TRAINOSYS

REACTJS & TYPESCRIPT

Day 3

HYBRID
TECHNOLOGY
COURSES

TRAINOSYS

Course Overview

- Introduction to ReactJS and TypeScript (Day 1)
- React Component Development (Day 2)
- React Routing and Data Fetching (Day 3)
- State Management with Redux and TypeScript (Day 4)
- Advanced TypeScript and Project Development (Day 5)

TRAINOSYS

Program

Day 02

- Recap of React component structure and lifecycle methods
- Creating functional components with TypeScript
- Working with props and prop types in TypeScript
- State management in React components using hooks with TypeScript
- Building a simple React component with TypeScript
- Handling events and form inputs in TypeScript-based components

Day 03

- Introduction to React Router for handling client-side routing
- Configuring routes in a TypeScript-based React Application
- Navigating between different routes with React Router
- Fetching data from APIs using TypeScript and React
- Displaying data fetched from an API in React components

TRAINOSYS

RECAP

HYBRID
TECHNOLOGY
COURSES

TRAINOSYS

REACT ROUTER

HYBRID
TECHNOLOGY
COURSES

TRAINOSYS

Multi-page applications

- Multiple HTML files, browser handles navigation
- Server-side processing is common
- Full page refresh during navigation
- Suitable for content-heavy websites
- Easier for beginners

Single-Page Applications (SPAs)

- Single HTML file, React handles navigation
- Client-side rendering using JavaScript frameworks
- Dynamic navigation within the same page
- Smoother user experience after initial load
- Great for interactive web applications
- SEO challenges, but improving
- App-like, seamless user experience

React Router Package

- You can use a package to make a Single Page Application in React
- npm install react-router-dom

React Router Example Code

```
import { BrowserRouter, Route, Routes } from 'react-router-dom';


<BrowserRouter>
  <Routes>
    <Route path="/" component={HomePage} />
    <Route path="/shop" component={ShopPage} />
    <Route path="/cart" component={CartPage} />
  </Routes>
</BrowserRouter>
```

TRAINOSYS

CODE ALONG

HYBRID
TECHNOLOGY
COURSES

TRAINOSYS

Generate a new project

```
npm create vite@latest / npm create vite@4.1.0
```

TRAINOSYS

Generate a new project

- ❑ Project Name: pokedex-react-router
- ❑ > React
- ❑ > TypeScript
- ❑ > cd pokedex-react-router

Install packages

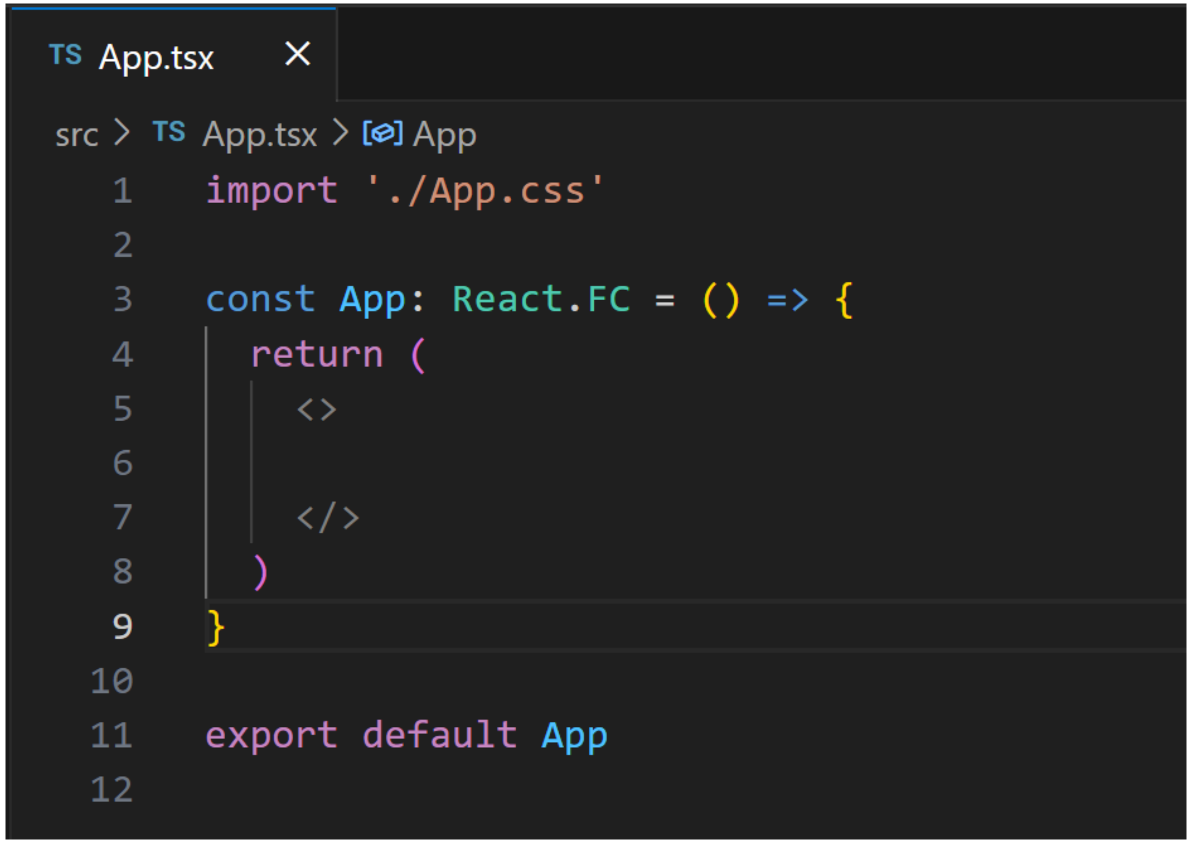
React Router Dom and Axios

```
npm install react-router-dom axios
```

