WebTech LAB

Name: Pranjal Khare

RollNo: 22CS2038

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.

App.vue

```
<template>
  <div id="app">
    <h1>Currency Converter</h1>

    <label for="amount">Enter Amount:</label>
    <input type="number" v-model="amount" id="amount">

    <label for="fromCurrency">From Currency:</label>
    <select v-model="fromCurrency" id="fromCurrency">
      <option value="USD">USD</option>
      <option value="EUR">EUR</option>
      <option value="GBP">GBP</option>
    </select>

    <label for="toCurrency">To Currency:</label>
    <select v-model="toCurrency" id="toCurrency">
      <option value="USD">USD</option>
      <option value="EUR">EUR</option>
      <option value="GBP">GBP</option>
    </select>

    <p>Converted Amount: {{ convertedAmount }}</p>

    <button @click="convert">Convert</button>
  </div>
</template>

<script>
export default {
  data() {
```

```
    return {
      amount: 1,
      fromCurrency: 'USD',
      toCurrency: 'USD',
      exchangeRate: 1,
    };
  },
  computed: {
    convertedAmount() {
      return (this.amount * this.exchangeRate).toFixed(2);
    },
  },
  methods: {
    convert() {
      const exchangeRates = {
        'USD': { 'USD': 1, 'EUR': 0.85, 'GBP': 0.75 },
        'EUR': { 'USD': 1.18, 'EUR': 1, 'GBP': 0.88 },
        'GBP': { 'USD': 1.33, 'EUR': 1.14, 'GBP': 1 },
      };

      this.exchangeRate = exchangeRates[this.fromCurrency][this.toCurrency];
    },
  },
};
</script>

<style scoped>
body {
  font-family: 'Arial', sans-serif;
  background-color: #f0f0f0;
  margin: 0;
  padding: 0;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

#app {
  background-color: #fff;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  padding: 20px;
```

```
  border-radius: 8px;
  text-align: center;
}

h1 {
  color: #333;
}

label {
  display: block;
  margin-bottom: 8px;
  color: #555;
}

input, select {
  width: 100%;
  padding: 8px;
  margin-bottom: 16px;
  box-sizing: border-box;
}

button {
  background-color: #4caf50;
  color: #fff;
  padding: 10px 20px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size: 16px;
}

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

p {
  font-size: 18px;
  margin-top: 16px;
  color: #333;
}
</style>
```

index.html

```
<!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>
  <link rel="stylesheet" href="./style.css">
</head>
<body>
  <div id="app"></div>
  <script type="module" src="./script.js"></script>
</body>
</html>
```

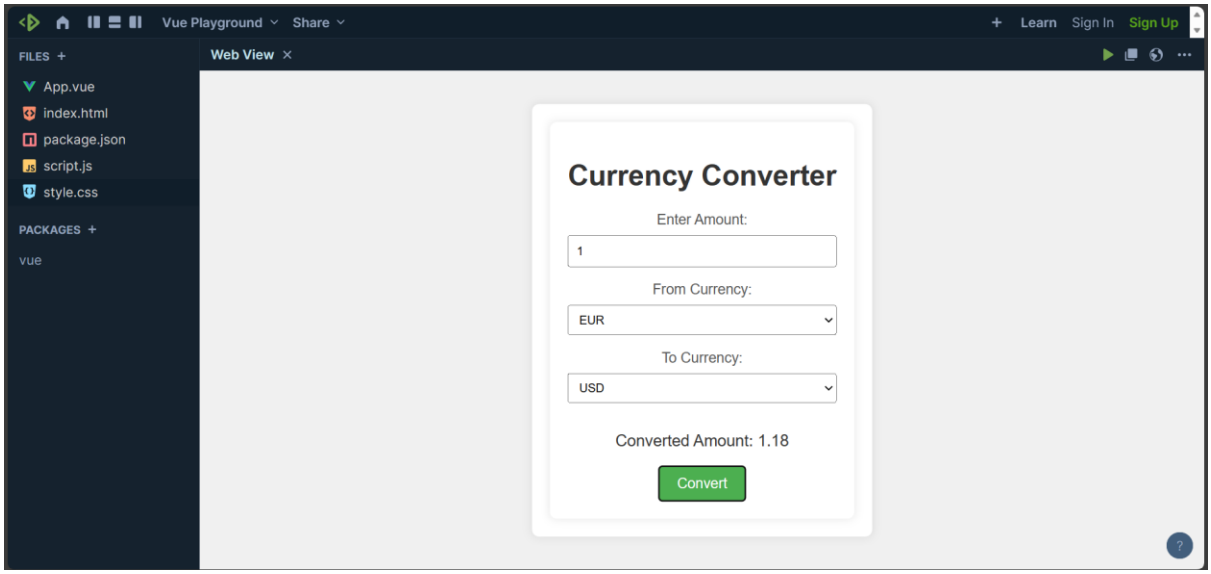
script.js

```
import { createApp } from 'vue';
import App from './App.vue';

createApp(App).mount('#app');
```

