

Assignment-2

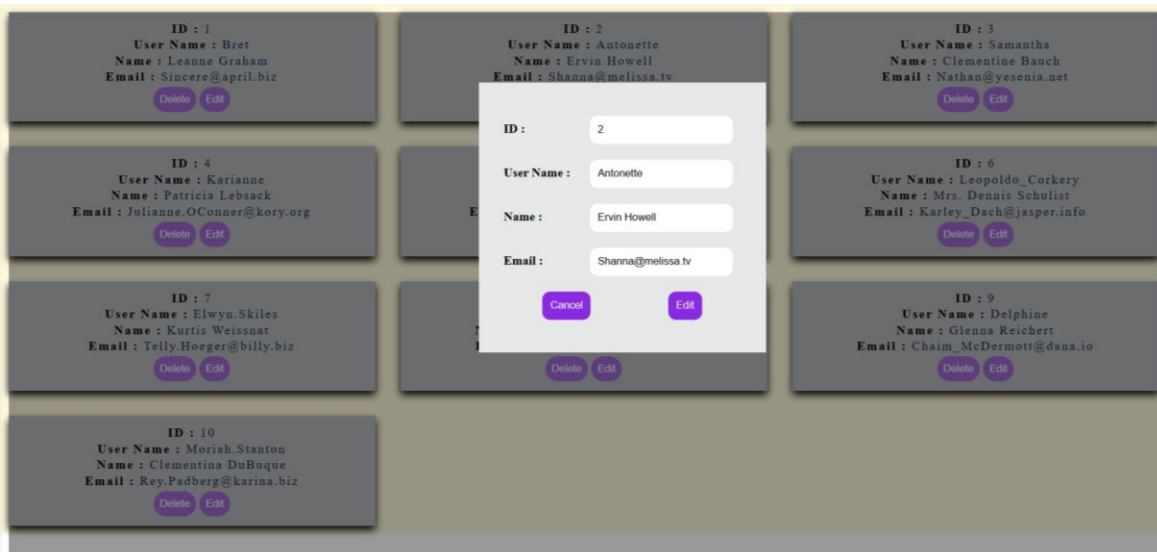
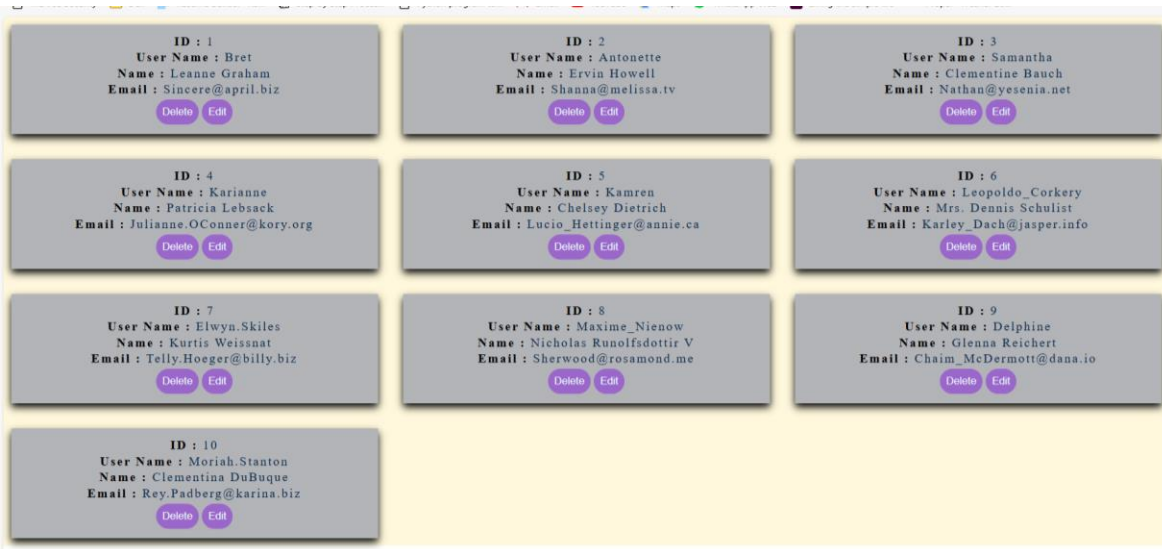
K.Krishna Pavan Kumar

