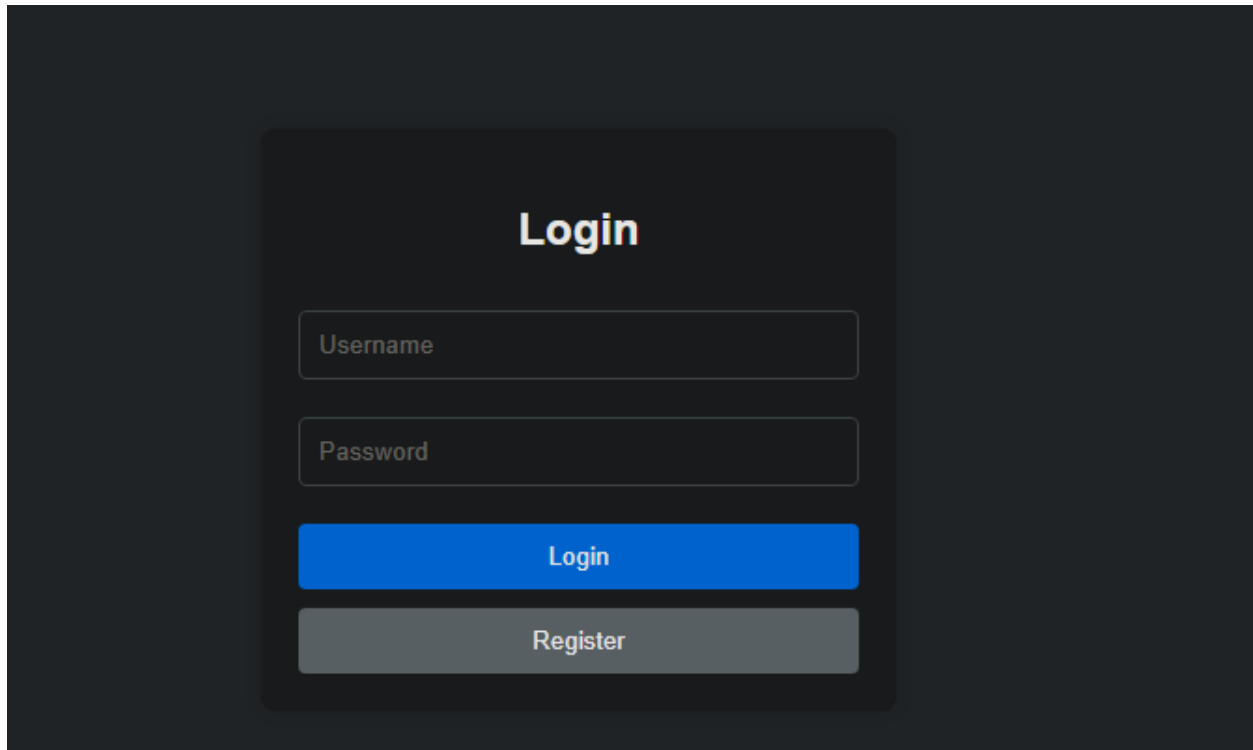


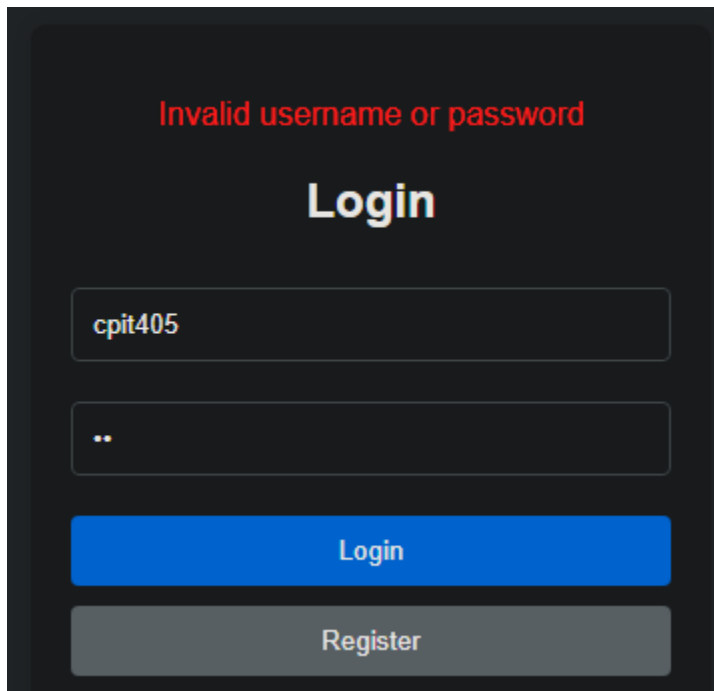
Im doing a calculator that makes you can (add , multiply , subtract , divide) and also can saved your results , so that you have create a user to save your results .

