Persistent Input

INTERVIEW QUESTIONS-112

Follow on in

@DUVVURU KISHORE

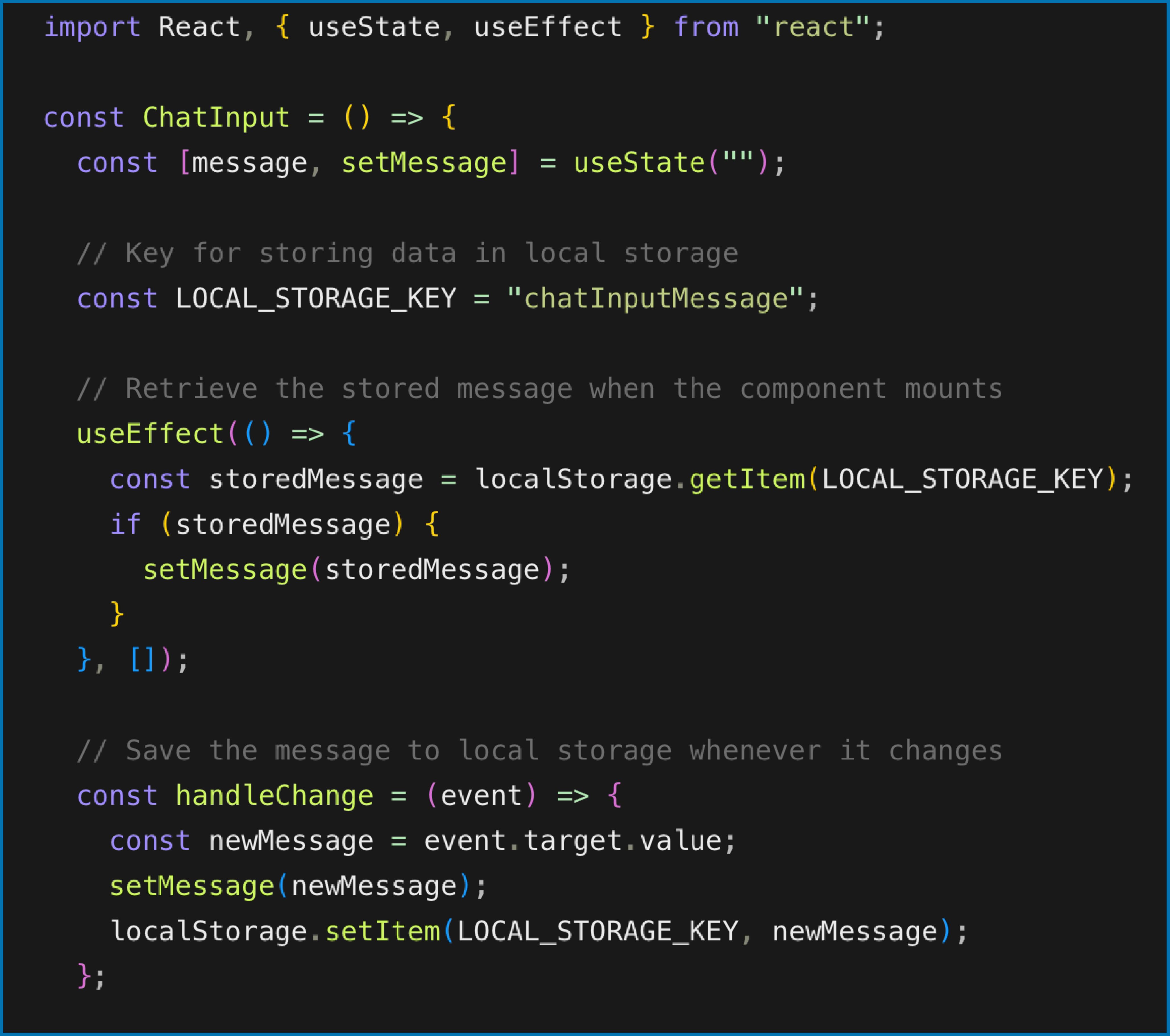
Interviewer

In applications like WhatsApp or Jira, when a user types a message or comment and then navigates away from the page, the input is still present when they return or on refresh. How can you develop this feature in a React application? What mechanisms ensure that the data persists even after closing the chat or the browser?

