LAB ASSIGNMENT – 8

Name – **Vivek Yadav**

Branch – **CSE**

Roll No. – **22CS3067**

CODE-1

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

function CurrencyConverter() {
  const [amount, setAmount] = useState(0);

  const [fromCurrency, setFromCurrency] = useState('USD');

  const [toCurrency, setToCurrency] = useState('EUR');

  const [convertedAmount, setConvertedAmount] = useState(0);

  // Hard-coded exchange rate

  const exchangeRate = 0.85; // 1 USD = 0.85 EUR

  const handleAmountChange = (e) => {

    const value = parseFloat(e.target.value);
```

```jsx
    setAmount(value);
  };

  const handleFromCurrencyChange = (e) => {
    setFromCurrency(e.target.value);
  };

  const handleToCurrencyChange = (e) => {
    setToCurrency(e.target.value);
  };

  const convertCurrency = () => {
    const result = amount * exchangeRate;
    setConvertedAmount(result.toFixed(2)); // Round to 2 decimal places
  };

  return (
    <div>
      <h2>Currency Converter</h2>
      <div>
        <label>
          Amount:
```

```jsx
        <input type="number" value={amount}
onChange={handleAmountChange} />
      </label>
    </div>
    <div>
      <label>
        From Currency:
        <select value={fromCurrency}
onChange={handleFromCurrencyChange}>
          <option value="USD">USD</option>
          <option value="EUR">EUR</option>
        </select>
      </label>
    </div>
    <div>
      <label>
        To Currency:
        <select value={toCurrency}
onChange={handleToCurrencyChange}>
          <option value="USD">USD</option>
          <option value="EUR">EUR</option>
        </select>
      </label>
    </div>
    <button onClick={convertCurrency}>Convert</button>
```

```
    <div>

      {amount} {fromCurrency} is equal to {convertedAmount}
{toCurrency}

    </div>

  </div>

 );