We will start with the login page



The image shows a dark-themed login page. At the top center, the word "Login" is displayed in a bold, white font. Below it, there are two input fields: "Username" and "Password", both with light gray borders. Under the "Password" field, there is a blue button labeled "Login" and a gray button labeled "Register".

If you dont have a user you will not able to login



The image shows the same dark-themed login page, but with an error message. At the top, the text "Invalid username or password" is displayed in red. Below it, the word "Login" is in white. The "Username" field now contains the text "cpit405". The "Password" field contains two dots. The blue "Login" button and the gray "Register" button are still present at the bottom.

) Whats

البريد الوارد

localhost:3000 says

User registered successfully!

OK

Register

Register

Login

Welcome, cpit405

Calculate

Previous Results

Logout

Calculation

Enter first number

Enter second number

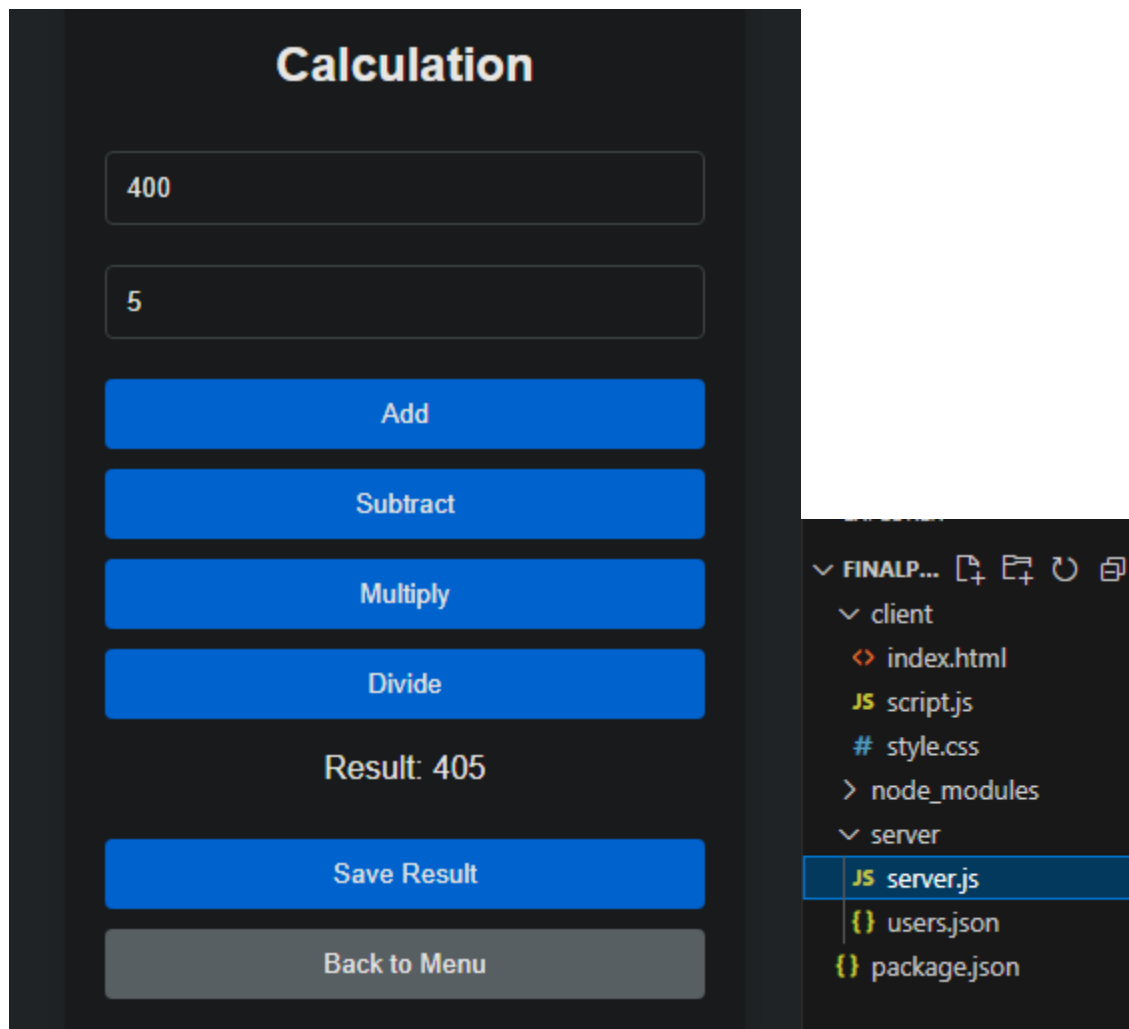
Add

Subtract

Multiply

Divide

Back to Menu



```
ent > <> index.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>React Auth and Calculation App</title>
7    <link rel="stylesheet" href="style.css">
8  </head>
9  <body>
10   <div id="root"></div>
11   <script src="https://unpkg.com/react/umd/react.development.js"></script>
12   <script src="https://unpkg.com/react-dom/umd/react-dom.development.js"></script>
13   <script src="https://unpkg.com/babel-standalone/babel.min.js"></script>
14   <script src="script.js" type="text/babel"></script>
15 </body>
16 </html>
17
```

```

1  class App extends React.Component {
2      constructor(props) {
3          super(props);
4          this.state = {
5              isLoggedIn: false,
6              registerUsername: '',
7              registerPassword: '',
8              loginUsername: '',
9              loginPassword: '',
10             num1: '',
11             num2: '',
12             result: null,
13             savedResults: [],
14             error: '',
15             username: '',
16             isRegisterView: false,
17             view: 'menu' // 'menu', 'calculate', 'results'
