WEB Technology assignment

Shubh Raheja

22cs2031

Q1.

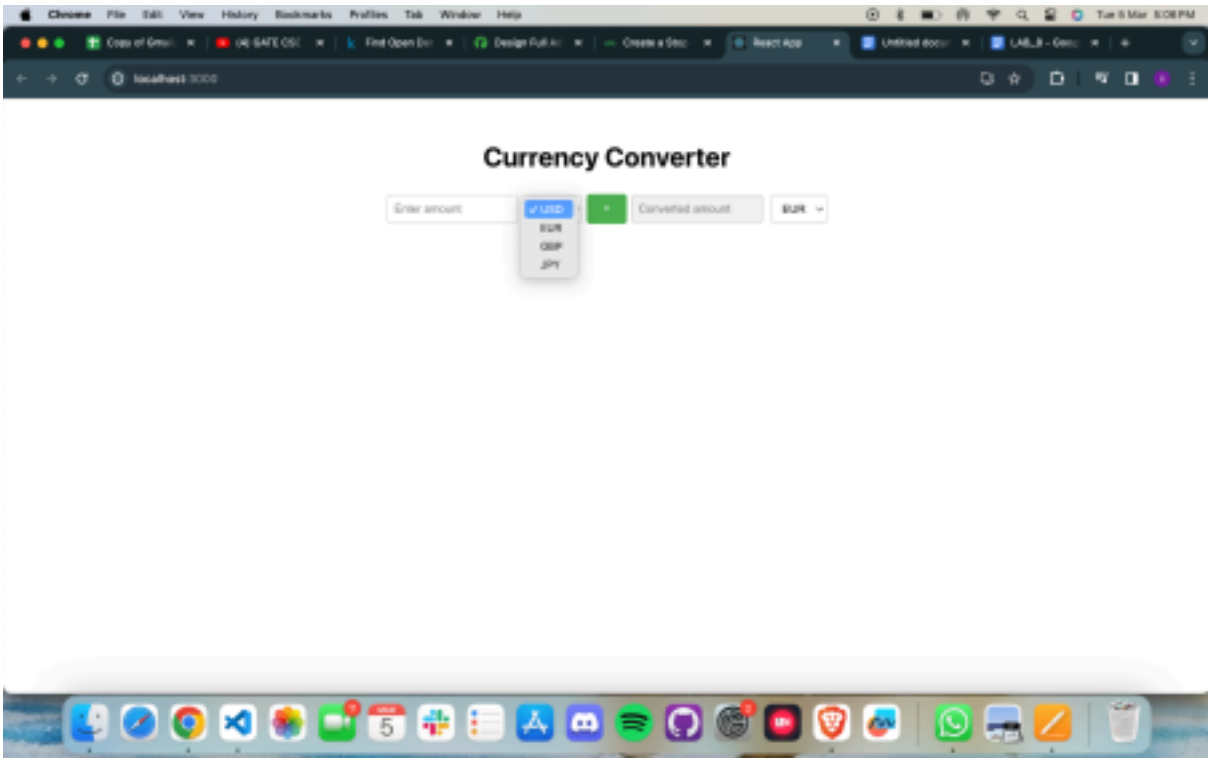
```
// App.js

import React, { useState } from 'react';

import './App.css';


function App() {

 const [amount, setAmount] = useState('');

 const [fromCurrency, setFromCurrency] = useState('USD');

 const [toCurrency, setToCurrency] = useState('EUR');

 const [convertedAmount, setConvertedAmount] = useState('');


 // Hard-coded exchange rate

 const exchangeRate = {

   USD: {

     EUR: 0.85,

     GBP: 0.72,

     JPY: 109.63,

   },

   EUR: {

     USD: 1.18,

     GBP: 0.85,
     JPY: 130.50,

   },

   GBP: {

     USD: 1.38,
```

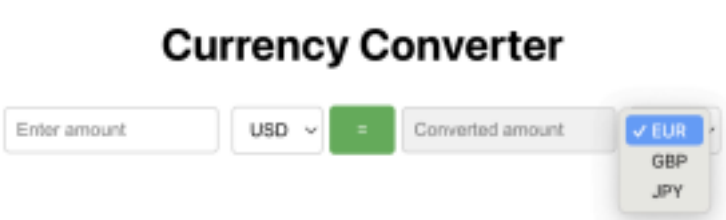
```
    EUR: 1.17,
    JPY: 152.38,
  },
  JPY: {
    USD: 0.0091,
    EUR: 0.0077,
    GBP: 0.0066,
  },
};

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

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

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

const convertCurrency = () => {
  const converted = amount *
  exchangeRate[fromCurrency][toCurrency];
  setConvertedAmount(converted.toFixed(2));
};
return (
  <div className="App">
    <h1>Currency Converter</h1>
    <div className="converter">
      <input
        type="number"
```