Generate a new project

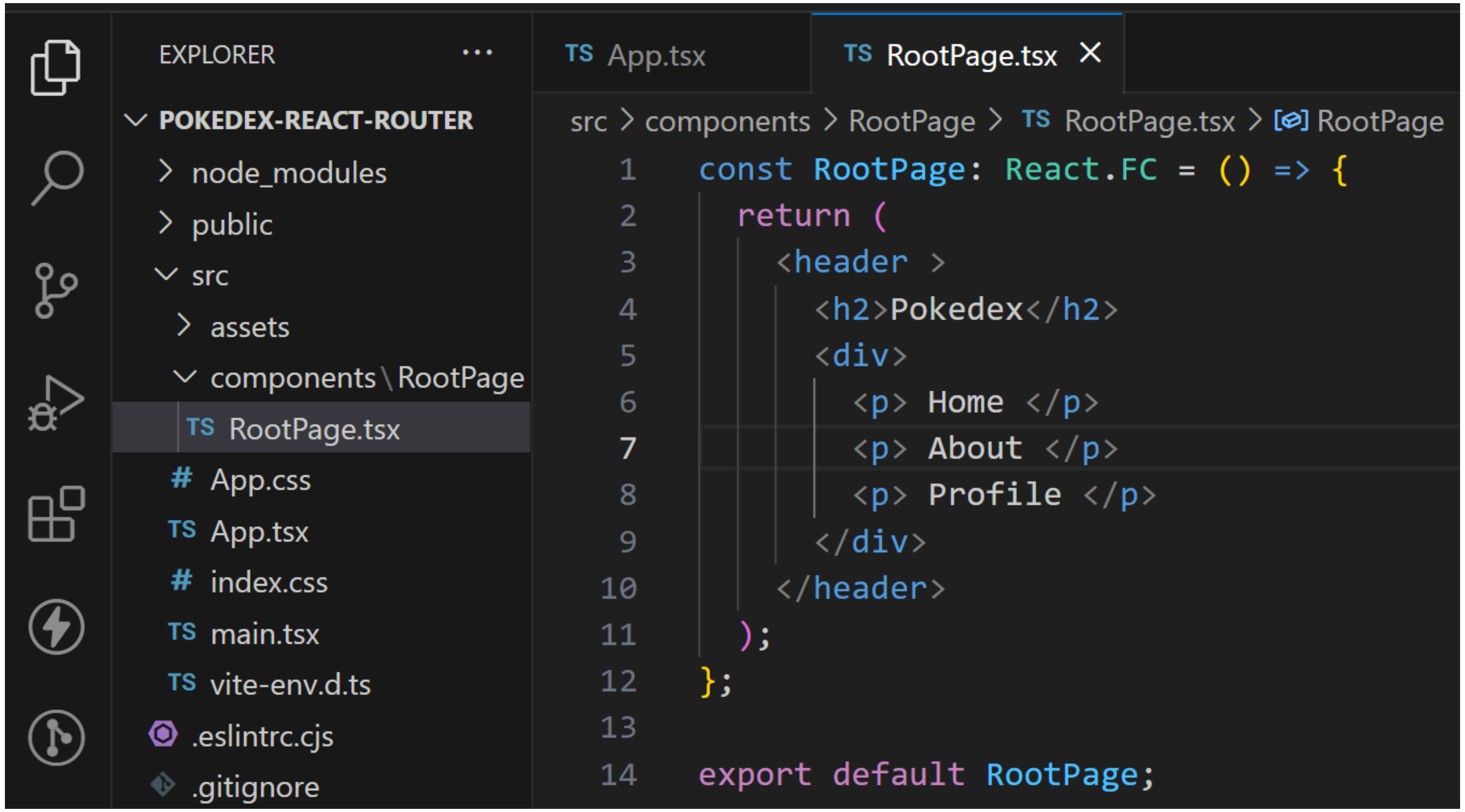
- ❑ > npm run dev

Delete jsx in App.tsx and other unnecessary lines of code

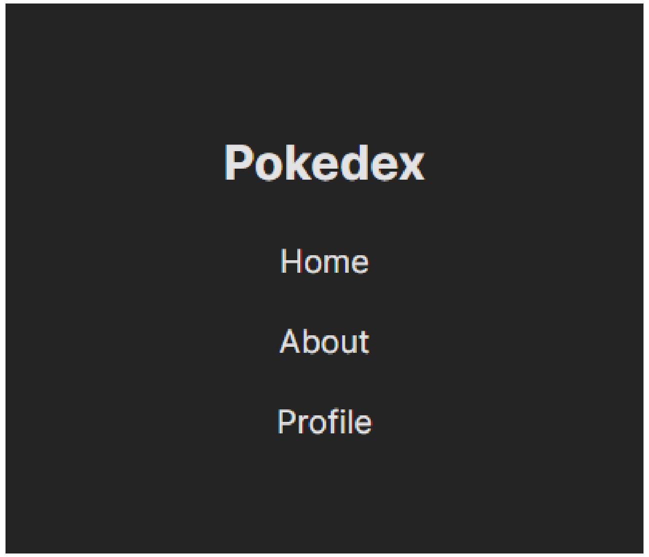


```tsx
import './App.css'

const App: React.FC = () => {
  return (
    <>


    </>
  )
}

export default App