18         };
19     }
20
21     handleInputChange = (e) => {
22         this.setState({ [e.target.name]: e.target.value });
23     }
24
25     toggleView = () => {
26         this.setState(prevState => ({
27             isRegisterView: !prevState.isRegisterView,
28             error: ''
29         }));
30     }
31
32     handleRegister = () => {
33         const { registerUsername, registerPassword } = this.state;
34         fetch('/register', {
35             method: 'POST',
36             headers: {
37                 'Content-Type': 'application/json'
38             },
39             body: JSON.stringify({ username: registerUsername, password: registerPassword })
40         })
41         .then(response => response.json())
42         .then(data => {
43             if (data.success) {
44                 alert('User registered successfully!');
45                 this.setState({ registerUsername: '', registerPassword: '', error: '', isRegisterView: false });
46             } else {
47                 this.setState({ error: data.message });
48             }
49         })
50     }
51 }

```

```

    });
  }

  handleLogin = () => {
    const { loginUsername, loginPassword } = this.state;
    fetch('/login', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ username: loginUsername, password: loginPassword })
    })
    .then(response => response.json())
    .then(data => {
      if (data.success) {
        this.setState({
          isLoggedIn: true,
          savedResults: data.results,
          error: '',
          loginUsername: '',
          loginPassword: '',
          username: loginUsername
        });
      } else {
        this.setState({ error: data.message });
      }
    });
  }

  handleLogout = () => {
    this.setState({ isLoggedIn: false, result: null, savedResults: [], username: '' });
  }

  handleCalculation = (operation) => {
    const { num1, num2 } = this.state;
    let result;
    const n1 = parseFloat(num1);
    const n2 = parseFloat(num2);

    if (isNaN(n1) || isNaN(n2)) {
      this.setState({ error: 'Please enter valid numbers' });
      return;
    }

    switch (operation) {