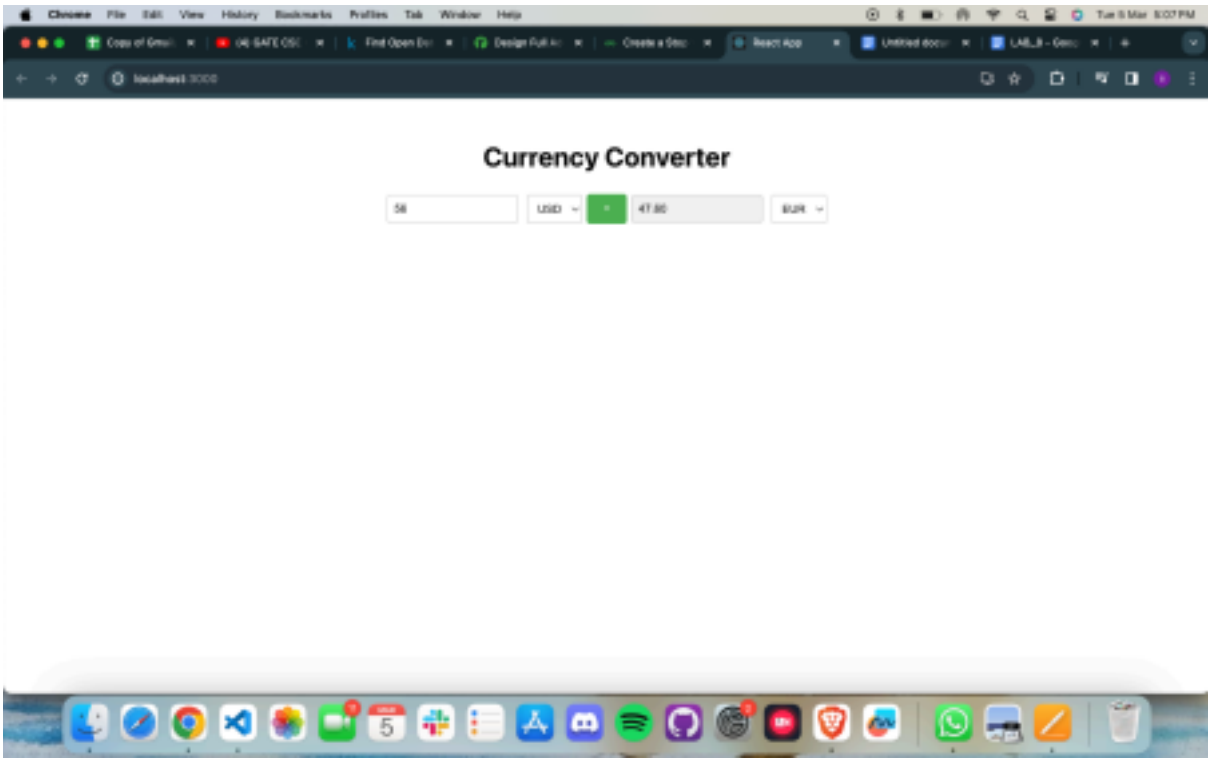
```
          value={amount}
          onChange={handleAmountChange}
          placeholder="Enter amount"
        />
        <select value={fromCurrency}
          onChange={handleFromCurrencyChange}>
          {Object.keys(exchangeRate).map((currency) => (
            <option key={currency} value={currency}>
              {currency}
            </option>
          ))}
        </select>
        <button onClick={convertCurrency}>=</button>
        <input
          type="text"
          value={convertedAmount}
          readOnly
          placeholder="Converted amount"
        />
        <select value={toCurrency} onChange={handleToCurrencyChange}>
          {Object.keys(exchangeRate[fromCurrency]).map((currency) =>
          ( <option key={currency} value={currency}>
              {currency}
            </option>
          ))}
        </select>
      </div>
    </div>
  );
}

export default App;
```

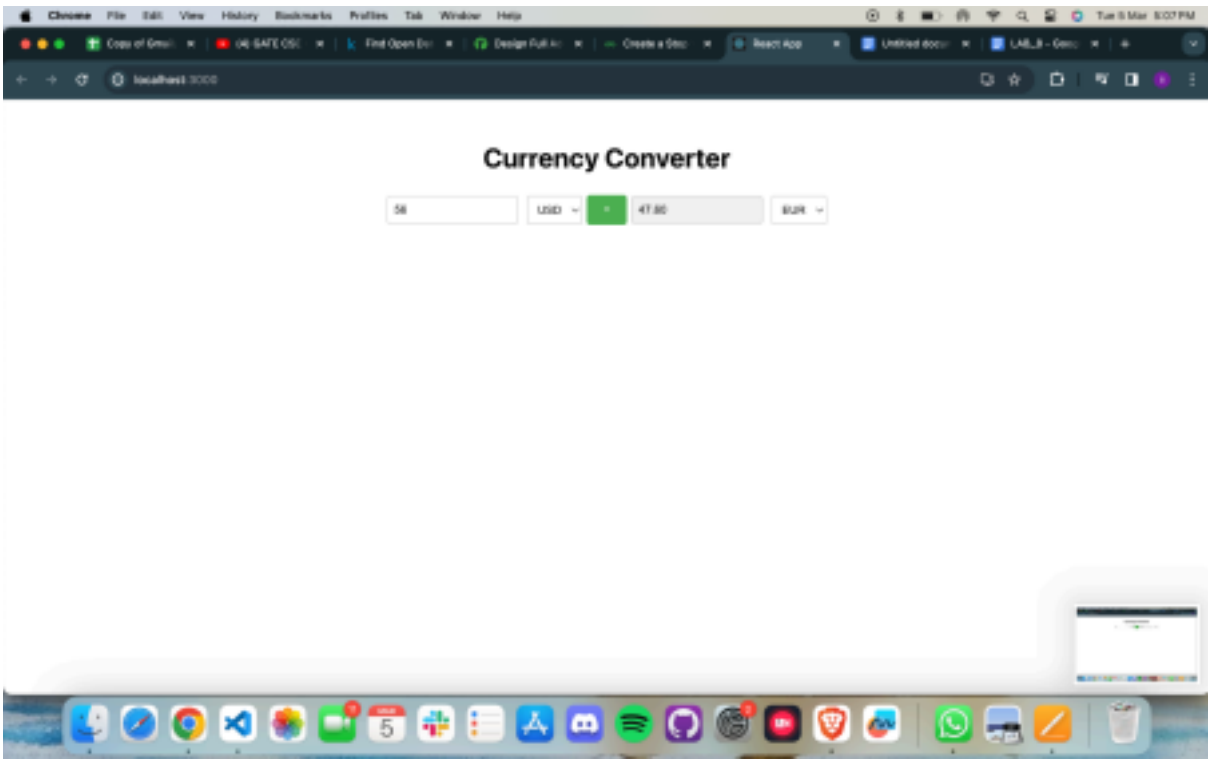
```css
/* App.css */

.App {

 text-align: center;

 margin-top: 50px;

}


.converter {

 display: flex;

 align-items: center;

 justify-content: center;

 margin-top: 20px;

}


input[type='number'],

input[type='text'],

select {

 margin: 5px;

 padding: 8px;

 border: 1px solid #ccc;
 border-radius: 4px;

}


button {

 background-color: #4caf50;

 color: white;

 border: none;

 border-radius: 4px;

 cursor: pointer;

 padding: 10px 20px;

}
```

