

React Assignments

1. Create a simple “Hello World” application in React JS

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure under "hello-world\src". The "App.js" file is open in the editor.
- Editor:** The code for "App.js" is displayed:

```
1 import './App.css';
2 import React from 'react';
3 function App() {
4   return (
5     <div className="App">
6       <h1>Hello World</h1>
7     </div>
8   );
9 }
10
11 export default App;
```

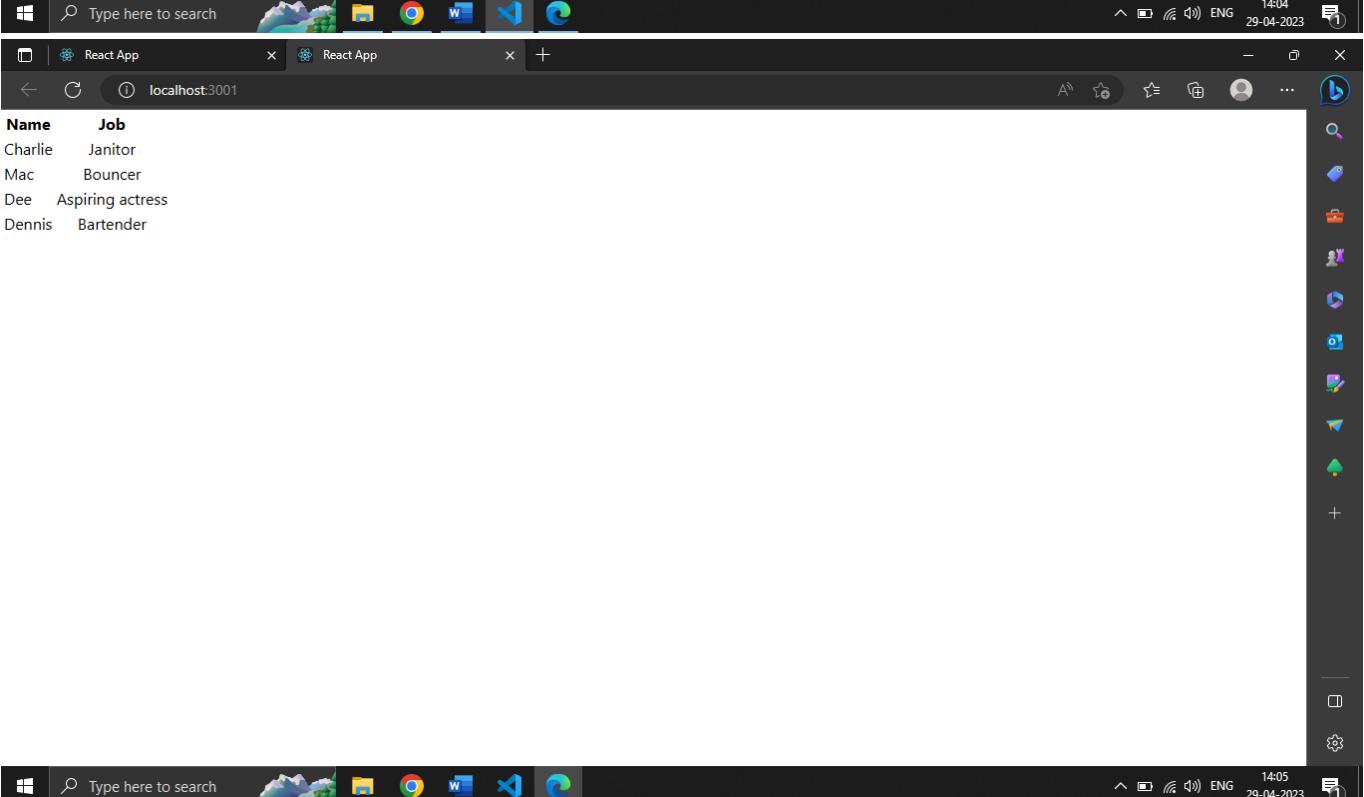
- Terminal:** Shows the output of the build process:

```
Compiled successfully!
You can now view hello-world in the browser.
Local: http://localhost:3000
On Your Network: http://192.168.1.6:3000
Note that the development build is not optimized.
To create a production build, use npm run build.
webpack compiled successfully
```
- Browser Preview:** A Microsoft Edge window shows the result of the build at "localhost:3000", displaying the text "Hello World".
- System Taskbar:** Shows the Windows taskbar with various pinned icons and the date/time "29-04-2023 14:01".

2) Create a React application that displays the list of employees.



```
File Edit Selection View Go Run Terminal Help • App.js - Assignment - Visual Studio Code
EXPLORER ... JS App.js M ●
ASSIGNMENT
> calculator
> clock
> employee
  > node_modules
  > public
  > src
# App.css
JS App.js M
JS App.test.js
# index.css
JS index.js
logo.svg
JS reportWebVitals.js
JS setupTests.js
.gitignore
{ package-lock.json
() package.json
① README.md
> employee2
> hello-world
> hello-worlda
> node_modules
> router6
> weather
{ package-lock.json
{ package.json
> OUTLINE
> TIMELINE
Ln 31, Col 2 Spaces: 2 UTF-8 LF () JavaScript Go Live
employee > src > JS App.js > App
1 import React, { useState } from 'react';
2 import './App.css';
3
4 function App() {
5   const [employees, setEmployees] = useState([
6     { name: 'Charlie', jobRole: 'Janitor' },
7     { name: 'Mac', jobRole: 'Bouncer' },
8     { name: 'Dee', jobRole: 'Aspiring actress' },
9     { name: 'Dennis', jobRole: 'Bartender' },
10    ]);
11
12  const employeeTable = employees.map((employee) => (
13    <tr>
14      <td>{employee.name}</td>
15      <td style={{ textAlign: 'center' }}>{employee.jobRole}</td>
16    </tr>
17  ));
18  return (
19    <div>
20      <table>
21        <thead>
22          <tr>
23            <th>Name</th>
24            <th>Job</th>
25          </tr>
26        </thead>
27        <tbody>{employeeTable}</tbody>
28      </table>
29    </div>
30  );
31 }
32
33 export default App;
```



3. Create a React application with following requirements.

- a. Add new employee. After successful addition of new employee, It will be displayed in the table.
 - b. Add few more employees.

c. If I click on the Delete button, record of that specific employee should be deleted. For example, I delete record of Alex Browning, then list of employees should look like this

Add new employee

Name: Job Role: Add Employee

List of employees

Name	Job Role	Action
Shivani Tiwari	Doctor	Delete
Avi	Engineer	Delete
Nikki	HR	Delete
Mikky	Developer	Delete
Alex Browning	Assistant	Delete

Add new employee

Name: Job Role: Add Employee

List of employees

Name	Job Role	Action
Shivani Tiwari	Doctor	Delete
Avi	Engineer	Delete
Nikki	HR	Delete
Mikky	Developer	Delete

App.js

File Edit Selection View Go Run Terminal Help App.js - Assignment - Visual Studio Code

```
employee2 > src > JS App.js employee2\... M X
1 import './App.css';
2 import React, { useState } from 'react';
3
4 function App() {
5   const [employees, setEmployees] = useState([
6     ]);
7   const [newEmployeeName, setNewEmployeeName] = useState('');
8   const [newEmployeeJobRole, setNewEmployeeJobRole] = useState('');
9   const handleAddEmployee = () => {
10     const newEmployee = {
11       id: employees.length + 1,
12       name: newEmployeeName,
13       jobRole: newEmployeeJobRole,
14     };
15     setEmployees([...employees, newEmployee]);
16     setNewEmployeeName('');
17     setNewEmployeeJobRole('');
18   };
19   const handleDeleteEmployee = (id) => {
20     const filteredEmployees = employees.filter((employee) => employee.id !== id);
21     setEmployees(filteredEmployees);
22   };
23   const employeeTable = employees.map((employee) => (
24     <tr key={employee.id}>
25       <td>{employee.name}</td>
26       <td>{employee.jobRole}</td>
27       <td>
28         <button className="delete-button" onClick={() => handleDeleteEmployee(employee.id)}>Delete</button>
29       </td>
30     </tr>
31   )));
32   return (
33     <div className="container">
34       <h2 style={{ margin: '30px' }}>Add new employee</h2>
35       <div className="input-row">
36         <label className="input-label">
37           Name:
38           <input
39             className="input-field"
40             type="text"
41             value={newEmployeeName}
42             onChange={(event) => setNewEmployeeName(event.target.value)}
43           />
44         </label>
45         <label className="input-label">
46           Job Role:
47           <input
48             className="input-field"
49             type="text"
50             value={newEmployeeJobRole}
51             onChange={(event) => setNewEmployeeJobRole(event.target.value)}
52           />
53         </label>
54         <button className="add-button" onClick={handleAddEmployee}>Add Employee</button>
55       </div>
56       <h2>List of employees</h2>
57       <table className="employee-table">
58         <thead>
59           <tr>
60             <th>Name</th>
61             <th>Job Role</th>
62             <th>Action</th>
63           </tr>
64         </thead>
65         <tbody>
66           {employeeTable}
67         </tbody>
68       </table>
69     </div>
70   );
71 }
```

Ln 1, Col 1 Spaces: 2 UTF-8 LF ⓘ JavaScript ⓘ Go Live ⌂ ⓘ

File Edit Selection View Go Run Terminal Help App.js - Assignment - Visual Studio Code

```
employee2 > src > JS App.js employee2\... M X
1 </tr>
2   );
3   return (
4     <div className="container">
5       <h2 style={{ margin: '30px' }}>Add new employee</h2>
6       <div className="input-row">
7         <label className="input-label">
8           Name:
9           <input
10             className="input-field"
11             type="text"
12             value={newEmployeeName}
13             onChange={(event) => setNewEmployeeName(event.target.value)}
14           />
15         </label>
16         <label className="input-label">
17           Job Role:
18           <input
19             className="input-field"
20             type="text"
21             value={newEmployeeJobRole}
22             onChange={(event) => setNewEmployeeJobRole(event.target.value)}
23           />
24         </label>
25         <button className="add-button" onClick={handleAddEmployee}>Add Employee</button>
26       </div>
27       <h2>List of employees</h2>
28       <table className="employee-table">
29         <thead>
30           <tr>
31             <th>Name</th>
32             <th>Job Role</th>
33             <th>Action</th>
34           </tr>
35         </thead>
36         <tbody>
37           {employeeTable}
38         </tbody>
39       </table>
40     </div>
41   );
42 }
```

Ln 1, Col 1 Spaces: 2 UTF-8 LF ⓘ JavaScript ⓘ Go Live ⌂ ⓘ

File Edit Selection View Go Run Terminal Help

App.js - Assignment - Visual Studio Code

EXPLORER

ASSIGNMENT

- > calculator
- > clock
- > employee
- < employee2
- > node_modules
- > public
- < src
 - # App.css M
 - JS App.js M
 - JS App.test.js
 - # index.css
 - JS index.js
 - logo.svg
 - JS reportWebVitals.js
 - JS setupTests.js
 - .gitignore
 - { package-lock.json
 - { package.json
 - README.md
- > hello-world
- > hello-worlda
- > node_modules
- > router6
- > weather
- { package-lock.json
- { package.json

> OUTLINE

> TIMELINE

JS App.js employee\src M ● JS App.js employee2\... M X

```
employee2 > src > JS App.js > ...
59         <tr>
60             <th>Name</th>
61             <th>Job Role</th>
62             <th>Action</th>
63         </tr>
64     </thead>
65     <tbody>{employeeTable}</tbody>
66   </table>
67 </div>
68 );
69 }
70
71 export default App;
```

Ln 1, Col 1 Spaces: 2 UTF-8 LF {} JavaScript Go Live ⚡ 🔍

Type here to search

Windows Taskbar icons: File Explorer, Edge, Chrome, File Manager, Task View, Taskbar settings.

