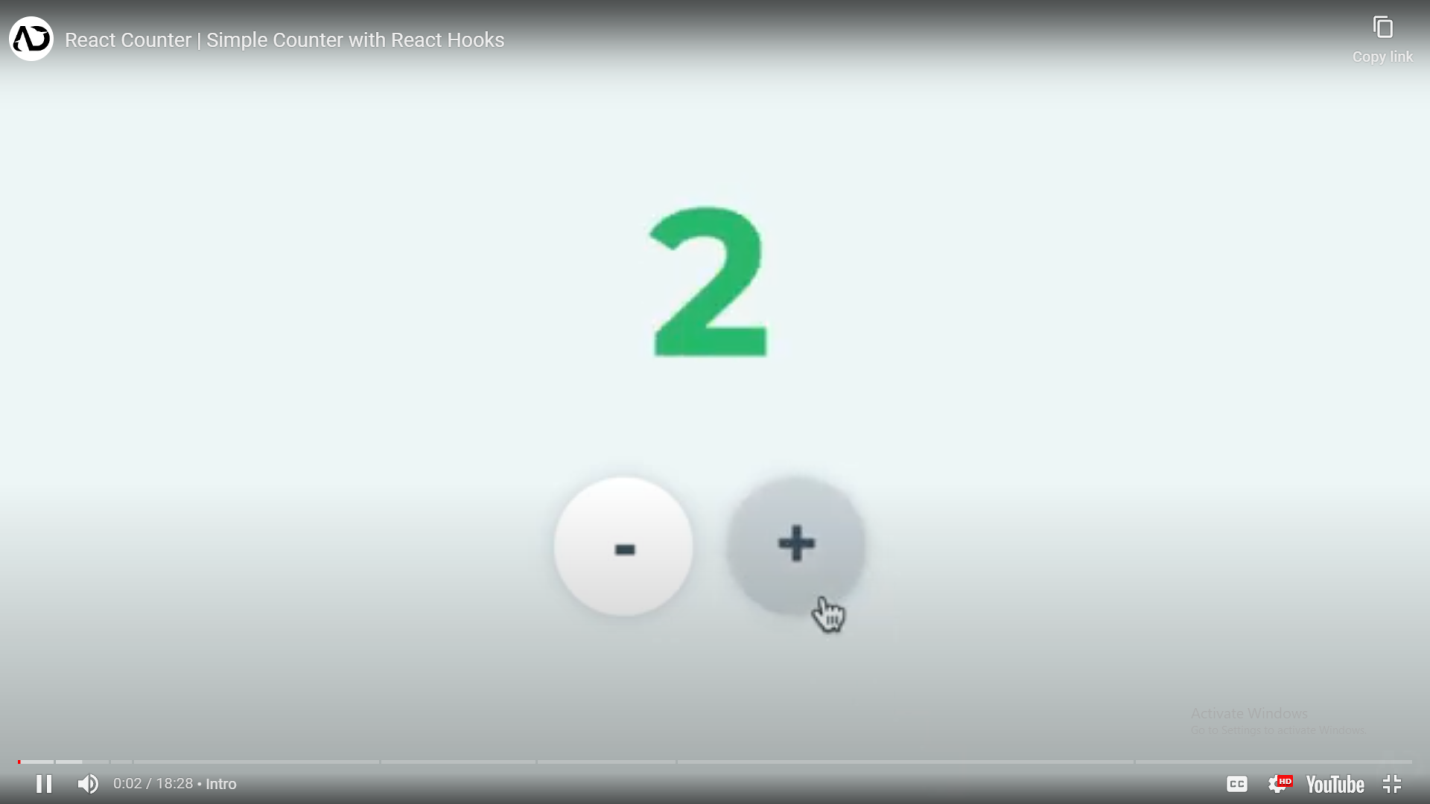
**Exercise 10**

## 1. Simple counter exercise

Creating a simple counter using React which increments or decrements count dynamically on-screen as the user clicks on the button. This exercise requires knowledge of fundamental React concepts such as State, Component, etc.

We can complete the simple counter exercise with the following steps:

1. Create React state to store the count value.
2. Declare JS functions to incement or decrement the value through **setState()**.
3. Add HTML buttons with onClick to JSX code.

**2. Filtering Array of objects in React JS**

Use filter for the array of objects in React JS. The React JS app for an E-Book shopping with have the following features:

1. Display a list of all Books with details and images.
2. Filter Book list by years ie displays all books with release\_year 2020.
3. Filter Book list by Publisher name from the dropdown, ie display all, Pearson, TMH etc.,