Answer

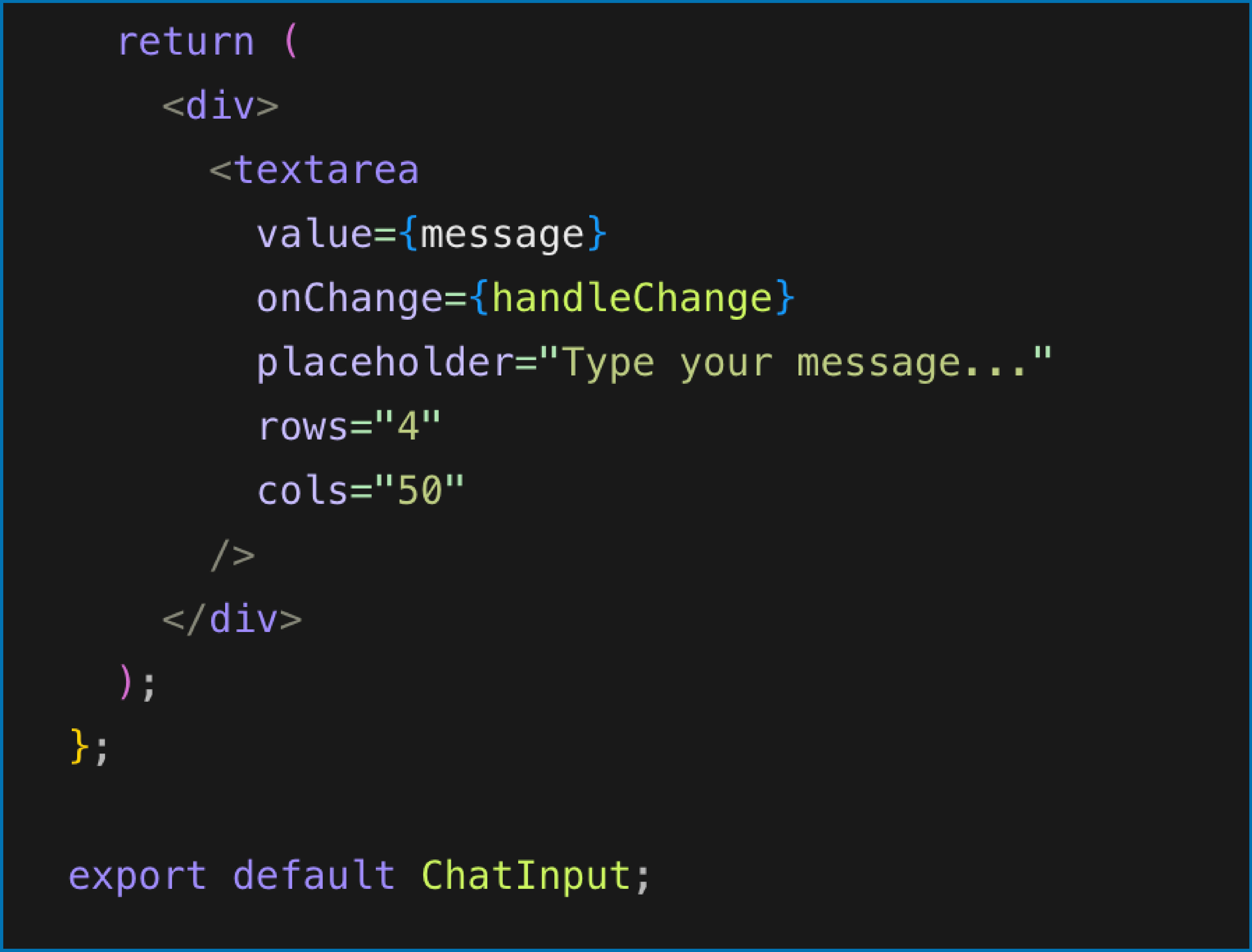
To develop a feature where the typed message or comment persists even after navigating away or closing the chat application, you can use browser storage options such as localStorage or sessionStorage