App.css

File Edit Selection View Go Run Terminal Help

App.css - Assignment - Visual Studio Code

EXPLORER

ASSIGNMENT

- > calculator
- > clock
- > employee
- < employee2
- > node_modules
- > public
- < src
 - # App.css M
 - JS App.js M
 - JS App.test.js
 - # index.css
 - JS index.js
 - logo.svg
 - JS reportWebVitals.js
 - JS setupTests.js
 - .gitignore
 - { package-lock.json
 - { package.json
 - README.md
- > hello-world
- > hello-worlda
- > node_modules
- > router6
- > weather
- { package-lock.json
- { package.json

> OUTLINE

> TIMELINE

JS App.js employee\src M ● JS App.js employee2\... M X

```
employee2 > src > # App.css > add-button
1 .container {
2     margin: 0 auto;
3     max-width: 800px;
4 }
5
6 .input-label {
7     margin-right: 100px;
8     font-weight: bold;
9     width: 90px;
10 }
11 .add-button {
12     margin-top: 20px;
13     padding: 5px 10px;
14     background-color: #0066CC;
15     color: #ffff;
16     border: none;
17     border-radius: 3px;
18     cursor: pointer;
19 }
20 .employee-table {
21     border-collapse: collapse;
22     width: 100%;
23 }
24
25 .employee-table th,
26 .employee-table td {
27     padding: 8px;
28     text-align: left;
29     border-bottom: 1px solid #ddd;
30 }
31 .employee-table th {
32     background-color: #0066CC;
33 }
```

Ln 19, Col 2 Spaces: 2 UTF-8 LF CSS Go Live ⚡ 🔍

Type here to search

Windows Taskbar icons: File Explorer, Edge, Chrome, File Manager, Task View, Taskbar settings.

File Edit Selection View Go Run Terminal Help

App.css - Assignment - Visual Studio Code

EXPLORER

ASSIGNMENT

- > calculator
- > clock
- > employee
- < employee2
- > node_modules
- > public
- < src
 - # App.css M
 - JS App.js M
 - JS App.test.js
 - # index.css
 - JS index.js
 - logo.svg
 - JS reportWebVitals.js
 - JS setupTests.js
 - .gitignore
 - { package-lock.json
 - { package.json
 - ① README.md
 - > hello-world
 - > hello-worlda
 - > node_modules
 - > router6
 - > weather
 - { package-lock.json
 - { package.json
- > OUTLINE
- > TIMELINE

App.css M X

```
employee2 > src > # App.css > add-button
  .add-button {
    background-color: #0066CC;
    color: #fff;
  }
  .employee-table tr:hover {
    background-color: #ddd;
  }
  .delete-button {
    padding: 5px 10px;
    background-color: #e60000;
    color: #fff;
    border: none;
    border-radius: 3px;
    cursor: pointer;
  }
  .delete-button:hover {
    background-color: #cc0000;
  }
  .employee-table tr:nth-child(even) {
    background-color: #f2f2f2;
  }
  .add-button:hover {
    background-color: #0059b3;
  }
  .input-field {
    padding: 5px;
    border: 1px solid #ccc;
  }
```

Ln 19, Col 2 Spaces: 2 UTF-8 LF CSS Go Live ⚙ 14:19 29-04-2023

Type here to search

File Edit Selection View Go Run Terminal Help

App.css - Assignment - Visual Studio Code

EXPLORER

ASSIGNMENT

- > calculator
- > clock
- > employee
- < employee2
- > node_modules
- > public
- < src
 - # App.css M
 - JS App.js M
 - JS App.test.js
 - # index.css
 - JS index.js
 - logo.svg
 - JS reportWebVitals.js
 - JS setupTests.js
 - .gitignore
 - { package-lock.json
 - { package.json
 - ① README.md
 - > hello-world
 - > hello-worlda
 - > node_modules
 - > router6
 - > weather
 - { package-lock.json
 - { package.json
- > OUTLINE
- > TIMELINE

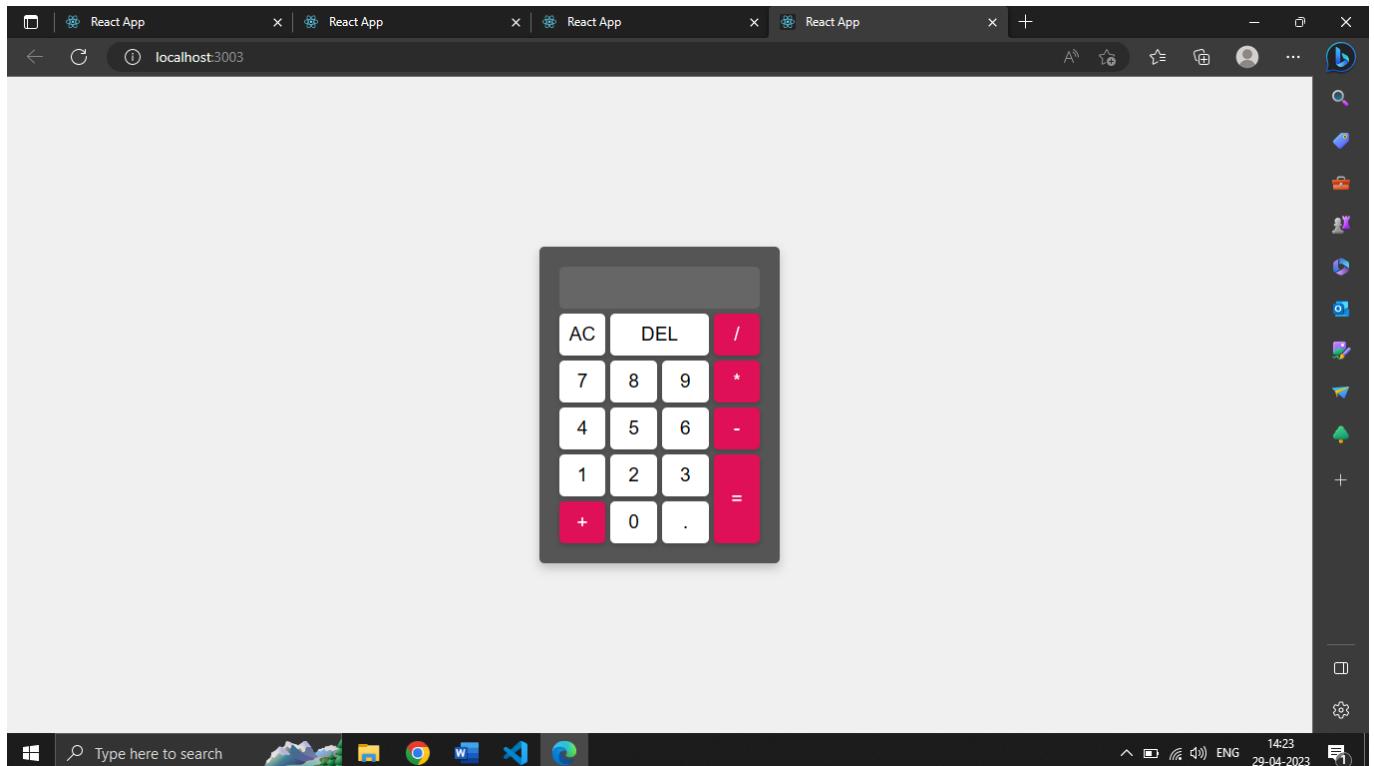
App.css M X

```
employee2 > src > # App.css > add-button
  .add-button {
    background-color: #0059b3;
  }
  .input-field {
    padding: 5px;
    border: 1px solid #ccc;
    border-radius: 3px;
    flex: 1;
  }
  .input-row {
    display: flex;
    align-items: center;
    margin-bottom: 20px;
  }
```

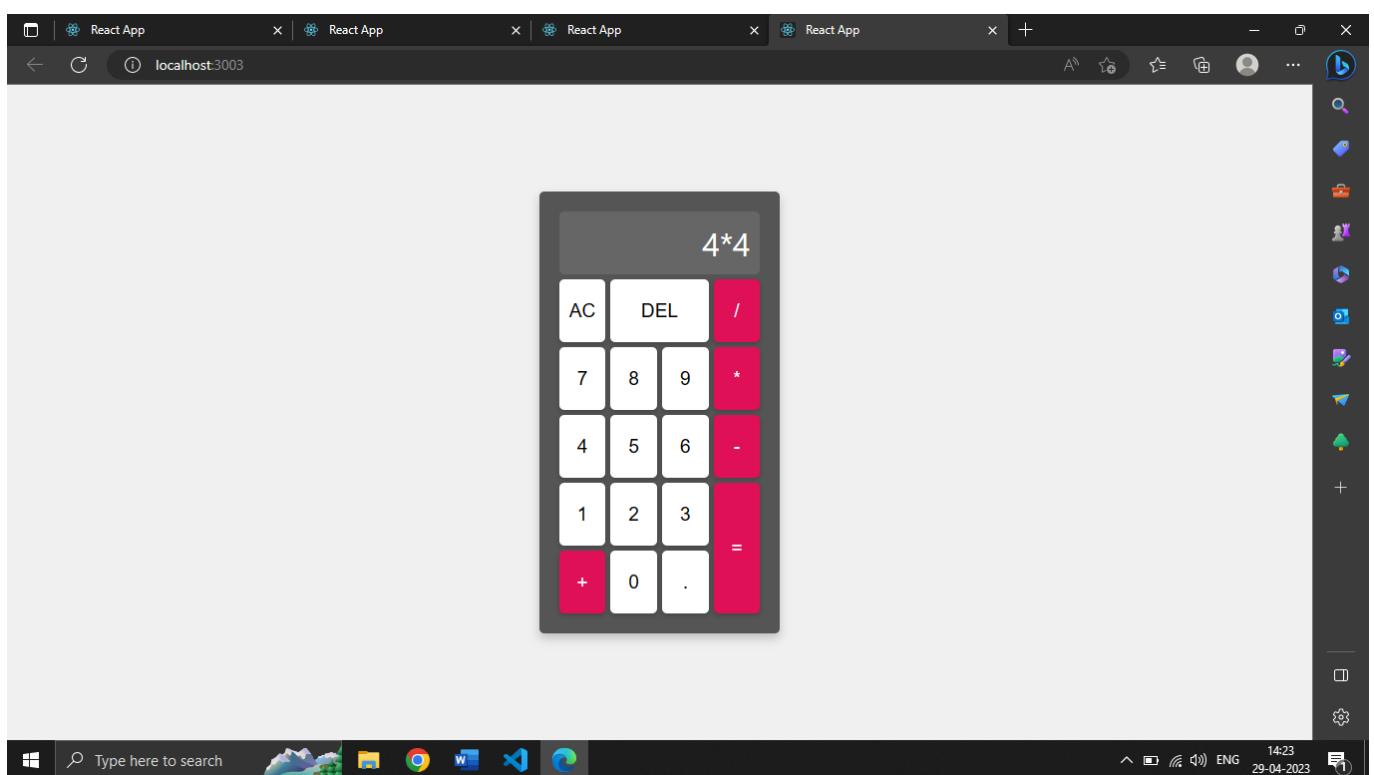
Ln 19, Col 2 Spaces: 2 UTF-8 LF CSS Go Live ⚙ 14:19 29-04-2023

Type here to search

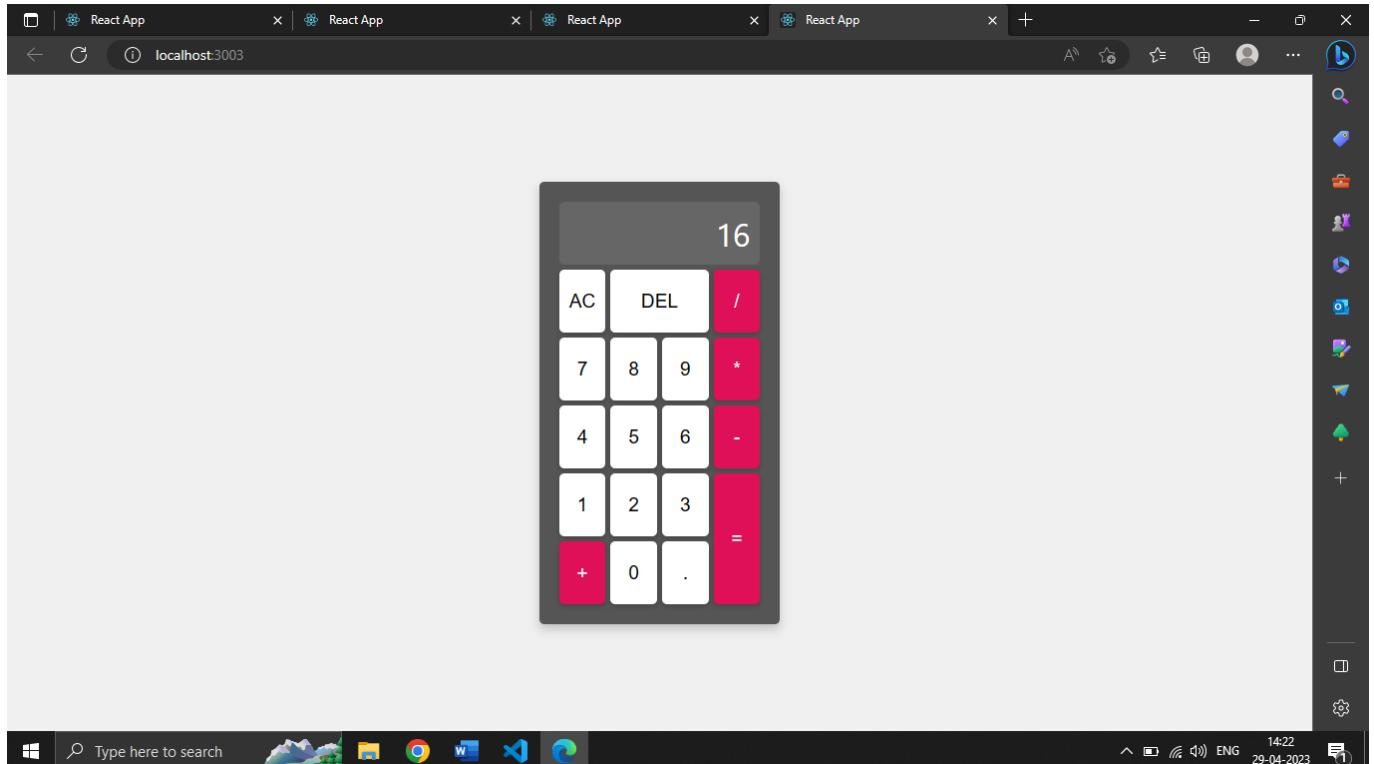
4. Create an application for simple calculator



