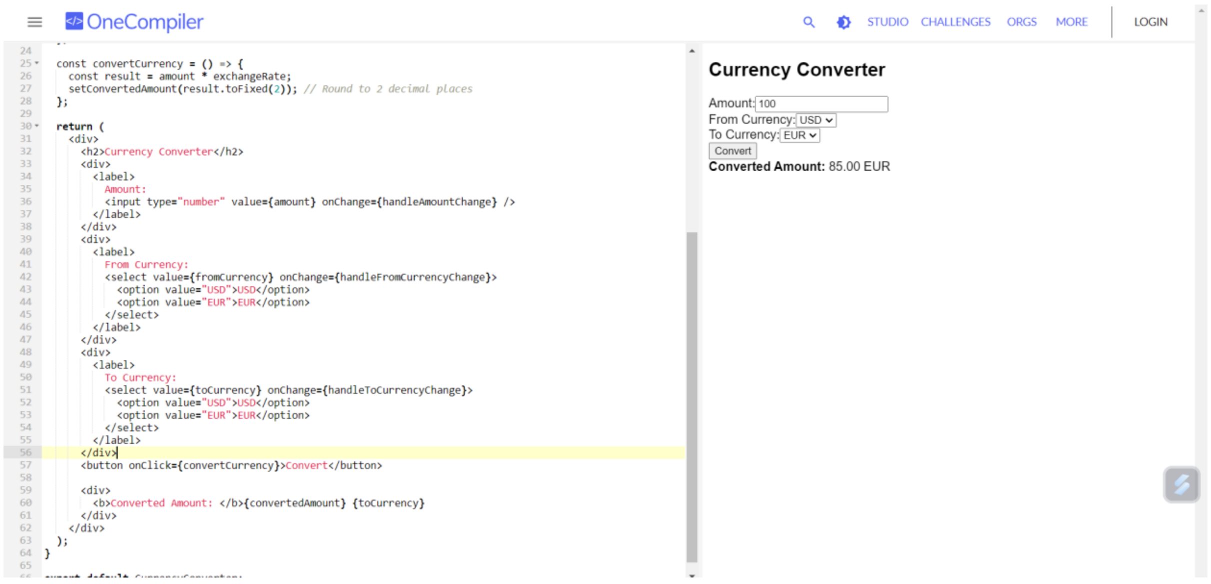
LAB – 08

T1.

```
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);
    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:
        <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>
          <b>Converted Amount: </b>{convertedAmount} {toCurrency}
        </div>
      </div>
  );
}

export default CurrencyConverter;
```



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

  return (
    <div>
      <h2>Currency Converter</h2>
      <div>
        <label>
          Amount:
          <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>
        <b>Converted Amount: </b>{convertedAmount} {toCurrency}
      </div>
    </div>
  );
}
```

Currency Converter

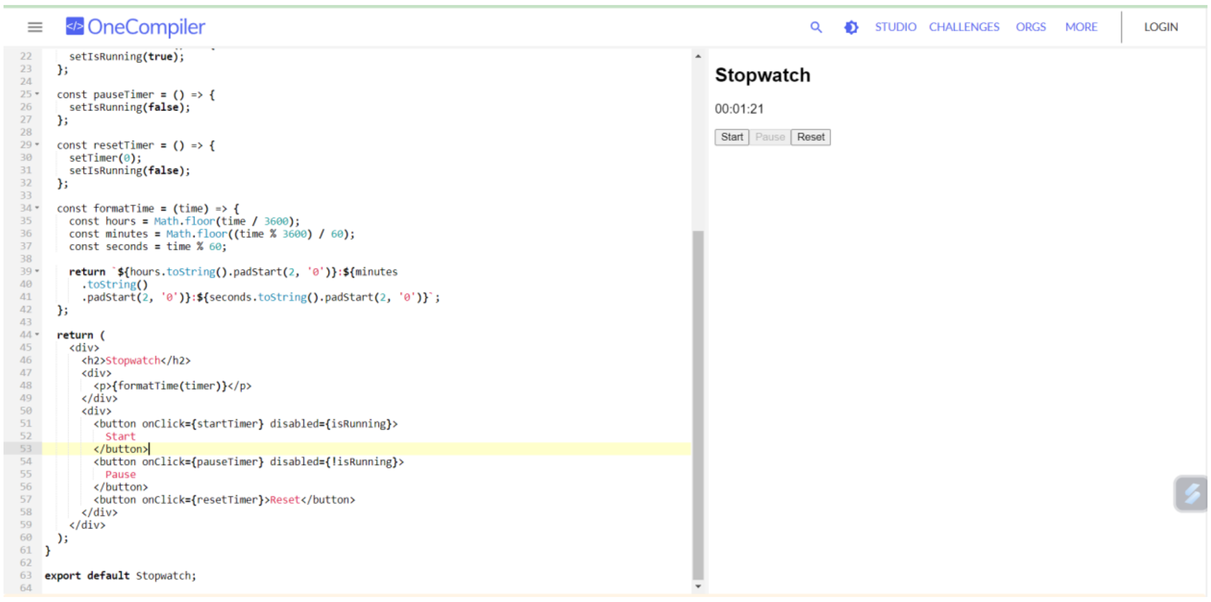
Amount: 100
From Currency: USD
To Currency: EUR
Convert
Converted Amount: 85.00 EUR

T2.