style.css

```
body {
  font-family: 'Arial', sans-serif;
  background-color: #f0f0f0;
  margin: 0;
  padding: 0;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

#app {
  background-color: #fff;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  padding: 20px;
  border-radius: 8px;
```

```
  text-align: center;
}

h1 {
  color: #333;
}

label {
  display: block;
  margin-bottom: 8px;
  color: #555;
}

input, select {
  width: 100%;
  padding: 8px;
  margin-bottom: 16px;
  box-sizing: border-box;
}

button {
  background-color: #4caf50;
  color: #fff;
  padding: 10px 20px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size: 16px;
}

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

p {
  font-size: 18px;
  margin-top: 16px;
  color: #333;
}
```



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.

App.vue

```
<template>
  <div id="app">
    <h1>Stopwatch</h1>
    <p>{{ formatTime }}</p>
    <button @click="startTimer" :disabled="isRunning">Start</button>
    <button @click="pauseTimer" :disabled="!isRunning">Pause</button>
    <button @click="resetTimer">Reset</button>
  </div>
</template>

<script>
export default {
  data() {
    return {
      time: 0,
      isRunning: false,
    };
  },
  computed: {
    formatTime() {
      const minutes = Math.floor(this.time / 60);
      const seconds = this.time % 60;
```

```
      return `${String(minutes).padStart(2,
'0')}:${String(seconds).padStart(2, '0')}`;
    },
  },
  methods: {
    startTimer() {
      this.isRunning = true;
      this.intervalId = setInterval(() => {
        this.time += 1;
      }, 1000);
    },
    pauseTimer() {
      this.isRunning = false;
      clearInterval(this.intervalId);
    },
    resetTimer() {
      this.isRunning = false;
      this.time = 0;
      clearInterval(this.intervalId);
    },
  },
};
</script>

<style scoped>
#app {
  font-family: 'Arial', sans-serif;
  text-align: center;
  padding: 20px;
}

h1 {
  color: #333;
}

p {
  font-size: 24px;
  margin: 20px 0;
}

button {
  font-size: 16px;
```

```
  padding: 10px 20px;
  margin: 5px;
  cursor: pointer;
  background-color: #4caf50;
  color: #fff;
  border: none;
  border-radius: 4px;
}

button:disabled {
  background-color: #ddd;
  cursor: not-allowed;
}
</style>
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Vue Playground Stopwatch</title>
  <link rel="stylesheet" href="./style.css">
</head>
<body>
  <div id="app"></div>
  <script type="module" src="./script.js"></script>
</body>
</html>
```