208X1A0538

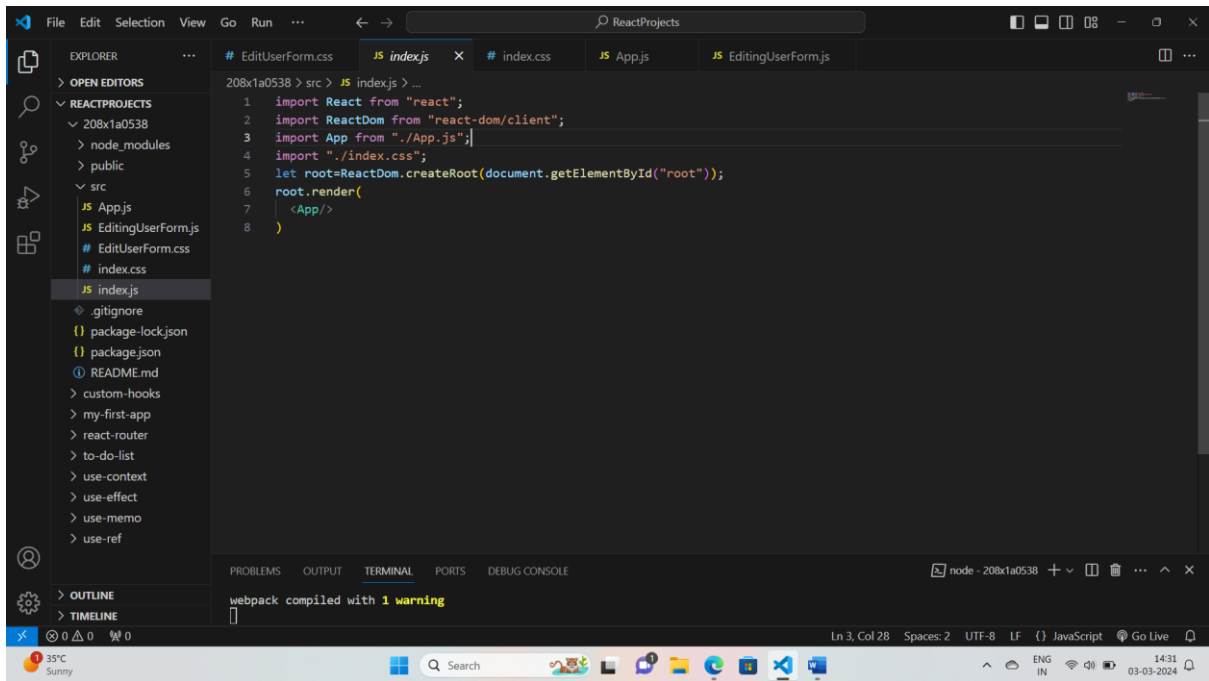
Kallam Haranadhareddy Institute of
Technology

- In this react application I had fetched users data through an api in App component.
- We can edit and delete the users data.
- I have used useEffect , useReducer hooks in the application.
- There are two components in this application.

