Name : SOURABH
Roll no.: 22CS3058
Lab-9

T1. 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.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Currency Converter</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }

        #app {
            max-width: 400px;
            margin: 50px auto;
            padding: 20px;
            background-color: #fff;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }
        h1 {
            text-align: center;
        }

        label {
            display: block;
            margin-bottom: 5px;
        }

        input,
        select {
            width: 100%;
            padding: 8px;
            margin-bottom: 15px;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }

        button {
            width: 100%;
            padding: 10px;
            background-color: #4CAF50;
            color: #fff;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }

        button:hover {
            background-color: #45a049;
        }

        .result {
            margin-top: 15px;
        }
    </style>
</head>

<body>
```

```
    <div id="app">
        <h1>Currency Converter</h1>
        <div>
            <label for="amount">Amount:</label>
            <input type="number" id="amount" v-model="amount"
placeholder="Enter amount">
        </div>
        <div>
            <label for="from">From:</label>
            <select id="from" v-model="fromCurrency">
                <option v-for="(rate, currency) in exchangeRates"
:value="currency">{{ currency }}</option>
            </select>
        </div>
        <div>
            <label for="to">To:</label>
            <select id="to" v-model="toCurrency">
                <option v-for="(rate, currency) in exchangeRates"
:value="currency">{{ currency }}</option>
            </select>
        </div>
        <div>
            <button @click="convert">Convert</button>
        </div>
        <div class="result" v-if="convertedAmount !== null">
            <p>{{ amount }} {{ fromCurrency }} is {{ convertedAmount }} {{
toCurrency }}</p>
        </div>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2"></script>
    <script>
        new Vue({
            el: '#app',
            data: {
                amount: null,
                fromCurrency: 'INR',
                toCurrency: 'USD',
                exchangeRates: {
                    USD: 1,
                    EUR: 0.83,
                    GBP: 0.72,
                    JPY: 109.36,
                    AUD: 1.30,
                    CAD: 1.26,
                    INR: 82.78
                },
                convertedAmount: null
            },
            methods: {
                convert() {
                    const fromRate =
this.exchangeRates[this.fromCurrency];
                    const toRate = this.exchangeRates[this.toCurrency];
                    this.convertedAmount = (this.amount * toRate /
fromRate).toFixed(2);
                }
            }
        });
    </script>
</body>

</html>
```

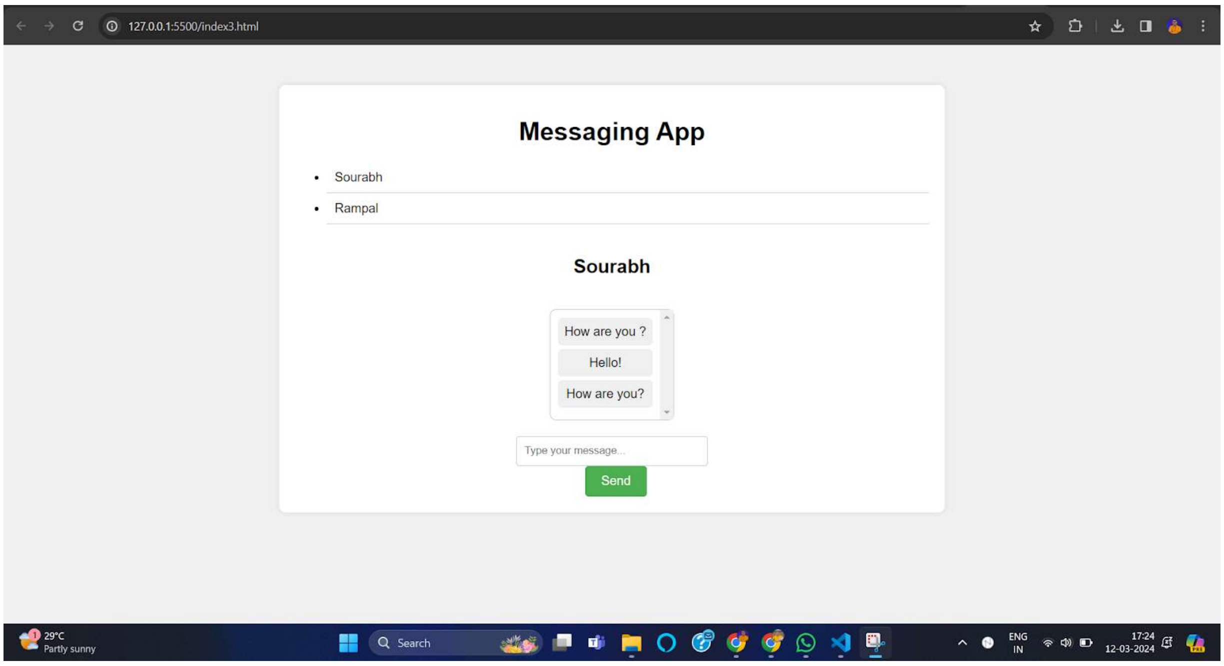
T2. Create a stopwatch application through which users can start, pause and reset the timer. Use React state, event handlers and the setTimeout or setInterval functions to manage the timer’s state and actions.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Stopwatch</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }

        #app {
            max-width: 400px;
            margin: 50px auto;
            padding: 20px;
            background-color: #fff;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            text-align: center;
        }

        h1 {
            margin-bottom: 20px;
        }

        .timer {
            font-size: 2em;
            margin-bottom: 20px;
        }

        button {
            padding: 10px 20px;
            margin: 0 10px;
            font-size: 1em;
            cursor: pointer;
            background-color: #4CAF50;
            color: white;
            border: none;
            border-radius: 4px;
        }

        button:hover {
            background-color: #45a049;
        }
    </style>
```

```
</head>

<body>
    <div id="app">
        <h1>Stopwatch</h1>
        <div class="timer">{{ formatTime }}</div>
        <div>
            <button @click="startStop">{{ isRunning ? 'Pause' : 'Start'
}}</button>
            <button @click="reset" :disabled="!isRunning && seconds ===
0">Reset</button>
        </div>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2"></script>
    <script>
        new Vue({
            el: '#app',
            data: {
                seconds: 0,
                isRunning: false
            },
            computed: {
                formatTime() {
                    const minutes = Math.floor(this.seconds / 60);
                    const seconds = this.seconds % 60;
                    return `${minutes.toString().padStart(2,
'0')}:${seconds.toString().padStart(2, '0')}`;
                }
            },
            methods: {
                startStop() {
                    if (this.isRunning) {
                        clearInterval(this.intervalId);
                    } else {
                        this.intervalId = setInterval(() => {
                            this.seconds++;
                        }, 1000);
                    }
                    this.isRunning = !this.isRunning;
                },
                reset() {
                    clearInterval(this.intervalId);
                    this.seconds = 0;
                    this.isRunning = false;
                }
            }
        });
    </script>
</body>

</html>
```