Operation



Result



App.js

```
calculator > src > JS App.js > App
1 import React, { useState } from 'react';
2 import './App.css';
3
4 function App() {
5   const [result, setResult] = useState('');
6
7   const handleClick = (e) => {
8     setResult(result.concat(e.target.name));
9   }
10
11   const handleClear = () => {
12     setResult('');
13   }
14
15   const handleDelete = () => {
16     setResult(result.slice(0, -1));
17   }
18
19   const handleEqual = () => {
20     try {
21       setResult(eval(result).toString());
22     } catch (err) {
23       setResult('Error');
24     }
25   }
26
27   return (
28     <div className="App">
29       <div className="calculator">
30         <div className="display">{result}</div>
31         <button className="clear" onClick={handleClear}>AC</button>
32         <button className="delete" onClick={handleDelete}>DEL</button>
33         <button className="operator" name="/" onClick={handleClick}>/</button>
```

File Edit Selection View Go Run Terminal Help App.js - Assignment - Visual Studio Code

EXPLORER

ASSIGNMENT

- calculator
 - node_modules
 - public
 - src
 - # App.css M
 - JS App.js M
- JS App.test.js
- # index.css
- JS index.js
- logo.svg
- JS reportWebVitals.js
- JS setupTests.js
- .gitignore
- package-lock.json
- package.json
- README.md
- clock
- employee
- employee2
- hello-world
- hello-worlda
- node_modules
- router6
- weather
- package-lock.json
- package.json

OUTLINE

TIMELINE

JS App.js M

```
calculator > src > JS App.js > App
```

```
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
```

```
    return (
      <div className="App">
        <div className="calculator">
          <div className="display">{result}</div>
          <button className="clear" onClick={handleClear}>AC</button>
          <button className="delete" onClick={handleDelete}>DEL</button>
          <button className="operator" name="/" onClick={handleClick}>/</button>
          <button name="7" onClick={handleClick}>7</button>
          <button name="8" onClick={handleClick}>8</button>
          <button name="9" onClick={handleClick}>9</button>
          <button className="operator" name="*" onClick={handleClick}>*</button>
          <button name="4" onClick={handleClick}>4</button>
          <button name="5" onClick={handleClick}>5</button>
          <button name="6" onClick={handleClick}>6</button>
          <button className="operator" name="-" onClick={handleClick}>-</button>
          <button name="1" onClick={handleClick}>1</button>
          <button name="2" onClick={handleClick}>2</button>
          <button name="3" onClick={handleClick}>3</button>
          <button className="operator" name="+" onClick={handleClick}>+</button>
          <button name="0" onClick={handleClick}>0</button>
          <button name=". " onClick={handleClick}>. </button>
          <button className="equal" onClick={handleEqual}>=</button>
        </div>
      </div>
    );
  }

  export default App;
```

Ln 14, Col 1 Spaces: 2 UTF-8 LF JavaScript Go Live

Type here to search 14:24 ENG 29-04-2023

App.css

File Edit Selection View Go Run Terminal Help App.css - Assignment - Visual Studio Code

EXPLORER

ASSIGNMENT

- calculator
 - node_modules
 - public
 - src
 - # App.css M
 - JS App.js M
- JS App.test.js
- # index.css
- JS index.js
- logo.svg
- JS reportWebVitals.js
- JS setupTests.js
- .gitignore
- package-lock.json
- package.json
- README.md
- clock
- employee
- employee2
- hello-world
- hello-worlda
- node_modules
- router6
- weather
- package-lock.json
- package.json

OUTLINE

TIMELINE

JS App.js M

```
calculator > src > # App.css > App
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
```

```
  .App {
    height: 100vh;
    display: flex;
    justify-content: center;
    align-items: center;
    background-color: #f0f0f0;
  }

  .calculator {
    display: grid;
    grid-template-columns: repeat(4, 1fr);
    grid-template-rows: repeat(6, 1fr);
    grid-gap: 5px;
    background-color: #555;
    border-radius: 5px;
    padding: 20px;
    box-shadow: 0 5px 10px rgba(0, 0, 0, 0.2);
  }

  .display {
    grid-column: 1 / -1;
    grid-row: 1 / 2;
    font-size: 2rem;
    color: #fff;
    text-align: right;
    padding: 10px;
    background-color: rgba(255, 255, 255, 0.1);
    border-radius: 5px;
    overflow: hidden;
    text-overflow: ellipsis;
    white-space: nowrap;
  }

  button {
    font-size: 1.2rem;
    padding: 10px;
  }
```

Ln 1, Col 1 Spaces: 2 UTF-8 LF CSS Go Live

Type here to search 14:25 ENG 29-04-2023

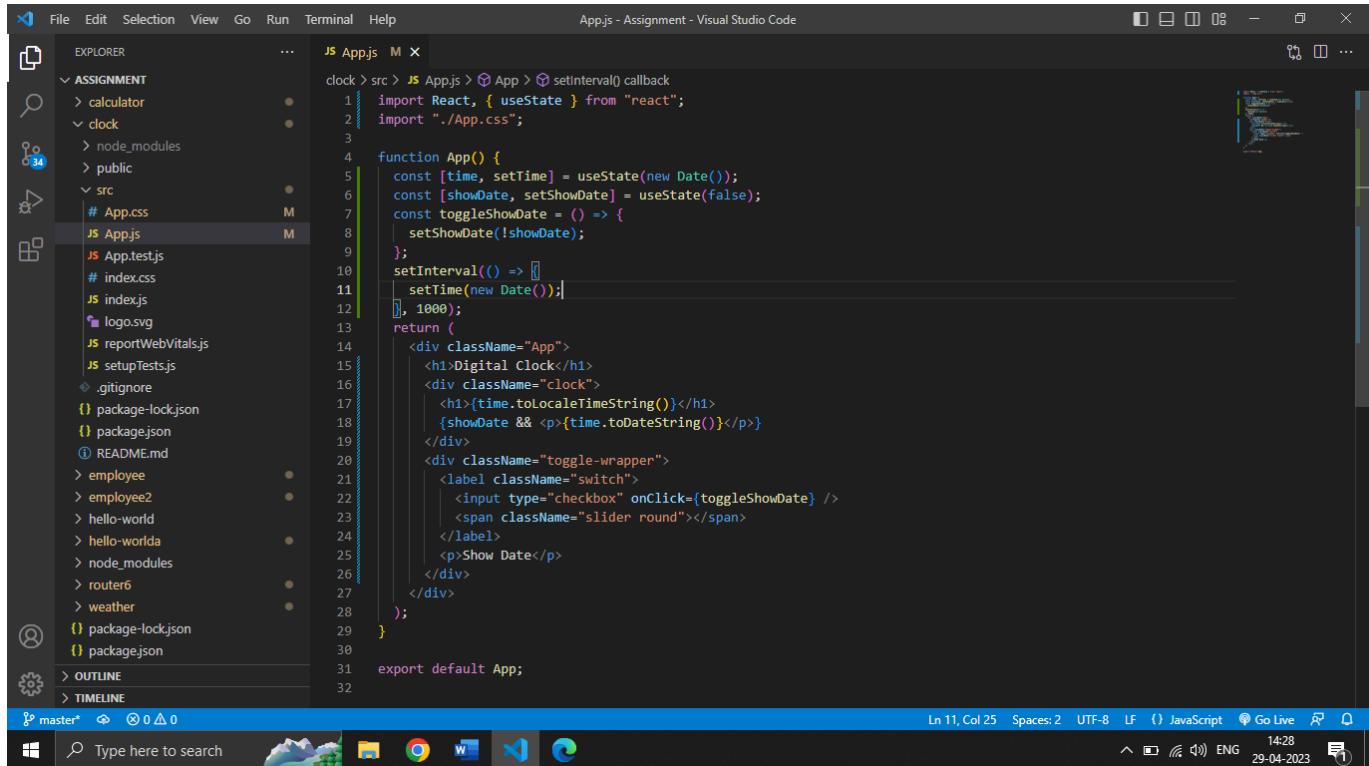
```
calculator > src > # App.css > .App
  padding: 10px;
  background-color: #ffff;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.2);
  transition: all 0.2s ease-in-out;
}
button:hover {
  transform: scale(1.1);
  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.2);
}
.clear {
  grid-column: 1 / 2;
  grid-row: 2 / 3;
}
.delete {
  grid-column: 2 / 4;
  grid-row: 2 / 3;
}
.operator {
  background-color: #df0f58;
  color: #ffff;
}
.equal {
  grid-column: 4 / -1;
  grid-row: 5 / -1;
  background-color: #df0f58;
  color: #ffff;
}
@media (max-width: 768px) {
  .calculator {
```

```
grid-row: 2 / 3;
}
.operator {
  background-color: #df0f58;
  color: #ffff;
}
.equal {
  grid-column: 4 / -1;
  grid-row: 5 / -1;
  background-color: #df0f58;
  color: #ffff;
}
@media (max-width: 768px) {
  .calculator {
    grid-template-columns: repeat(3, 1fr);
    grid-template-rows: repeat(7, 1fr);
    padding: 10px;
  }
  .display {
    font-size: 1.5rem;
    padding: 5px;
  }
  button {
    font-size: 1rem;
    padding: 5px;
  }
}
```

5) Create React application to develop digital clock.

If I toggle the button, it should display the date.