```
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]);

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 (
    <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;
```



```
    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 (
    <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;
```

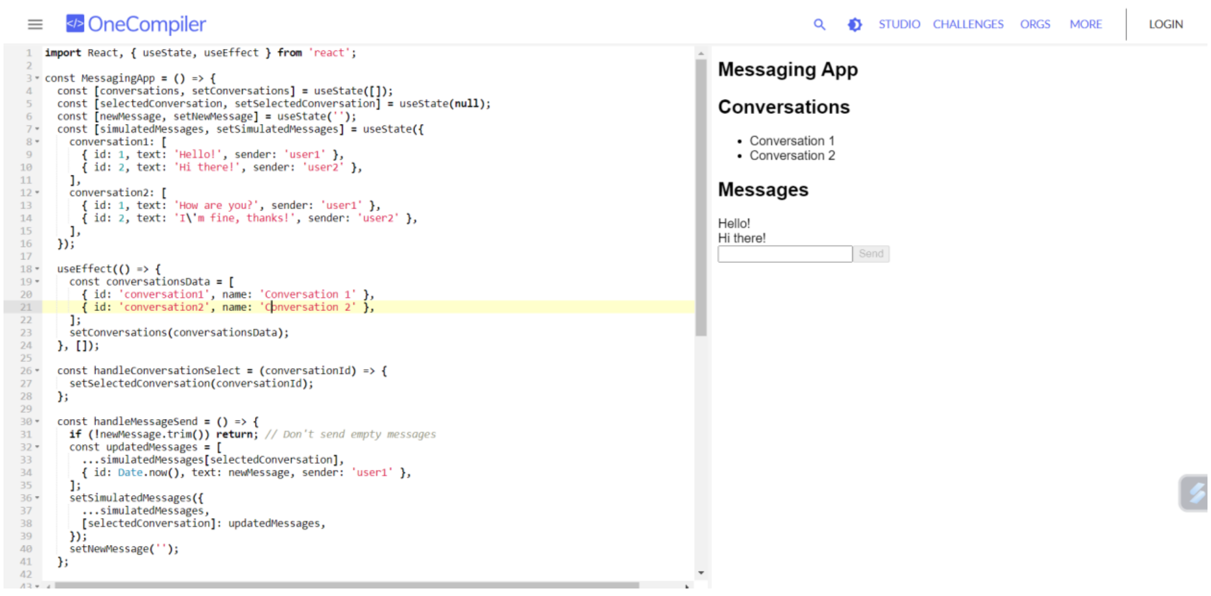
```
Stopwatch

00:01:21
```

T3.

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


const MessagingApp = () => {
  const [conversations, setConversations] = useState([]);
  const [selectedConversation, setSelectedConversation] = useState(null);
  const [newMessage, setNewMessage] = useState('');
  const [simulatedMessages, setSimulatedMessages] = useState({
    conversation1: [
      { id: 1, text: 'Hello!', sender: 'user1' },
      { id: 2, text: 'Hi there!', sender: 'user2' },
    ],
    conversation2: [
      { id: 1, text: 'How are you?', sender: 'user1' },
      { id: 2, text: 'I\'m fine, thanks!', sender: 'user2' },
    ],
  });
```

```
useEffect(() => {
  const conversationsData = [
    { id: 'conversation1', name: 'Conversation 1' },
    { id: 'conversation2', name: 'Conversation 2' },
  ];
  setConversations(conversationsData);
}, []);

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

const handleMessageSend = () => {
  if (!newMessage.trim()) return; // Don't send empty messages
  const updatedMessages = [
    ...simulatedMessages[selectedConversation],
    { id: Date.now(), text: newMessage, sender: 'user1' },
  ];
  setSimulatedMessages({
    ...simulatedMessages,
    [selectedConversation]: updatedMessages,
  });
  setNewMessage('');
};

return (
  <div>
    <h1>Messaging App</h1>
    <div className="conversations">
      <h2>Conversations</h2>
      <ul>
```

```
          {conversations.map((conversation) => (
            <li key={conversation.id} onClick={() => handleConversationSelect(conversation.id)}>
              {conversation.name}
            </li>
          ))}
        </ul>
      </div>
      <div className="messages">
        <h2>Messages</h2>
        {selectedConversation && (
          <div>
            {simulatedMessages[selectedConversation].map((message) => (
              <div key={message.id} className={message.sender === 'user1' ? 'message sent' : 'message
received'}>
                {message.text}
              </div>
            ))}
          </div>
        )}
        <div className="message-input">
          <input type="text" value={newMessage} onChange={(e) => setNewMessage(e.target.value)} />
          <button onClick={handleMessageSend} disabled={!newMessage.trim()}>Send</button>
        </div>
      </div>
    </div>
  );
};

export default MessagingApp;
```