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

input[type='number']::-webkit-inner-spin-button {
 -webkit-appearance: none;
 margin: 0;
}

input[type='number'] {
 -moz-appearance: textfield;
}

input[readonly] {
 background-color: #f0f0f0;
}
```




Currency Converter
Enter amount
USD
=
Converted amount
EUR
GBP
JPY


Currency Converter


Currency Converter

Q2.

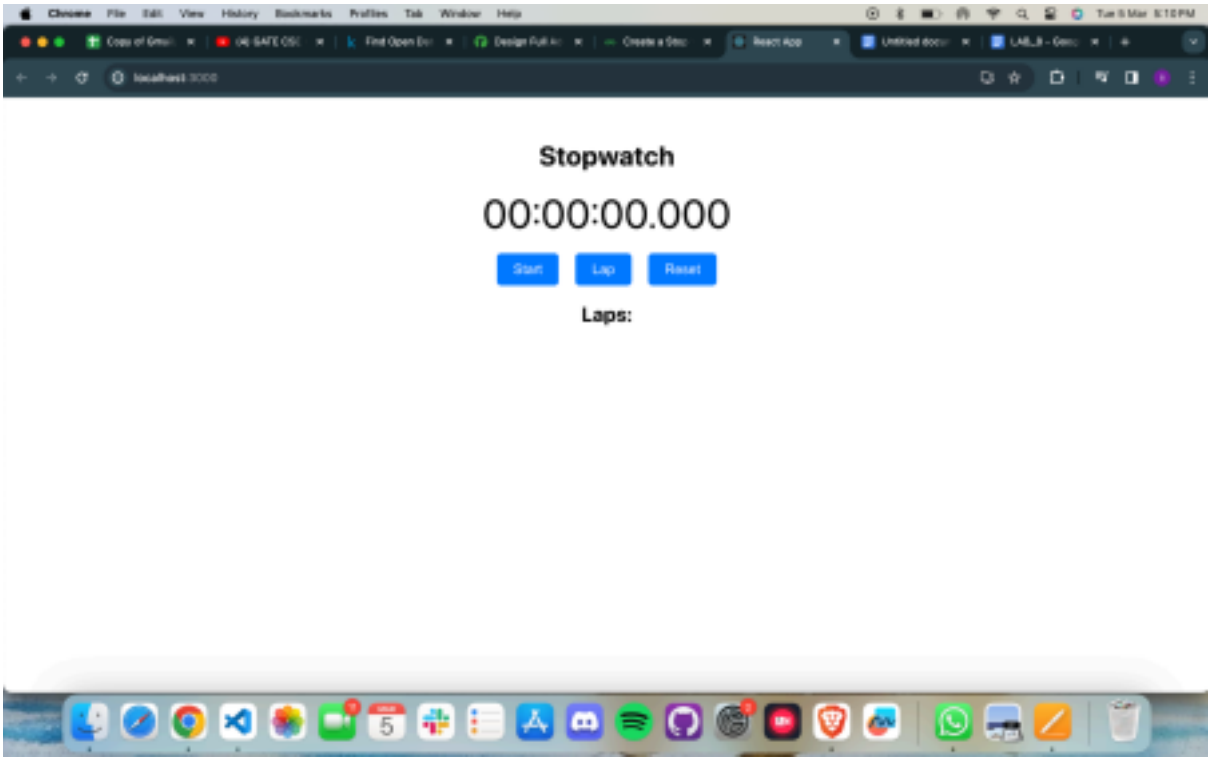
```
// App.js
import React, { useState, useEffect } from 'react';
import './App.css';

function App() {
 const [time, setTime] = useState(0);
 const [isRunning, setIsRunning] = useState(false);
 const [laps, setLaps] = useState([]);

 useEffect(() => {
   let intervalId;

   if (isRunning) {
     intervalId = setInterval(() => {
       setTime(prevTime => prevTime + 10); // Increment by 10
     milliseconds }, 10);
   } else {
     clearInterval(intervalId);
   }

   return () => clearInterval(intervalId);
 }, [isRunning]);
 const handleStartPause = () => {
   setIsRunning(prevIsRunning => !prevIsRunning);
 };

