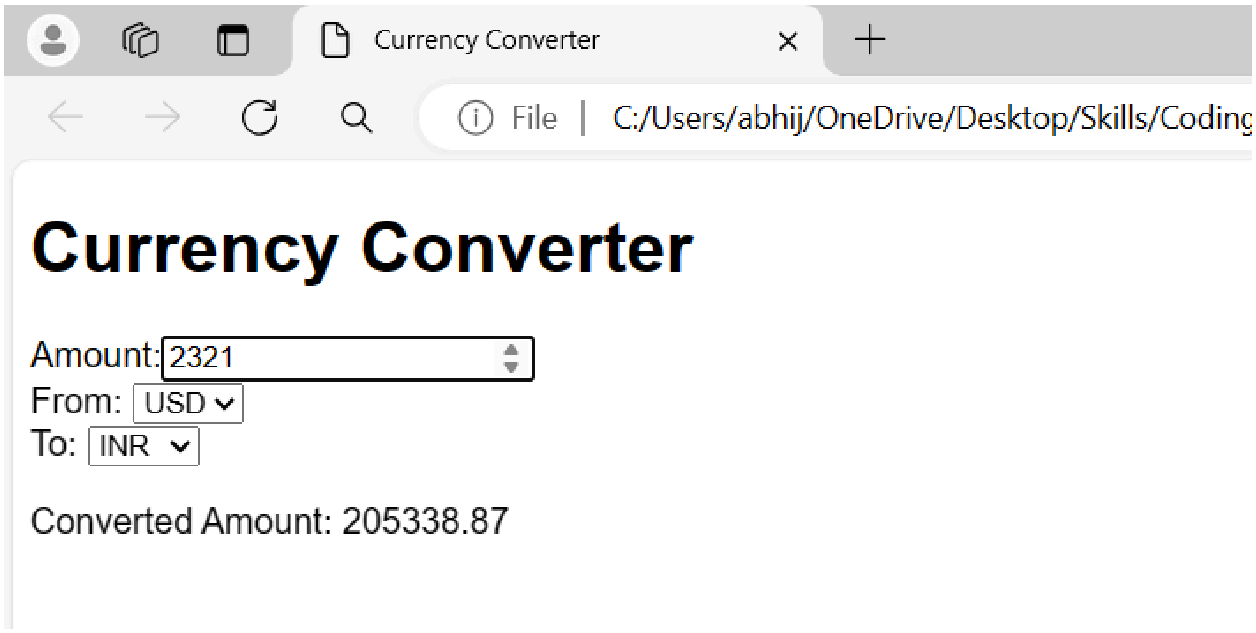
# Web Tech Lab-8

Name:- Abhijeet Jadhav
Branch :- Mathematics and Computing
Roll:- 22mc3002

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

const CurrencyConverter = () => {
  const [amount, setAmount] = useState('');
  const [fromCurrency, setFromCurrency] = useState('USD');
  const [toCurrency, setToCurrency] = useState('INR');
  const exchangeRate = 88.47;

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

  const handleFromCurrencyChange = (e) => {
    setFromCurrency(e.target.value);
  };

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

  const convertCurrency = () => {
    const convertedAmount = amount * exchangeRate;
    return convertedAmount.toFixed(2);
  };

  return (
    <div>
      <h2>Currency Converter</h2>
      <div>
        <label htmlFor="amount">Amount:</label>
        <input type="number" id="amount" value={amount}
onChange={handleAmountChange} />
      </div>
      <div>
        <label htmlFor="fromCurrency">From Currency:</label>
        <select id="fromCurrency" value={fromCurrency}
onChange={handleFromCurrencyChange}>
          <option value="USD">USD</option>
          <option value="EUR">EUR</option>
          {/* Add more currencies as needed */}
        </select>
      </div>
      <div>
        <label htmlFor="toCurrency">To Currency:</label>
        <select id="toCurrency" value={toCurrency}
onChange={handleToCurrencyChange}>
          <option value="USD">USD</option>
          <option value="INR">EUR</option>
          {/* Add more currencies as needed */}
        </select>
      </div>
      <div>
        <button onClick={convertCurrency}>Convert</button>
      </div>
      <div>
        {amount && (
          <p>
            {amount} {fromCurrency} is equal to {convertCurrency()}
{toCurrency}
          </p>
        )}
      </div>
    </div>
  );
};