Output



Index.js

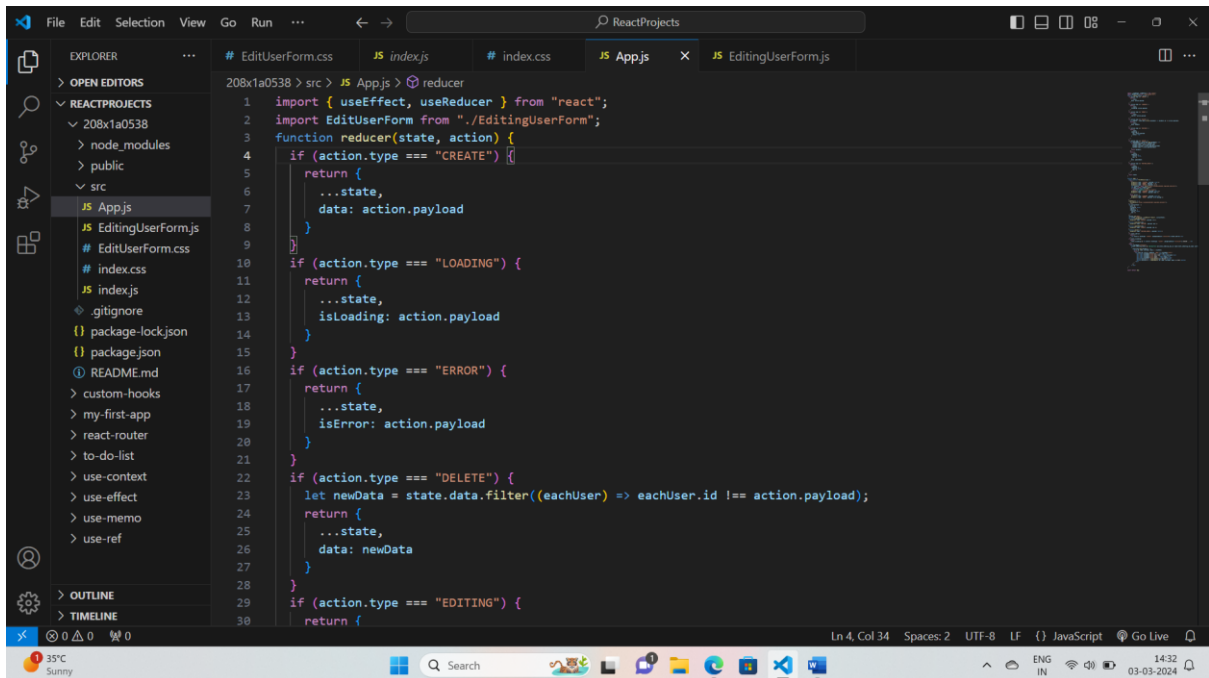


The screenshot shows the VS Code editor with the file explorer on the left. The file explorer shows a project structure with a 'src' directory containing 'App.js', 'EditingUserForm.js', 'EditUserForm.css', 'index.css', and 'index.js'. The 'index.js' file is selected and its content is displayed in the editor. The terminal at the bottom shows 'webpack compiled with 1 warning'.

```
1 import React from "react";
2 import ReactDOM from "react-dom/client";
3 import App from "../App.js";
4 import "../index.css";
5 let root=ReactDOM.createRoot(document.getElementById("root"));
6 root.render(
7   <App/>
8 )
```

Ln 3, Col 28 Spaces: 2 UTF-8 LF JavaScript Go Live

App.js



The screenshot shows the VS Code editor with the file explorer on the left. The file explorer shows a project structure with a 'src' directory containing 'App.js', 'EditingUserForm.js', 'EditUserForm.css', 'index.css', and 'index.js'. The 'App.js' file is selected and its content is displayed in the editor. The terminal at the bottom shows 'webpack compiled with 1 warning'.

```
1 import { useEffect, useReducer } from "react";
2 import EditUserForm from "../EditingUserForm";
3 function reducer(state, action) {
4   if (action.type === "CREATE") {
5     return {
6       ...state,
7       data: action.payload
8     }
9   }
10  if (action.type === "LOADING") {
11    return {
12      ...state,
13      isLoading: action.payload
14    }
15  }
16  if (action.type === "ERROR") {
17    return {
18      ...state,
19      isError: action.payload
20    }
21  }
22  if (action.type === "DELETE") {
23    let newData = state.data.filter((eachUser) => eachUser.id !== action.payload);
24    return {
25      ...state,
26      data: newData
27    }
28  }
29  if (action.type === "EDITING") {
30    return {
```

Ln 4, Col 34 Spaces: 2 UTF-8 LF JavaScript Go Live

```
function reducer(state, action) {
  if (action.type === "EDITING_CANCEL") {
    return {
      ...state,
      isEditing: {
        status: false,
        obj: {}
      }
    }
  }
  return state;
}

function App() {
  async function fetchDetails(api) {
    try {
      dispatch({ type: "LOADING", payload: true });
      dispatch({ type: "ERROR", payload: "" });
      dispatch({ type: "CREATE", payload: [] });
      let response = await fetch("https://jsonplaceholder.typicode.com/users");
      let data = await response.json();
      if (response.status === 404) {
        throw new TypeError("Data not found");
      }
      dispatch({ type: "LOADING", payload: false });
      dispatch({ type: "CREATE", payload: data });
    } catch (error) {
      dispatch({ type: "LOADING", payload: false });
      dispatch({ type: "ERROR", payload: error.message });
    }
  }
}
```

```
function reducer(state, action) {
  if (action.type === "EDITING") {
    return {
      ...state,
      isEditing: {
        obj: action.payload,
        status: true
      }
    }
  }
  if (action.type === "UPDATE") {
    let updatedData = state.data.map(eachUser => {
      if (action.payload.id === eachUser.id) {
        eachUser.name = action.payload.name;
        eachUser.username = action.payload.username;
        eachUser.email = action.payload.email;
      }
      return eachUser;
    });
    return {
      ...state,
      isEditing: {
        status: false,
        obj: {}
      },
      data: updatedData
    }
  }
  if (action.type === "EDITING_CANCEL") {