 const handleReset = () => {
   setTime(0);
   setIsRunning(false);
   setLaps([]);
 };
```

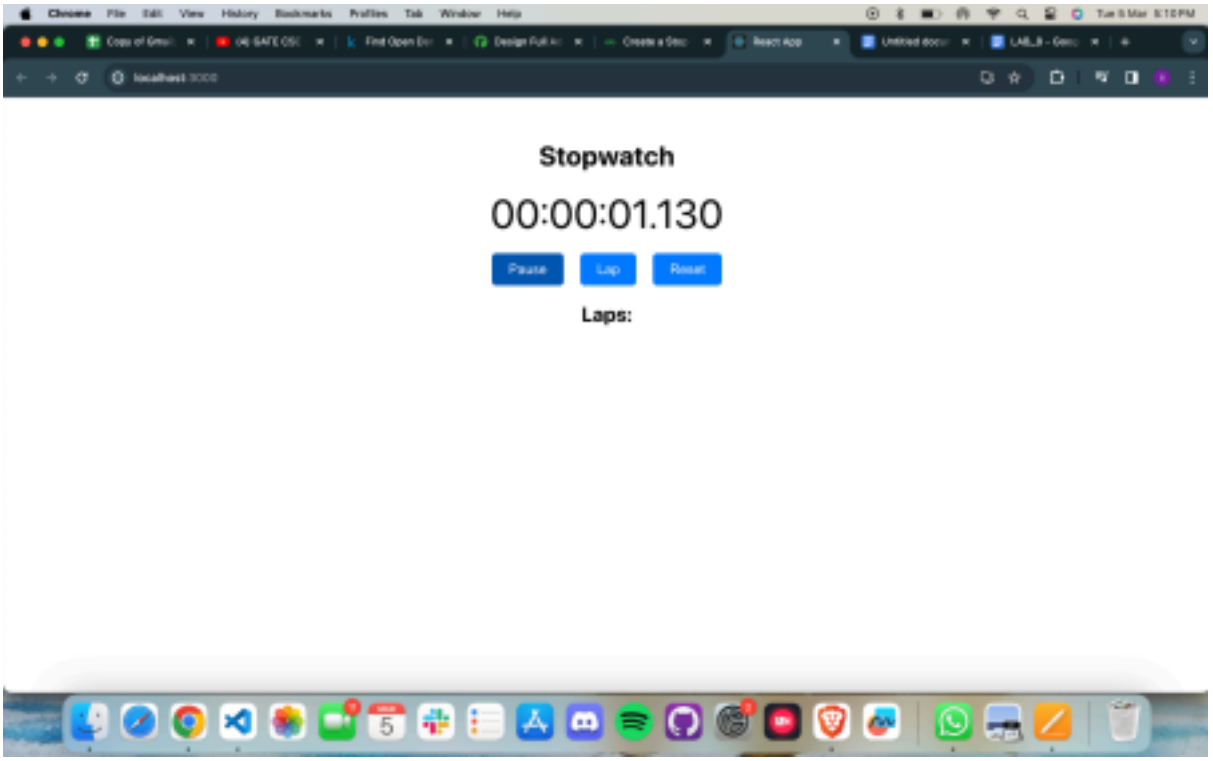
```
const handleLap = () => {
  setLaps(prevLaps => [...prevLaps, time]);
};

const formatTime = timeInMilliseconds => {
  const milliseconds = timeInMilliseconds % 1000;
  const seconds = Math.floor(timeInMilliseconds / 1000) % 60;
  const minutes = Math.floor(timeInMilliseconds / (1000 * 60)) %
  60; const hours = Math.floor(timeInMilliseconds / (1000 * 60 *
  60));

  const formattedHours = String(hours).padStart(2, '0');
  const formattedMinutes = String(minutes).padStart(2, '0');
  const formattedSeconds = String(seconds).padStart(2, '0');
  const formattedMilliseconds = String(milliseconds).padStart(3, '0');

  return
`${formattedHours}:${formattedMinutes}:${formattedSeconds}.${formattedMilliseconds
} `;
};

return (
  <div className="App">
    <h1>Stopwatch</h1>
    <div className="time">{formatTime(time)}</div>
    <div className="buttons">
      <button onClick={handleStartPause}>{isRunning ? 'Pause' :
      'Start'}</button> <button onClick={handleLap}
      disabled={!isRunning}>Lap</button> <button
      onClick={handleReset}>Reset</button>
    </div>
    <div className="laps">
```

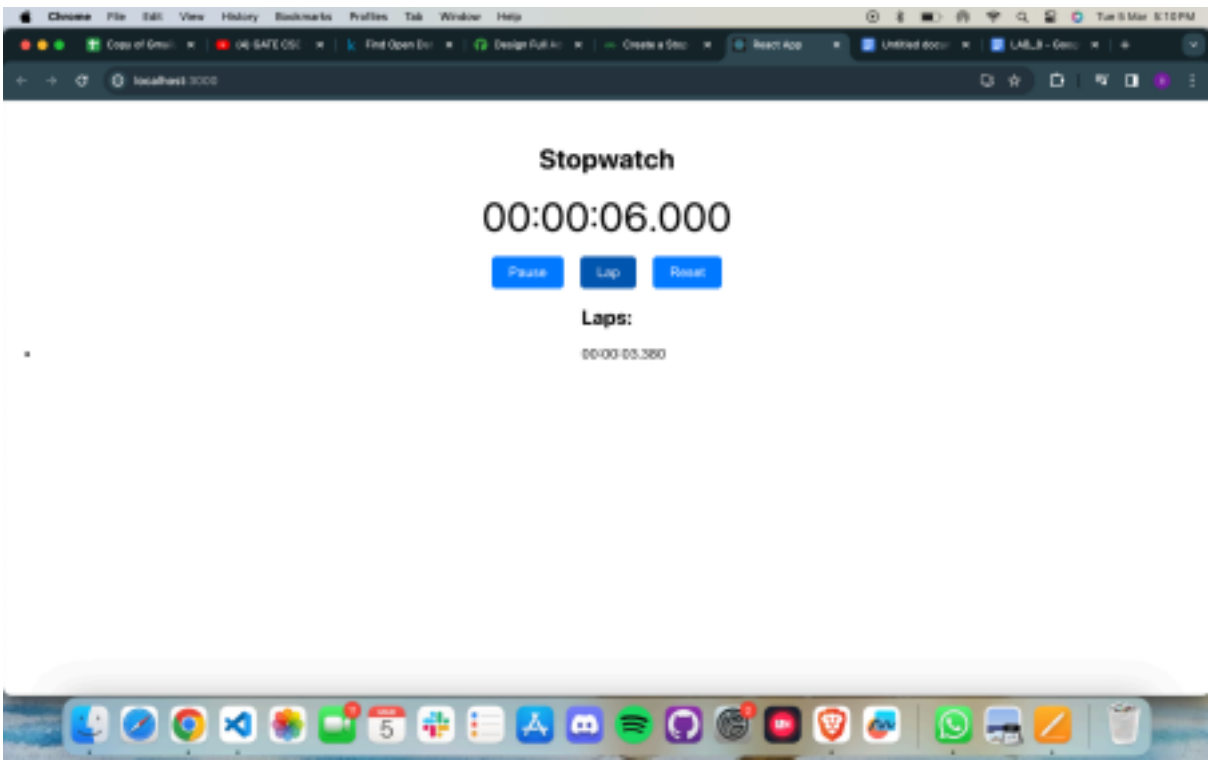
```
        <h2>Laps:</h2>
        <ul>
          {laps.map((lap, index) => (
            <li key={index}>{formatTime(lap)}</li>
          ))}
        </ul>
      </div>
    </div>
  );
}