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

const MessagingApp = () => {
  const [conversations, setConversations] = useState([]);
  const [selectedConversation, setSelectedConversation] = useState(null);
  const [newMessage, setNewMessage] = useState('');
  const [simulatedMessages, setSimulatedMessages] = useState({
    conversation1: [
      { id: 1, text: 'Hello!', sender: 'user1' },
      { id: 2, text: 'Hi there!', sender: 'user2' },
    ],
    conversation2: [
      { id: 1, text: 'How are you?', sender: 'user1' },
      { id: 2, text: 'I\'m fine, thanks!', sender: 'user2' },
    ],
  });

  useEffect(() => {
    const conversationsData = [
      { id: 'conversation1', name: 'Conversation 1' },
      { id: 'conversation2', name: 'Conversation 2' },
    ];
    setConversations(conversationsData);
  }, []);

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

  const handleMessageSend = () => {
    if (!newMessage.trim()) return; // Don't send empty messages
    const updatedMessages = [
      ...simulatedMessages[selectedConversation],
      { id: Date.now(), text: newMessage, sender: 'user1' },
    ];
    setSimulatedMessages({
      ...simulatedMessages,
      [selectedConversation]: updatedMessages,
    });
    setNewMessage('');
  };

```

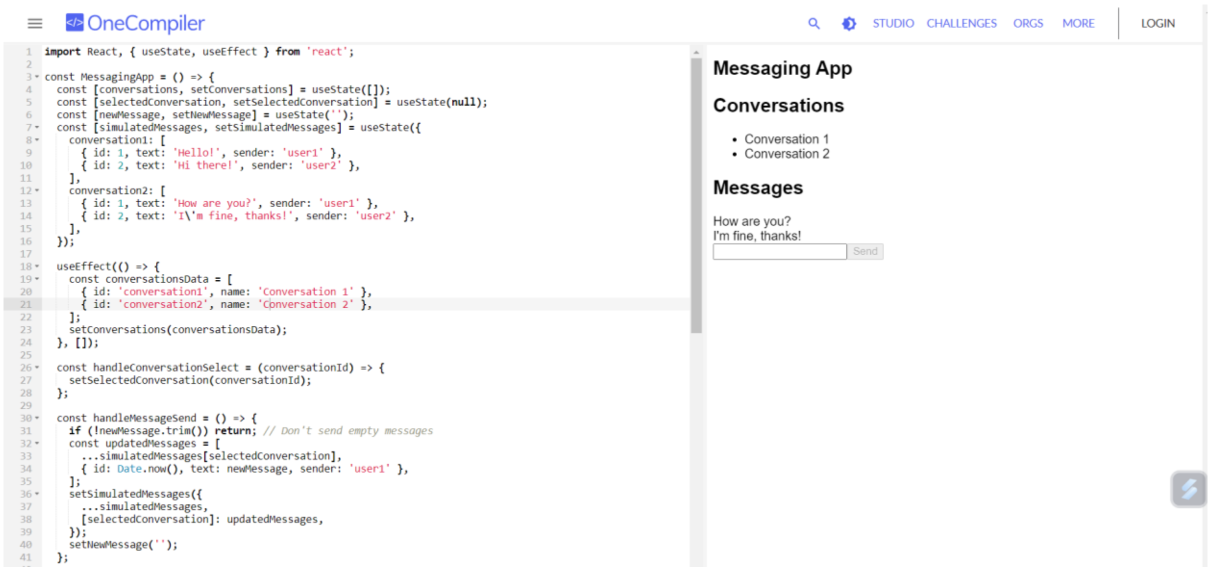
Messaging App

Conversations

- Conversation 1
- Conversation 2

Messages

Hello!
Hi there!



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

const MessagingApp = () => {
  const [conversations, setConversations] = useState([]);
  const [selectedConversation, setSelectedConversation] = useState(null);
  const [newMessage, setNewMessage] = useState('');
  const [simulatedMessages, setSimulatedMessages] = useState({
    conversation1: [
      { id: 1, text: 'Hello!', sender: 'user1' },
      { id: 2, text: 'Hi there!', sender: 'user2' },
    ],
    conversation2: [
      { id: 1, text: 'How are you?', sender: 'user1' },
      { id: 2, text: 'I\'m fine, thanks!', sender: 'user2' },
    ],
  });

  useEffect(() => {
    const conversationsData = [
      { id: 'conversation1', name: 'Conversation 1' },
      { id: 'conversation2', name: 'Conversation 2' },
    ];
    setConversations(conversationsData);
  }, []);

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

  const handleMessageSend = () => {
    if (!newMessage.trim()) return; // Don't send empty messages
    const updatedMessages = [
      ...simulatedMessages[selectedConversation],
      { id: Date.now(), text: newMessage, sender: 'user1' },
    ];
    setSimulatedMessages({
      ...simulatedMessages,
      [selectedConversation]: updatedMessages,
    });
    setNewMessage('');
  };
```

Messaging App

Conversations

- Conversation 1
- Conversation 2

Messages

How are you?
I'm fine, thanks!