

## WEBTECH LAB-8

By: - Abhivir Singh(22CS2017)

Q1. Develop a currency converter application that allows users to input an amount in one currency and convert it to another. For the sake of this challenge, you can use a hard-coded exchange rate. Take advantage of React state and event handlers to manage the input and conversion calculations.

The image displays two screenshots of a web browser and code editor, illustrating the development of a Currency Converter application.

**Top Screenshot:** The code editor shows the initial setup of the application. The code defines a `CurrencyConverter` component using `useState` to manage the `amount`, `fromCurrency`, and `toCurrency` states. A hard-coded `exchangeRate` of 0.85 is used. The `handleAmountChange`, `handleFromCurrencyChange`, and `handleToCurrencyChange` functions are defined to update the respective states. The `convertCurrency` function calculates the converted amount using the exchange rate. The component returns a simple HTML structure with a heading and a label for the from currency.

The browser view shows the rendered application, titled "Currency Converter". It displays the "From: USD" dropdown menu and the "To: EUR" dropdown menu. The converted amount is shown as 38.25.

**Bottom Screenshot:** The code editor shows the completed application. The `return` statement is updated to include the full HTML structure, including the `fromCurrency` and `toCurrency` dropdown menus, the `amount` input field, and the `convertCurrency` function call. The `export default CurrencyConverter;` statement is added at the bottom.

The browser view shows the rendered application, titled "Currency Converter". It displays the "From: USD" dropdown menu and the "To: EUR" dropdown menu. The converted amount is shown as 38.25.

Q2. Create a stopwatch application through which users can start, pause and reset the timer. Use React state, event handlers and the `setTimeout` or `setInterval` functions to manage the timer's state and actions.

```
code > src > App.jsx > ...
1  import React, { useState, useEffect } from 'react';
2
3  const Stopwatch = () => {
4    const [time, setTime] = useState(0);
5    const [isRunning, setIsRunning] = useState(false);
6
7    useEffect(() => {
8      let timer;
9      if (isRunning) {
10        timer = setInterval(() => {
11          setTime((prevTime) => prevTime + 1);
12        }, 1000);
13      } else {
14        clearInterval(timer);
15      }
16      return () => clearInterval(timer);
17    }, [isRunning]);
18
19    const startTimer = () => {
20      setIsRunning(true);
21    };
22
23    const pauseTimer = () => {
24      setIsRunning(false);
25    };
26
27    const resetTimer = () => {
28      setTime(0);
29      setIsRunning(false);
30    };
31  };
32
33  export default Stopwatch;
```

Stopwatch

00:03

Start Reset

```
code > src > App.jsx > ...
43  setIsRunning(false);
44  };
45
46  const formatTime = (timeInSeconds) => {
47    const minutes = Math.floor(timeInSeconds / 60);
48    const seconds = timeInSeconds % 60;
49    return `${minutes < 10 ? '0' : ''}${minutes}:${seconds < 10 ? '0' : ''}${seconds}`;
50  };
51
52  return (
53    <div>
54      <h2>Stopwatch</h2>
55      <div>
56        <span>{formatTime(time)}</span>
57      </div>
58      <div>
59        {isRunning ? (
60          <button onClick={startTimer}>Start</button>
61        ) : (
62          <button onClick={pauseTimer}>Pause</button>
63        )}
64        <button onClick={resetTimer}>Reset</button>
65      </div>
66    </div>
67  );
68
69  export default Stopwatch;
```

Stopwatch

00:03

Start Reset