export default App;
```

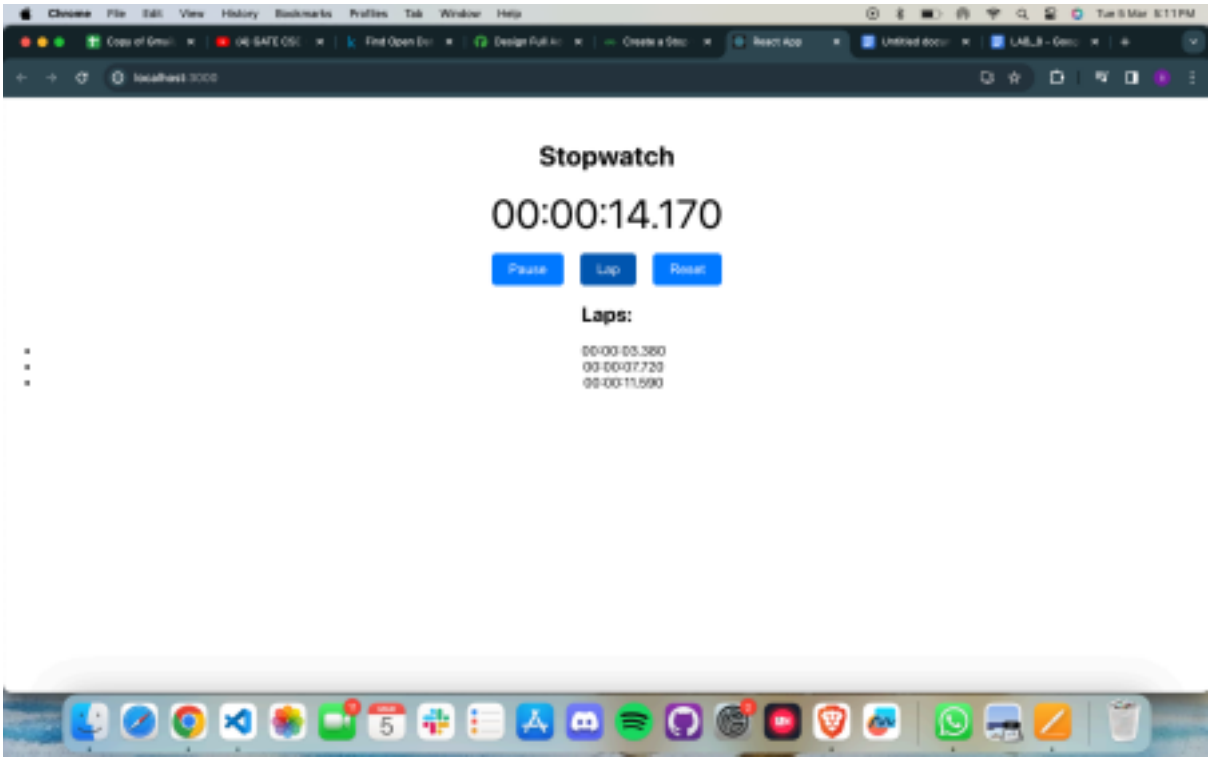
```
/* App.css */
.App {
  text-align: center;
  margin-top: 50px;
}
.time {
  font-size: 3em;
}

.buttons {
  margin-top: 20px;
}

button {
  font-size: 1em;
  padding: 10px 20px;
  margin: 0 10px;
  cursor: pointer;
```

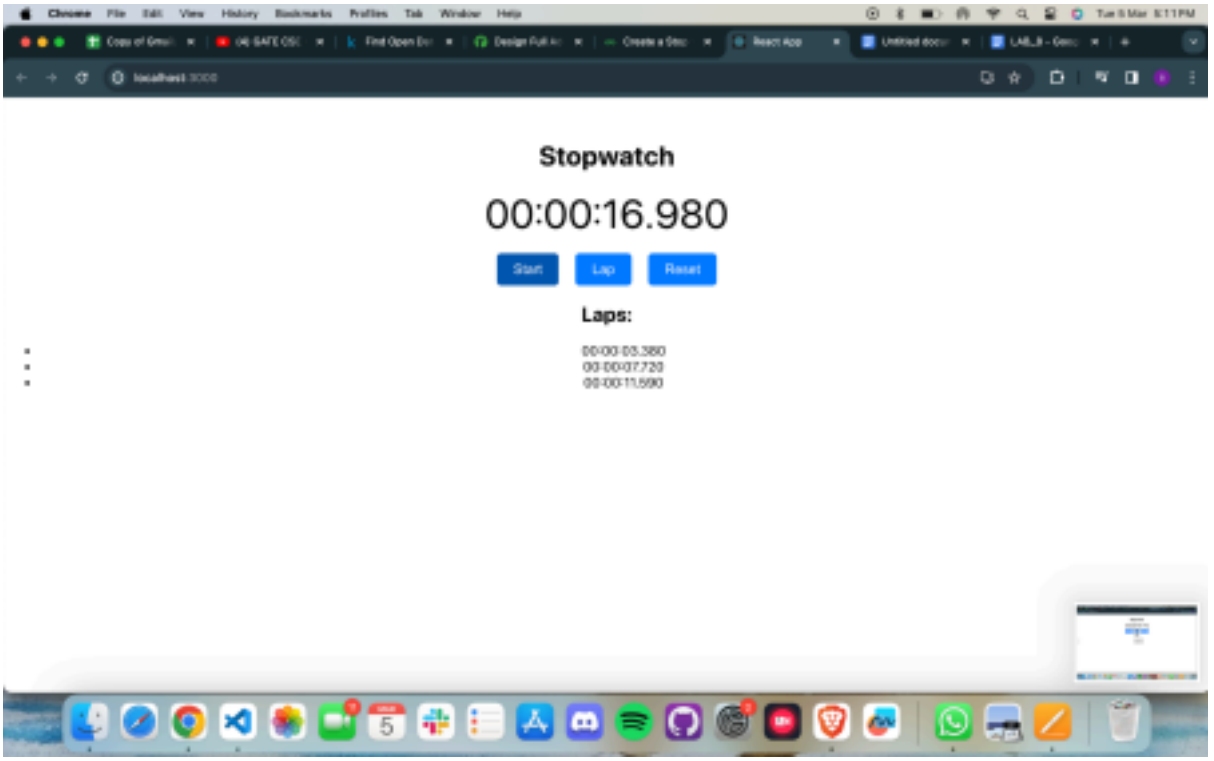
```
  border: none;
  border-radius: 5px;
  background-color: #007bff;
  color: white;
}

