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>
    /* CSS styles */
    body {
      font-family: Verdana;
    }
    label {
      margin-right: 8px;
    }
    select {
      margin-left: 5px;
    }
  </style>
</head>
<body>
  <div id="root"></div>
  <script src="https://unpkg.com/react@17/umd/react.development.js"></script>
  <script src="https://unpkg.com/react-dom@17/umd/react-
dom.development.js"></script>
  <script>
    // ReactJS code
    const { useState } = React;


    function CurrencyConverter() {
      const [amount, setAmount] = useState('');
      const [fromCurrency, setFromCurrency] = useState('USD');
      const [toCurrency, setToCurrency] = useState('EUR');
      const exchangeRate = 0.83; // Example exchange rate (1 USD = 0.85 EUR)


      const handleAmountChange = (event) => {
        setAmount(event.target.value);
      };


      const handleFromCurrencyChange = (event) => {
        setFromCurrency(event.target.value);
```

```
  };

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

  const convertCurrency = () => {
    const convertedAmount = parseFloat(amount) * exchangeRate;
    return isNaN(convertedAmount) ? '' : convertedAmount.toFixed(2);
  };

  return React.createElement(
    'div',
    null,
    React.createElement('h1', null, 'Currency Converter'),
    React.createElement(
      'div',
      null,
      React.createElement(
        'label',
        null,
        'Amount:',
        React.createElement('input', { type: 'number', value: amount,
onChange: handleAmountChange })
      )
    ),
    React.createElement(
      'div',
      null,
      React.createElement(
        'label',
        null,
        'From:',
        React.createElement(
          'select',
          { value: fromCurrency, onChange: handleFromCurrencyChange },
          React.createElement('option', { value: 'USD' }, 'USD'),
          React.createElement('option', { value: 'EUR' }, 'EUR')
        )
      )
    ),
    React.createElement(
      'div',
      null,
      React.createElement(
        'label',
```

```
            null,
            'To:',
            React.createElement(
              'select',
              { value: toCurrency, onChange: handleToCurrencyChange },
              React.createElement('option', { value: 'USD' }, 'USD'),
              React.createElement('option', { value: 'EUR' }, 'EUR')
            )
          )
        ),
        React.createElement(
          'div',
          null,
          React.createElement('p', null, 'Converted Amount: ',
convertCurrency())
        )
      );
    }


    ReactDOM.render(React.createElement(CurrencyConverter),
document.getElementById('root'));
  </script>
</body>
</html>
```

Currency Converter

Amount: 5

From: USD

To: EUR

Converted Amount: 4.15

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>
    /* CSS styles */
    body {
      font-family: Verdana, Tahoma, sans-serif;
    }
    button {
      margin-right: 8px;
    }
  </style>
</head>
<body>
  <div id="root"></div>
  <script src="https://unpkg.com/react@17/umd/react.development.js"></script>
  <script src="https://unpkg.com/react-dom@17/umd/react-
dom.development.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/babel-
standalone/6.26.0/babel.min.js"></script>
  <script type="text/babel">
    // ReactJS code
    const { useState, useEffect } = React;


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


      useEffect(() => {
        let interval;
        if (isActive) {
          interval = setInterval(() => {
            setTimer((prevTimer) => prevTimer + 1);
          }, 1000);
        } else {
          clearInterval(interval);
        }
        return () => clearInterval(interval);
      }, [isActive]);
```

```
    const handleStart = () => {
      setIsActive(true);
    };


    const handlePause = () => {
      setIsActive(false);
    };


    const handleReset = () => {
      setIsActive(false);
      setTimer(0);
    };


    return (
      <div>
        <h1>Stopwatch</h1>
        <div>
          <p>{timer}s</p>
        </div>
        <div>
          {!isActive ? (
            <button onClick={handleStart}>Start</button>
          ) : (
            <button onClick={handlePause}>Pause</button>
          )}
          <button onClick={handleReset}>Reset</button>
        </div>
      </div>
    );
  }


  ReactDOM.render(<Stopwatch />, document.getElementById('root'));
</script>
</body>
</html>
```