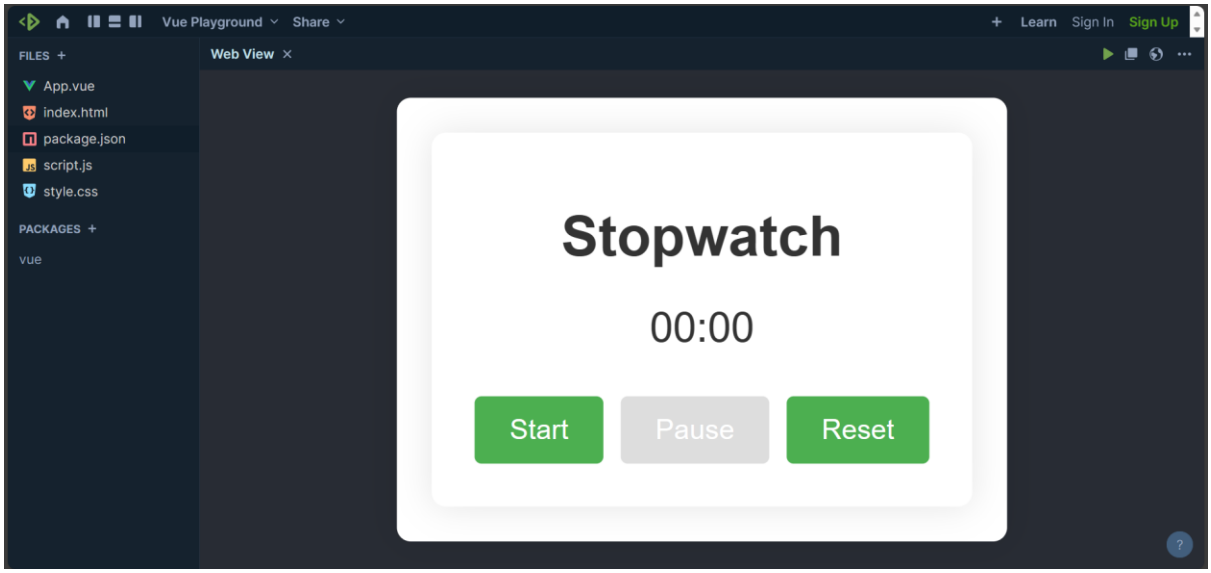
script.js

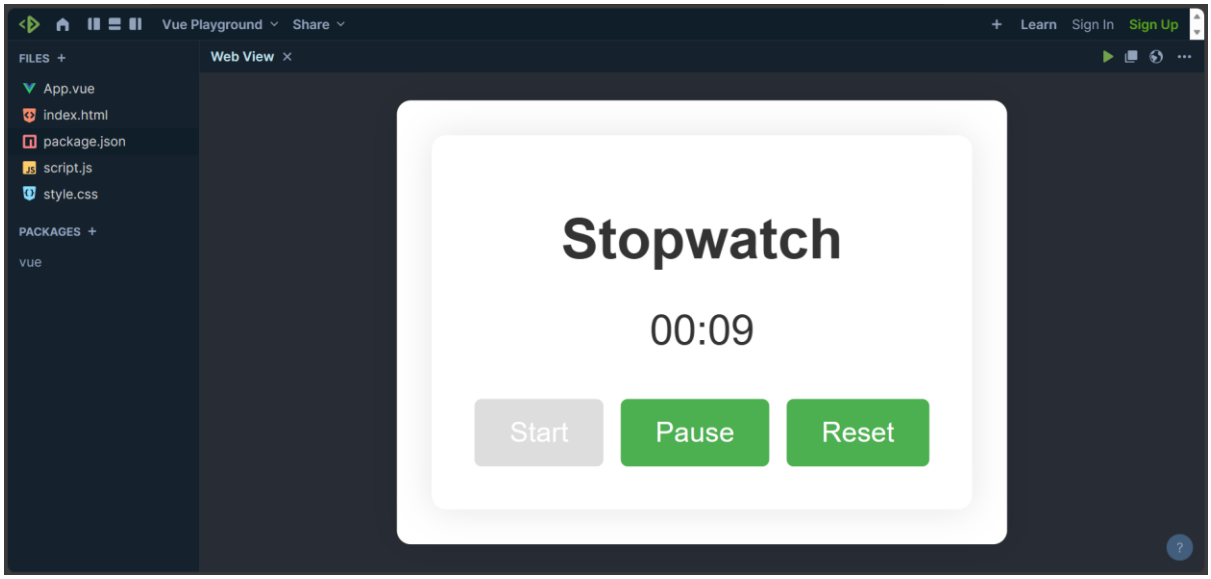
```
import { createApp } from 'vue';
import App from './App.vue';

createApp(App).mount('#app');
```

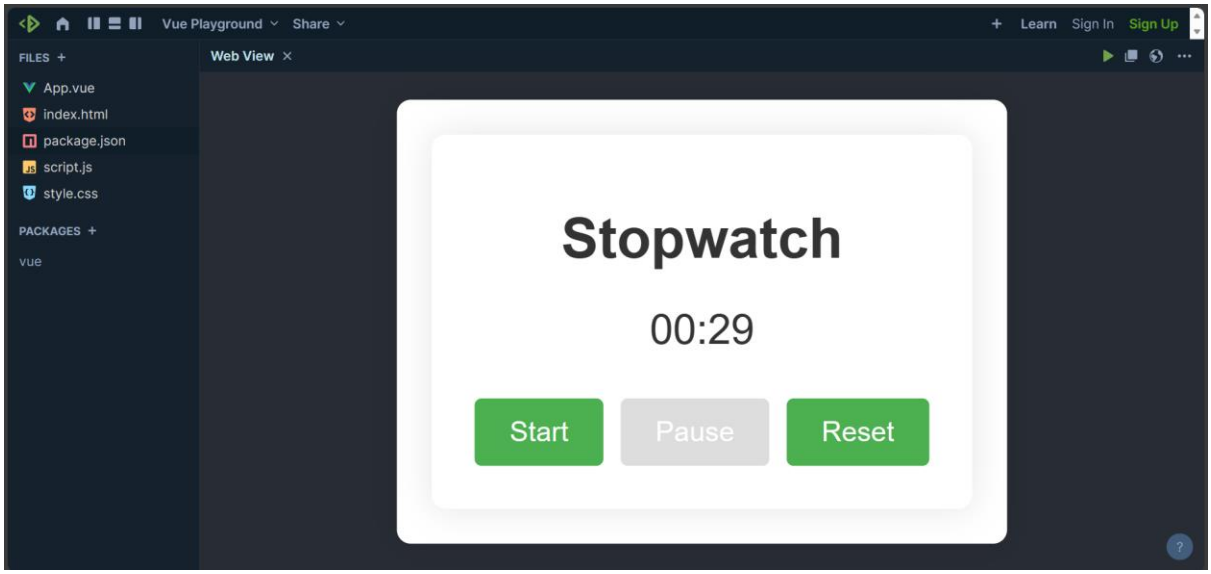
style.css

```css
body {
  margin: 0;
  padding: 0;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
  background-color: #282c34;
}

#app {
  font-family: 'Arial', sans-serif;
  text-align: center;
  padding: 20px;
  background-color: #fff;
  border-radius: 8px;
  box-shadow: 0 0 20px rgba(0, 0, 0, 0.1);
}

h1 {
  color: #61dafb;
  margin-bottom: 20px;
}

p {
  font-size: 36px;
  margin: 20px 0;
  color: #333;
}

button {
  font-size: 18px;
  padding: 12px 24px;
  margin: 10px;
  cursor: pointer;
  background-color: #4caf50;
  color: #fff;
  border: none;
  border-radius: 4px;
  transition: background-color 0.3s;
}
```

```css
button:disabled {
  background-color: #ddd;
  cursor: not-allowed;
}

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







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.

App.vue