button:hover {
  background-color: #0056b3;
}
```




Stopwatch
00:00:01.130
Pause
Lap
Reset
Laps:


Stopwatch
00:00:06.000
Pause
Lap
Reset
Laps:
00:00:03.380





Q3.

```
// App.js

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

import './App.css';


function App() {

 const [conversations, setConversations] = useState([]);

 const [selectedConversation, setSelectedConversation] = useState(null);
```

```
const [newMessage, setNewMessage] = useState('');

useEffect(() => {
  // Simulating fetching conversations from backend
  // In a real scenario, you would make an API call to fetch
  conversations const sampleConversations = [
    { id: 1, name: 'Friend1', messages: ['Hello', 'Hi there!'] },
    { id: 2, name: 'Friend2', messages: ['Hey', 'How are you?'] },
    // Add more sample conversations if needed
  ];
  setConversations(sampleConversations);
}, []);

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

const handleSendMessage = () => {
  if (newMessage.trim() === '') return;
  const updatedConversations = conversations.map((conv) => {
    if (conv.id === selectedConversation.id) {
      return {
        ...conv,
        messages: [...conv.messages, { text: newMessage, sender: 'user' }],
      };
    }
    return conv;
  });
  setConversations(updatedConversations);
  setNewMessage('');
};

return (
```

```
    <div className="App">
      <div className="sidebar">
        <h2>Conversations</h2>
        <ul>
          {conversations.map((conversation) => (
            <li
              key={conversation.id}
              className={selectedConversation && selectedConversation.id
=== conversation.id ? 'active' : ''}
              onClick={() => handleConversationClick(conversation)}
            >
              {conversation.name}
            </li>
          ))}
        </ul>
      </div>
      <div className="chatbox">
        {selectedConversation ? (
          <>
            <div className="chat-header">
              <h2>{selectedConversation.name}</h2>
            </div>
            <div className="messages">
              {selectedConversation.messages.map((message, index) => (
                <div key={index} className={`message ${message.sender}`}>
                  {message.text}
                </div>
              ))}
            </div>
            <div className="input">
              <input
                type="text"
                placeholder="Type your message..."
                value={newMessage}
```

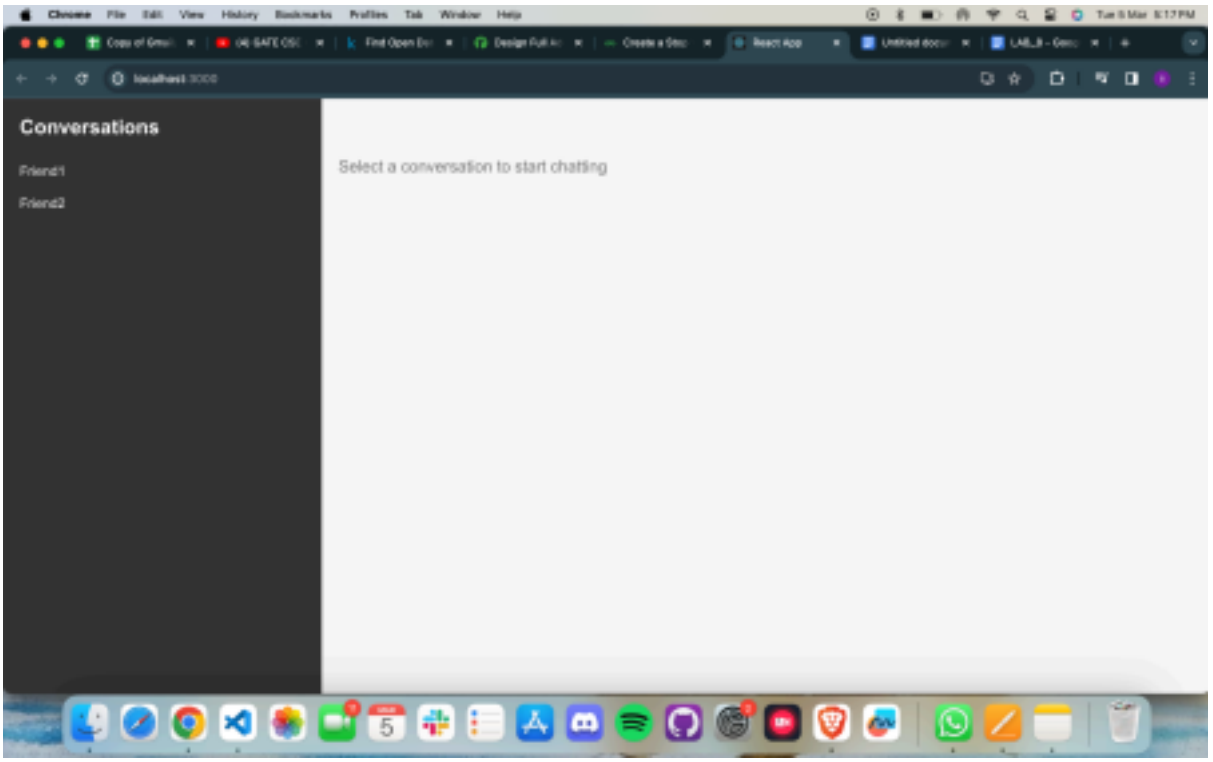
```
                onChange={(e) => setNewMessage(e.target.value)}
              />
              <button onClick={handleSendMessage}>Send</button>
            </div>
          </>
        ) : (
          <p className="no-conversation">Select a conversation to start chatting</p>
        )}
      </div>
    </div>
  );
}

export default App;
/* App.css */
body {
  font-family: Arial,
 sans-serif; margin: 0;
 padding: 0;
 background-color: #f5f5f5;
}