}


export default CurrencyConverter;
```
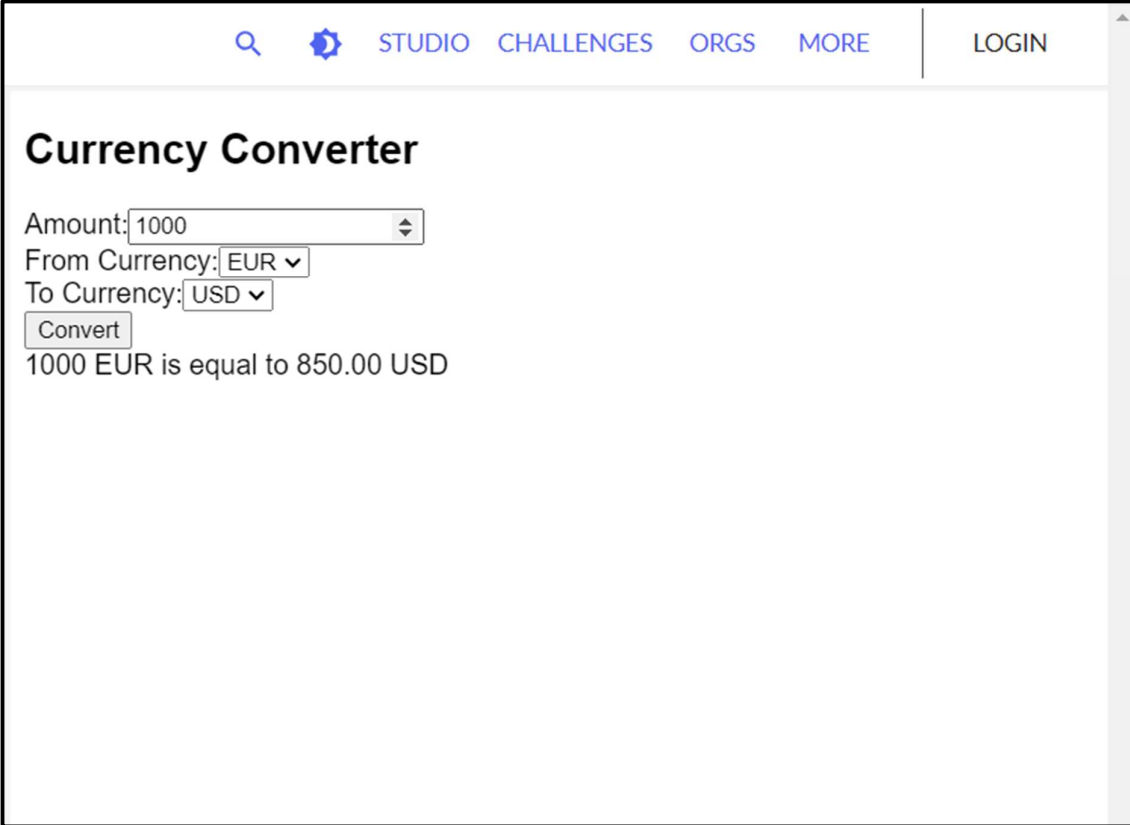
OUTPUT

CODE-2

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

function Stopwatch() {
  const [timer, setTimer] = useState(0);
  const [isRunning, setIsRunning] = useState(false);

  useEffect(() => {
    let intervalId;

    if (isRunning) {
      intervalId = setInterval(() => {
        setTimer((prevTimer) => prevTimer + 1);
      }, 1000);
    } else {
      clearInterval(intervalId);
    }

    return () => clearInterval(intervalId);
  }, [isRunning]);
```

```javascript
const startTimer = () => {
  setIsRunning(true);
};


const pauseTimer = () => {
  setIsRunning(false);
};


const resetTimer = () => {
  setTimer(0);
  setIsRunning(false);
};


const formatTime = (time) => {
  const hours = Math.floor(time / 3600);
  const minutes = Math.floor((time % 3600) / 60);
  const seconds = time % 60;

  return `${hours.toString().padStart(2, '0')}:${minutes
    .toString()
    .padStart(2, '0')}:${seconds.toString().padStart(2, '0')}`;
};


return (
```
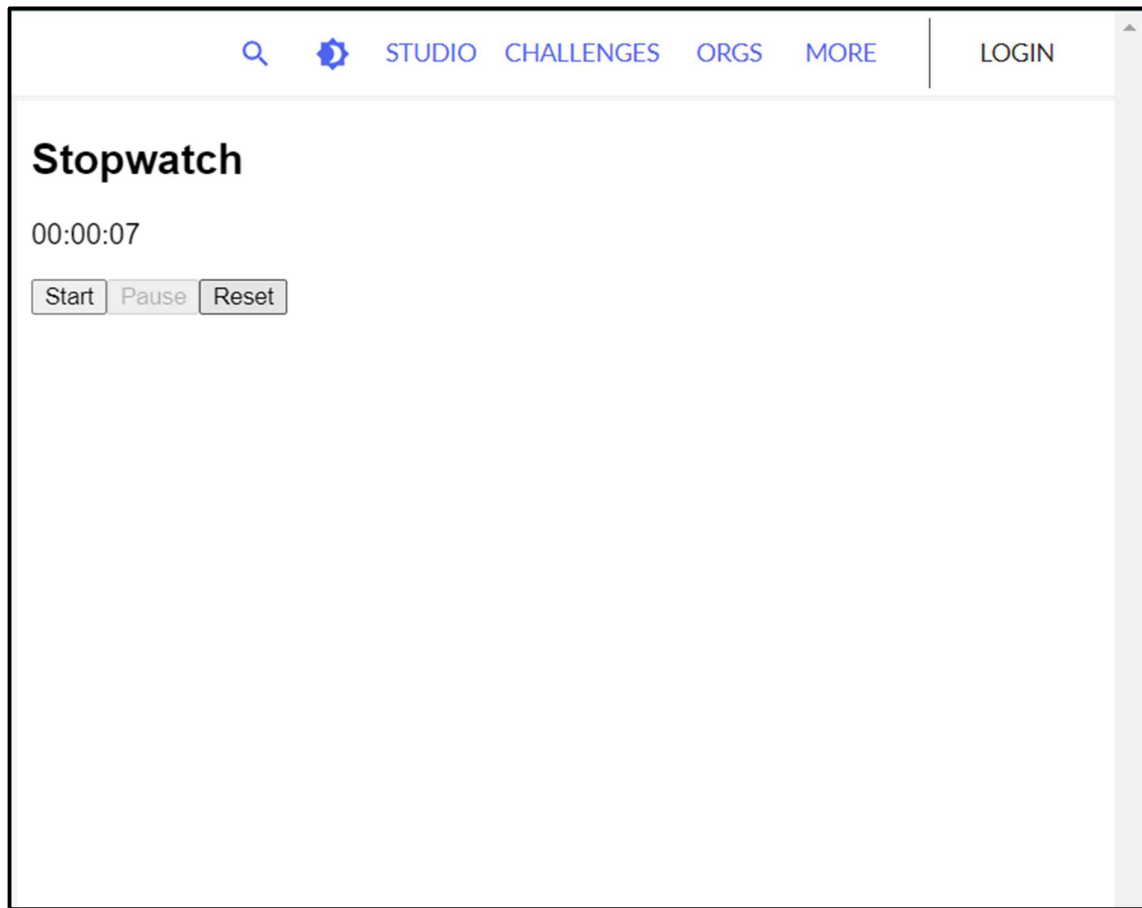
```jsx
  <div>
    <h2>Stopwatch</h2>
    <div>
      <p>{formatTime(timer)}</p>
    </div>
    <div>
      <button onClick={startTimer} disabled={isRunning}>
        Start
      </button>
      <button onClick={pauseTimer} disabled={!isRunning}>
        Pause
      </button>
      <button onClick={resetTimer}>Reset</button>
    </div>
  </div>
  );
}

export default Stopwatch;
```