Implementation



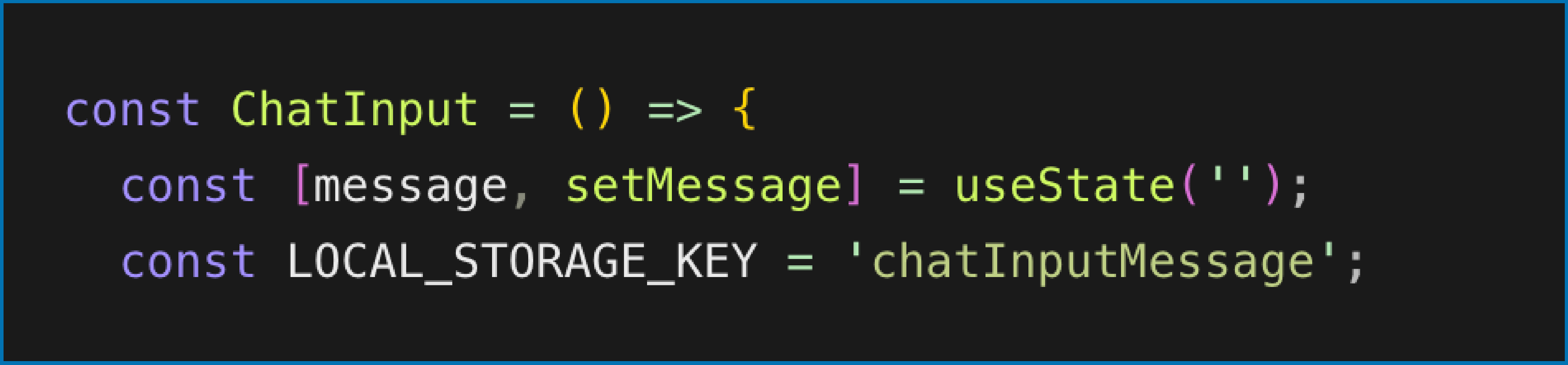
```
import React, { useState, useEffect } from "react";

const ChatInput = () => {
  const [message, setMessage] = useState("");

  // Key for storing data in local storage
  const LOCAL_STORAGE_KEY = "chatInputMessage";

  // Retrieve the stored message when the component mounts
  useEffect(() => {
    const storedMessage = localStorage.getItem(LOCAL_STORAGE_KEY);
    if (storedMessage) {
      setMessage(storedMessage);
    }
  }, []);

  // Save the message to local storage whenever it changes
  const handleChange = (event) => {
    const newMessage = event.target.value;
    setMessage(newMessage);
    localStorage.setItem(LOCAL_STORAGE_KEY, newMessage);
  };
```

explanation for the code



```
  return (
    <div>
      <textarea
        value={message}
        onChange={handleChange}
        placeholder="Type your message..."
        rows="4"
        cols="50"
      />
    </div>
  );
};

export default ChatInput;
```

Explanation

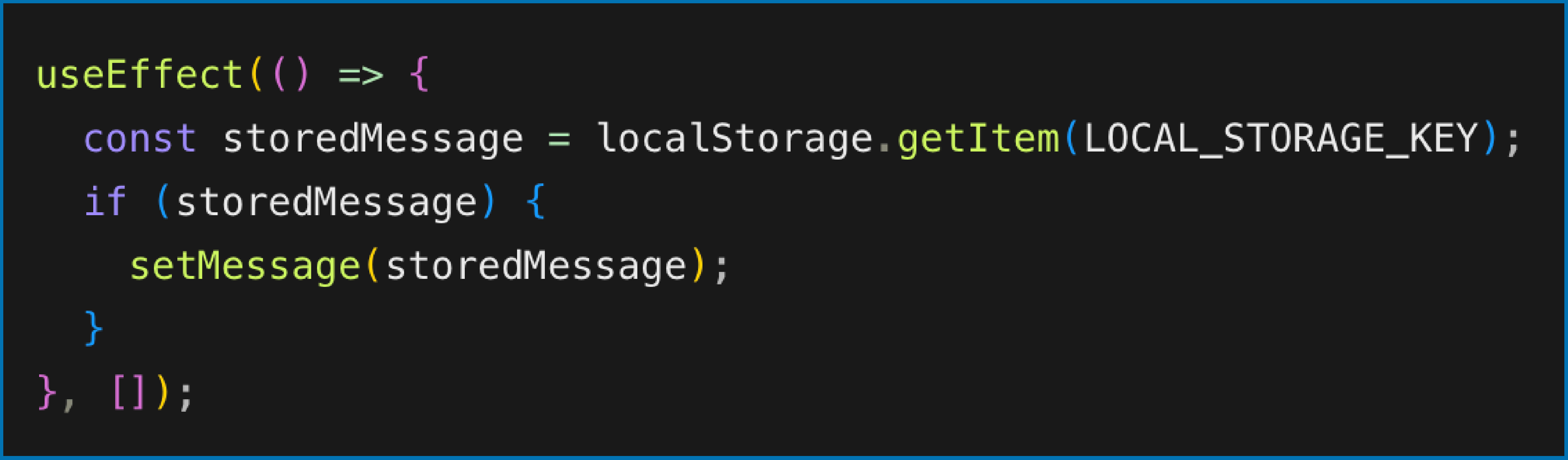


```
const ChatInput = () => {
  const [message, setMessage] = useState('');
  const LOCAL_STORAGE_KEY = 'chatInputMessage';
```