.App {
 display: flex;
 height: 100vh;
}

.sidebar {
 flex: 1;
 background-color: #333;
 color: white;
 padding: 20px;
}
```

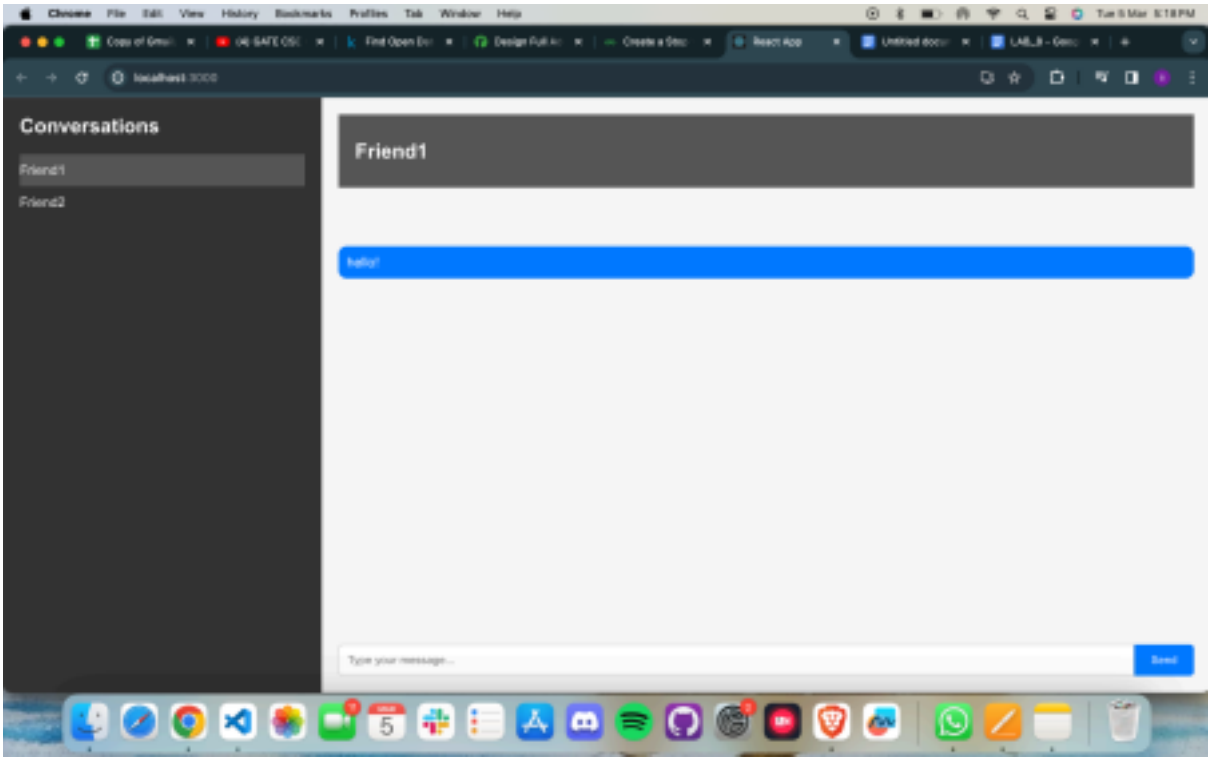
```
.sidebar h2 {
 margin-top: 0;
 font-size: 1.5rem;
}

ul {
 list-style-type: none;
 padding: 0;
}
ul li {
 cursor: pointer;
 padding: 10px 0;
}

ul li.active {
 background-color: #555;
}

.chatbox {
 flex: 3;
 padding: 20px;
 display: flex;
 flex-direction:
column; }

.chat-header {
 background-color: #555;
 color: white;
 padding: 10px 20px;
}

.messages {
```

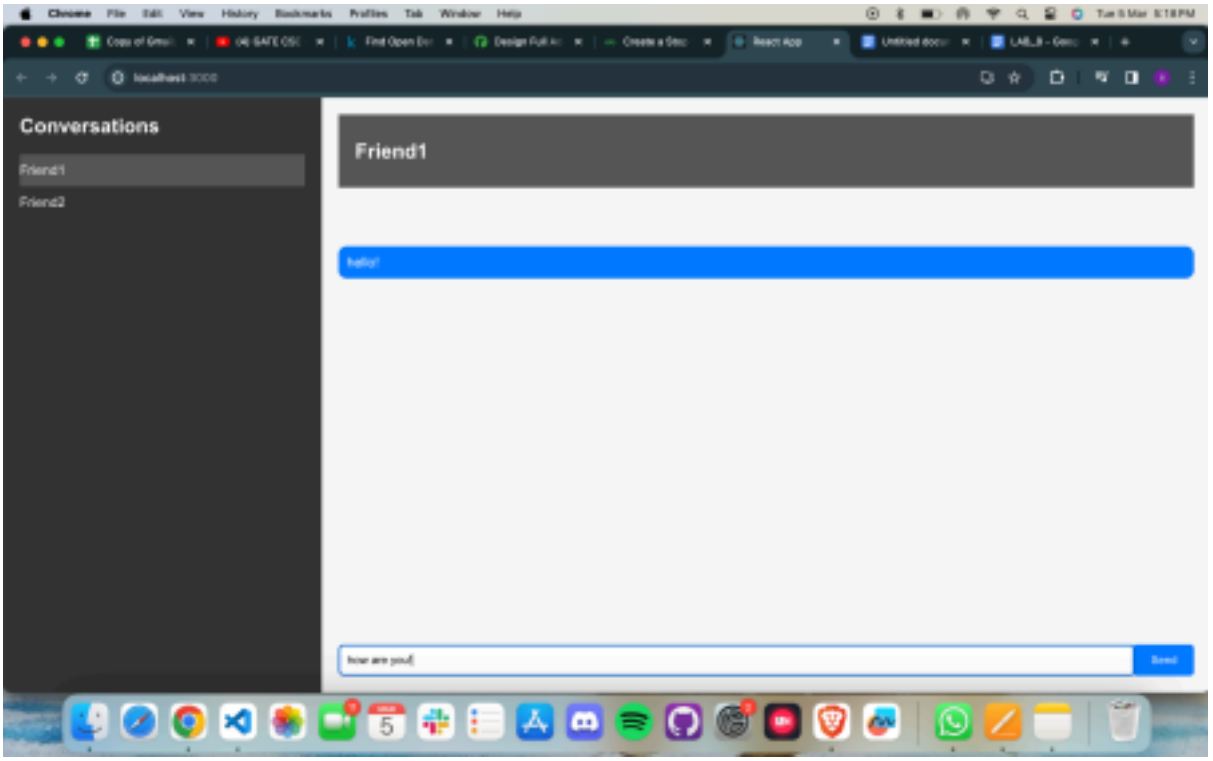
```
 flex: 1;
 overflow-y:
 scroll; margin-top:
 10px;
}