export default CurrencyConverter;
```



Currency Converter

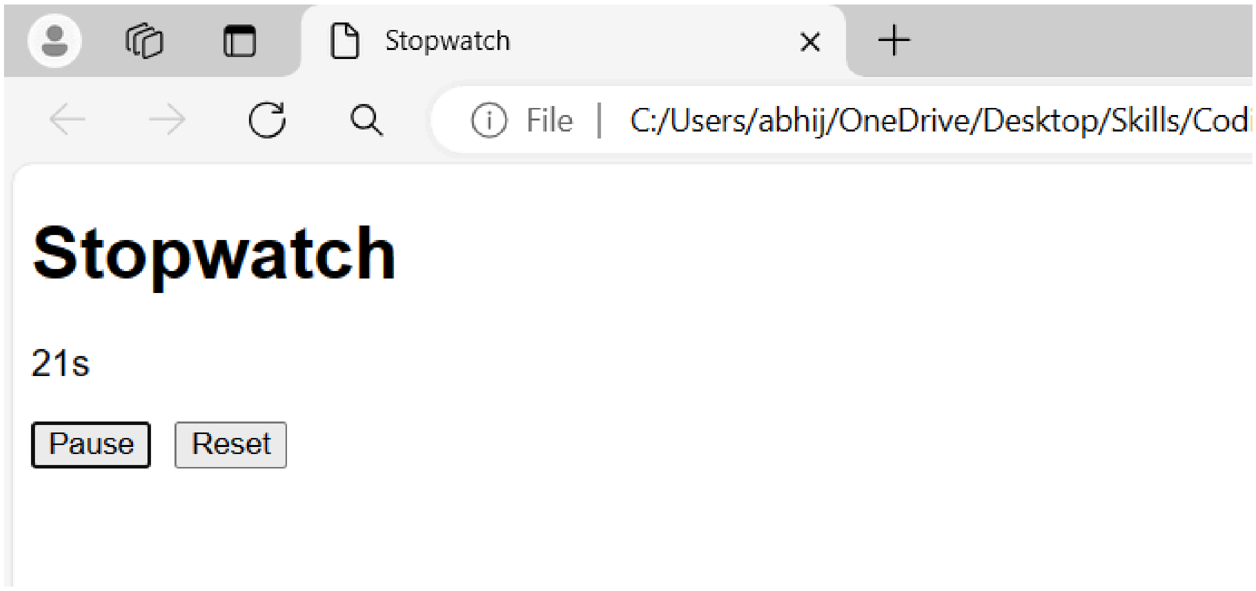
Amount: 2321
From: USD
To: INR

Converted Amount: 205338.87

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.

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

const Stopwatch = () => {
  const [time, setTime] = useState(0);
  const [isRunning, setIsRunning] = useState(false);
  const intervalRef = useRef(null);

  const startStopwatch = () => {
    if (!isRunning) {
      setIsRunning(true);
      intervalRef.current = setInterval(() => {
        setTime(prevTime => prevTime + 1);
      }, 1000);
    }
  };

  const pauseStopwatch = () => {
    clearInterval(intervalRef.current);
    setIsRunning(false);
  };

  const resetStopwatch = () => {
    clearInterval(intervalRef.current);
    setTime(0);
    setIsRunning(false);
  };

  const formatTime = (timeInSeconds) => {
    const hours = Math.floor(timeInSeconds / 3600);
    const minutes = Math.floor((timeInSeconds % 3600) / 60);
    const seconds = timeInSeconds % 60;

    return `${hours.toString().padStart(2,
'0')}:${minutes.toString().padStart(2,
'0')}:${seconds.toString().padStart(2, '0')}`;
  };

  return (
    <div>
      <h2>Stopwatch</h2>
      <p>{formatTime(time)}</p>
      <div>
        {!isRunning ? (
          <button onClick={startStopwatch}>Start</button>
        ) : (
          <button onClick={pauseStopwatch}>Pause</button>
        )}
        <button onClick={resetStopwatch}>Reset</button>
      </div>
    </div>
  );
};

export default Stopwatch;
```