App.js

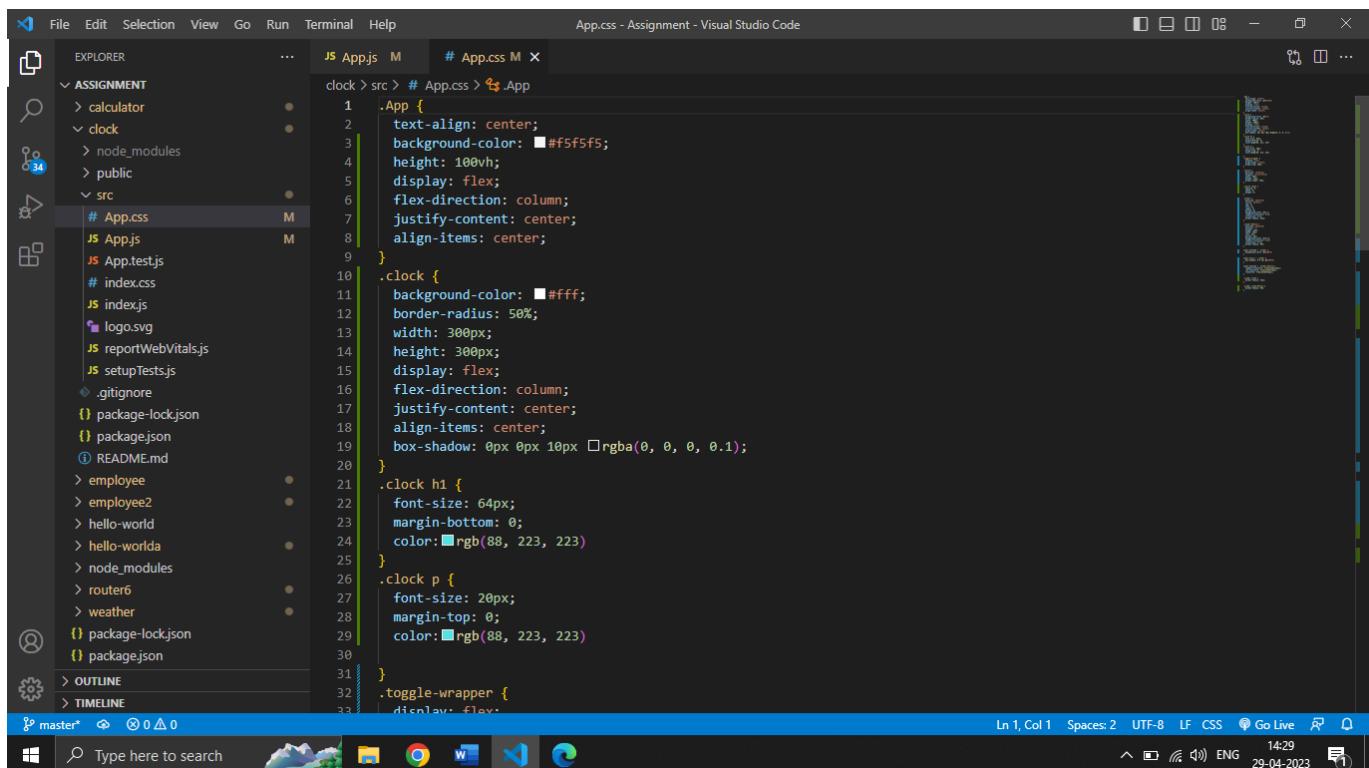


```
File Edit Selection View Go Run Terminal Help
JS App.js M X
clock > src > JS App.js > App > setInterval() callback
1 import React, { useState } from "react";
2 import "./App.css";
3
4 function App() {
5   const [time, setTime] = useState(new Date());
6   const [showDate, setShowDate] = useState(false);
7   const toggleShowDate = () => {
8     setShowDate(!showDate);
9   };
10  setInterval(() => [
11    setTime(new Date()),
12    , 1000);
13
14  return (
15    <div className="App">
16      <h1>Digital Clock</h1>
17      <div className="clock">
18        <h1>{time.toLocaleTimeString()}</h1>
19        {showDate && <p>{time.toDateString()}</p>}
20      </div>
21      <div className="toggle-wrapper">
22        <label className="switch">
23          <input type="checkbox" onClick={toggleShowDate} />
24          <span className="slider round"></span>
25        </label>
26        <p>Show Date</p>
27      </div>
28    );
29
30  export default App;
```

Ln 11, Col 25 Spaces: 2 UTF-8 ⓘ JavaScript ⓘ Go Live ⓘ Q

Type here to search ENG 14:28 29-04-2023 ⓘ

App.css



```
File Edit Selection View Go Run Terminal Help
# App.css M X
clock > src > # App.css > App
1 .App {
2   text-align: center;
3   background-color: #f5f5f5;
4   height: 100vh;
5   display: flex;
6   flex-direction: column;
7   justify-content: center;
8   align-items: center;
9 }
10 .clock {
11   background-color: #ffff;
12   border-radius: 50%;
13   width: 300px;
14   height: 300px;
15   display: flex;
16   flex-direction: column;
17   justify-content: center;
18   align-items: center;
19   box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
20 }
21 .clock h1 {
22   font-size: 64px;
23   margin-bottom: 0;
24   color: #rgb(88, 223, 223);
25 }
26 .clock p {
27   font-size: 20px;
28   margin-top: 0;
29   color: #rgb(88, 223, 223);
30 }
31 .toggle-wrapper {
32   display: flex;
```

Ln 1, Col 1 Spaces: 2 UTF-8 LF CSS ⓘ Go Live ⓘ Q

Type here to search ENG 14:29 29-04-2023 ⓘ

File Edit Selection View Go Run Terminal Help

EXPLORER

ASSIGNMENT

- > calculator
- < clock
- > node_modules
- > public
- < src
 - # App.css
 - JS App.js
 - JS App.test.js
 - # index.css
 - JS index.js
 - logo.svg
 - JS reportWebVitals.js
 - JS setupTests.js
 - .gitignore
 - { package-lock.json
 - { package.json
 - README.md
- > employee
- > employee2
- > hello-world
- > hello-worlda
- > node_modules
- > router6
- > weather

{ package-lock.json

{ package.json

> OUTLINE

> TIMELINE

App.css - Assignment - Visual Studio Code

```
clock > src > # App.css > App.css
```

```
30 }
31 }
32 .toggle-wrapper {
33   display: flex;
34   align-items: center;
35   margin-top: 20px;
36 }

37 .switch {
38   position: relative;
39   display: inline-block;
40   width: 60px;
41   height: 34px;
42   margin-right: 10px;
43 }

44 .switch input {
45   opacity: 0;
46   width: 0;
47   height: 0;
48 }

49 .slider {
50   position: absolute;
51   cursor: pointer;
52   top: 0;
53   left: 0;
54   right: 0;
55   bottom: 0;
56   background-color: #ccc;
57   -webkit-transition: 0.4s;
58   transition: 0.4s;
59   border-radius: 34px;
60 }
```

Ln 1, Col 1 Spaces: 2 UTF-8 LF CSS Go Live ⚙ 14:30 29-04-2023

Type here to search

File Edit Selection View Go Run Terminal Help

EXPLORER

ASSIGNMENT

- > calculator
- < clock
- > node_modules
- > public
- < src
 - # App.css
 - JS App.js
 - JS App.test.js
 - # index.css
 - JS index.js
 - logo.svg
 - JS reportWebVitals.js
 - JS setupTests.js
 - .gitignore
 - { package-lock.json
 - { package.json
 - README.md
- > employee
- > employee2
- > hello-world
- > hello-worlda
- > node_modules
- > router6
- > weather

{ package-lock.json

{ package.json

> OUTLINE

> TIMELINE

App.css - Assignment - Visual Studio Code

```
clock > src > # App.css > App.css
```

```
51 transition: 0.4s;
52 border-radius: 34px;
53 }
54 }

55 .slider:before {
56   position: absolute;
57   content: "";
58   height: 26px;
59   width: 26px;
60   left: 4px;
61   bottom: 4px;
62   background-color: white;
63   -webkit-transition: 0.4s;
64   transition: 0.4s;
65   border-radius: 50%;
66 }

67 input:checked + .slider {
68   background-color: #2196f3;
69 }

71 input:focus + .slider {
72   box-shadow: 0 0 1px #2196f3;
73 }

76 input:checked + .slider:before {
77   -webkit-transform: translateX(26px);
78   -ms-transform: translateX(26px);
79   transform: translateX(26px);
80 }

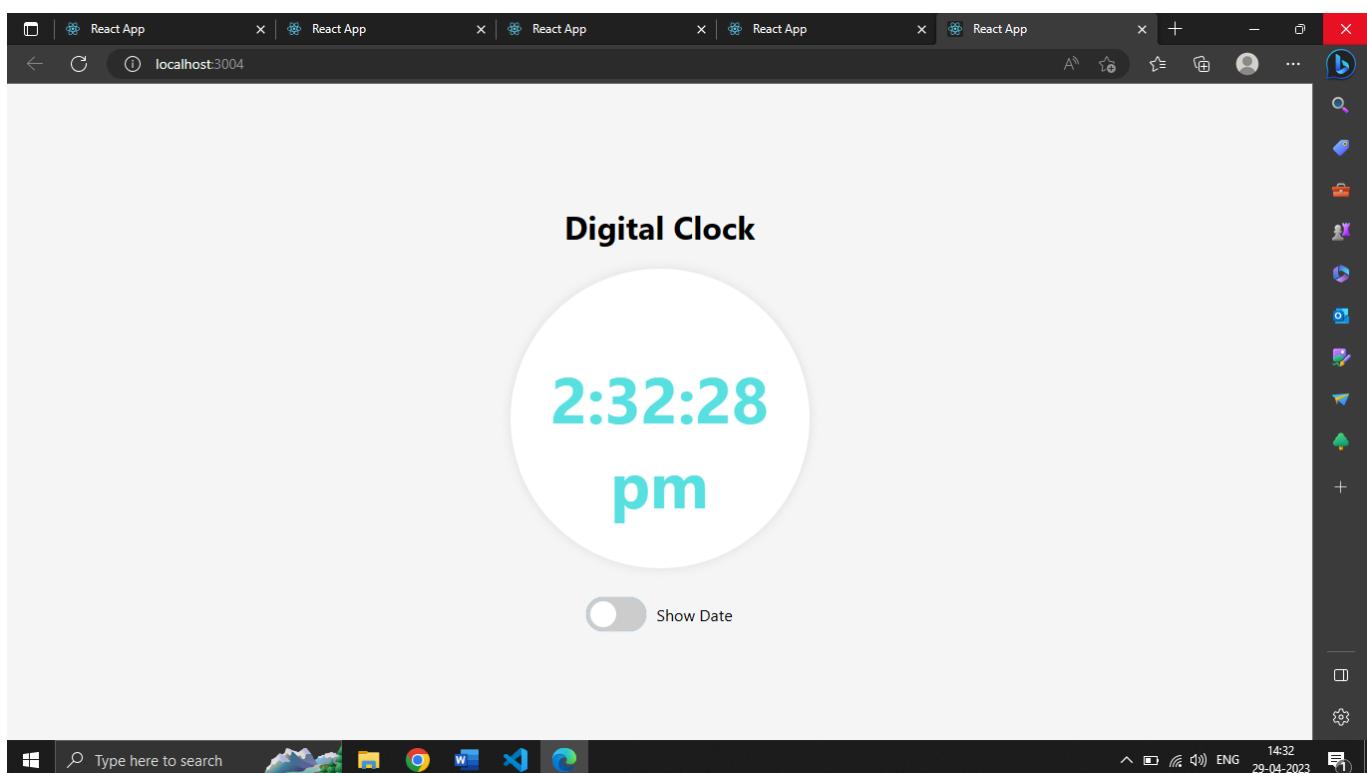
83 .slider.round {
84   border-radius: 34px;
85 }
```

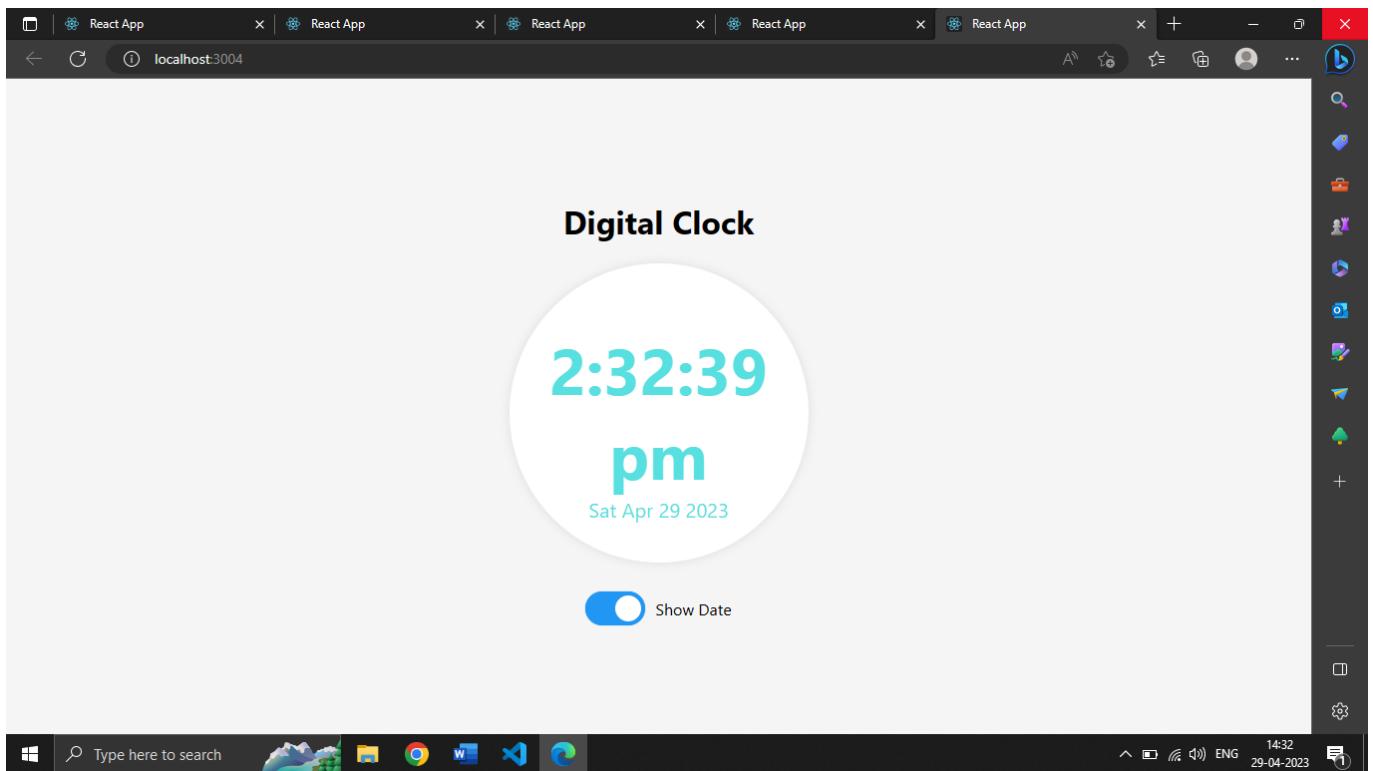
Ln 1, Col 1 Spaces: 2 UTF-8 LF CSS Go Live ⚙ 14:30 29-04-2023