```

```

    if (isNaN(n1) || isNaN(n2)) {
      this.setState({ error: 'Please enter valid numbers' });
      return;
    }

    switch (operation) {
      case 'add':
        result = n1 + n2;
        break;
      case 'subtract':
        result = n1 - n2;
        break;
      case 'multiply':
        result = n1 * n2;
        break;
      case 'divide':
        result = n1 / n2;
        break;
      default:
        result = null;
    }

    this.setState({ result, error: '' });
  }

  handleSaveResult = () => {
    const { result, username } = this.state;
    fetch('/save-result', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ username, result })
    })
      .then(response => response.json())
      .then(data => {
        if (data.success) {
          this.setState(prevState => ({
            savedResults: [...prevState.savedResults, prevState.result]
          }));
        } else {
          this.setState({ error: data.message });
        }
      })
  }
}

```



```

162         name="num1"
163         placeholder="Enter first number"
164         value={num1}
165         onChange={this.handleInputChange}
166     />
167     <input
168     type="text"
169     name="num2"
170     placeholder="Enter second number"
171     value={num2}
172     onChange={this.handleInputChange}
173     />
174     <div>
175         <button onClick={() => this.handleCalculation('add')}>Add</button>
176         <button onClick={() => this.handleCalculation('subtract')}>Subtract</button>
177         <button onClick={() => this.handleCalculation('multiply')}>Multiply</button>
178         <button onClick={() => this.handleCalculation('divide')}>Divide</button>
179     </div>
180     {result !== null && (
181     <div>
182         <p>Result: {result}</p>
183         <button onClick={this.handleSaveResult}>Save Result</button>
184     </div>
185     )}
186     <button className="switch-button" onClick={() => this.setView('menu')}>Back to Menu</button>
187 </div>
188 );
189 } else if (view === 'results') {
190     return (
191     <div className="container">
192         <h2>Saved Results</h2>
193         <ul>
194             {savedResults.map((result, index) => (
195                 <li key={index}>{result}</li>
196             ))}
197         </ul>
198         <button className="switch-button" onClick={() => this.setView('menu')}>Back to Menu</button>
199     </div>
200     );
201 }
202 }
203
204 return (
205 <div className="container">
206     {error && <p className="error">{error}</p>}
207     <div className={this.registerView} > '' : 'hidden'>

```

```

206 {error && <p className="error">{error}</p>}
207 <div className={!isRegisterView ? '' : 'hidden'}>
208   <h2>Login</h2>
209   <input
210     type="text"
211     name="loginUsername"
212     placeholder="Username"
213     value={loginUsername}
214     onChange={this.handleInputChange}
215   />
216   <input
217     type="password"
218     name="loginPassword"
219     placeholder="Password"
220     value={loginPassword}
221     onChange={this.handleInputChange}
222   />
223   <button onClick={this.handleLogin}>Login</button>
224   <button className="switch-button" onClick={this.toggleView}>Register</button>
225 </div>
226 <div className={isRegisterView ? '' : 'hidden'}>
227   <h2>Register</h2>
228   <input
229     type="text"
230     name="registerUsername"
231     placeholder="Username"
232     value={registerUsername}
233     onChange={this.handleInputChange}
234   />
235   <input
236     type="password"
237     name="registerPassword"
238     placeholder="Password"
239     value={registerPassword}
240     onChange={this.handleInputChange}
241   />
242   <button onClick={this.handleRegister}>Register</button>
243   <button className="switch-button" onClick={this.toggleView}>Login</button>
244 </div>
245 </div>
246 );
247 }
248 }
249