Stopwatch

0s

Start Reset

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>
  <head>
    <meta charset="UTF-8" />
    <title>Simple Messaging App</title>
    <style>
      body {
        font-family: Arial, sans-serif;
      }
      #conversations {
        list-style-type: none;
        padding: 0;
        margin: 0;
      }
      #conversations li {
        padding: 10px;
        border-bottom: 1px solid #ddd;
        cursor: pointer;
      }
      #conversations li:hover {
        background-color: #f5f5f5;
      }
      #messages {
        list-style-type: none;
        padding: 0;
        margin: 0;
        height: 300px;
        overflow-y: auto;
      }
      #messages li {
        padding: 10px;
        border-bottom: 1px solid #ddd;
      }
      #message-input {
        width: 100%;
        padding: 10px;
      }
      #send-button {
        padding: 10px;
        background-color: #4CAF50;
        color: white;
```

```
        border: none;
        cursor: pointer;
      }
      #send-button:hover {
        background-color: #45a049;
      }
    </style>
  </head>
  <body>
    <h1>Simple Messaging App</h1>
    <ul id="conversations"></ul>
    <ul id="messages"></ul>
    <input id="message-input" type="text" placeholder="Type a message..." />
    <button id="send-button">Send</button>
    <script
src="https://unpkg.com/react@16.13.1/umd/react.development.js"></script>
    <script src="https://unpkg.com/react-dom@16.13.1/umd/react-
dom.development.js"></script>
    <script src="https://unpkg.com/babel-
standalone@6.26.0/babel.min.js"></script>
    <script src="https://unpkg.com/push.js@1.0.12/push.min.js"></script>
    <script type="text/babel">
      class Conversation extends React.Component {
        constructor(props) {
          super(props);
          this.state = {
            messages: [],
          };
          this.sendMessage = this.sendMessage.bind(this);
        }
        componentDidMount() {
          this.getMessages();
          this.interval = setInterval(() => this.getMessages(), 1000);
        }
        componentWillUnmount() {
          clearInterval(this.interval);
        }
        getMessages() {
          fetch(`/api/conversations/${this.props.conversation.id}/messages`)
           .then((response) => response.json())
           .then((data) => {
              this.setState({ messages: data });
           });
        }
        sendMessage() {
          const message = {
            text: document.getElementById("message-input").value,
          };
```

```
      fetch(`/api/conversations/${this.props.conversation.id}/messages`, {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
        },
        body: JSON.stringify(message),
      })
       .then((response) => response.json())
       .then((data) => {
          this.getMessages();
          document.getElementById("message-input").value = "";
          Push.create("New message", {
            body: data.text,
            timeout: 4000,
          });
        });
    }
    render() {
      return (
        <div>
          <h2>{this.props.conversation.name}</h2>
          <ul id="messages">
            {this.state.messages.map((message) => (
             <li key={message.id}>{message.text}</li>
            ))}
          </ul>
          <input
            id="message-input"
            type="text"
            placeholder="Type a message..."
            onKeyPress={(event) => {
              if (event.key === "Enter") {
                this.sendMessage();
              }
            }}
          />
          <button id="send-button" onClick={this.sendMessage}>
            Send
          </button>
        </div>
      );
    }
  }
  class Conversations extends React.Component {
    constructor(props) {
      super(props);
      this.state = {
        conversations: [],
```

```
        };
      }
      componentDidMount() {
        fetch("/api/conversations")
         .then((response) => response.json())
         .then((data) => {
           this.setState({ conversations: data });
         });
      }
      render() {
        return (
          <ul id="conversations">
            {this.state.conversations.map((conversation) => (
              <li key={conversation.id} onClick={() =>
this.props.onSelect(conversation)}>
                {conversation.name}
              </li>
            ))}
          </ul>
        );
      }
    }
    class App extends React.Component {
      constructor(props) {
        super(props);
        this.state = {
          selectedConversation: null,
        };
        this.onSelectConversation = this.onSelectConversation.bind(this);
      }
      onSelectConversation(conversation) {
        this.setState({ selectedConversation: conversation });
      }
      render() {
        return (
          <div>
            <Conversations onSelect={this.onSelectConversation} />
            {this.state.selectedConversation && (
              <Conversation conversation={this.state.selectedConversation}
/>

            )}
          </div>
        );
      }
    }
    ReactDOM.render(<App />, document.getElementById("root"));
  </script>
</body>
```

```
</html>
```