```

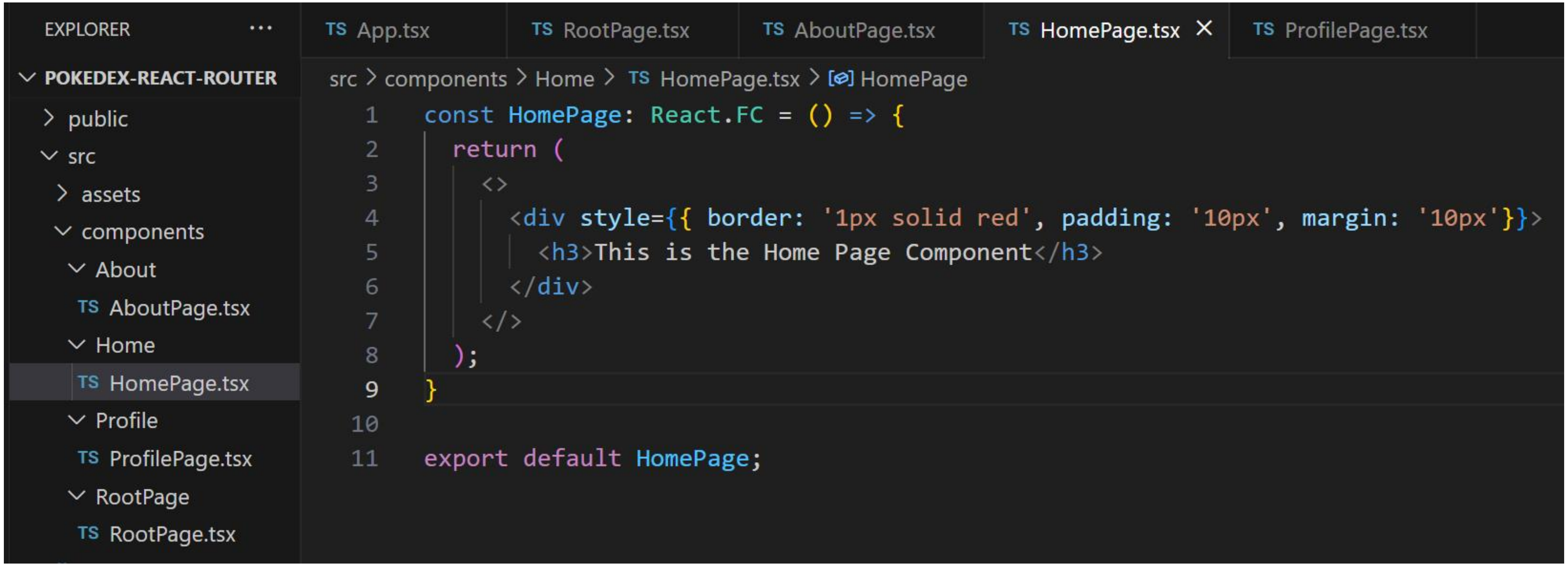
Create components folder, *RootPage* folder and <*RootPage* />



```tsx
const RootPage: React.FC = () => {
  return (
    <header >
      <h2>Pokedex</h2>
      <div>
        <p> Home </p>
        <p> About </p>
        <p> Profile </p>
      </div>
    </header>
  );
};

export default RootPage;
```



