

WEB TECHNOLOGY

LAB 9

ASSIGNMENT

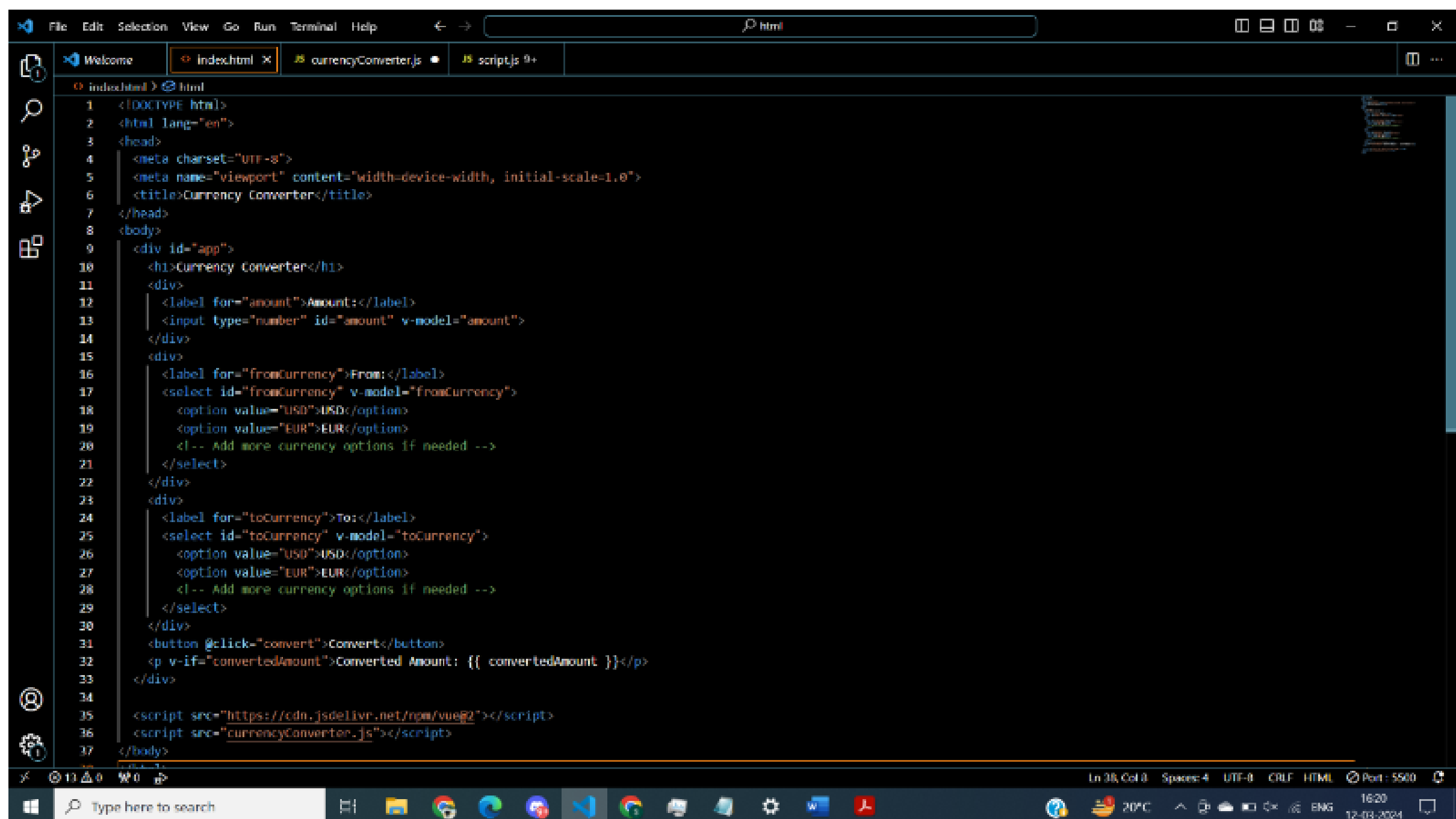
LOKESH

22CS3036

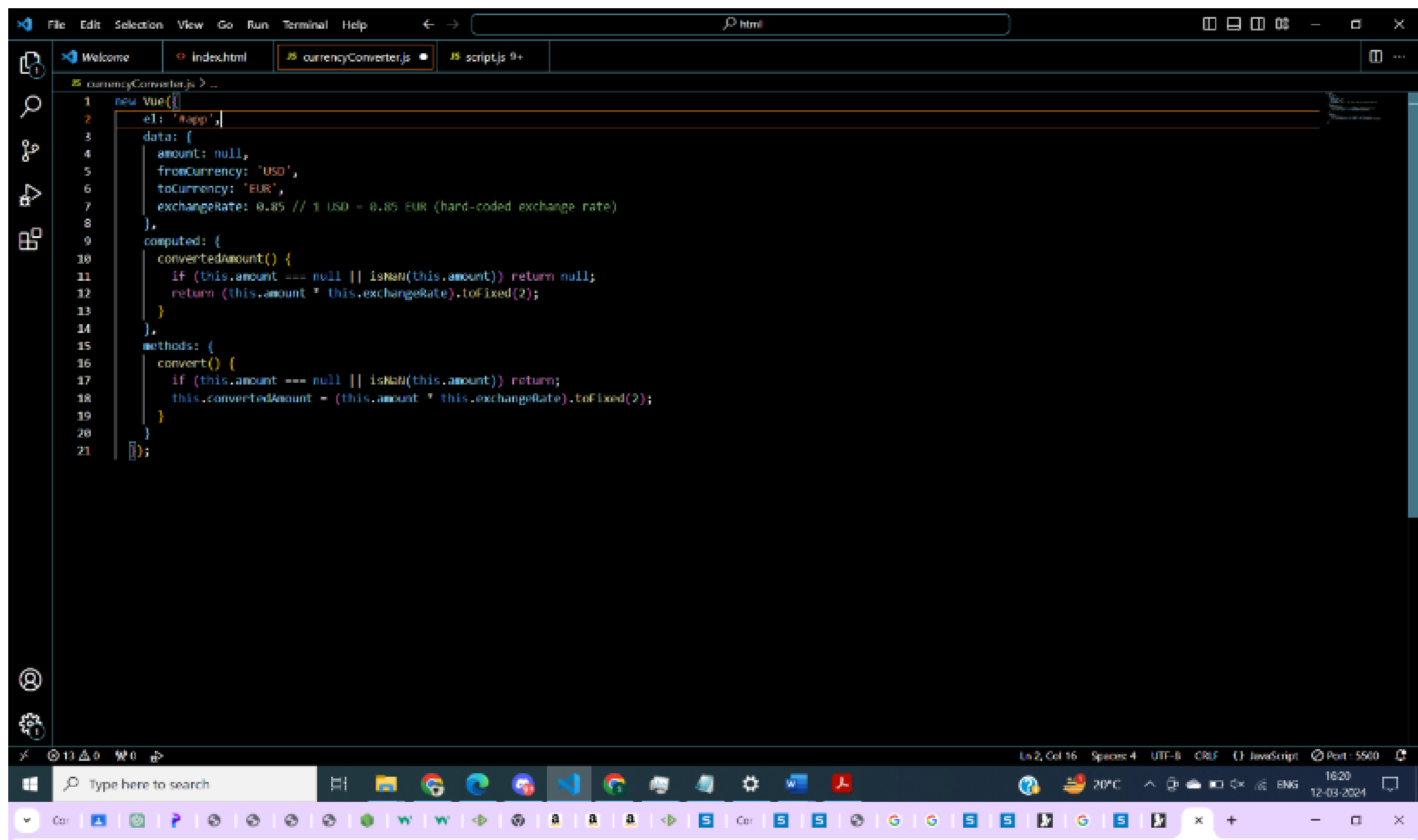
CSE

Q1 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.

Note: Everything has to be done using Vue.js



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Currency Converter</title>
7 </head>
8 <body>
9   <div id="app">
10    <h1>Currency Converter</h1>