Type here to search

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left lists files and folders, including 'ASSIGNMENT' containing 'calculator', 'clock', 'node_modules', 'public', and 'src'. Inside 'src', there are files like 'App.js', 'App.css', 'index.js', 'logo.svg', 'reportWebVitals.js', 'setupTests.js', '.gitignore', 'package-lock.json', 'package.json', and 'README.md'. The 'clock' folder contains 'App.css' which is open in the code editor. The code in 'App.css' is a CSS file for a digital clock component, defining styles for a circular container and its content.

```
clock > src > # App.css > App.css
  83 |   input:checked + .slider:before {
  84 |     box-shadow: 0 0 1px #2196f3;
  85 |
  86 |   input:checked + .slider:after {
  87 |     -webkit-transform: translateX(26px);
  88 |     -ms-transform: translateX(26px);
  89 |     transform: translateX(26px);
  90 |
  91 |   .slider.round {
  92 |     border-radius: 34px;
  93 |
  94 |   .slider.round:after {
  95 |     border-radius: 50%;
```





6) Create React application to demonstrate routing. Create a navigation bar with 4 links.

Navigate between the links.

App.js

```
File Edit Selection View Go Run Terminal Help App.js - Assignment - Visual Studio Code

EXPLORER
ASSIGNMENT
calculator
clock
employee
employee2
hello-world
hello-worlds
node_modules
router6
public
src
component
# App.css
JS App.js M
JS App.test.js
# index.css
JS index.js
logo.svg
JS reportWebVitals.js
JS setupTests.js
.gitignore
() package-lock.json M
() package.json M
 README.md
weather
{} package-lock.json
OUTLINE
TIMELINE

router6 > src > JS App.js > App
  1 import './App.css';
  2 import React from "react";
  3 import { BrowserRouter, Routes, Route } from "react-router-dom";
  4 import Home from "./component/Home";
  5 import Projects from "./component/Projects";
  6 import Services from "./component/Services";
  7 import Contact from "./component/Contact";
  8 import NavBar from "./component/NavBar";
  9 function App() {
10   return (
11     <div className='App'>
12       <BrowserRouter>
13         <NavBar />
14         <Routes>
15           <Route path="/" element={<Home />} />
16           <Route path="/projects" element={<Projects />} />
17           <Route path="/services" element={<Services />} />
18           <Route path="/contact" element={<Contact />} />
19         </Routes>
20       </BrowserRouter>
21     </div>
22   );
23 }
24
25 export default App;
26
```

master* 0 0 △ 0 Type here to search 14:36 29-04-2023

App.css

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure under "ASSIGNMENT" with files like calculator, clock, employee, employee2, hello-world, hello-worlda, node_modules, router6, public, component, App.css, App.js, App.test.js, index.css, index.js, logo.svg, reportWebVitals.js, setupTests.js, .gitignore, package-lock.json, package.json, and README.md.
- Code Editor:** Displays the content of the App.css file:

```
.navbar {  
    background-color: #4d86d5;  
    overflow: hidden;  
}  
  
.navbar a {  
    float: left;  
    color: #f2f2f2;  
    text-align: center;  
    padding: 14px 16px;  
    text-decoration: none;  
    font-size: 17px;  
}  
  
.navbar a:hover {  
    background-color: #ddd;  
    color: black;  
}  
  
.navbar a.active {  
    background-color: #5b0656;  
    color: white;  
}  
  
h1{  
    text-align: center;  
}
```
- Bottom Status Bar:** Shows "Ln 1, Col 1" and "Spaces: 2" and "UTF-8".
- Taskbar:** Shows icons for File Explorer, Search, Task List, and others.

Components

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure under "ASSIGNMENT" with files like calculator, clock, employee, employee2, hello-world, hello-worlda, node_modules, router6, public, component, Contact.js, Home.js, NavBar.js, Projects.js, Services.js, App.css, App.js, App.test.js, index.css, index.js, logo.svg, reportWebVitals.js, setupTests.js, and .gitignore.
- Code Editor:** Displays the content of the Contact.js file:

```
import React from "react";  
  
function Contact() {  
    return<>  
        <h1>You Choose Contact!</h1>  
    </></>  
}  
  
export default Contact;
```
- Bottom Status Bar:** Shows "Ln 1, Col 1" and "Spaces: 2" and "UTF-8".
- Taskbar:** Shows icons for File Explorer, Search, Task List, and others.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure. The `src` folder contains `component`, `Contact.js`, `Home.js` (which is currently selected), `NavBar.js`, `Projects.js`, and `Services.js`. Other files like `App.css`, `index.css`, and `reportWebVitals.js` are also listed.
- Code Editor (Center):** Displays the `Home.js` file content:

```
import React from "react";
function Home() {
  return(
    <h1>You Choose Home!</h1>
  )
}
export default Home;
```
- Status Bar (Bottom):** Shows the file path as `Home.js - Assignment - Visual Studio Code`, the current position as `Ln 1, Col 1`, and the file type as `JavaScript`.

File Edit Selection View Go Run Terminal Help Projects.js - Assignment - Visual Studio Code

EXPLORER ... JS App.js M # App.css M JS Home.js U JS NavBar.js U JS Projects.js U X

ASSIGNMENT router6 > src > component > JS Projects.js > ...

```
1 import React from "react";
2
3 function Projects() {
4     return<>
5         <h1>You Choose Projects!</h1>;
6     </>>
7
8 export default Projects;
```

calculator
clock
employee
employee2
hello-world
hello-worlda
node_modules
router6
node_modules
public
src
component
Contact.js
Home.js
NavBar.js
Projects.js
Services.js
App.css
App.js
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore

> OUTLINE
> TIMELINE

master* Type here to search 14:38 29-04-2023

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF {} JavaScript Go Live ⚡

File Edit Selection View Go Run Terminal Help Services.js - Assignment - Visual Studio Code

EXPLORER ... JS App.js M # App.css M JS Home.js U JS NavBar.js U JS Projects.js U X JS Services.js U X

ASSIGNMENT router6 > src > component > JS Services.js > ...

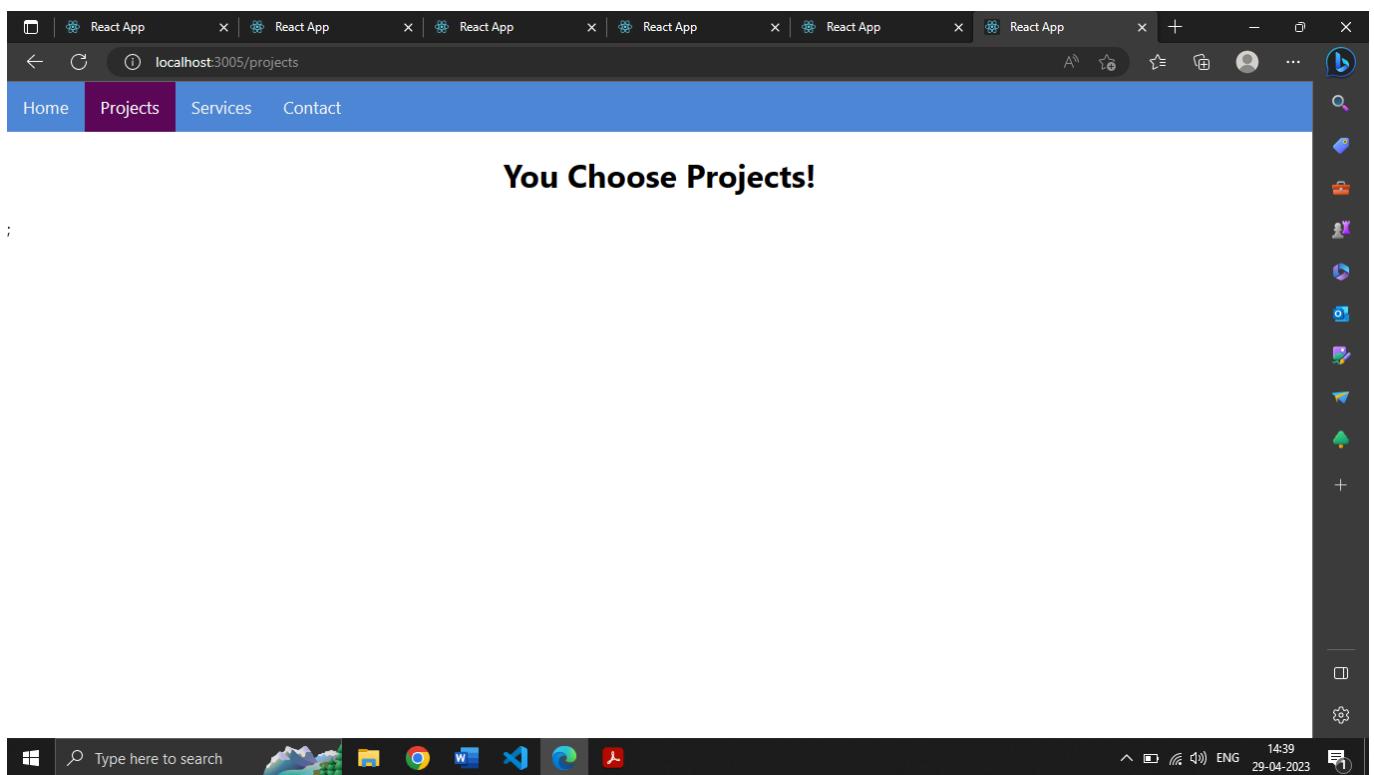
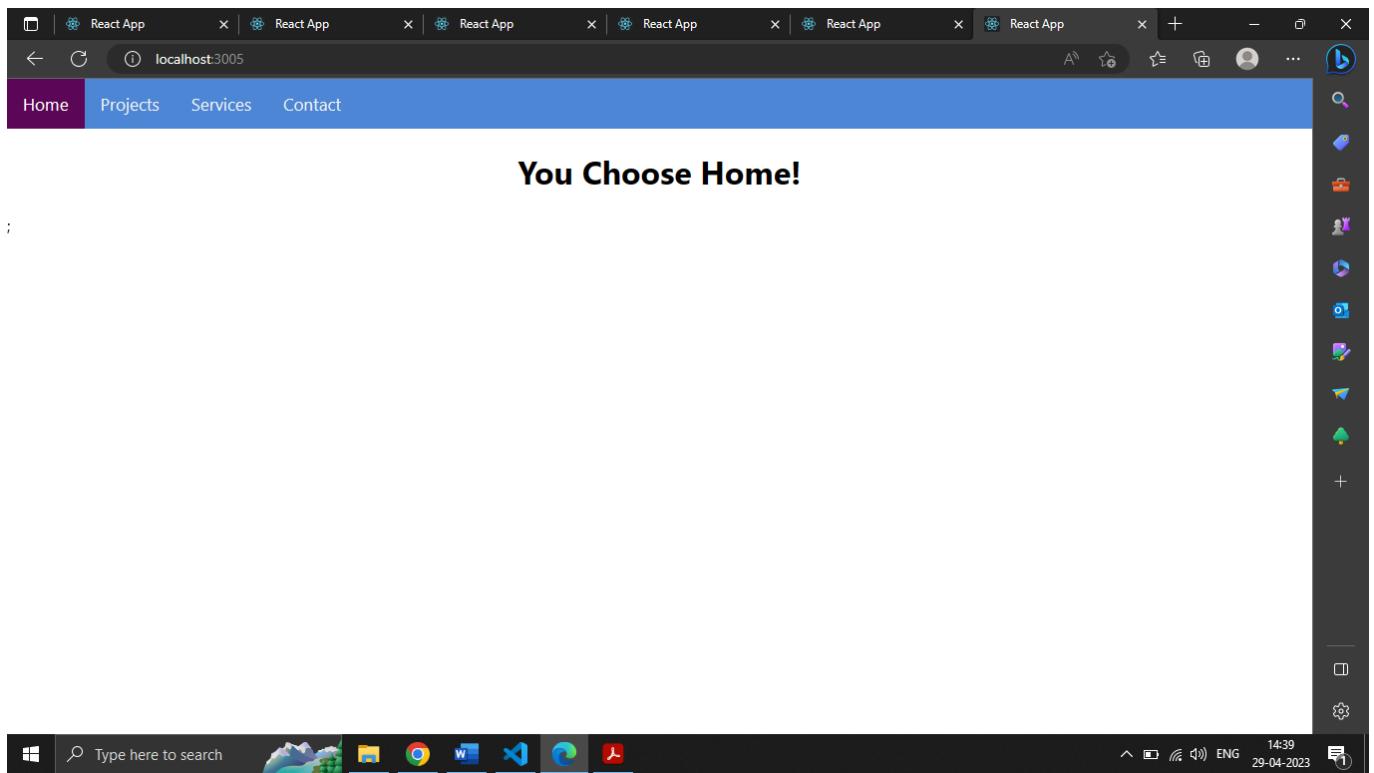
```
1 import React from "react";
2
3 function Services() {
4     return<>
5         <h1>You Choose Services!</h1>;
6     </>>
7
8 export default Services;
```