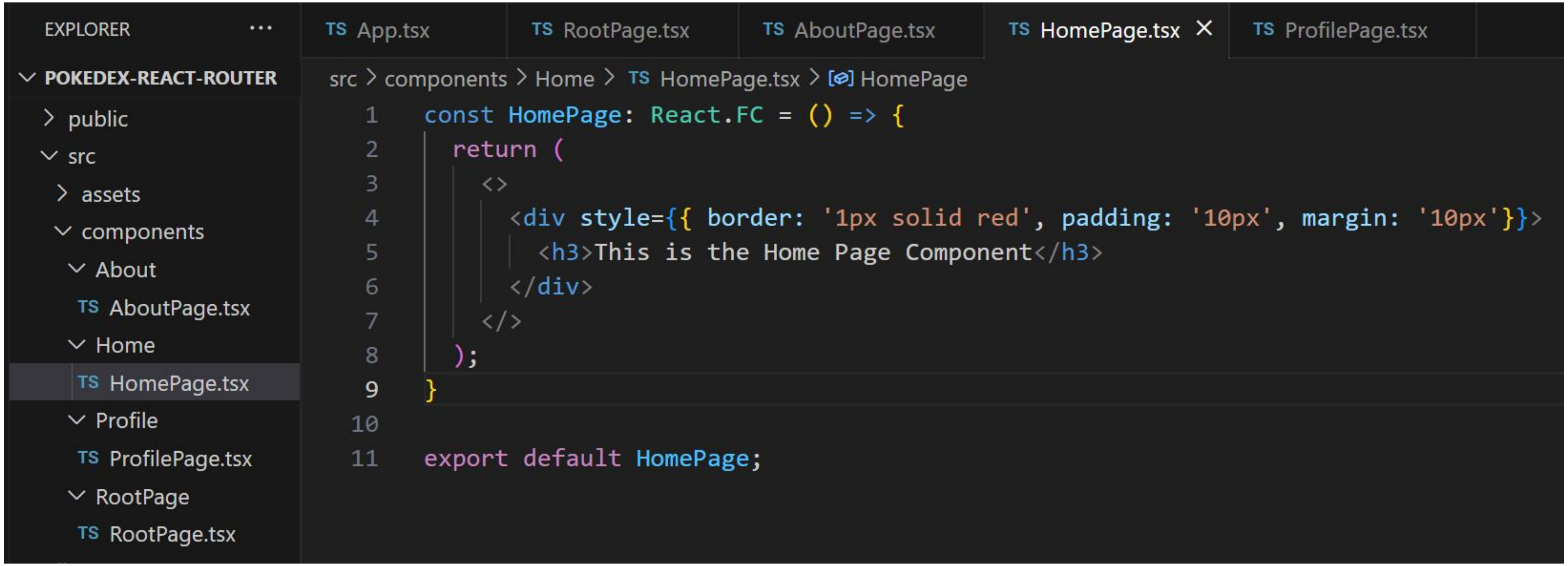
Create folders and files for *<Home />, <About /> and <Profile />*



```tsx
const HomePage: React.FC = () => {
  return (
    <>
      <div style={{ border: '1px solid red', padding: '10px', margin: '10px'}}>
        <h3>This is the Home Page Component</h3>
      </div>
    </>
  );
}

export default HomePage;
```