11    <div>
12      <label for="amount">Amount:</label>
13      <input type="number" id="amount" v-model="amount">
14    </div>
15    <div>
16      <label for="fromCurrency">From:</label>
17      <select id="fromCurrency" v-model="fromCurrency">
18        <option value="USD">USD</option>
19        <option value="EUR">EUR</option>
20        <!-- Add more currency options if needed -->
21      </select>
22    </div>
23    <div>
24      <label for="toCurrency">To:</label>
25      <select id="toCurrency" v-model="toCurrency">
26        <option value="USD">USD</option>
27        <option value="EUR">EUR</option>
28        <!-- Add more currency options if needed -->
29      </select>
30    </div>
31    <button @click="convert">Convert</button>
32    <p v-if="convertedAmount">Converted Amount: {{ convertedAmount }}</p>
33  </div>
34  <script src="https://cdn.jsdelivr.net/npm/vue@2"></script>
35  <script src="currencyConverter.js"></script>
36 </body>
```



Currency Converter

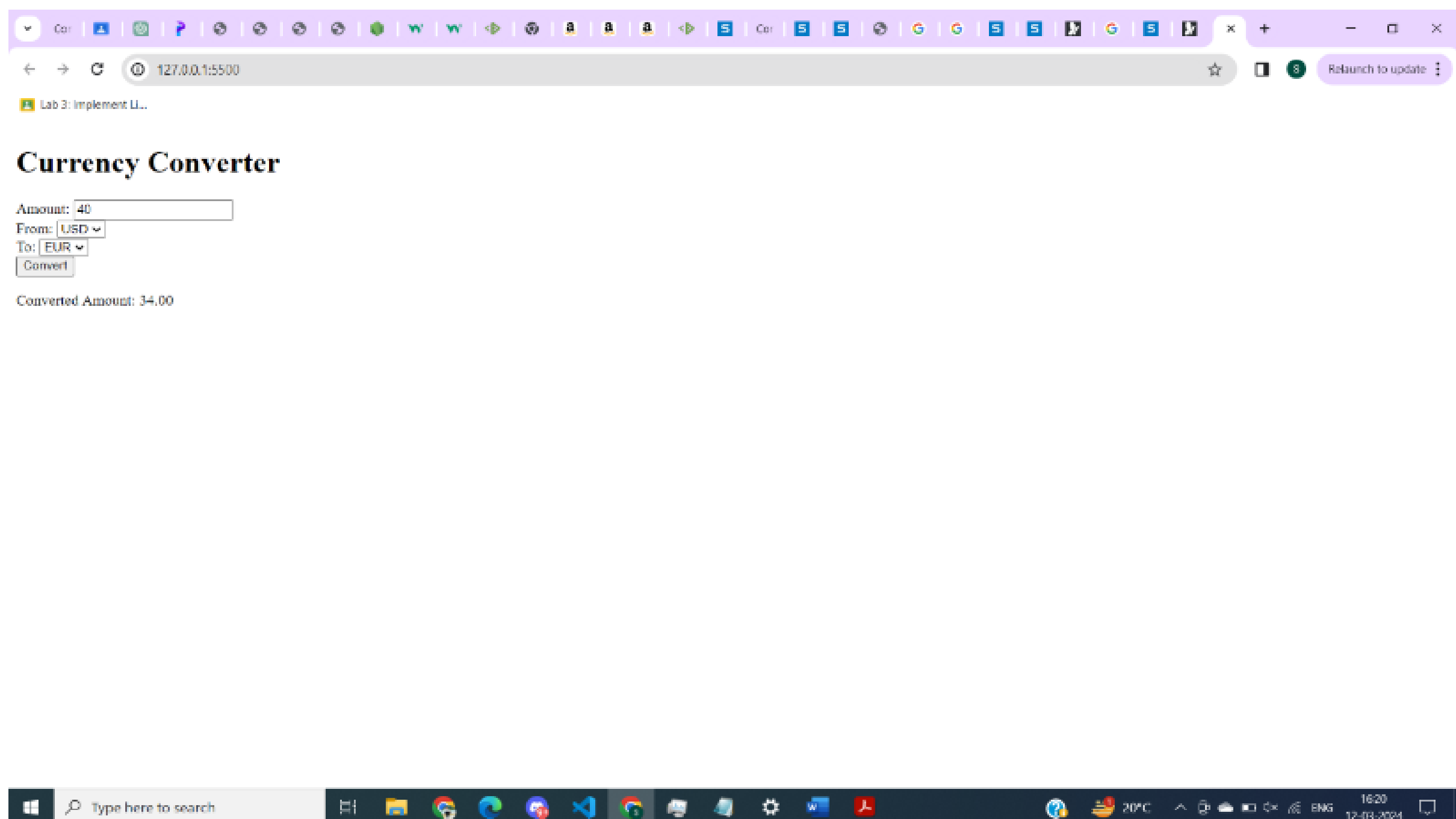
Amount:

From:

To:

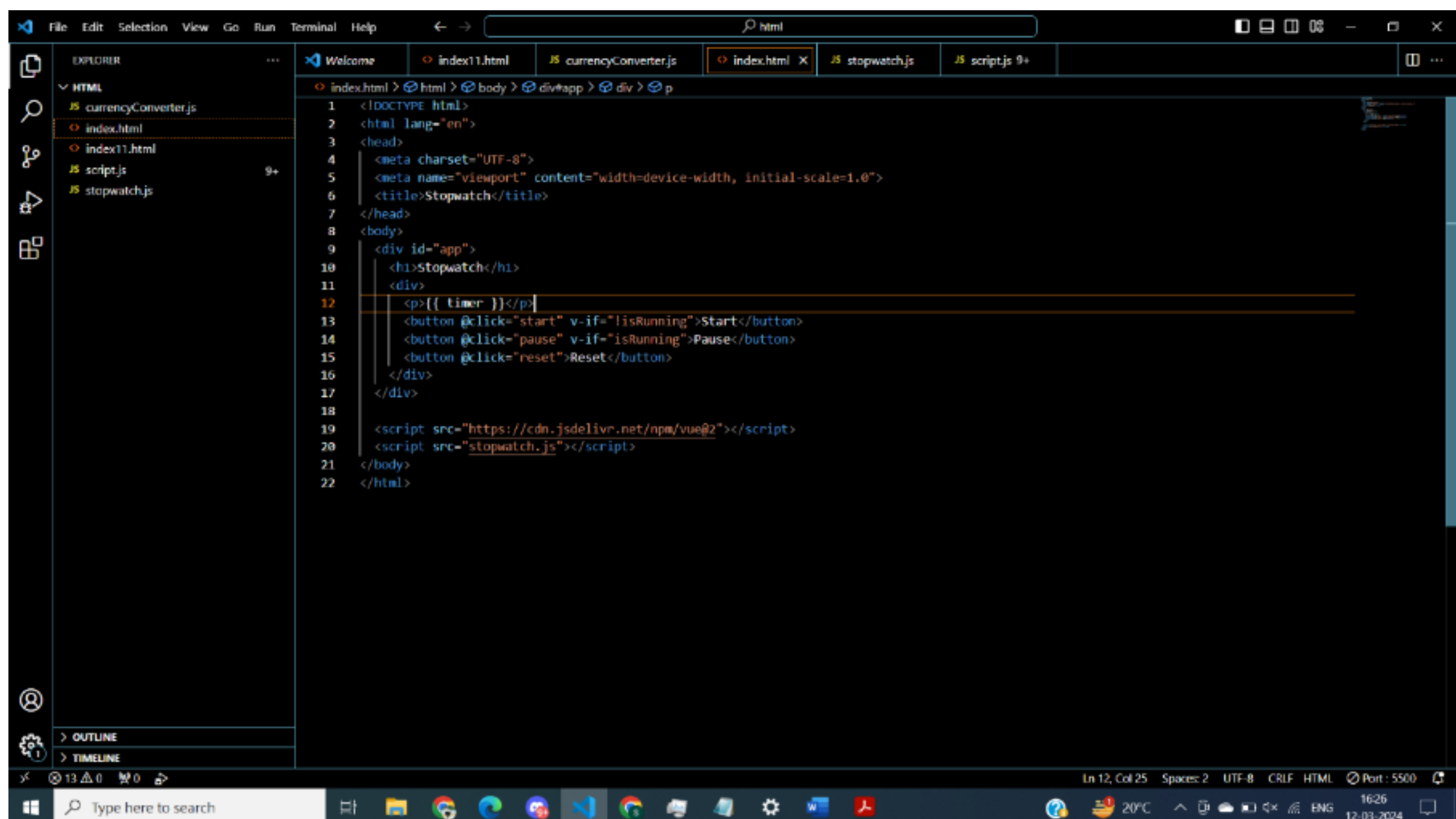
Converted Amount: 38.25

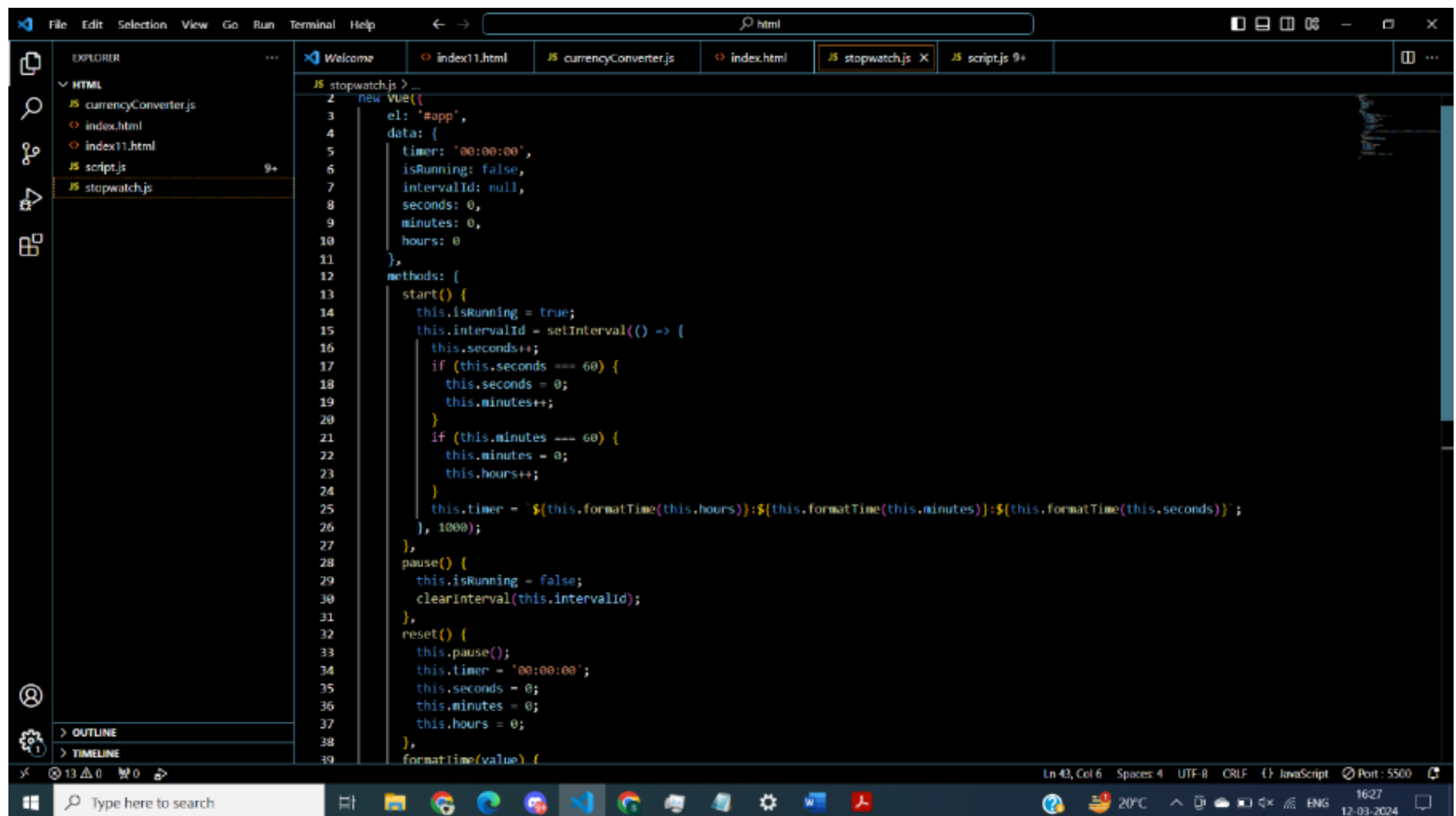