```

```
function App() {
  async function fetchDetails(api) {
    try {
      dispatch({ type: "LOADING", payload: false });
      dispatch({ type: "ERROR", payload: error.message });
    } catch (error) {
      dispatch({ type: "LOADING", payload: false });
      dispatch({ type: "ERROR", payload: error.message });
    }
  }

  useEffect(() => {
    fetchDetails("https://jsonplaceholder.typicode.com/users");
  }, []);

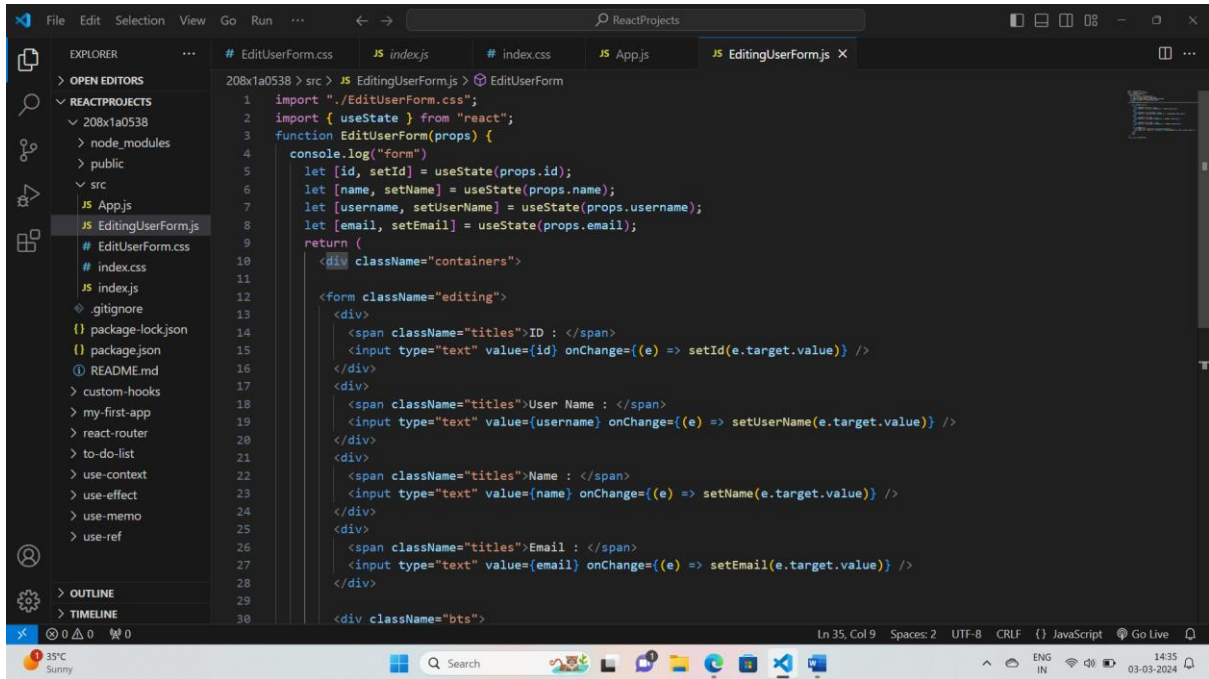
  let initialState = {
    data: [],
    isLoading: false,
    isError: "",
    isEditing: {
      status: false,
      obj: {}
    }
  };

  console.log("App");
  let [state, dispatch] = useReducer(reducer, initialState);
  function handleDelete(id) {
    dispatch({ type: "DELETE", payload: id });
  }
  function handleEdit(obj) {
    dispatch({ type: "EDITING", payload: obj });
  }
  function updateEdit(obj) {
    dispatch({ type: "UPDATE", payload: obj });
  }
}
```

