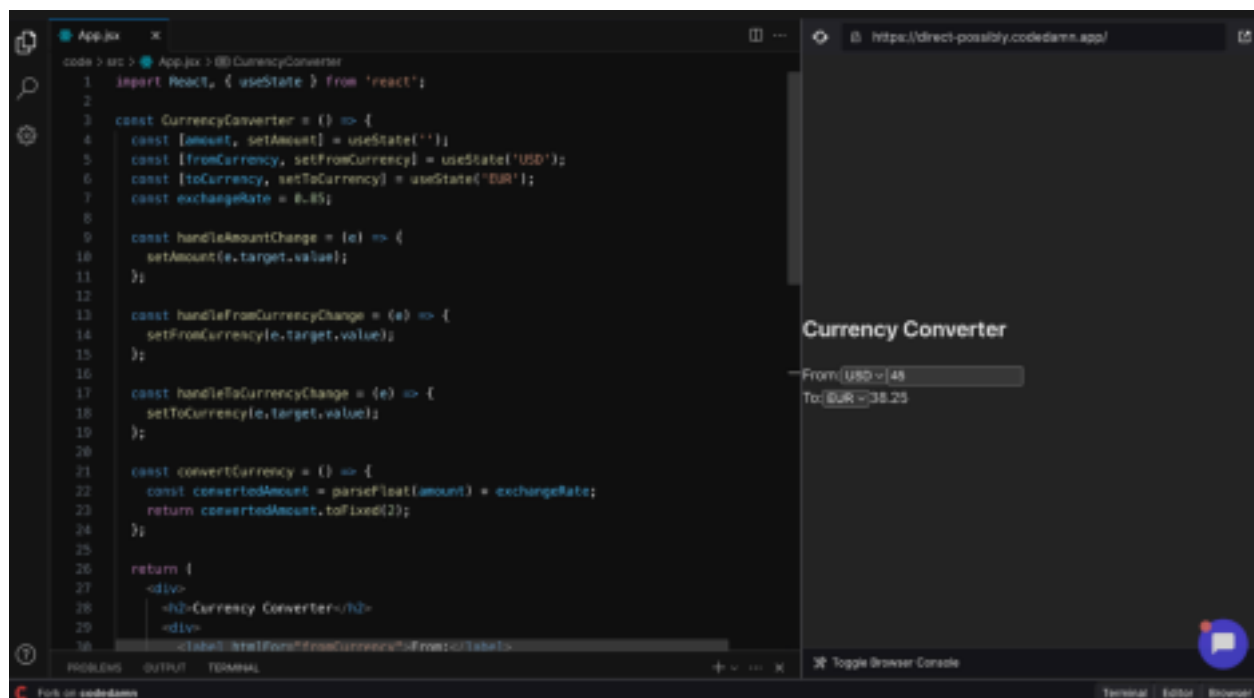


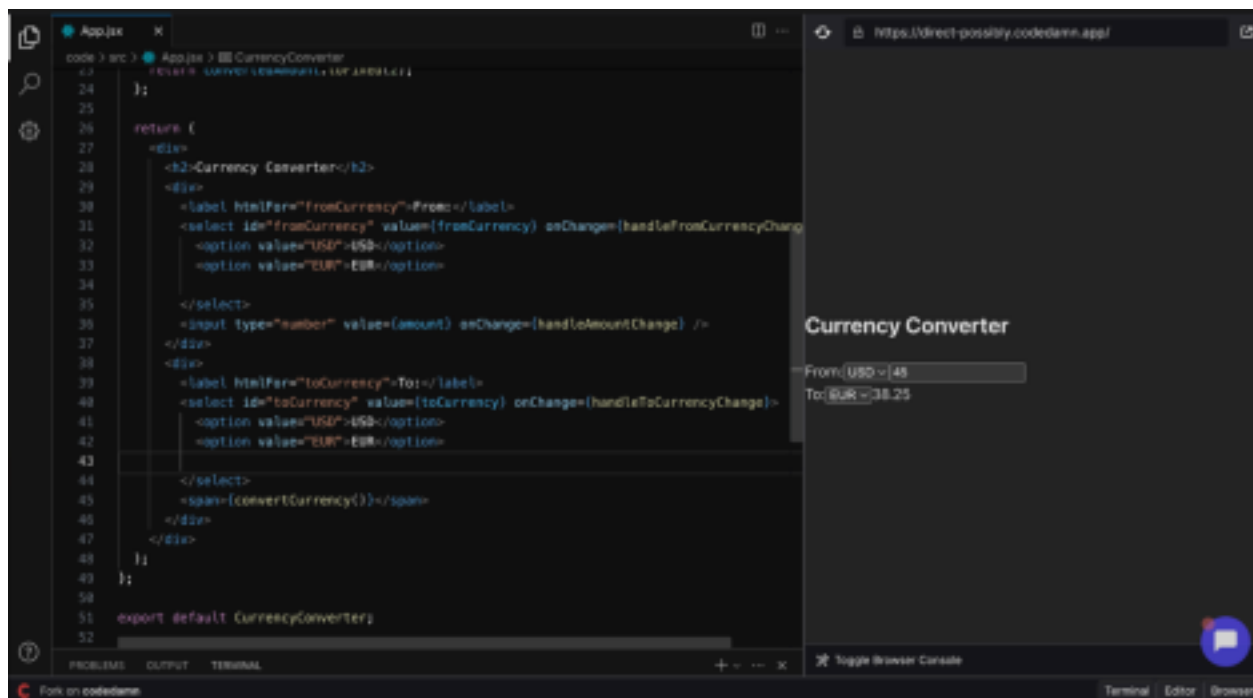
NAME : ARYAN KHAIWAL

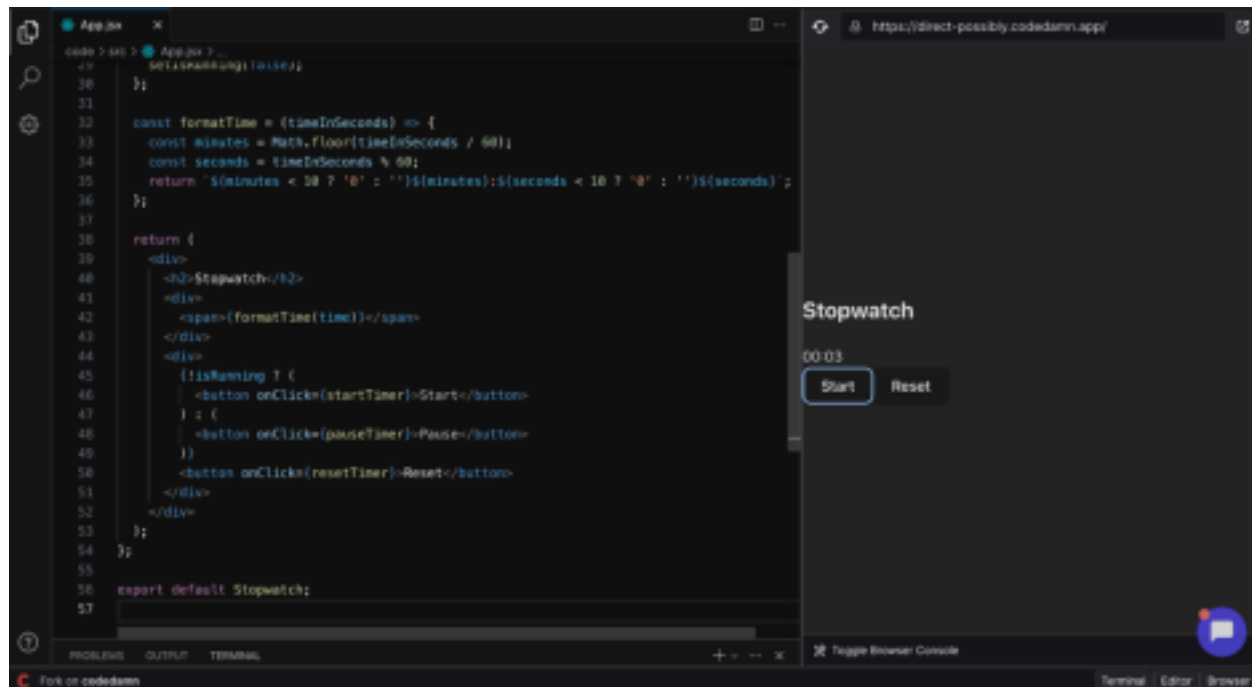
ROLL NO. : 22cs2029

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.



```
code > src > App.jsx > @ CurrencyConverter
1  import React, { useState } from 'react';
2
3  const CurrencyConverter = () => {
4    const [amount, setAmount] = useState('');
5    const [fromCurrency, setFromCurrency] = useState('USD');
6    const [toCurrency, setToCurrency] = useState('EUR');
7    const exchangeRate = 0.85;
8
9    const handleAmountChange = (e) => {
10     setAmount(e.target.value);
11   };
12
13   const handleFromCurrencyChange = (e) => {
14     setFromCurrency(e.target.value);
15   };
16
17   const handleToCurrencyChange = (e) => {
18     setToCurrency(e.target.value);
19   };
20
21   const convertCurrency = () => {
22     const convertedAmount = parseFloat(amount) * exchangeRate;
23     return convertedAmount.toFixed(2);
24   };
25
26   return (
27     <div>
28       <h2>Currency Converter</h2>
29       <div>
30         <label>From</label> <input type="text" value={fromCurrency} />
31         <label>To</label> <input type="text" value={toCurrency} />
32         <button>Convert</button>
33       </div>
34       <div>
35         <input type="text" value={amount} />
36         <input type="text" value={convertedAmount} />
37       </div>
38     </div>
39   );
40 }
```





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.

```
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>

```

```
        </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;
```

```
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%;
```