- **ChatInput** is a functional React component.

- message is a state variable initialized with an empty string. **setMessage** is the function to update message.

- **LOCAL_STORAGE_KEY** is a constant string key used to store and retrieve the message in/from local storage.

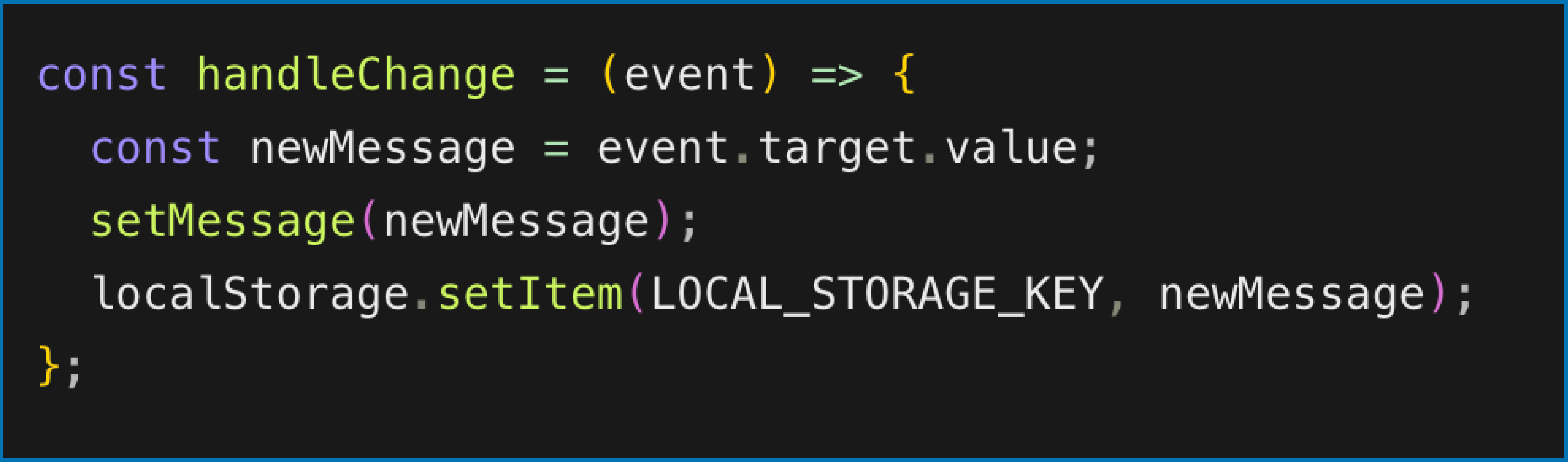


```
useEffect(() => {
  const storedMessage = localStorage.getItem(LOCAL_STORAGE_KEY);
  if (storedMessage) {
    setMessage(storedMessage);
  }
}, []);
```

- This useEffect hook runs once when the component mounts (due to the empty dependency array []).

- It attempts to retrieve a stored message from localStorage using the key LOCAL_STORAGE_KEY.

- If a stored message is found, it updates the message state with this value using setMessage.



```
const handleChange = (event) => {
  const newMessage = event.target.value;
  setMessage(newMessage);
  localStorage.setItem(LOCAL_STORAGE_KEY, newMessage);
};
```

- **handleChange** is a function that gets called whenever the user types in the textarea.

- It receives an **event** object, event, which contains the new value of the **textarea** (event.target.value).

- The function updates the **message** state with the new value and also saves this value to **localStorage** using the same key, **LOCAL_STORAGE_KEY**.

By using this approach, the message will be saved locally on the user's device, ensuring persistence between sessions unless the user clears their browser data or specifically removes the stored item.

Follow on in

@Duvvuru Kishore