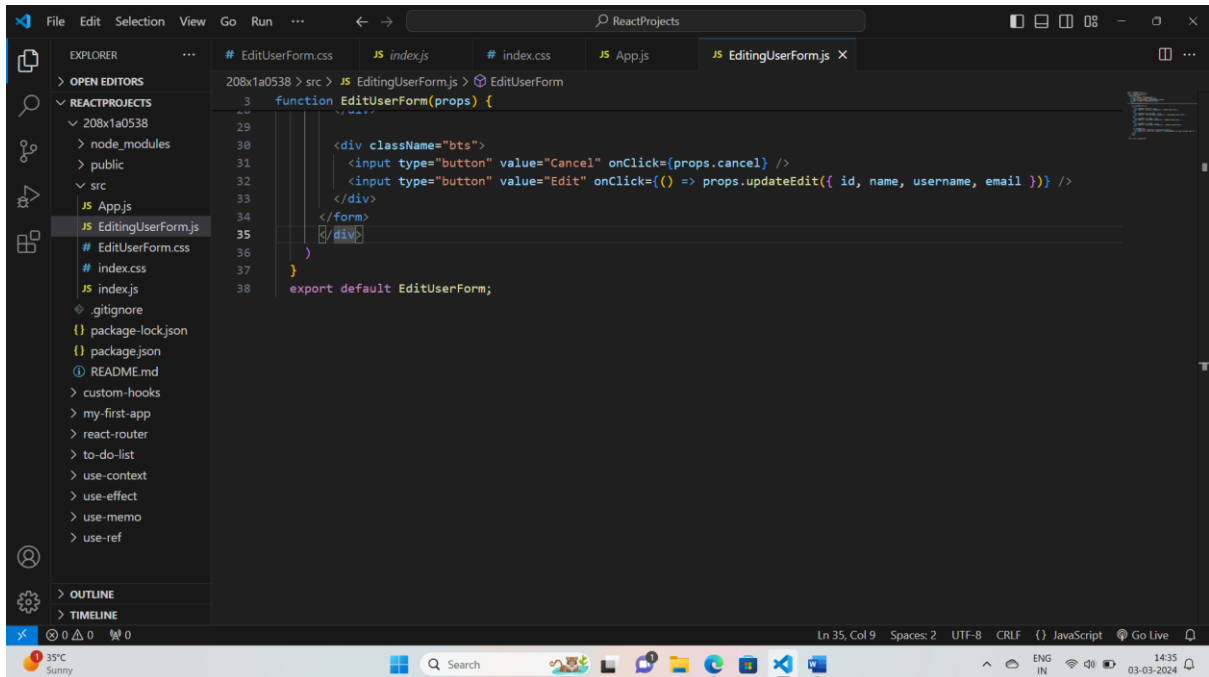
```
208x1a0538 > src > JS App.js > reducer
69 function App() {
104   function handleEdit(obj) {
105     dispatch({ type: "EDITING", payload: obj });
106   }
107   function updateEdit(obj) {
108     dispatch({ type: "UPDATE", payload: obj });
109   }
110   function cancel() {
111     dispatch({ type: "EDITING_CANCEL", payload: false });
112   }
113   if (state.isError)
114     return (
115       <h1 style={{ textAlign: "center", backgroundColor: "cornsilk" }}>{state.isError}</h1>
116     );
117   if (state.isLoading)
118     return (
119       state.isLoading && <h1 style={{ textAlign: "center", backgroundColor: "cornsilk" }}>LOADING ...</h1>
120     );
121   return (
122     <div className="container">
123       {state.isEditing.status && <EditUserForm id={state.isEditing.obj.id} name={state.isEditing.obj.name} username={
124         {
125           state.data.map((eachUser) => {
126             let { id, name, username, email } = eachUser;
127             return (
128               <div key={id} style={{ padding: "10px" }} className="users">
129                 <div><span className="title">ID : </span>{id}</div>
130                 <div><span className="title">User Name : </span>{username}</div>
131                 <div><span className="title">Name : </span>{name}</div>
132                 <div><span className="title">Email : </span>{email}</div>
```

```
208x1a0538 > src > JS App.js > reducer
69 function App() {
125   state.data.map((eachUser) => {
127     return (
128       <div key={id} style={{ padding: "10px" }} className="users">
129         <div><span className="title">ID : </span>{id}</div>
130         <div><span className="title">User Name : </span>{username}</div>
131         <div><span className="title">Name : </span>{name}</div>
132         <div><span className="title">Email : </span>{email}</div>
133         <button onClick={() => handleDelete(id)}>Delete</button>
134         <button onClick={() => handleEdit({ id, name, username, email })}>Edit</button>
135       </div>
136     );
137   });
138 }
139 }
140 }
141 }
142 export default App;
```