Create folders and files for *<Home />, <About /> and <Profile />*

TRAINOSYS

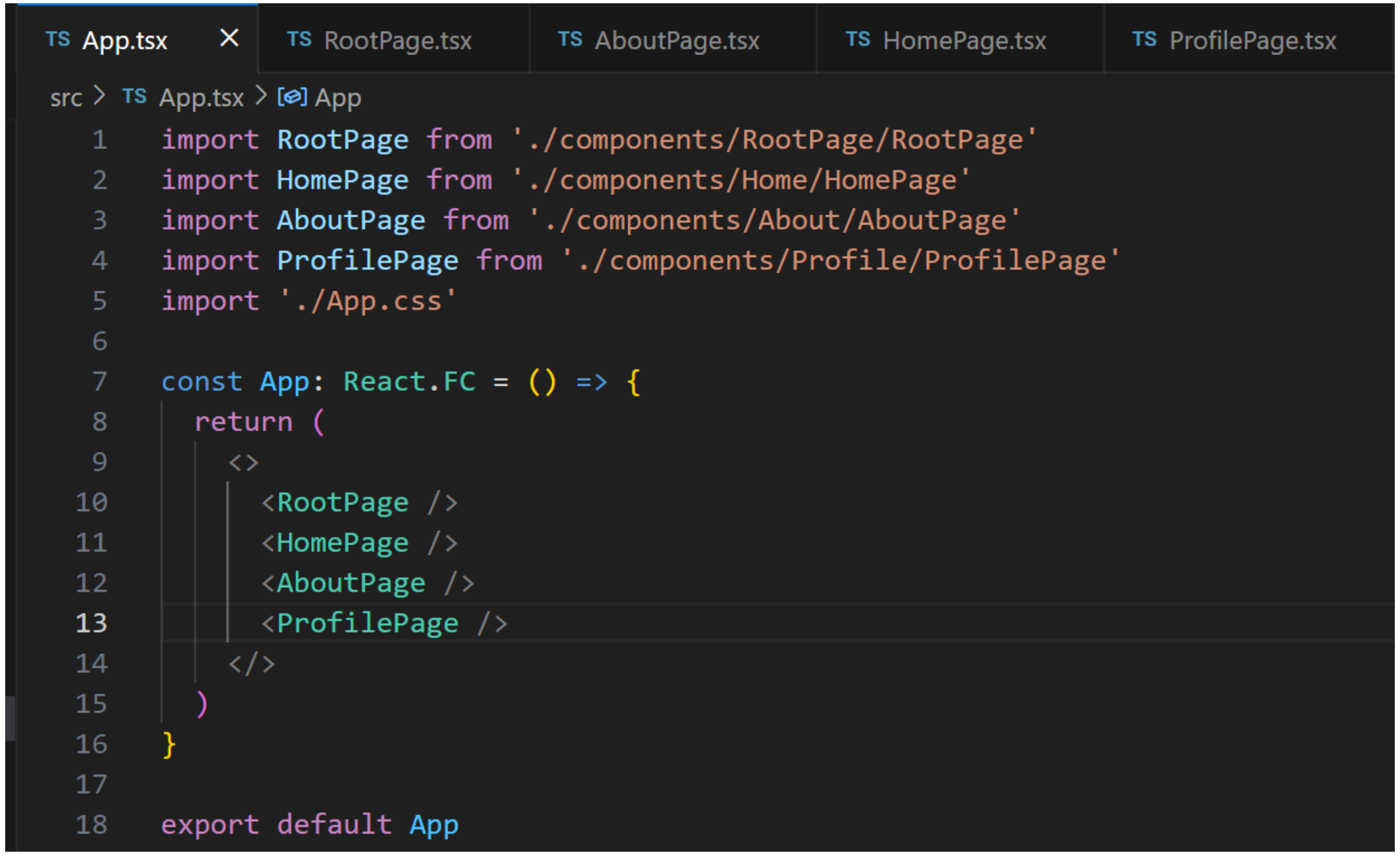


```tsx
const HomePage: React.FC = () => {
  return (
    <>
      <div style={{ border: '1px solid red', padding: '10px', margin: '10px'}}>
        <h3>This is the Home Page Component</h3>
      </div>
    </>
  );
}

export default HomePage;
```