```
<template>
  <div id="app">
    <conversation-list :conversations="conversations" />
    <router-view @sendMessage="sendMessage" />
  </div>
</template>

<script>
import ConversationList from './ConversationList.vue';

export default {
  name: 'App',
  components: {
    ConversationList,
  },
  data() {
    return {
      conversations: [
```

```
        { id: 1, title: 'Friend 1' },
        { id: 2, title: 'Friend 2' },
        // Add more conversations as needed
      ],
    };
  },
  methods: {
    sendMessage(message) {
      // Implement logic to handle sending messages (e.g., update messages
array)
      console.log('Message sent:', message);
    },
  },
};
</script>

<style scoped>
#app {
  font-family: Avenir, Helvetica, Arial, sans-serif;
  text-align: center;
  color: #2c3e50;
  margin-top: 60px;
}
</style>
```

ConversationList.vue

```
<template>
  <div>
    <h2>Conversations</h2>
    <ul>
      <li v-for="conversation in conversations" :key="conversation.id">
        <router-link :to="'/conversation/' + conversation.id">{{
conversation.title }}</router-link>
      </li>
    </ul>
  </div>
</template>

<script>
export default {
```

```
  props: ['conversations'],
};
</script>

<style scoped>
/* Add styles if needed */
</style>
```

ConversationDetail.vue

```
<template>
  <div>
    <h2>{{ currentConversation.title }}</h2>
    <div v-for="message in messages" :key="message.id" class="message">
      <span>{{ message.sender }}:</span> {{ message.text }}
    </div>
    <input v-model="newMessage" @keyup.enter="sendMessage" placeholder="Type a
message..." />
  </div>
</template>

<script>
export default {
  data() {
    return {
      currentConversation: { id: 1, title: 'Friend 1' },
      messages: [
        { id: 1, sender: 'Friend 1', text: 'Hello!' },
        // Add more messages as needed
      ],
      newMessage: '',
    };
  },
  methods: {
    sendMessage() {
      if (this.newMessage.trim() !== '') {
        const message = {
          id: this.messages.length + 1,
          sender: 'You',
          text: this.newMessage.trim(),
        };
```

```
        this.messages.push(message);
        this.$emit('sendMessage', message); // Emit the sendMessage event to
App.vue
        this.newMessage = '';
      }
    },
  },
};
</script>

<style scoped>
/* Add styles if needed */
.message {
  margin: 10px 0;
}
</style>
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Messaging App</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div id="app"></div>

  <!-- Vue.js and Vue Router -->
  <script src="https://cdn.jsdelivr.net/npm/vue@2.6.14"></script>
  <script src="https://cdn.jsdelivr.net/npm/vue-router@3.5.2"></script>

  <!-- Your main script file -->
  <script src="script.js"></script>
</body>
</html>
```

script.js

```
import Vue from 'vue';
import VueRouter from 'vue-router';
import App from './App.vue';
import ConversationDetail from './ConversationDetail.vue';

Vue.use(VueRouter);

const routes = [
  { path: '/', component: App },
  { path: '/conversation/:id', component: ConversationDetail },
];

const router = new VueRouter({
  routes,
});

new Vue({
  el: '#app',
  router,
  render: (h) => h(App),
});
```

style.css

```
body {
  color: #fcbe24;
  padding: 0 24px;
  margin: 0;
}
```

package.json

```
{
  "dependencies": {
    "vue": "3.2.33",
    "vue-router": "4.3.0",
    "vue-socket.io": "3.0.10"
  }
}
```