EditingUserForm.js

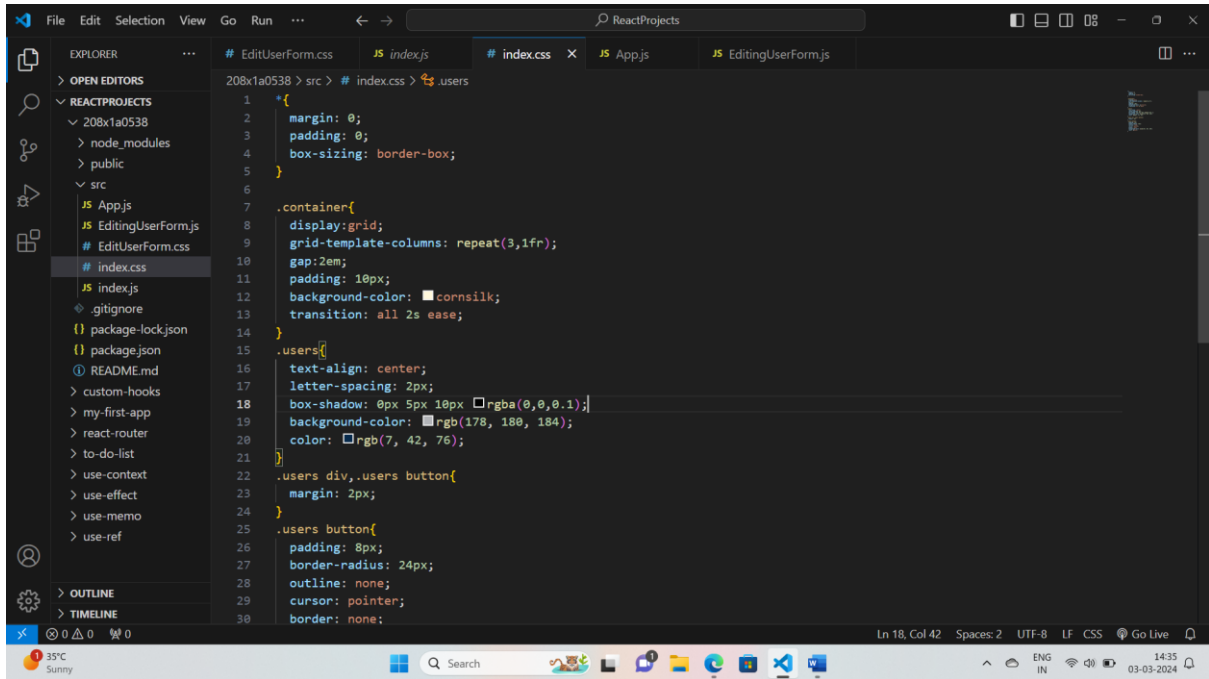


```
1 import './EditUserForm.css';
2 import { useState } from 'react';
3 function EditUserForm(props) {
4   console.log("form")
5   let [id, setId] = useState(props.id);
6   let [name, setName] = useState(props.name);
7   let [username, setUsername] = useState(props.username);
8   let [email, setEmail] = useState(props.email);
9   return (
10     <div className="containers">
11
12       <form className="editing">
13         <div>
14           <span className="titles">ID : </span>
15           <input type="text" value={id} onChange={(e) => setId(e.target.value)} />
16         </div>
17         <div>
18           <span className="titles">User Name : </span>
19           <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} />
20         </div>
21         <div>
22           <span className="titles">Name : </span>
23           <input type="text" value={name} onChange={(e) => setName(e.target.value)} />
24         </div>
25         <div>
26           <span className="titles">Email : </span>
27           <input type="text" value={email} onChange={(e) => setEmail(e.target.value)} />
28         </div>
29       </form>
30     </div>
31   );
32 }
```



```
1 function EditUserForm(props) {
2   console.log("form")
3   let [id, setId] = useState(props.id);
4   let [name, setName] = useState(props.name);
5   let [username, setUsername] = useState(props.username);
6   let [email, setEmail] = useState(props.email);
7   return (
8     <div className="containers">
9
10       <form className="editing">
11         <div>
12           <span className="titles">ID : </span>
13           <input type="text" value={id} onChange={(e) => setId(e.target.value)} />
14         </div>
15         <div>
16           <span className="titles">User Name : </span>
17           <input type="text" value={username} onChange={(e) => setUsername(e.target.value)} />
18         </div>
19         <div>
20           <span className="titles">Name : </span>
21           <input type="text" value={name} onChange={(e) => setName(e.target.value)} />
22         </div>
23         <div>
24           <span className="titles">Email : </span>
25           <input type="text" value={email} onChange={(e) => setEmail(e.target.value)} />
26         </div>
27         <div>
28           <input type="button" value="Cancel" onClick={props.cancel} />
29           <input type="button" value="Edit" onClick={() => props.updateEdit({ id, name, username, email })} />
30         </div>
31       </form>
32     </div>
33   );
34 }
35 export default EditUserForm;
```

Index.css



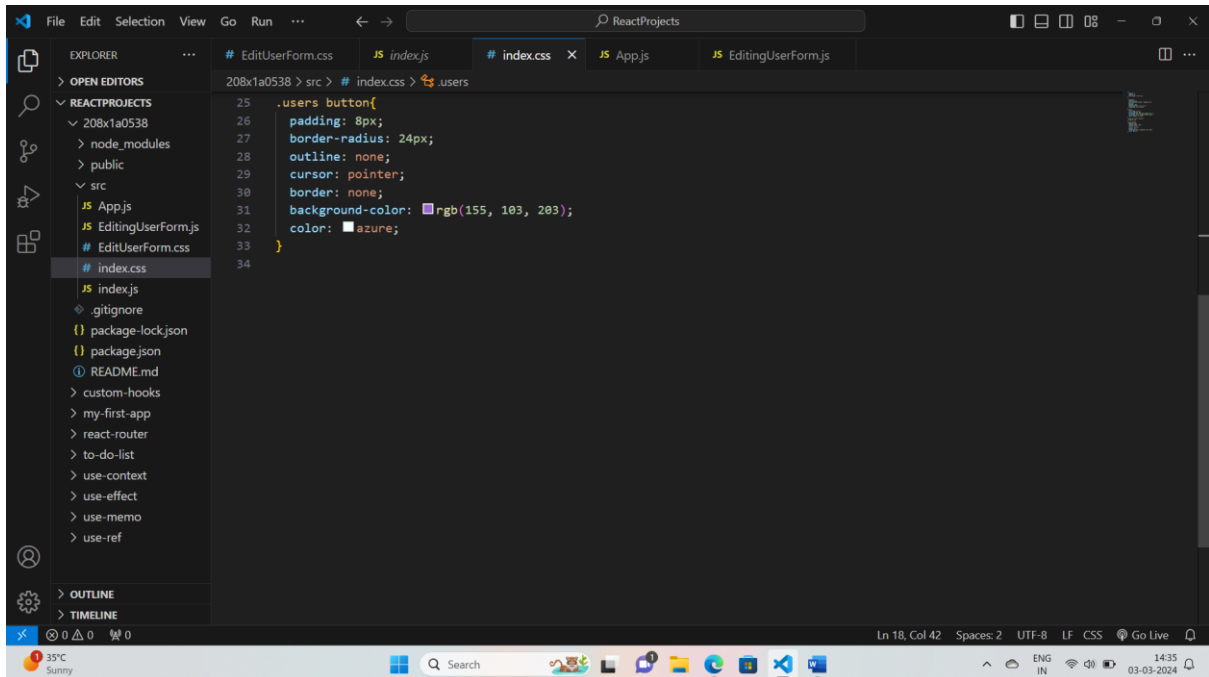
This screenshot shows the VS Code editor with the file explorer on the left and the editor window in the center. The file explorer shows the project structure with the following files and folders:

- 208x1a0538
 - node_modules
 - public
 - src
 - App.js
 - EditingUserForm.js
 - EditUserForm.css
 - index.css
 - index.js
 - .gitignore
 - package-lock.json
 - package.json
 - README.md
 - custom-hooks
 - my-first-app
 - react-router
 - to-do-list
 - use-context
 - use-effect
 - use-memo
 - use-ref
- OUTLINE
- TIMELINE

The editor window shows the following CSS code in index.css:

```
1 {  
2   margin: 0;  
3   padding: 0;  
4   box-sizing: border-box;  
5 }  
6  
7 .container{  
8   display: grid;  
9   grid-template-columns: repeat(3,1fr);  
10  gap: 2em;  
11  padding: 10px;  
12  background-color: #cornsilk;  
13  transition: all 2s ease;  
14 }  
15  
16 .users{  
17   text-align: center;  
18   letter-spacing: 2px;  
19   box-shadow: 0px 5px 10px #rgba(0,0,0,0.1);  
20   background-color: #rgb(178, 180, 184);  
21   color: #rgb(7, 42, 76);  
22 }  
23  
24 .users div, .users button{  
25   margin: 2px;  
26 }  
27  
28 .users button{  
29   padding: 8px;  
30   border-radius: 24px;  
31   outline: none;  
32   cursor: pointer;  
33   border: none;  
34 }
```

The status bar at the bottom shows the file is at line 18, column 42, with 2 spaces, UTF-8 encoding, LF line endings, and CSS syntax highlighting. The system tray shows a temperature of 35°C and the date 03-03-2024.



This screenshot shows the VS Code editor with the file explorer on the left and the editor window in the center. The file explorer shows the project structure with the following files and folders:

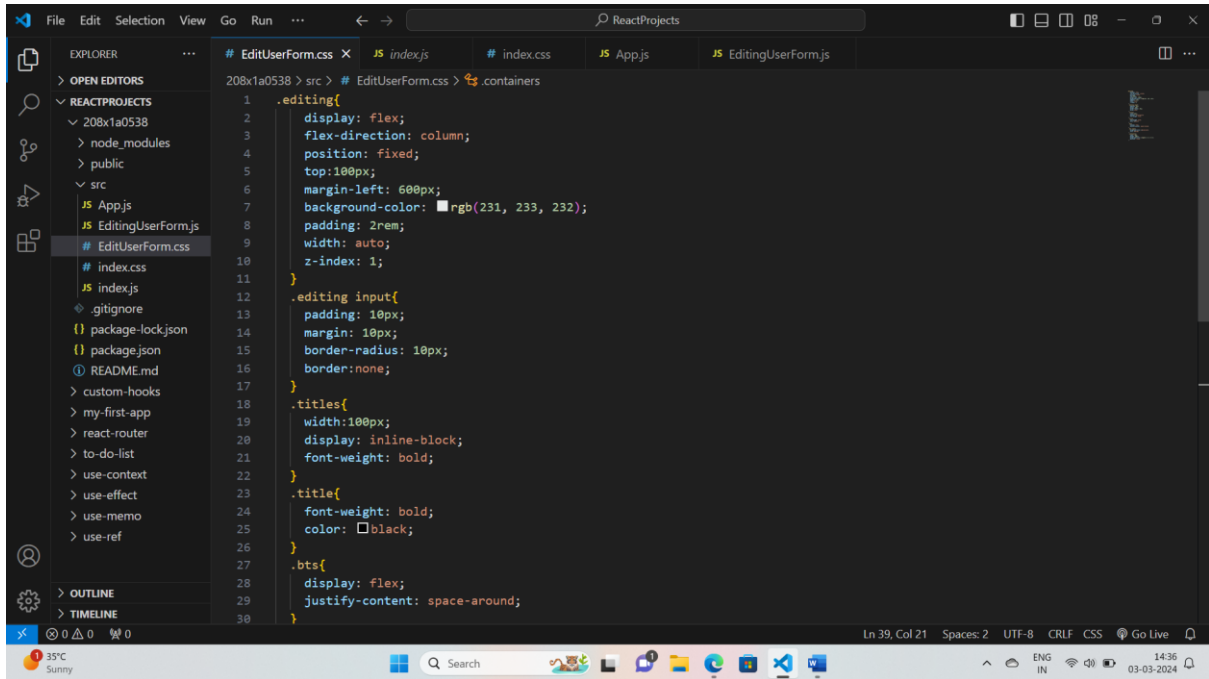
- 208x1a0538
 - node_modules
 - public
 - src
 - App.js
 - EditingUserForm.js
 - EditUserForm.css
 - index.css
 - index.js
 - .gitignore
 - package-lock.json
 - package.json
 - README.md
 - custom-hooks
 - my-first-app
 - react-router
 - to-do-list
 - use-context
 - use-effect
 - use-memo
 - use-ref
- OUTLINE
- TIMELINE