Q2 T2. Create a stopwatch application through which users can start, pause and reset the timer. Use React state, event handlers and the setTimeout or setInterval functions to manage the timer' s state and actions.

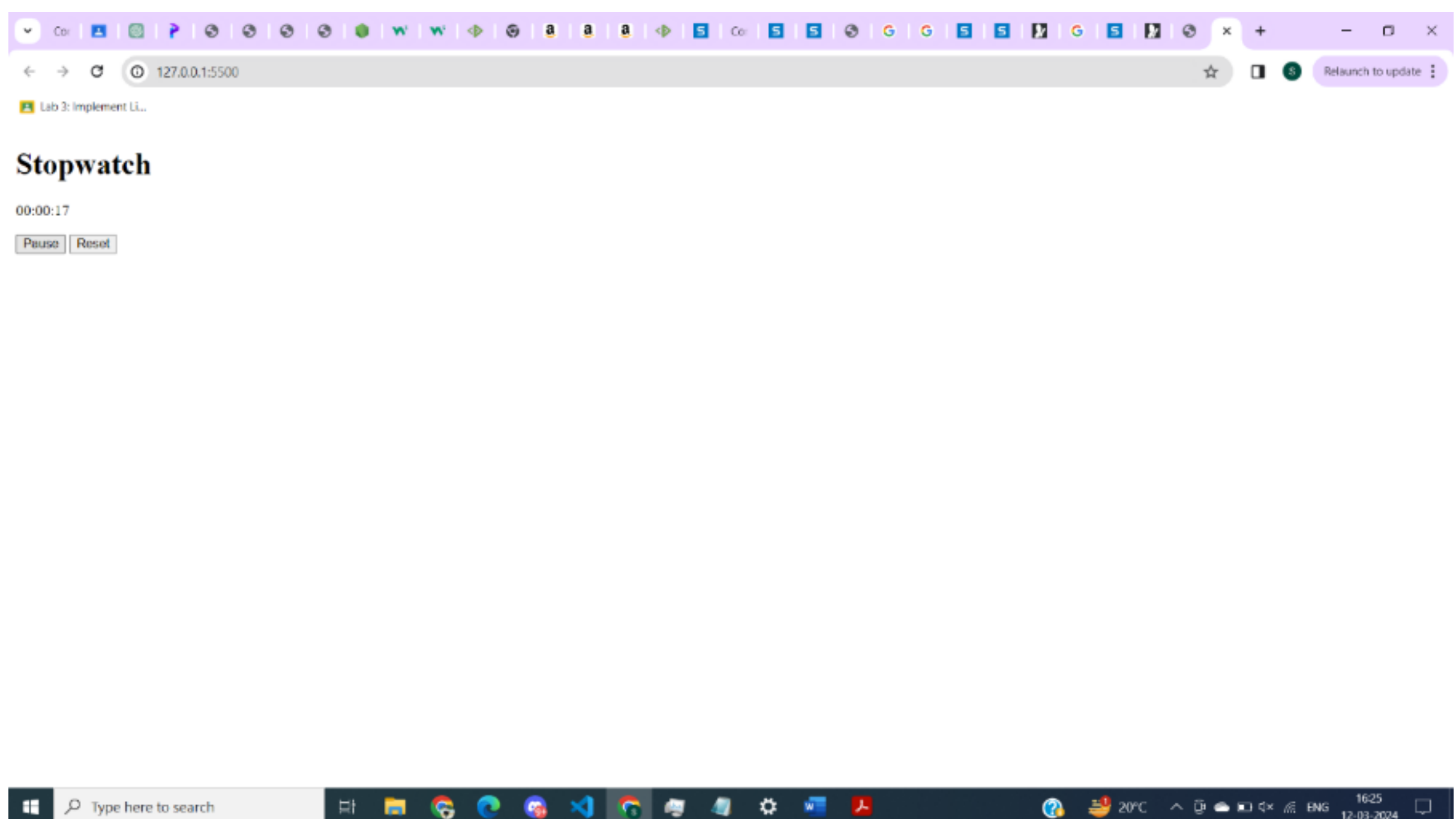
Note: Everything has to be done using Vue.js





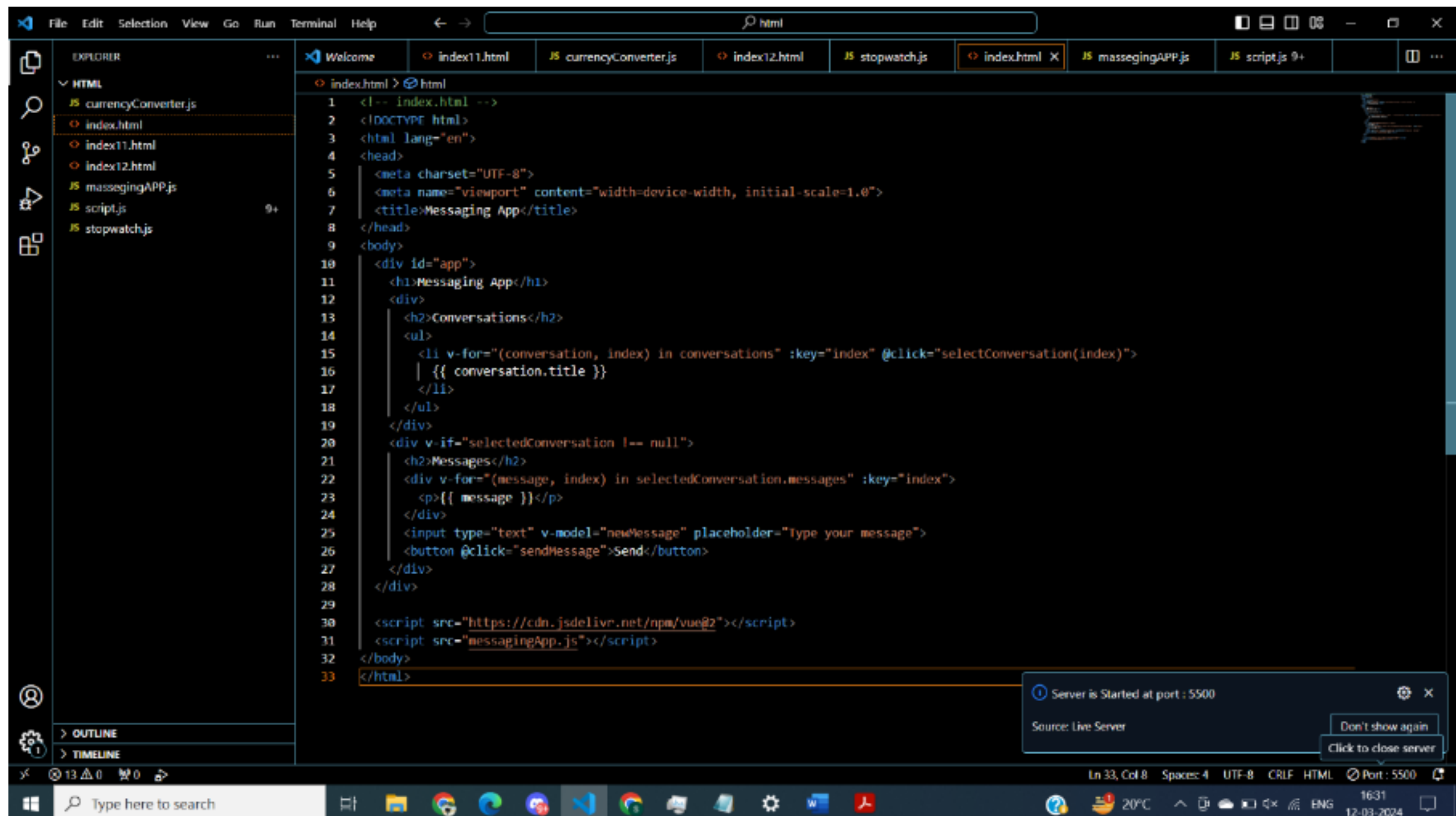
The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a project structure with files like currencyConverter.js, index.html, and stopwatch.js. The main editor window is open to stopwatch.js, showing a Vue.js component with data and methods for a stopwatch. The code includes variables for timer, isRunning, intervalId, seconds, minutes, and hours, along with methods for start, pause, reset, and formatTime.