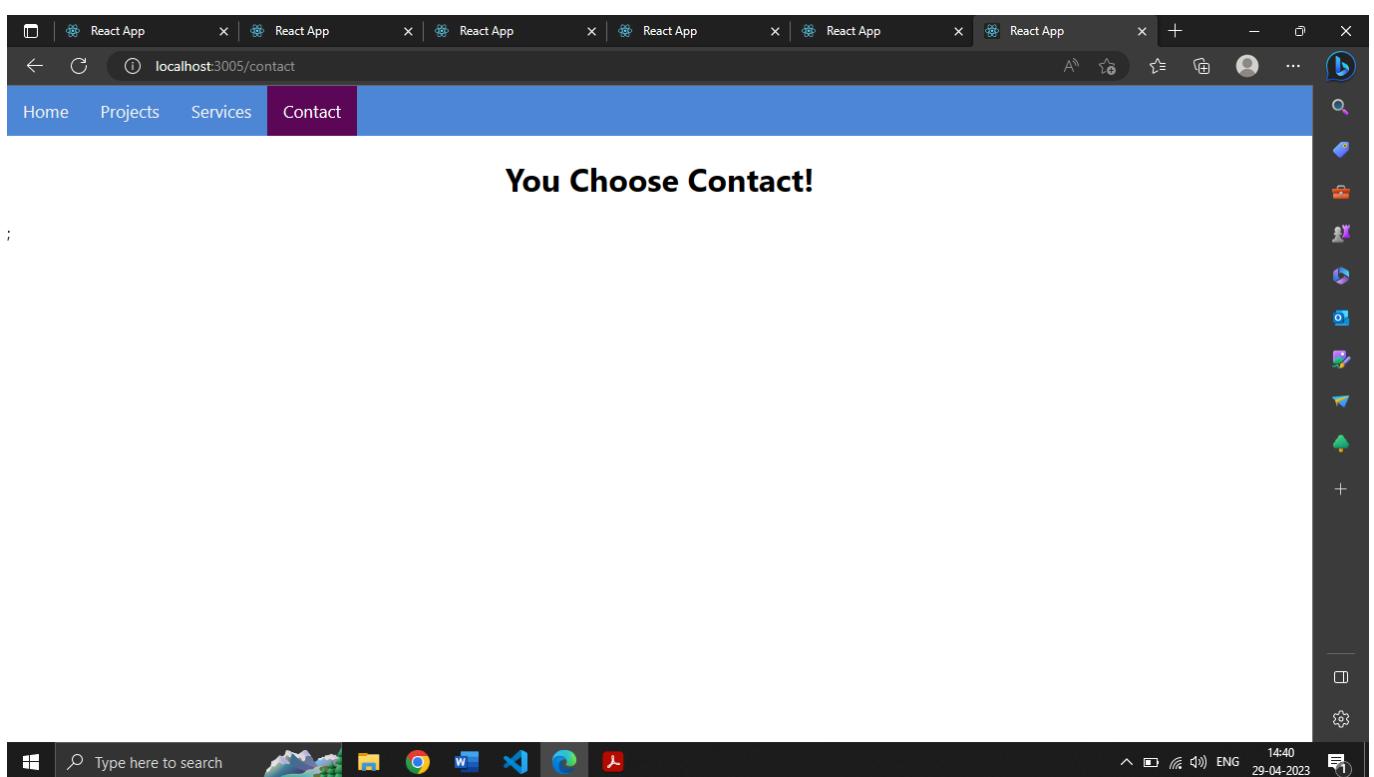
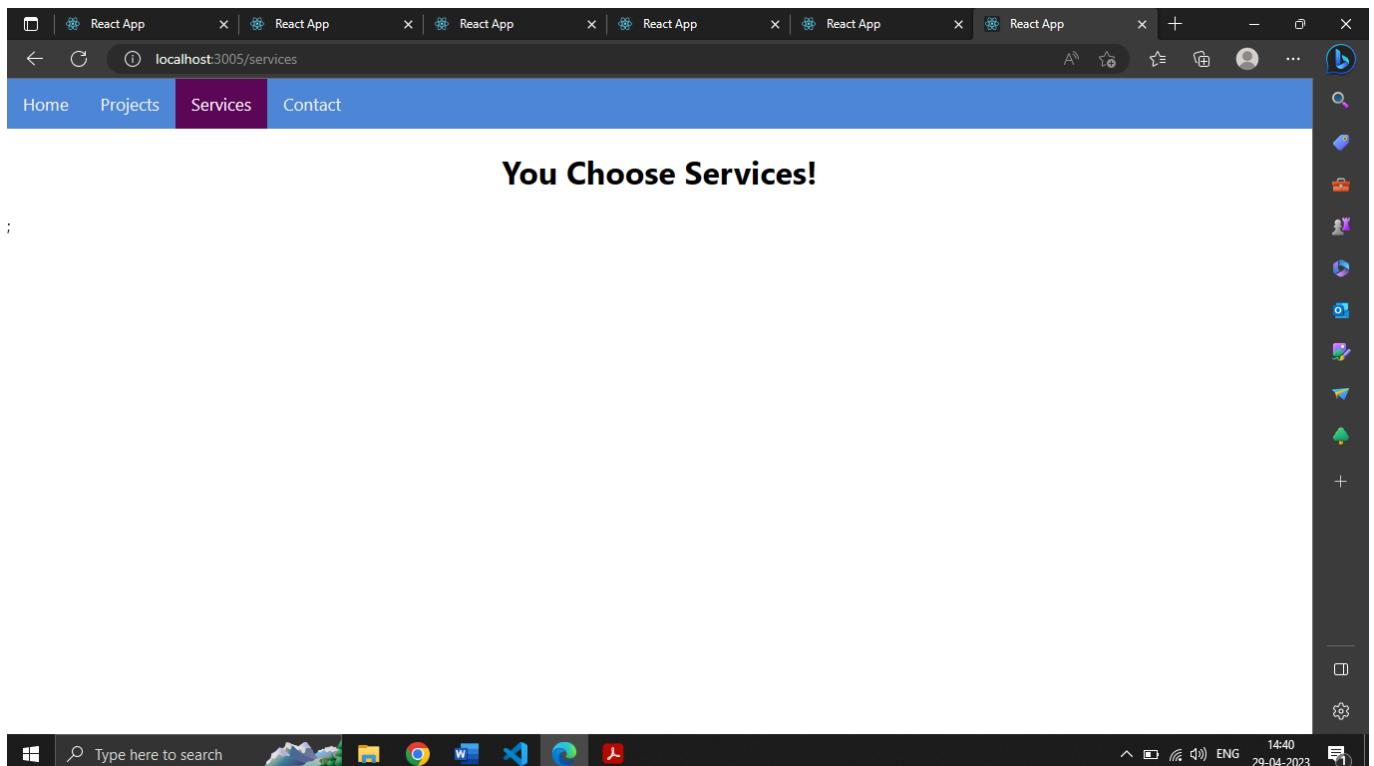
calculator
clock
employee
employee2
hello-world
hello-worlda
node_modules
router6
node_modules
public
src
component
Contact.js
Home.js
NavBar.js
Projects.js
Services.js
App.css
App.js
App.test.js
index.css
index.js
logo.svg
reportWebVitals.js
setupTests.js
.gitignore

> OUTLINE
> TIMELINE

master* Type here to search 14:38 29-04-2023

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF {} JavaScript Go Live ⚡





7) Demonstration of consuming services over HTTP. Design a weather application that displays the current weather, daily forecasts, and hourly forecasts based on your current geolocation.

The weather application is composed of the following components:

- Header - A heading that displays application title

- WeatherDashboard - The primary (root) component that manages state for all underlying components. It is also responsible for connecting to and retrieving data from a weather and geolocation service.
- CurrentWeatherDisplay - Displays weather information for the current point in time based on current location.
- DailyWeatherDisplay - Displays a 7 day weather forecast in the form of a scrollable carousel.
- DailyWeatherForecastCard - Displays weather summary for a given day
- HourlyWeatherDisplay - Displays a 24 hour weather forecast in the form of a scrollable carousel.
- HourlyWeatherForecastCard - Displays weather summary for a given hour

The following services are used to obtain weather and location data:

- WeatherService - A wrapper that is responsible for integrating with the OpenWeather API. It provides an interface that allows one to obtain current weather, daily forecast, and hourly forecast information.
- GeolocationService - A wrapper that is responsible for integrating with the Google Geolocation API. It provides an interface that allows one to obtain the current GPS coordinates. These coordinates are used by the WeatherService to obtain weather information.

Features:

- Display current weather
- Display 7 day weather forecast
- Display 24 hour weather forecast

APIs should be used in project

1) Open weather API : <https://openweathermap.org/>

2) Google Geolocation API: <https://developers.google.com/maps/documentation/geolocation/intro>

CurrentWeatherDisplay.js

The screenshot shows the Visual Studio Code interface with the file `CurrentWeatherDisplay.js` open in the editor. The code defines a component that returns a `div` with a `h1` and a `h2` element if there is weather data available.

```
1 import React from "react";
2 const CurrentWeatherDisplay = ({ weather }) => {
3   return (
4     <div className="current-weather">
5       {weather && (
6         <React.Fragment>
7           <h1>Current Weather</h1>
8           <h2>{weather.temperature} °C</h2>
9           <p>{weather.description}</p>
10        </React.Fragment>
11      )}
12    </div>
13  );
14}
15 export default CurrentWeatherDisplay;
```

The Explorer sidebar shows the project structure with files like `DailyWeatherDisplay.js`, `DailyWeatherForecastCard.js`, and `Header.js`. The bottom status bar indicates the file is in JavaScript mode, has 14:48 remaining, and was last modified on 29-04-2023.

Header.js

The screenshot shows the Visual Studio Code interface with the file `Header.js` open in the editor. The code defines a component that returns a `div` with a `h1` element containing the text "Weather App".

```
1 const Header = () => {
2   return (
3     <div className="header">
4       <h1>Weather App</h1>
5     </div>
6   );
7 }
8 export default Header;
```

The Explorer sidebar shows the project structure with files like `CurrentWeatherDisplay.js`, `DailyWeatherDisplay.js`, and `DailyWeatherForecastCard.js`. The bottom status bar indicates the file is in JavaScript mode, has 14:48 remaining, and was last modified on 29-04-2023.