TRAINING THE FUTURE TODAY
www.trainosys.com

HYBRID
TECHNOLOGY
COURSES

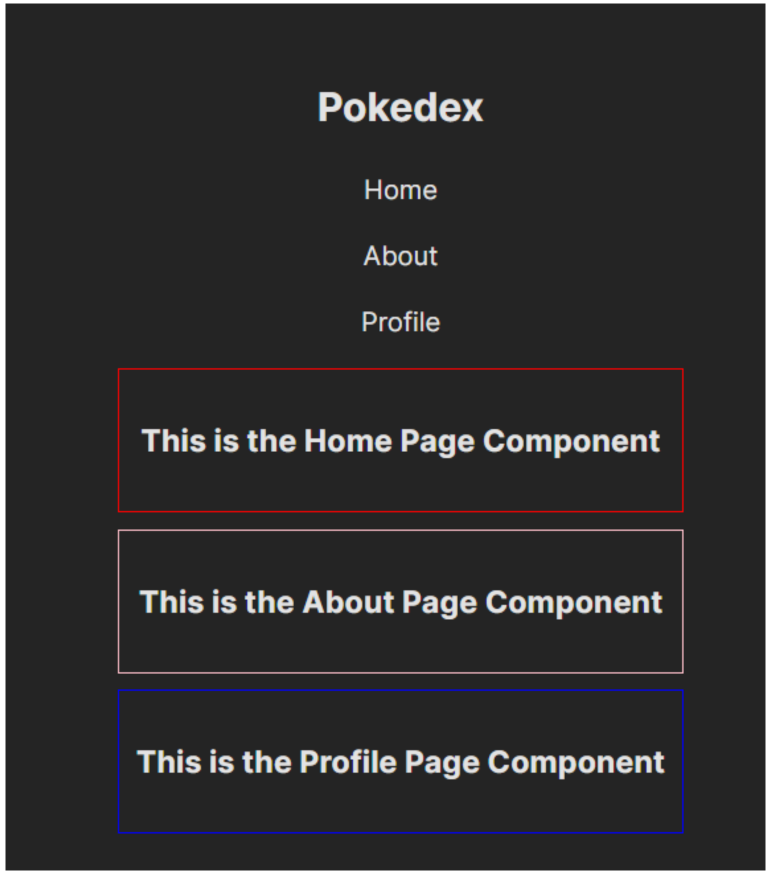
Import components in App.tsx



TS App.tsx × | TS RootPage.tsx | TS AboutPage.tsx | TS HomePage.tsx | TS ProfilePage.tsx

src > TS App.tsx > App

```tsx
import RootPage from './components/RootPage/RootPage'
import HomePage from './components/Home/HomePage'
import AboutPage from './components/About/AboutPage'
import ProfilePage from './components/Profile/ProfilePage'
import './App.css'

const App: React.FC = () => {
  return (
    <>
      <RootPage />
      <HomePage />
      <AboutPage />
      <ProfilePage />
    </>
  )
}

export default App
```



