LAB-9

Name -Utkarsh Raj **Roll -22CS2024**

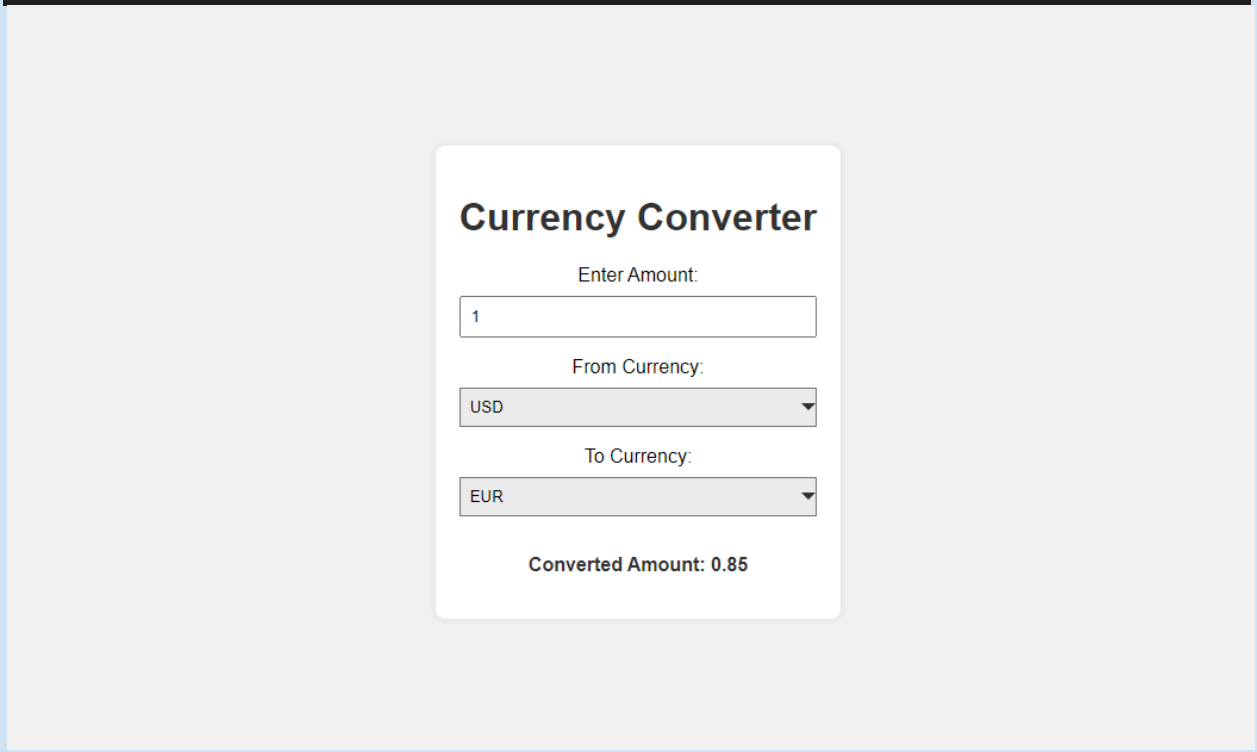
Q1.Develop a currency converter application that allows users to input an amount in one currency and convert it to another. For the sake of this challenge, you can use a hard-coded exchange rate. Take advantage of React state and event handlers to manage the input and conversion calculations.

```
<!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>
  <script src="https://cdn.jsdelivr.net/npm/vue@2"></script>

  <style>
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 0;
      display: flex;
      align-items: center;
      justify-content: center;
      height: 100vh;
    }

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

    h1 {
      color: #333;
    }
```

```
    label {
      display: block;
      margin-bottom: 8px;
    }

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

    select {
      appearance: none;
      -webkit-appearance: none;
      -moz-appearance: none;
      background: url('data:image/svg+xml;utf8,<svg
xmlns="http://www.w3.org/2000/svg" width="12" height="6" viewBox="0 0 12
6"><polygon fill="%23333" points="0,0 12,0 6,6"/></svg>') no-repeat right
#eee;
      background-size: 12px 6px;
    }

    p {
      margin-top: 16px;
      font-weight: bold;
      color: #333;
    }
  </style>
</head>
<body>

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

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

```
  </div>

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

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

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

<script>
new Vue({
  el: '#app',
  data: {
    amount: 1,
    fromCurrency: 'USD',
    toCurrency: 'EUR',
    exchangeRate: 0.85,
  },
  computed: {
    convertedAmount: function () {
      return (this.amount * this.exchangeRate).toFixed(2);
    },
  },
  methods: {
```