DailyWeatherDisplay.js

A screenshot of Visual Studio Code showing the code editor with the file `DailyWeatherDisplay.js` open. The code implements a component that displays daily weather forecast cards. It checks if there is daily weather data and maps it to cards if present, or shows a message if none.

```
1 import React from 'react';
2 import DailyWeatherForecastCard from './DailyWeatherForecastCard';
3
4 const DailyWeatherDisplay = ({ dailyWeather }) => {
5   return (
6     <div className="daily-weather-display">
7       <h2>7-Day Forecast</h2>
8       {dailyWeather && dailyWeather.length > 0 ? (
9         <div className="daily-weather-cards">
10           {dailyWeather.map((forecast, index) => (
11             <DailyWeatherForecastCard key={index} forecast={forecast} />
12           ))
13         </div>
14       ) : (
15         <div>No daily weather data available</div>
16       )
17     </div>
18   );
19 };
20
21 export default DailyWeatherDisplay;
```

DailyWeatherForecastCard.js

A screenshot of Visual Studio Code showing the code editor with the file `DailyWeatherForecastCard.js` open. The code defines a component that takes a forecast object and displays its details. It calculates the date, minimum temperature, maximum temperature, and the URL for the weather icon based on the forecast data.

```
1 import React from 'react';
2 import moment from 'moment';
3
4 const DailyWeatherForecastCard = ({ forecast }) => {
5   const date = moment.unix(forecast.dt).format('ddd, MMMM D');
6   const minTemp = Math.round(forecast.temp.min);
7   const maxTemp = Math.round(forecast.temp.max);
8   const iconUrl = `https://openweathermap.org/img/w/${forecast.weather[0].icon}.png`;
9
10   return (
11     <div className="daily-forecast-card">
12       <div className="daily-forecast-date">{date}</div>
13       <div className="daily-forecast-icon">
14         <img src={iconUrl} alt={forecast.weather[0].description} />
15       </div>
16       <div className="daily-forecast-temp">
17         {(maxTemp - minTemp) / 5}°
18       </div>
19       <div className="daily-forecast-description">{forecast.weather[0].description}</div>
20     </div>
21   );
22 }
23
24 export default DailyWeatherForecastCard;
```

HourlyWeatherDisplay.js

The screenshot shows the Visual Studio Code interface with the file 'HourlyWeatherDisplay.js' open. The code defines a component that maps weather data to hourly forecast cards.

```
1 import HourlyWeatherForecastCard from './HourlyWeatherForecastCard';
2 const HourlyWeatherDisplay = ({ weather }) => {
3   return (
4     <div className="hourly-weather">
5       <h3>Hourly Forecast</h3>
6       <div className="hourly-forecast">
7         {weather.map((hour) => (
8           <HourlyWeatherForecastCard key={hour.date} weather={hour} />
9         ))}
10      </div>
11    </div>
12  );
13}
14 export default HourlyWeatherDisplay;
```

The Explorer sidebar shows the project structure, including files like 'CurrentWeatherDisplay.js', 'Header.js', 'DailyWeatherDisplay.js', 'DailyWeatherForecastCard.js', and 'HourlyWeatherDisplay.js'. The bottom status bar shows the file is in master branch, has 0 changes, and is in JavaScript mode.

HourlyWeatherForecastCard.js

The screenshot shows the Visual Studio Code interface with the file 'HourlyWeatherForecastCard.js' open. The code defines a component that displays weather information in a card format.

```
1 const HourlyWeatherForecastCard = ({ weather }) => {
2   return (
3     <div className="hourly-forecast-card">
4       <p>{weather.time}</p>
5       <p>{weather.description}</p>
6       <p>{weather.temperature} °C</p>
7     </div>
8   );
9 }
10 export default HourlyWeatherForecastCard;
```

The Explorer sidebar shows the project structure, including files like 'CurrentWeatherDisplay.js', 'Header.js', 'DailyWeatherDisplay.js', 'DailyWeatherForecastCard.js', and 'HourlyWeatherDisplay.js'. The bottom status bar shows the file is in master branch, has 0 changes, and is in JavaScript mode.

WeatherDashboard.js

This screenshot shows the Visual Studio Code interface with the file `WeatherDashboard.js` open. The code implements a weather dashboard component using React. It imports `React`, `GeolocationService`, `WeatherService`, and `CurrentWeatherDisplay`. The component state includes latitude, longitude, currentWeather, dailyWeather, and hourlyWeather. It uses `componentDidMount` to get the user's location and then calls `getWeather` with the position. The `getWeather` function uses `GeolocationService` to get the city name and `WeatherService` to get current, daily, and hourly weather information.

```
1 import React from "react";
2 import GeolocationService from "../services/GeolocationService";
3 import WeatherService from "../services/WeatherService";
4 import CurrentWeatherDisplay from "./CurrentWeatherDisplay";
5 import DailyWeatherDisplay from "./DailyWeatherDisplay";
6 import HourlyWeatherDisplay from "./HourlyWeatherDisplay";
7
8 class WeatherDashboard extends React.Component {
9   constructor(props) {
10     super(props);
11     this.state = {
12       latitude: null,
13       longitude: null,
14       currentWeather: null,
15       dailyWeather: [],
16       hourlyWeather: []
17     };
18   }
19
20   componentDidMount() {
21     this.getLocation();
22   }
23
24   getLocation = () => {
25     if (navigator.geolocation) {
26       navigator.geolocation.getCurrentPosition(this.getWeather);
27     } else {
28       alert("Geolocation is not supported by this browser.");
29     }
30   };
31
32   getWeather = (position) => {
33     const latitude = position.coords.latitude;
34     const longitude = position.coords.longitude;
35
36     // geolocation API to get location
37     GeolocationService.getCityName(latitude, longitude)
38       .then((city) => {
39         this.setState({
40           city: city,
41           latitude: latitude,
42           longitude: longitude
43         });
44
45         // weather API to get current, daily, and hourly weather information
46         WeatherService.getCurrentWeather(latitude, longitude)
47           .then((currentWeather) => {
48             this.setState({ currentWeather: currentWeather });
49           });
50
51         WeatherService.getDailyWeather(latitude, longitude)
52           .then((dailyWeather) => {
53             this.setState({ dailyWeather: dailyWeather });
54           });
55
56         WeatherService.getHourlyWeather(latitude, longitude)
57           .then((hourlyWeather) => {
58             this.setState({ hourlyWeather: hourlyWeather });
59           });
60       })
61       .catch((error) => {
62         console.log(error);
63       });
64   };
65 }
```

This screenshot shows the same Visual Studio Code interface with additional code added to the `getWeather` function. The new code uses `WeatherService` to get daily and hourly weather information, and then sets the state with the results. The code is identical to the one in the first screenshot, but the additional code is now present.

```
1 import React from "react";
2 import GeolocationService from "../services/GeolocationService";
3 import WeatherService from "../services/WeatherService";
4 import CurrentWeatherDisplay from "./CurrentWeatherDisplay";
5 import DailyWeatherDisplay from "./DailyWeatherDisplay";
6 import HourlyWeatherDisplay from "./HourlyWeatherDisplay";
7
8 class WeatherDashboard extends React.Component {
9   constructor(props) {
10     super(props);
11     this.state = {
12       latitude: null,
13       longitude: null,
14       currentWeather: null,
15       dailyWeather: [],
16       hourlyWeather: []
17     };
18   }
19
20   componentDidMount() {
21     this.getLocation();
22   }
23
24   getLocation = () => {
25     if (navigator.geolocation) {
26       navigator.geolocation.getCurrentPosition(this.getWeather);
27     } else {
28       alert("Geolocation is not supported by this browser.");
29     }
30   };
31
32   getWeather = (position) => {
33     const latitude = position.coords.latitude;
34     const longitude = position.coords.longitude;
35
36     // geolocation API to get location
37     GeolocationService.getCityName(latitude, longitude)
38       .then((city) => {
39         this.setState({
40           city: city,
41           latitude: latitude,
42           longitude: longitude
43         });
44
45         // weather API to get current, daily, and hourly weather information
46         WeatherService.getCurrentWeather(latitude, longitude)
47           .then((currentWeather) => {
48             this.setState({ currentWeather: currentWeather });
49           });
50
51         WeatherService.getDailyWeather(latitude, longitude)
52           .then((dailyWeather) => {
53             this.setState({ dailyWeather: dailyWeather });
54           });
55
56         WeatherService.getHourlyWeather(latitude, longitude)
57           .then((hourlyWeather) => {
58             this.setState({ hourlyWeather: hourlyWeather });
59           });
60       })
61       .catch((error) => {
62         console.log(error);
63       });
64   };
65 }
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure under the "weather" folder, including components like CurrentWeatherDisplay.js, DailyWeatherDisplay.js, DailyWeatherForecastCard.js, Header.js, HourlyWeatherDisplay.js, HourlyWeatherForecastCard.js, and WeatherDashboard.js.
- Code Editor (Center):** Displays the code for WeatherDashboard.js. The code uses React components and the WeatherService to fetch weather data.
- Bottom Status Bar:** Shows the file path "weather > src > components > JS WeatherDashboard.js", line 1, column 1, spaces: 2, encoding: UTF-8, and a Go Live button.

```
play.js U JS DailyWeatherForecastCard.js U JS HourlyWeatherDisplay.js U JS HourlyWeatherForecastCard.js U JS WeatherDashboard.js U
... weather > src > components > JS WeatherDashboard.js > ...
54      });
55      WeatherService.getHourlyWeather(latitude, longitude)
56        .then((hourlyWeather) => {
57          this.setState({ hourlyWeather: hourlyWeather });
58        });
59      })
60    );
61    .catch((error) => {
62      console.log(error);
63    });
64  }
65
66 render() {
67   return (
68     <div className="weather-dashboard">
69       <CurrentWeatherDisplay weather={this.state.currentWeather} />
70       <DailyWeatherDisplay weather={this.state.dailyWeather} />
71       <HourlyWeatherDisplay weather={this.state.hourlyWeather} />
72     </div>
73   );
74 }
75
76 export default WeatherDashboard;
```

GeolocationService.js

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure under the "weather" folder, including components like CurrentWeatherDisplay.js, DailyWeatherDisplay.js, DailyWeatherForecastCard.js, Header.js, HourlyWeatherDisplay.js, HourlyWeatherForecastCard.js, and WeatherDashboard.js, along with services like GeolocationService.js and WeatherService.js.
- Code Editor (Center):** Displays the code for GeolocationService.js. It defines a class GeolocationService with methods to get current position and city name using the Geocoding API.
- Bottom Status Bar:** Shows the file path "weather > src > services > JS GeolocationService.js", line 1, column 1, spaces: 2, encoding: UTF-8, and a Go Live button.

```
forecastCard.js U JS HourlyWeatherDisplay.js U JS HourlyWeatherForecastCard.js U JS WeatherDashboard.js U JS GeolocationService.js U
... weather > src > services > JS GeolocationService.js > ...
1  const API_KEY = 'a43988fafe6d496fb2e60939afe1a6c1';
2
3  class GeolocationService {
4    async getCurrentPosition() {
5      return new Promise((resolve, reject) => {
6        navigator.geolocation.getCurrentPosition(resolve, reject);
7      });
8    }
9
10   async getCityName() {
11     try {
12       const position = await this.getCurrentPosition();
13       const { latitude, longitude } = position.coords;
14       const url = `https://api.opencagedata.com/geocode/v1/json?key=${API_KEY}&q=${latitude},${longitude}&pretty`;
15       const response = await fetch(url);
16       const data = await response.json();
17       if (data.results && data.results.length > 0) {
18         const { components } = data.results[0];
19         if (components && components.city) {
20           return components.city;
21         }
22       }
23     } catch (error) {
24       console.error(`Unable to retrieve city name: ${error}`);
25     }
26     return null;
27   }
28
29   export default new GeolocationService();
```

WeatherService.js

File Edit Selection View Go Run Terminal Help WeatherService.js - Assignment - Visual Studio Code