```
JS stopwatch.js > ...
2 new Vue({
3   el: '#app',
4   data: {
5     timer: '00:00:00',
6     isRunning: false,
7     intervalId: null,
8     seconds: 0,
9     minutes: 0,
10    hours: 0
11  },
12  methods: {
13    start() {
14      this.isRunning = true;
15      this.intervalId = setInterval(() => {
16        this.seconds++;
17        if (this.seconds === 60) {
18          this.seconds = 0;
19          this.minutes++;
20        }
21        if (this.minutes === 60) {
22          this.minutes = 0;
23          this.hours++;
24        }
25        this.timer = `${this.formatTime(this.hours)}:${this.formatTime(this.minutes)}:${this.formatTime(this.seconds)}`;
26      }, 1000);
27    },
28    pause() {
29      this.isRunning = false;
30      clearInterval(this.intervalId);
31    },
32    reset() {
33      this.pause();
34      this.timer = '00:00:00';
35      this.seconds = 0;
36      this.minutes = 0;
37      this.hours = 0;
38    },
39    formatTime(value) {
```



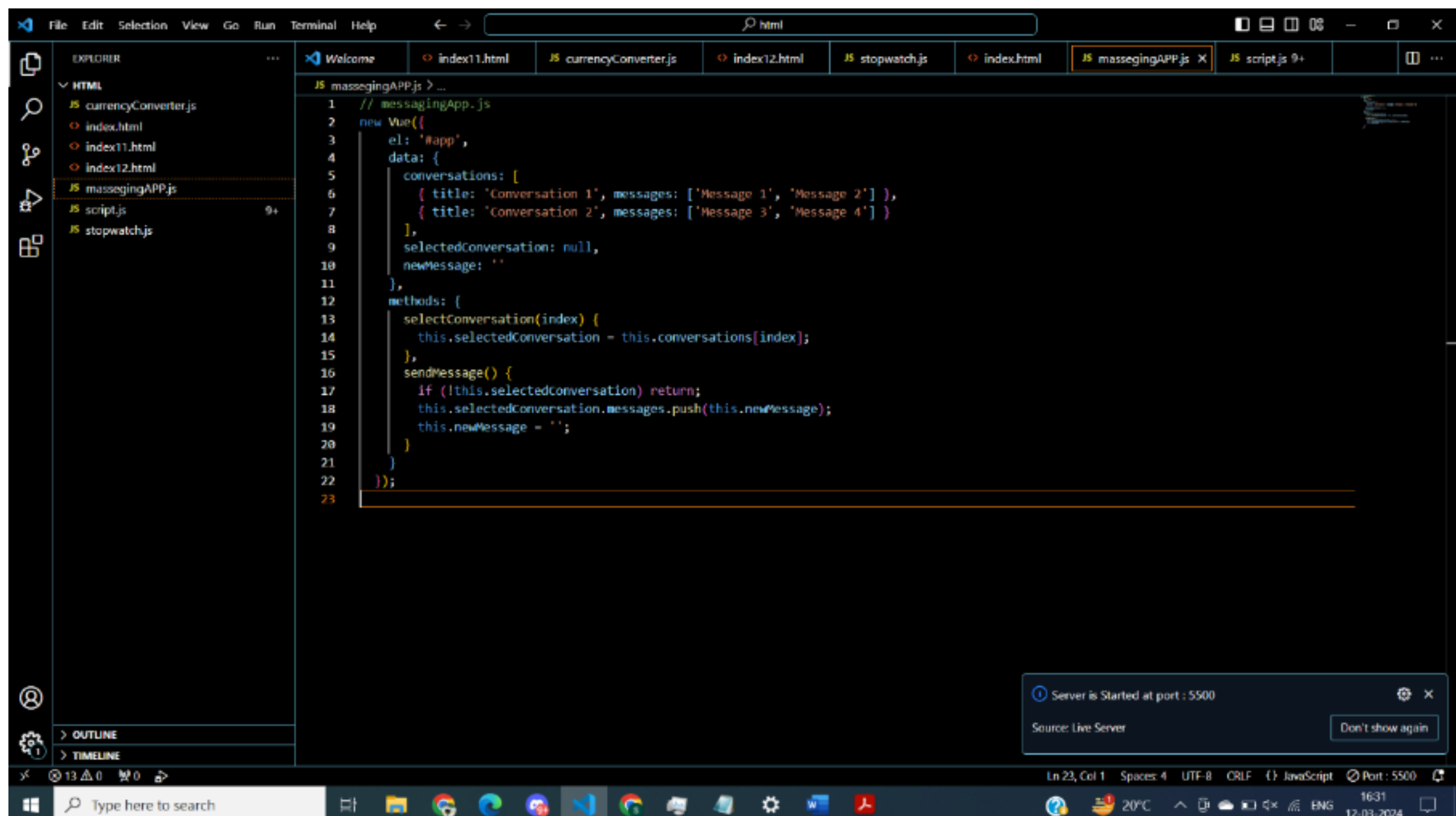
Q2 T3. Develop a messaging application that allows users to send and receive messages in real time. The application should display a list of conversations and allow the user to select a specific conversation to view its messages. The messages should be displayed in a chat interface with the most recent message at the top. Users should be able to send new messages and receive push notifications.

Note: Everything has to be done using Vue.js



This screenshot shows the Visual Studio Code editor with the `index.html` file open. The Explorer sidebar on the left shows a project structure with files like `currencyConverter.js`, `index.html`, `index11.html`, `index12.html`, `massegingAPP.js`, `script.js`, and `stopwatch.js`. The main editor area displays the HTML code for `index.html`, which includes a Vue.js application structure with a `div id="app"` containing a `h1` and a `div` with a `ul` of conversations and a `div` for messages. The code also includes a `script` tag for the `massegingAPP.js` file. A status bar at the bottom indicates the server is running on port 5500.

```