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.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Messaging App</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }
        #app {
            max-width: 800px;
            margin: 50px auto;
            padding: 20px;
            background-color: #fff;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            text-align: center;
        }

        h1 {
            margin-bottom: 20px;
        }

        .conversation-list {
            list-style: none;
            padding: 0;
            text-align: left;
            margin-bottom: 20px;
        }

        .conversation-list li {
            padding: 10px;
            border-bottom: 1px solid #ccc;
            cursor: pointer;
        }

        .conversation-list li:hover {
            background-color: #f0f0f0;
        }

        .chat-container {
            display: flex;
            flex-direction: column;
            align-items: center;
        }

        .message-container {
            max-width: 600px;
            margin-top: 20px;
            padding: 10px;
            border: 1px solid #ccc;
            border-radius: 8px;
            overflow-y: scroll;
            max-height: 300px;
        }

        .message {
            background-color: #f0f0f0;
            padding: 8px;
            margin-bottom: 5px;
            border-radius: 5px;
            word-wrap: break-word;
        }

        .input-container {
            margin-top: 20px;
        }

        input[type="text"] {
            width: 80%;
            padding: 10px;
            border-radius: 4px;
            border: 1px solid #ccc;
        }

        button {
            padding: 10px 20px;
            margin-left: 10px;
            font-size: 1em;
            cursor: pointer;
            background-color: #4CAF50;
            color: white;
            border: none;
            border-radius: 4px;
        }

        button:hover {
            background-color: #45a049;
        }
    </style>
</head>

<body>
    <div id="app">
        <h1>Messaging App</h1>
        <div class="conversation-list">
            <ul>
                <li v-for="(conversation, index) in conversations"
:key="index" @click="selectConversation(index)">
                    {{ conversation.name }}
                </li>
            </ul>
        </div>
        <div class="chat-container" v-if="selectedConversation !== null">
            <h2>{{ selectedConversation.name }}</h2>
            <div class="message-container">
                <div class="message" v-for="(message, index) in
selectedConversation.messages" :key="index">
                    {{ message }}
                </div>
            </div>
            <div class="input-container">
                <input type="text" v-model="newMessage" placeholder="Type
your message...">
                <button @click="sendMessage">Send</button>
            </div>
        </div>
    </div>

    <script src="https://cdn.jsdelivr.net/npm/vue@2"></script>
    <script>
        new Vue({
            el: '#app',
            data: {
                conversations: [
                    {
                        name: 'Ankur',
                        messages: ['Hello!', 'How are you?']
                    },
                    {
                        name: 'Rishabh',
                        messages: ['Hi there!', 'I\'m good, thanks.']
                    }
                    // Add more conversations as needed
                ],
                selectedConversation: null,
                newMessage: ''
            },
            methods: {
                selectConversation(index) {
                    this.selectedConversation = this.conversations[index];
                },
                sendMessage() {
                    if (this.newMessage.trim() === '') return;

this.selectedConversation.messages.unshift(this.newMessage);
                    this.newMessage = '';
                    // Code to send message to backend or update database
                }
            }
        });
    </script>
</body>

</html>
```



Messaging App

- Sourabh
- Rampal

Sourabh

How are you ?

Hello!

How are you?