```
    convertCurrency: function () {
      this.convertedAmount = (this.amount * this.exchangeRate).toFixed(2);
    },
  },
});
</script>

</body>
</html>
```



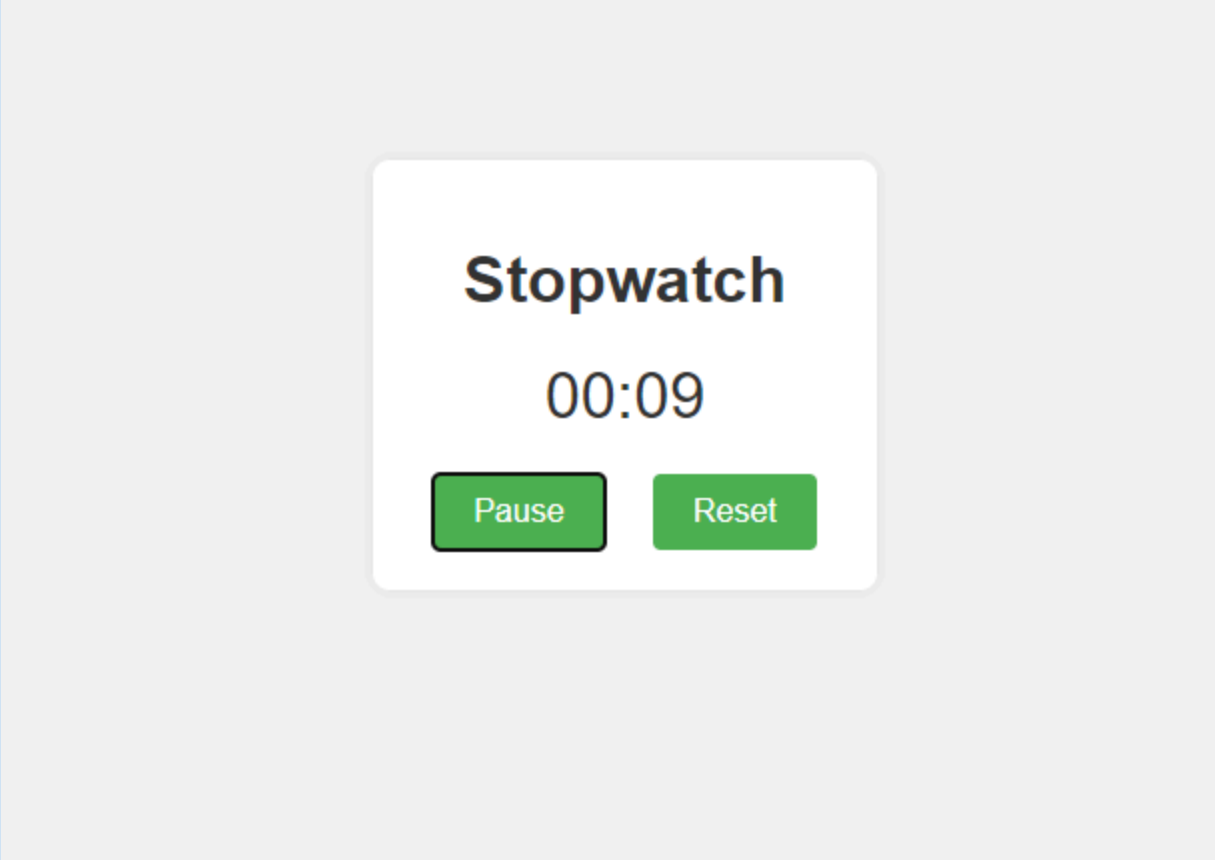
Q2.T2. Create a stopwatch application through which users can start, pause and reset the timer.
Use React state, event handlers and the setTimeout or setInterval functions to manage the timer's state and actions.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Stopwatch</title>
  <script src="https://cdn.jsdelivr.net/npm/vue@2"></script>
```

```
<style>
  body {
    font-family: 'Arial', sans-serif;
    background-color: #f4f4f4;
    margin: 0;
    padding: 0;
    display: flex;
    align-items: center;
    justify-content: center;
    height: 100vh;
  }

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

  h1 {
    color: #333;
  }

  p {
    font-size: 2em;
    margin: 20px 0;
    color: #333;
  }

  button {
    background-color: #4caf50;
    color: #fff;
    padding: 10px 20px;
    font-size: 1em;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    margin: 0 10px;
```

```
      transition: background-color 0.3s;
    }

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

<div id="app">
  <h1>Stopwatch</h1>

  <div>
    <p>{{ formatTime }}</p>
  </div>

  <div>
    <button @click="startPauseTimer">{{ isRunning ? 'Pause' : 'Start'
}}</button>
    <button @click="resetTimer">Reset</button>
  </div>
</div>

<script>
new Vue({
  el: '#app',
  data: {
    timer: 0,
    isRunning: false,
  },
  computed: {
    formatTime: function () {
      const minutes = Math.floor(this.timer / 60);
      const seconds = this.timer % 60;
      return `${this.padNumber(minutes)}:${this.padNumber(seconds)}`;
    },
  },
  methods: {
    startPauseTimer: function () {
```

```
      if (this.isRunning) {
        clearInterval(this.timerInterval);
      } else {
        this.timerInterval = setInterval(() => {
          this.timer++;
        }, 1000);
      }
      this.isRunning = !this.isRunning;
    },
    resetTimer: function () {
      clearInterval(this.timerInterval);
      this.timer = 0;
      this.isRunning = false;
    },
    padNumber: function (number) {
      return number.toString().padStart(2, '0');
    },
  },
});
</script>

</body>
</html>
```