1 <!-- index.html -->
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Messaging App</title>
8 </head>
9 <body>
10  <div id="app">
11    <h1>Messaging App</h1>
12    <div>
13      <h2>Conversations</h2>
14      <ul>
15        <li v-for="(conversation, index) in conversations" :key="index" @click="selectConversation(index)">
16          {{ conversation.title }}
17        </li>
18      </ul>
19    </div>
20    <div v-if="selectedConversation !== null">
21      <h2>Messages</h2>
22      <div v-for="(message, index) in selectedConversation.messages" :key="index">
23        <p>{{ message }}</p>
24      </div>
25      <input type="text" v-model="newMessage" placeholder="Type your message">
26      <button @click="sendMessage">Send</button>
27    </div>
28  </div>
29
30  <script src="https://cdn.jsdelivr.net/npm/vue@2"></script>
31  <script src="massegingAPP.js"></script>
32 </body>
33 </html>
```



This screenshot shows the Visual Studio Code editor with the `massegingAPP.js` file open. The Explorer sidebar on the left shows the same project structure as the previous screenshot. The main editor area displays the JavaScript code for `massegingAPP.js`, which defines a Vue.js instance with a `data` object containing `conversations`, `selectedConversation`, and `newMessage`. It also defines `selectConversation` and `sendMessage` methods. The code is wrapped in an immediately invoked function expression (IIFE). A status bar at the bottom indicates the server is running on port 5500.

```
1 // massegingAPP.js
2 new Vue({
3   el: '#app',
4   data: {
5     conversations: [
6       { title: 'Conversation 1', messages: ['Message 1', 'Message 2'] },
7       { title: 'Conversation 2', messages: ['Message 3', 'Message 4'] }
8     ],
9     selectedConversation: null,
10    newMessage: ''
11  },
12  methods: {
13    selectConversation(index) {
14      this.selectedConversation = this.conversations[index];
15    },
16    sendMessage() {
17      if (!this.selectedConversation) return;
18      this.selectedConversation.messages.push(this.newMessage);
19      this.newMessage = '';
20    }
21  }
22 });
```