```
1 import moment from 'moment';
2 class WeatherService {
3     static API_KEY = "2c3ef3dd2b56f85c90c1ac1f69f5c30b";
4
5     static getCurrentWeather(latitude, longitude) {
6         const url = `https://api.openweathermap.org/data/2.5/weather?lat=${latitude}&lon=${longitude}&appid=${this.API_KEY}&units=metric`;
7         return fetch(url)
8             .then((response) => response.json())
9             .then((data) => {
10                 return {
11                     temperature: data.main.temp,
12                     description: data.weather[0].description
13                 };
14             });
15     }
16     static getHourlyWeather(latitude, longitude) {
17         const url = `https://api.openweathermap.org/data/2.5/forecast?lat=${latitude}&lon=${longitude}&appid=${this.API_KEY}&units=metric`;
18         return fetch(url)
19             .then((response) => response.json())
20             .then((data) => {
21                 return data.list.map((hour) => {
22                     return {
23                         time: moment.unix(hour.dt).format("h:mm a"),
24                         description: hour.weather[0].description,
25                         temperature: hour.main.temp
26                     };
27                 });
28             });
29     }
30
31     static getDailyWeather(latitude, longitude) {
32         const url = `https://api.openweathermap.org/data/2.5/forecast/daily?lat=${latitude}&lon=${longitude}&cnt=7&appid=${this.API_KEY}&units=metric`;
33         return fetch(url)
34             .then((response) => response.json())
35             .then((data) => {
36                 return data.list.map((day) => {
37                     return {
38                         date: moment.unix(day.dt).format("ddd, MMM D"),
39                         description: day.weather[0].description,
40                         maxTemp: day.temp.max,
41                         minTemp: day.temp.min
42                     };
43                 });
44             });
45         }
46     }
47
48     }
49 }
50 export default WeatherService;
```

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF () JavaScript Go Live 14:52 29-04-2023

File Edit Selection View Go Run Terminal Help WeatherService.js - Assignment - Visual Studio Code

```
1 import moment from 'moment';
2 class WeatherService {
3     static API_KEY = "2c3ef3dd2b56f85c90c1ac1f69f5c30b";
4
5     static getCurrentWeather(latitude, longitude) {
6         const url = `https://api.openweathermap.org/data/2.5/weather?lat=${latitude}&lon=${longitude}&appid=${this.API_KEY}&units=metric`;
7         return fetch(url)
8             .then((response) => response.json())
9             .then((data) => {
10                 return {
11                     time: moment.unix(hour.dt).format("h:mm a"),
12                     description: hour.weather[0].description,
13                     temperature: hour.main.temp
14                 };
15             });
16     }
17     static getHourlyWeather(latitude, longitude) {
18         const url = `https://api.openweathermap.org/data/2.5/forecast?lat=${latitude}&lon=${longitude}&appid=${this.API_KEY}&units=metric`;
19         return fetch(url)
20             .then((response) => response.json())
21             .then((data) => {
22                 return data.list.map((hour) => {
23                     return {
24                         time: moment.unix(hour.dt).format("h:mm a"),
25                         description: hour.weather[0].description,
26                         temperature: hour.main.temp
27                     };
28                 });
29             });
30     }
31     static getDailyWeather(latitude, longitude) {
32         const url = `https://api.openweathermap.org/data/2.5/forecast/daily?lat=${latitude}&lon=${longitude}&cnt=7&appid=${this.API_KEY}&units=metric`;
33         return fetch(url)
34             .then((response) => response.json())
35             .then((data) => {
36                 return data.list.map((day) => {
37                     return {
38                         date: moment.unix(day.dt).format("ddd, MMM D"),
39                         description: day.weather[0].description,
40                         maxTemp: day.temp.max,
41                         minTemp: day.temp.min
42                     };
43                 });
44             });
45         }
46     }
47     static getWeatherForecast(latitude, longitude) {
48         const url = `https://api.openweathermap.org/data/2.5/forecast?lat=${latitude}&lon=${longitude}&appid=${this.API_KEY}&units=metric`;
49         return fetch(url)
50             .then((response) => response.json())
51             .then((data) => {
52                 return data.list.map((forecast) => {
53                     return {
54                         date: moment.unix(forecast.dt).format("ddd, MMM D"),
55                         description: forecast.weather[0].description,
56                         maxTemp: forecast.temp.max,
57                         minTemp: forecast.temp.min
58                     };
59                 });
60             });
61         }
62     }
63     static getLocationServices() {
64         return new Promise((resolve, reject) => {
65             if ('geolocation' in navigator) {
66                 navigator.geolocation.getCurrentPosition(resolve, reject);
67             } else {
68                 reject('Geolocation is not supported by your browser');
69             }
70         });
71     }
72 }
73 export default WeatherService;
```

Ln 45, Col 9 Spaces: 2 UTF-8 CRLF () JavaScript Go Live 14:53 29-04-2023

App.js

The screenshot shows the Visual Studio Code interface with the file `App.js` open. The code implements a React application for displaying weather information. It uses `useState` and `useEffect` hooks to handle geolocation and city name retrieval. The code is well-structured with comments explaining the logic.

```
weather > src > JS App.js > ...
1 import React, { useState, useEffect } from 'react';
2 import './App.css';
3 import WeatherDashboard from './components/WeatherDashboard';
4 import GeolocationService from './services/GeolocationService';
5
6 function App() {
7   const [latitude, setLatitude] = useState(null);
8   const [longitude, setLongitude] = useState(null);
9   const [cityName, setCityName] = useState(null);
10
11   useEffect(() => {
12     navigator.geolocation.getCurrentPosition((position) => {
13       setLatitude(position.coords.latitude);
14       setLongitude(position.coords.longitude);
15     });
16   }, []);
17
18   useEffect(() => {
19     if (latitude && longitude) {
20       GeolocationService.getCityName(latitude, longitude)
21         .then((city) => setCityName(city))
22         .catch((error) => console.error(error));
23     }
24   }, [latitude, longitude]);
25
26   return (
27     <div className="App">
28       <header className="App-header">
29         <h1>Weather App</h1>
30       </header>
31       <div className="weather-dashboard">
32         {cityName ?
33           <WeatherDashboard
34             latitude={latitude}
35             longitude={longitude}
36             cityName={cityName}
37           /> :
38           <p>Loading...</p>
39         }
40       </div>
41     );
42   }
43
44   export default App;
45 
```

Below the code editor, the status bar displays: Ln 1, Col 1 Spaces: 2 UTF-8 CRLF () JavaScript Go Live 14:43 29-04-2023

This screenshot shows the same Visual Studio Code interface, but the code in `App.js` has been modified. The `return` statement now contains a single conditional block that only renders the `WeatherDashboard` component if a city name is available. The loading state is no longer present.

```
weather > src > JS App.js > ...
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45 
```

Below the code editor, the status bar displays: Ln 1, Col 1 Spaces: 2 UTF-8 CRLF () JavaScript Go Live 14:44 29-04-2023

App.css

The screenshot shows the Visual Studio Code interface with the file `App.css` open in the center editor pane. The code is a CSS file with the following content:

```
/* Header styles */
.App-header {
  color: #rgb(210, 128, 21);
  text-align: center;
  font-size: 2rem;
  margin: 1rem;
}

/* WeatherDashboard styles */
.weather-dashboard {
  display: flex;
  flex-direction: column;
  align-items: center;
  font-size: 1rem;
  margin: 10px;
}

/* CurrentWeatherDisplay styles */
.current-weather {
  font-size: 1rem;
  display: inline-flex;
  width: 255px;
  text-align: center;
  height: 200px;
  flex-direction: column;
  align-items: center;
  margin: 10px;
  padding: 10px;
  background-color: #f0f0f0;
  border-radius: 0.5rem;
  box-shadow: 0 0.5rem 1rem rgba(0, 0, 0, 0.2);
}
```

The left sidebar shows a project structure under the `ASSIGNMENT` folder, including files like `index.js`, `App.js`, and `App.test.js`. The bottom status bar indicates the file is 1444 bytes large and was last modified on 29-04-2023.

The screenshot shows the Visual Studio Code interface with the file `App.css` open in the center editor pane. The code has been modified to remove specific styling rules:

```
/* Header styles */
.App-header {
  color: #rgb(210, 128, 21);
  text-align: center;
  font-size: 2rem;
  margin: 1rem;
}

/* DailyWeatherDisplay styles */
.daily-weather {
  display: flex;
  flex-direction: column;
  align-items: center;
  margin: 1rem;
}

.daily-weather h2 {
  font-size: 1.5rem;
}

.daily-weather-display .carousel {
  display: flex;
  flex-wrap: nowrap;
  overflow-x: auto;
  padding: 0.5rem;
}

.daily-weather-forecast-card {
  display: flex;
  flex-direction: column;
  align-items: center;
  margin: 0.5rem;
  padding: 1rem;
  background-color: #f0f0f0;
  border-radius: 0.5rem;
}
```

The left sidebar shows the same project structure as the first screenshot. The bottom status bar indicates the file is 1444 bytes large and was last modified on 29-04-2023.

File Edit Selection View Go Run Terminal Help

EXPLORER

ASSIGNMENT

- > router6
- < weather
- > node_modules
- > public
- < src
 - < components
 - JS CurrentWeatherDisplay.js
 - JS DailyWeatherDisplay.js
 - JS DailyWeatherForecastCard.js
 - JS Header.js
 - JS HourlyWeatherDisplay.js
 - JS HourlyWeatherForecastCard.js
 - JS WeatherDashboard.js
 - < services
 - JS GeolocationService.js
 - JS WeatherService.js
 - # App.css M
 - JS App.js M
 - JS App.test.js
 - # index.css
 - JS index.js M
 - logo.svg
 - JS reportWebVitals.js
 - JS setupTests.js
 - .gitignore
 - { package-lock.json M
- > OUTLINE
- > TIMELINE

App.css - Assignment - Visual Studio Code

```
weather > src > # App.css > ...
  border-radius: 0.5rem;
  box-shadow: 0 0.5rem 1rem rgba(0, 0, 0, 0.2);
}

.daily-forecast-card p {
  font-size: 1.2rem;
}

/* HourlyWeatherDisplay styles */
.hourly-weather-display {
  display: flex;
  flex-direction: column;
  align-items: center;
  margin: 1rem;
}

.hourly-weather h2 {
  font-size: 1.5rem;
  text-align: center;
  justify-content: center;
  display: flex;
}

.hourly-weather .carousel {
  display: flex;
  flex-wrap: nowrap;
  overflow-x: auto;
  padding: 0.5rem;
}

.hourly-forecast-card {
  display: inline-flex;
  width: 135px;
}
```

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF CSS Go Live ⚙ 14:45 29-04-2023

Type here to search

File Edit Selection View Go Run Terminal Help

EXPLORER

ASSIGNMENT

- > router6
- < weather
- > node_modules
- > public
- < src
 - < components
 - JS CurrentWeatherDisplay.js
 - JS DailyWeatherDisplay.js
 - JS DailyWeatherForecastCard.js
 - JS Header.js
 - JS HourlyWeatherDisplay.js
 - JS HourlyWeatherForecastCard.js
 - JS WeatherDashboard.js
 - < services
 - JS GeolocationService.js
 - JS WeatherService.js
 - # App.css M
 - JS App.js M
 - JS App.test.js
 - # index.css
 - JS index.js M
 - logo.svg
 - JS reportWebVitals.js
 - JS setupTests.js
 - .gitignore
 - { package-lock.json M
- > OUTLINE
- > TIMELINE

App.css - Assignment - Visual Studio Code

```
weather > src > # App.css > ...
  }
}

.hourly-weather .carousel {
  display: flex;
  flex-wrap: nowrap;
  overflow-x: auto;
  padding: 0.5rem;
}

.hourly-forecast-card {
  display: inline-flex;
  width: 135px;
  text-align: center;
  height: 200px;
  flex-direction: column;
  align-items: center;
  margin: 0.5rem;
  padding: 1rem;
  background-color: #f0f0f0;
  border-radius: 0.5rem;
  box-shadow: 0 0.5rem 1rem rgba(0, 0, 0, 0.2);
}

.hourly-forecast-card p {
  font-size: 1.2rem;
}
```

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF CSS Go Live ⚙ 14:45 29-04-2023

Type here to search

localhost:3006

Weather App

Current Weather

27.88 °C

scattered clouds

7-Day Forecast

No daily weather data available

Hourly Forecast

5:30 pm	8:30 pm	11:30 pm	2:30 am	5:30 am	8:30 am
broken clouds	broken clouds	overcast clouds	light rain	overcast clouds	broken clouds
29.27 °C	25.63 °C	23.66 °C	23.94 °C	23.16 °C	26.43 °C

localhost:3006

No daily weather data available

Hourly Forecast

5:30 pm	8:30 pm	11:30 pm	2:30 am	5:30 am	8:30 am
broken clouds	broken clouds	overcast clouds	light rain	overcast clouds	broken clouds
29.27 °C	25.63 °C	23.66 °C	23.94 °C	23.16 °C	26.43 °C

11:30 am	2:30 pm	5:30 pm	8:30 pm	11:30 pm	2:30 am
scattered clouds	overcast clouds	broken clouds	few clouds	light rain	clear sky
30.82 °C	31.78 °C	31.75 °C	27.79 °C	24.43 °C	23.16 °C

5:30 am	8:30 am	11:30 am	2:30 pm	5:30 pm	8:30 pm