Output:-



Stopwatch

21s

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.

React

```
import React, { useState, useEffect } from 'react';
import firebase from 'firebase/app';
import 'firebase/database';

const firebaseConfig = {
  // Your Firebase configuration
};

firebase.initializeApp(firebaseConfig);

const MessagingApp = () => {
  const [conversations, setConversations] = useState([]);
  const [selectedConversation, setSelectedConversation] = useState(null);
  const [newMessage, setNewMessage] = useState('');

  useEffect(() => {
    const conversationsRef = firebase.database().ref('conversations');
    conversationsRef.on('value', (snapshot) => {
      const data = snapshot.val();
      if (data) {
        setConversations(Object.values(data));
      }
    });
  }, []);

  const selectConversation = (conversation) => {
    setSelectedConversation(conversation);
  };

  const sendMessage = () => {
    if (newMessage.trim() === '') return;

    const conversationRef =
firebase.database().ref(`conversations/${selectedConversation.id}/messages
`);
    conversationRef.push({
      text: newMessage,
      sender: 'user', // or you can set it to the user's ID if you have
user authentication
      timestamp: firebase.database.ServerValue.TIMESTAMP
    });

    setNewMessage('');
  };

  return (
    <div>
      <h2>Conversations</h2>
      <ul>
        {conversations.map(conversation => (
          <li key={conversation.id} onClick={() =>
selectConversation(conversation)}>
            {conversation.title}
          </li>
        ))}
      </ul>

      {selectedConversation && (
        <div>
          <h3>{selectedConversation.title}</h3>
          <div>
            {selectedConversation.messages.map(message => (
              <div key={message.id}>
                <p>{message.text}</p>
                <small>{message.sender}</small>
              </div>
            ))}
          </div>
          <input type="text" value={newMessage} onChange={(e) =>
setNewMessage(e.target.value)} />
          <button onClick={sendMessage}>Send</button>
        </div>
      )}
    </div>
  );
};

export default MessagingApp;
```

Veu.js

```
<template>
  <div>
    <h2>Conversations</h2>
    <ul>
      <li v-for="conversation in conversations" :key="conversation.id"
@click="selectConversation(conversation)">
        {{ conversation.title }}
      </li>
    </ul>

    <div v-if="selectedConversation">
      <h3>{{ selectedConversation.title }}</h3>
      <div v-for="message in selectedConversation.messages"
:key="message.id">
        <p>{{ message.text }}</p>
        <small>{{ message.sender }}</small>
      </div>
      <input type="text" v-model="newMessage" />
      <button @click="sendMessage">Send</button>
    </div>
  </div>
</template>

<script>
import { db } from './firebase';
export default {
  data() {
    return {
      conversations: [],
      selectedConversation: null,
      newMessage: ''
    };
  },
  firestore() {
    return {
      conversations: db.collection('conversations')
    };
  },
  methods: {
    selectConversation(conversation) {
      this.selectedConversation = conversation;
    },
    sendMessage() {
      if (this.newMessage.trim() === '') return;


db.collection(`conversations/${this.selectedConversation.id}/messages`).ad
d({
        text: this.newMessage,
        sender: 'user',
        timestamp: firebase.firestore.FieldValue.serverTimestamp()
      });

      this.newMessage = '';
    }
  }
};
</script>
```