LAB ASSIGNMENT – 8

Name – SOURABH

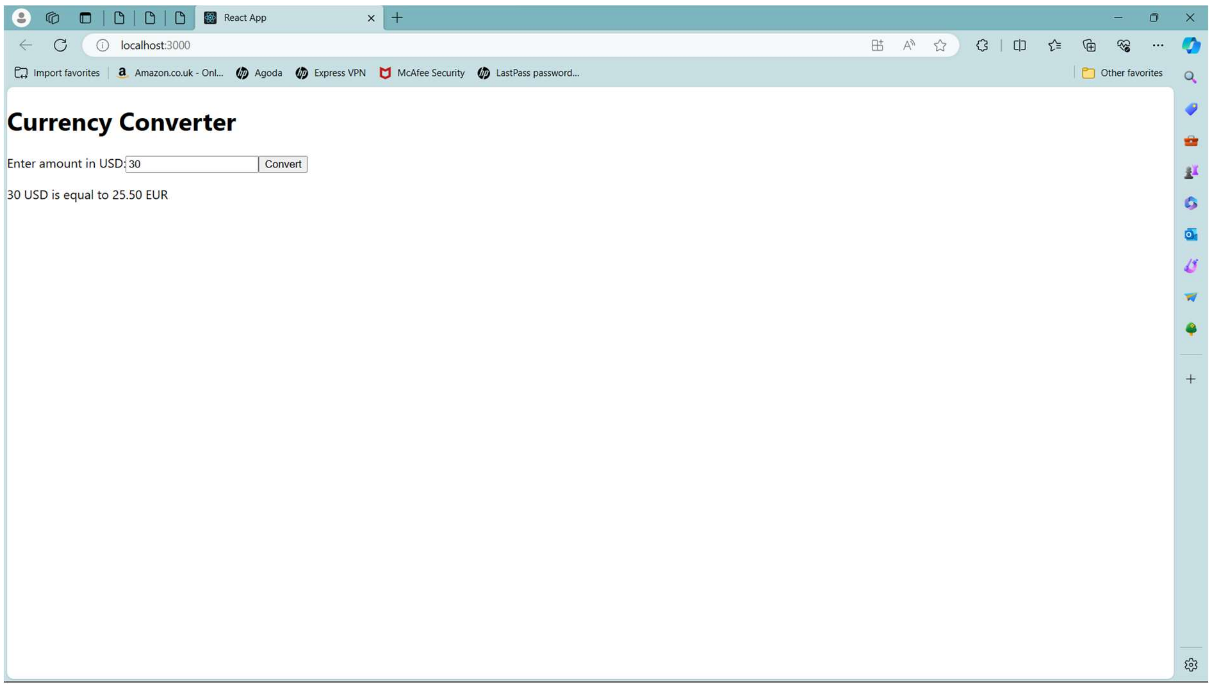
Branch – CSE

Roll No. – 22CS3058

O1.) 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.

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

const CurrencyConverter = () => {
  const [amount, setAmount] = useState(0);
  const [convertedAmount, setConvertedAmount] = useState(0);
  const exchangeRate = 0.85; // Replace this with the actual
exchange rate

  const handleAmountChange = (e) => {
    const inputAmount = parseFloat(e.target.value);
    setAmount(isNaN(inputAmount) ? 0 : inputAmount);
  };

  const convertCurrency = () => {
    const result = amount * exchangeRate;
    setConvertedAmount(result.toFixed(2));
  };

  return (
    <div>
      <h1>Currency Converter</h1>
      <label>
        Enter amount in USD:
        <input type="number" value={amount}
onChange={handleAmountChange} />
      </label>
      <button onClick={convertCurrency}>Convert</button>
      {convertedAmount > 0 && (
        <p>
          {amount} USD is equal to {convertedAmount} EUR
        </p>
      )}
    </div>
  );
};