.message {
 background-color: #f9f9f9;
 padding: 10px;
 border-radius: 10px;
 margin-bottom: 10px;
}

.message.user {
 align-self: flex-end;
 background-color: #007bff;
 color: white;
}

.message.user::after {
 content: '';
 position: absolute;
 width: 0;
 height: 0;
 border-top: 8px solid transparent;
 border-bottom: 8px solid transparent;
 border-right: 8px solid #007bff;
 left: -8px;
 top: 50%;
 margin-top: -8px;
}

.message:not(.user)::after {
```

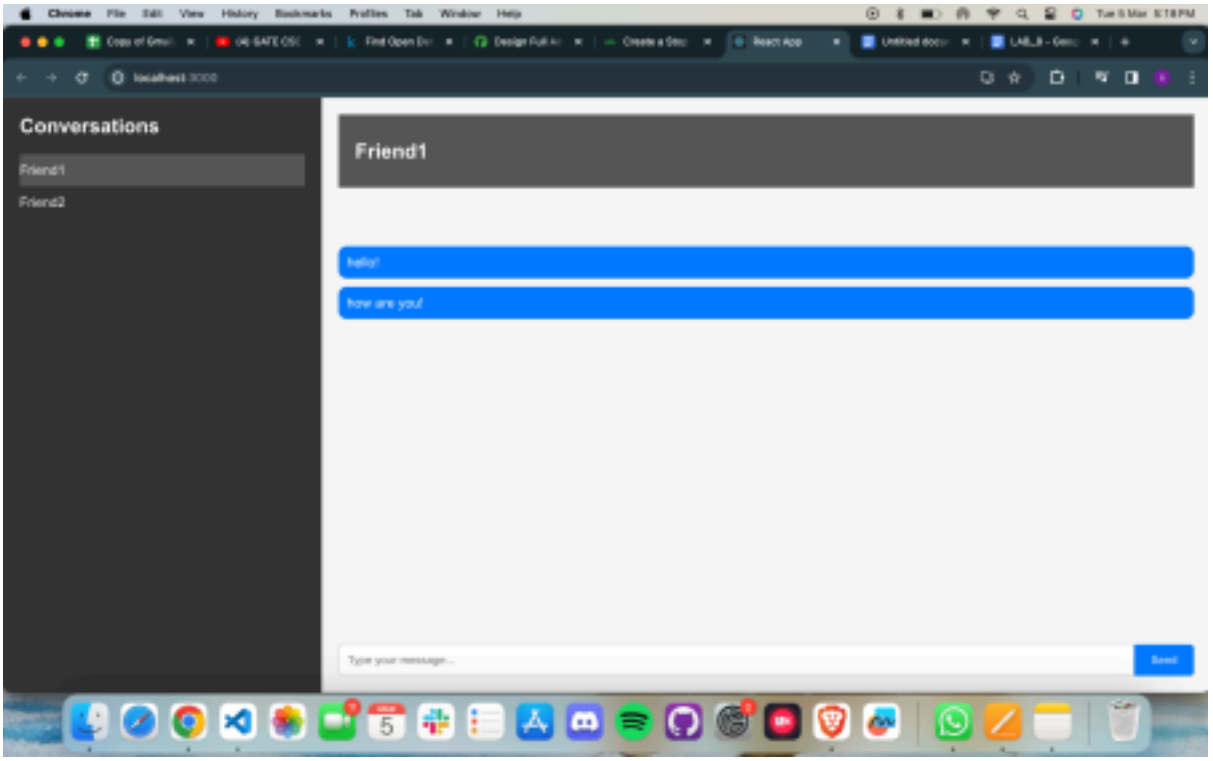
```
  content: '';
  position: absolute;
  width: 0;
  height: 0;
  border-top: 8px solid transparent;
  border-bottom: 8px solid transparent;
  border-left: 8px solid #f9f9f9;
  right: -8px;
  top: 50%;
  margin-top: -8px;
}

.input {
  display: flex;
  margin-top: 10px;
}

.input input {
  flex: 1;
  padding: 10px;
  border-radius: 5px 0 0 5px;
  border: 1px solid #ccc;
}

.input button {
  padding: 10px 20px;
  border: none;
  border-radius: 0 5px 5px 0;
  background-color: #007bff;
  color: white;
  cursor: pointer;
}
```

```
.input button:hover {
 background-color: #0056b3;

}


.no-conversation {

 margin-top: 50px;

 font-size: 1.2rem;

 color: #777;

}
```




Conversations
Friend1
Friend2
Friend1
hello!
Type your message...
Send


Conversations
Friend1
Friend2
Friend1
hello!
how are you|
Send


Conversations
Friend1
Friend2
Friend1
hello!
how are you!
Type your message...
Send