OUTPUT

Stopwatch

00:00:07

Start | Pause | Reset

CODE-3

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

const MessagingApp = () => {
  const [conversations, setConversations] = useState([
    { id: 1, name: 'Family', messages: [] },
    { id: 2, name: 'Friends', messages: [] },
  ]);
  const [selectedConversation, setSelectedConversation] =
useState(null);
  const [newMessage, setNewMessage] = useState('');
```

```javascript
const handleConversationClick = (conversationId) => {

  setSelectedConversation(conversationId);

};


const handleSendMessage = () => {

  if (newMessage.trim() !== '' && selectedConversation) {

    const updatedConversations = conversations.map((conversation) =>

      conversation.id === selectedConversation

        ? {

            ...conversation,

            messages: [

              { text: newMessage, timestamp: new Date().toLocaleTimeString() },

              ...conversation.messages,

            ],

          }

        : conversation

    );

    setConversations(updatedConversations);

    setNewMessage('');

  }

};
```

```jsx
  return (
    <div>
      <h1>Messaging App</h1>


      <div style={{ display: 'flex' }}>
        {/* List of Conversations */}
        <div style={{ flex: '1', borderRight: '1px solid #ccc', padding:
'10px' }}>
          <h2>Conversations</h2>
          <ul>
            {conversations.map((conversation) => (
              <li key={conversation.id} onClick={() =>
handleConversationClick(conversation.id)}>
                {conversation.name}
              </li>
            ))}
          </ul>
        </div>


        {/* Chat Interface */}
        <div style={{ flex: '3', padding: '10px' }}>
          {selectedConversation && (
            <div>
```

```jsx
      <h2>Chat with {conversations.find((conv) => conv.id ===
selectedConversation).name}</h2>
      <div style={{ maxHeight: '300px', overflowY: 'auto', border:
'1px solid #ccc', padding: '10px' }}>
        {conversations
          .find((conv) => conv.id === selectedConversation)
          .messages.map((message, index) => (
            <div key={index} style={{ borderBottom: '1px solid #eee',
paddingBottom: '5px' }}>
              <strong>{message.timestamp}</strong>: {message.text}
            </div>
          ))}
      </div>
      <textarea
        rows="3"
        value={newMessage}
        onChange={(e) => setNewMessage(e.target.value)}
      />
      <button onClick={handleSendMessage}>Send</button>
    </div>
    )}
   </div>
  </div>
 );
```

```
};
```

export default MessagingApp;

OUTPUT