export default CurrencyConverter;
```

OUTPUT



Currency Converter

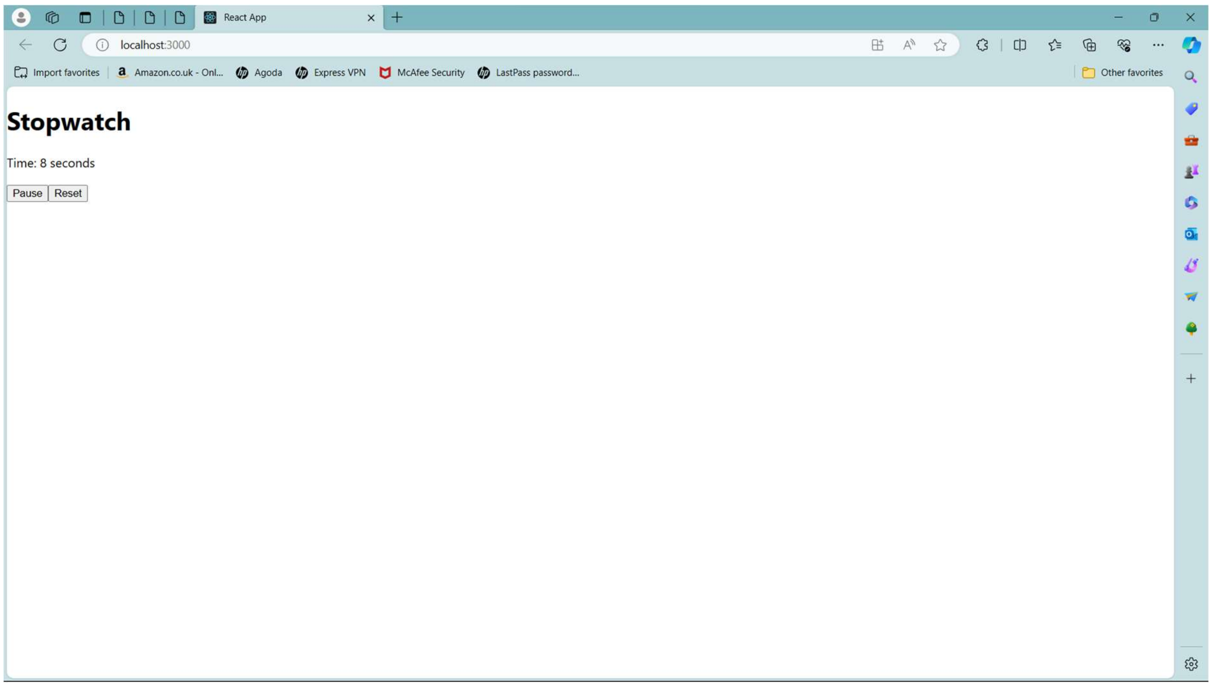
Enter amount in USD: 30 Convert

30 USD is equal to 25.50 EUR

T2. Create a stopwatch application through which users can start, pause and reset the timer. Use React state, event handlers and the set Timeout or

setInterval functions to manage the timer's state and actions.

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

const Stopwatch = () => {
  const [seconds, setSeconds] = useState(0);
  const [isRunning, setIsRunning] = useState(false);

  useEffect(() => {
    let interval;

    if (isRunning) {
      interval = setInterval(() => {
        setSeconds((prevSeconds) => prevSeconds + 1);
      }, 1000);
    }

    return () => {
      clearInterval(interval);
    };
  }, [isRunning]);

  const handleStartPause = () => {
    setIsRunning((prevIsRunning) => !prevIsRunning);
  };

  const handleReset = () => {
    setSeconds(0);
    setIsRunning(false);
  };

  return (
    <div>
      <h1>Stopwatch</h1>
      <p>Time: {seconds} seconds</p>
      <button onClick={handleStartPause}>
        {isRunning ? 'Pause' : 'Start'}
      </button>
      <button onClick={handleReset}>Reset</button>
    </div>
  );
};

export default Stopwatch;
```