The editor window shows the following CSS code in index.css:

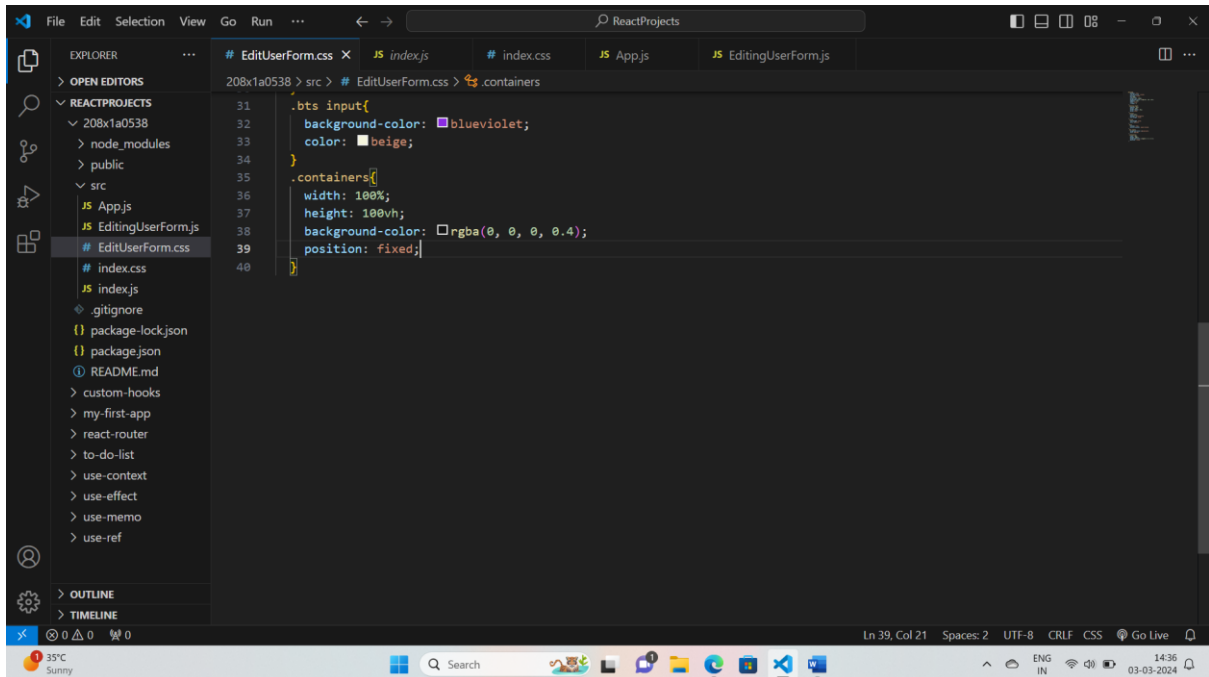
```
25 .users button{  
26   padding: 8px;  
27   border-radius: 24px;  
28   outline: none;  
29   cursor: pointer;  
30   border: none;  
31   background-color: #rgb(155, 103, 203);  
32   color: #azure;  
33 }  
34
```

The status bar at the bottom shows the file is at line 18, column 42, with 2 spaces, UTF-8 encoding, LF line endings, and CSS syntax highlighting. The system tray shows a temperature of 35°C and the date 03-03-2024.

EditUserForm.css



```
1 .editing{
2   display: flex;
3   flex-direction: column;
4   position: fixed;
5   top: 100px;
6   margin-left: 600px;
7   background-color: rgb(231, 233, 232);
8   padding: 2rem;
9   width: auto;
10  z-index: 1;
11 }
12 .editing input{
13   padding: 10px;
14   margin: 10px;
15   border-radius: 10px;
16   border: none;
17 }
18 .titles{
19   width: 100px;
20   display: inline-block;
21   font-weight: bold;
22 }
23 .title{
24   font-weight: bold;
25   color: black;
26 }
27 .bts{
28   display: flex;
29   justify-content: space-around;
30 }
```



```
31 .bts input{
32   background-color: blueviolet;
33   color: beige;
34 }
35 .containers{
36   width: 100%;
37   height: 100vh;
38   background-color: rgba(0, 0, 0, 0.4);
39   position: fixed;
40 }
```