```

```
client > # style.css > ...
1  ∨ body {
2      font-family: Arial, sans-serif;
3      display: flex;
4      justify-content: center;
5      align-items: center;
6      height: 100vh;
7      background-color: #f0f0f0;
8      margin: 0;
9  }
10
11 ∨ .container {
12     background: white;
13     padding: 20px;
14     border-radius: 8px;
15     box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
16     width: 300px;
17     text-align: center;
18 }
19
20 ∨ input {
21     width: 100%;
22     padding: 10px;
23     margin: 10px 0;
24     border: 1px solid #ccc;
25     border-radius: 4px;
26     box-sizing: border-box;
27 }
28
29 ∨ button {
30     width: 100%;
31     padding: 10px;
32     background-color: #007BFF;
33     border: none;
34     color: white;
35     border-radius: 4px;
36     cursor: pointer;
37     margin-top: 10px;
38 }
39
40 ∨ button:hover {
41     background-color: #0056b3;
42 }
43
44 ∨ .error {
45     color: red;
46     margin-bottom: 10px;
47 }
48
49 ∨ .switch-button {
```

```
43
44 .error {
45     color: red;
46     margin-bottom: 10px;
47 }
48
49 .switch-button {
50     background-color: #6c757d;
51 }
52
53 .switch-button:hover {
54     background-color: #5a6268;
55 }
56
57 .hidden {
58     display: none;
59 }
60
61 .menu {
62     display: flex;
63     flex-direction: column;
64     gap: 10px;
65 }
66
67 .menu button {
68     width: 100%;
69 }
70
```

```

server > JS server.js > ...
1  const express = require('express');
2  const fs = require('fs');
3  const path = require('path');
4  const bodyParser = require('body-parser');
5  const app = express();
6  const port = 3000;
7
8  // Middleware
9  app.use(bodyParser.json());
10 app.use(express.static(path.join(__dirname, '../client')));
11
12 // Serve the index.html file at the root URL
13 app.get('/', (req, res) => {
14   res.sendFile(path.join(__dirname, '../client/index.html'));
15 });
16
17 let users = [];
18
19 // Load existing users from users.json
20 const usersFilePath = path.join(__dirname, 'users.json');
21 if (fs.existsSync(usersFilePath)) {
22   users = JSON.parse(fs.readFileSync(usersFilePath));
23 }
24
25 // Registration endpoint
26 app.post('/register', (req, res) => {
27   const { username, password } = req.body;
28   if (users.find(user => user.username === username)) {
29     return res.json({ success: false, message: 'Username already exists' });
30   }
31   users.push({ username, password, results: [] });
32   fs.writeFileSync(usersFilePath, JSON.stringify(users));
33   res.json({ success: true });
34 });
35
36 // Login endpoint
37 app.post('/login', (req, res) => {
38   const { username, password } = req.body;
39   const user = users.find(user => user.username === username && user.password === password);
40   if (user) {
41     res.json({ success: true, results: user.results });
42   } else {
43     res.json({ success: false, message: 'Invalid username or password' });
44   }
45 });
46
47 // Save result endpoint
48 app.post('/save-result', (req, res) => {
49   const { username, result } = req.body;

```

```

    });

    // Save result endpoint
    app.post('/save-result', (req, res) => {
      const { username, result } = req.body;
      const user = users.find(user => user.username === username);
      if (user) {
        user.results.push(result);
        fs.writeFileSync(usersFilePath, JSON.stringify(users));
        res.json({ success: true });
      } else {
        res.json({ success: false, message: 'User not found' });
      }
    });

    app.listen(port, () => {
      console.log(`Server running at http://localhost:${port}`);
    });

```

index.html # style.css JS script.js JS server.js {} users.json X {} package.json

server > {} users.json > {} 1

```
1 [{"username": "as", "password": ""}, {"username": "Mohannad", "password": "asd"}, {"username": "cpit405", "password": "aa"}]
```