Q3.T2. 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>
  <script src="https://cdn.jsdelivr.net/npm/vue@2"></script>

  <style>
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f5f5f5;
      margin: 0;
```

```
  padding: 0;
  display: flex;
  align-items: center;
  justify-content: center;
  height: 100vh;
}

#app {
  background-color: #fff;
  border-radius: 8px;
  box-shadow: 0 0 20px rgba(0, 0, 0, 0.1);
  width: 400px;
  overflow: hidden;
}

.header {
  background-color: #2196F3;
  color: #fff;
  padding: 10px;
  text-align: center;
}

.conversation-list {
  overflow-y: auto;
  max-height: 300px;
}

.conversation {
  padding: 15px;
  border-bottom: 1px solid #ddd;
  cursor: pointer;
  transition: background-color 0.3s;
}

.conversation:hover {
  background-color: #f9f9f9;
}

.chat-area {
  padding: 20px;
```

```
    height: 400px;
    overflow-y: auto;
  }

  .message {
    margin-bottom: 15px;
  }

  .message.sent {
    text-align: right;
  }

  .message.received {
    text-align: left;
  }

  .input-area {
    padding: 15px;
    border-top: 1px solid #ddd;
    display: flex;
    justify-content: space-between;
    align-items: center;
  }

  input[type="text"] {
    flex: 1;
    padding: 10px;
    margin-right: 10px;
    border: 1px solid #ddd;
    border-radius: 4px;
  }

  button {
    background-color: #4caf50;
    color: #fff;
    padding: 10px 15px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    transition: background-color 0.3s;
```

```
    }

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

<div id="app">
  <div class="header">
    <h2>Messaging App</h2>
  </div>

  <div class="conversation-list">
    <div v-for="conversation in conversations" class="conversation"
@click="selectConversation(conversation.id)">
      {{ conversation.name }}
    </div>
  </div>

  <div class="chat-area">
    <div v-for="message in selectedConversation.messages" :class="{
'message': true, 'sent': message.sentBy === 'user', 'received':
message.sentBy === 'contact' }">
      {{ message.text }}
    </div>
  </div>

  <div class="input-area">
    <input type="text" v-model="newMessage" placeholder="Type your
message...">
    <button @click="sendMessage">Send</button>
  </div>
</div>

<script>
new Vue({
  el: '#app',
  data: {
```

```
    conversations: [
      { id: 1, name: 'John Doe', messages: [] },
      { id: 2, name: 'Jane Doe', messages: [] },
    ],
    selectedConversation: null,
    newMessage: '',
  },
  methods: {
    selectConversation: function (conversationId) {
      this.selectedConversation = this.conversations.find(conversation =>
conversation.id === conversationId);
    },
    sendMessage: function () {
      if (this.selectedConversation) {
        const newMessage = { text: this.newMessage, sentBy: 'user' };
        this.selectedConversation.messages.push(newMessage);
        setTimeout(() => {
          const responseMessage = { text: 'Response from contact', sentBy:
'contact' };
          this.selectedConversation.messages.push(responseMessage);
        }, 1000);
        this.newMessage = '';
      }
    },
  },
});
</script>

</body>
</html>
```

Messaging App

Friend 1

Friend 2

Friend 2

friend: Hey!

user: How are you?

Type your message...

Send