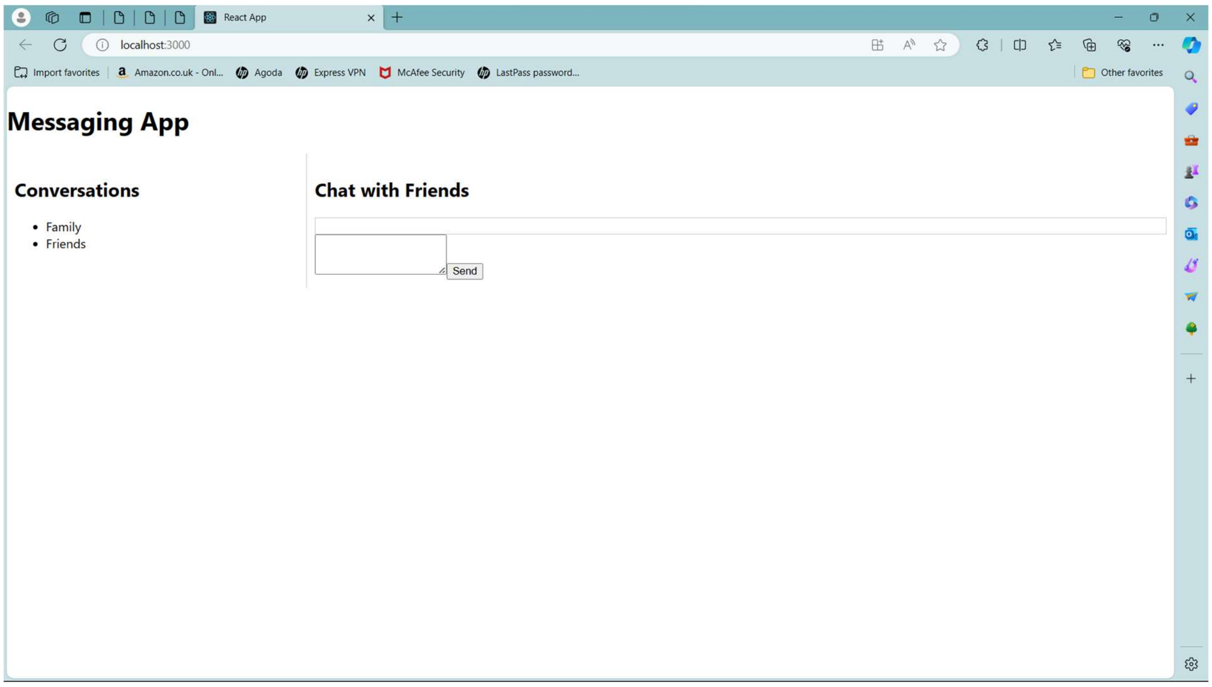
Stopwatch

Time: 8 seconds

T3. Develop a messaging application that allows users to send and receive messages in real time. The application should display a list of conversations and allow the user to select a specific conversation to view its messages. The messages should be displayed in a chat interface with the most recent message at the top. Users should be able to send new messages and receive push notifications.

```
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('');

  const handleConversationClick = (conversationId) => {
    setSelectedConversation(conversationId);
  };

  const handleSendMessage = () => {
    if (newMessage.trim() !== '' && selectedConversation) {
      setConversations((prevConversations) =>
        prevConversations.map((conversation) =>
          conversation.id === selectedConversation
            ? {
                ...conversation,
                messages: [
                  { text: newMessage, timestamp: new
Date().toLocaleTimeString() },
                  ...conversation.messages,
                ],
              }
            : conversation
        )
      );

      setNewMessage('');
    }
  };

  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>
              <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>
    </div>
  );
};

export default MessagingApp;
```



Messaging App

Conversations

- Family
- Friends

Chat with Friends