Pokedex

Home

About

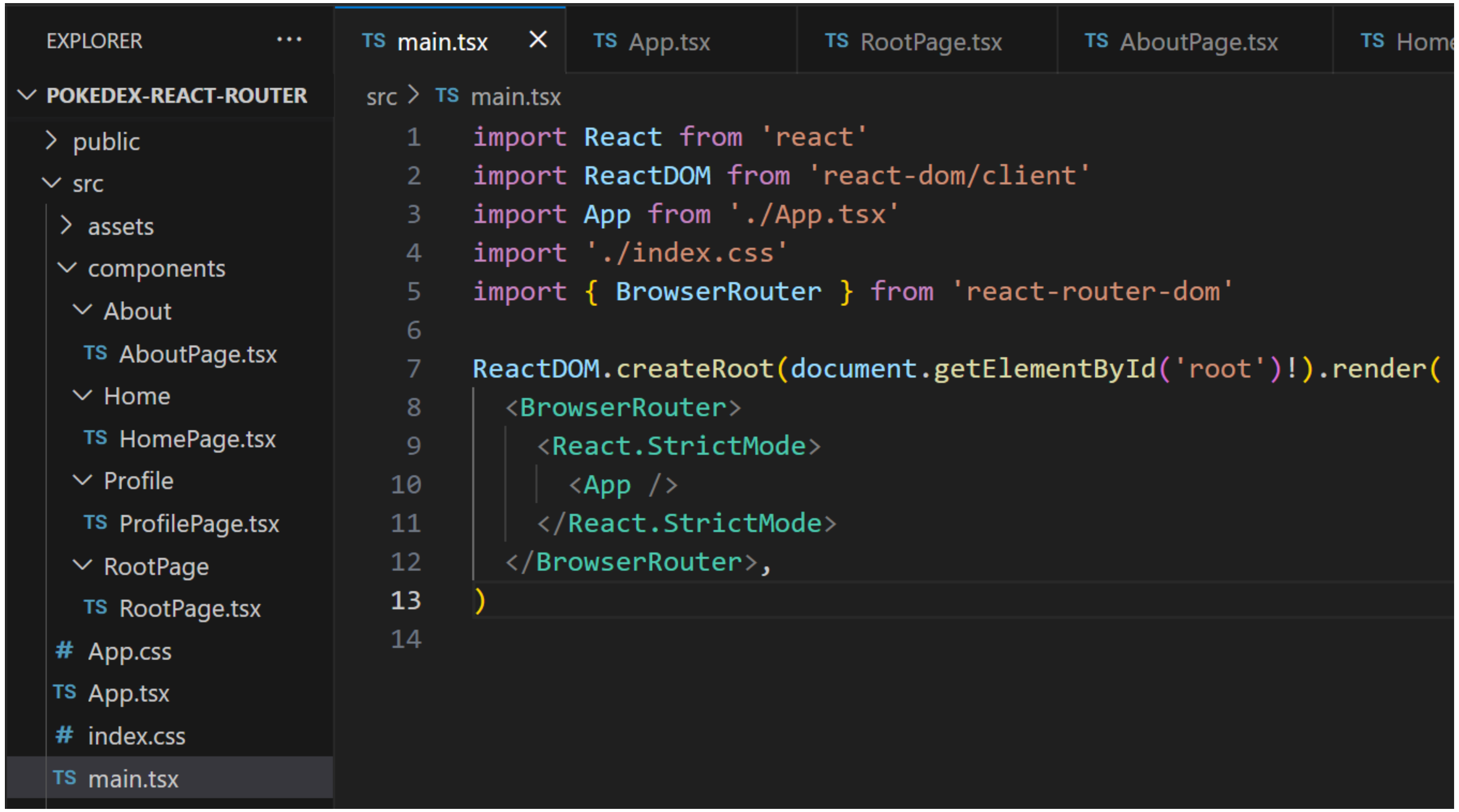
Profile

This is the Home Page Component

This is the About Page Component

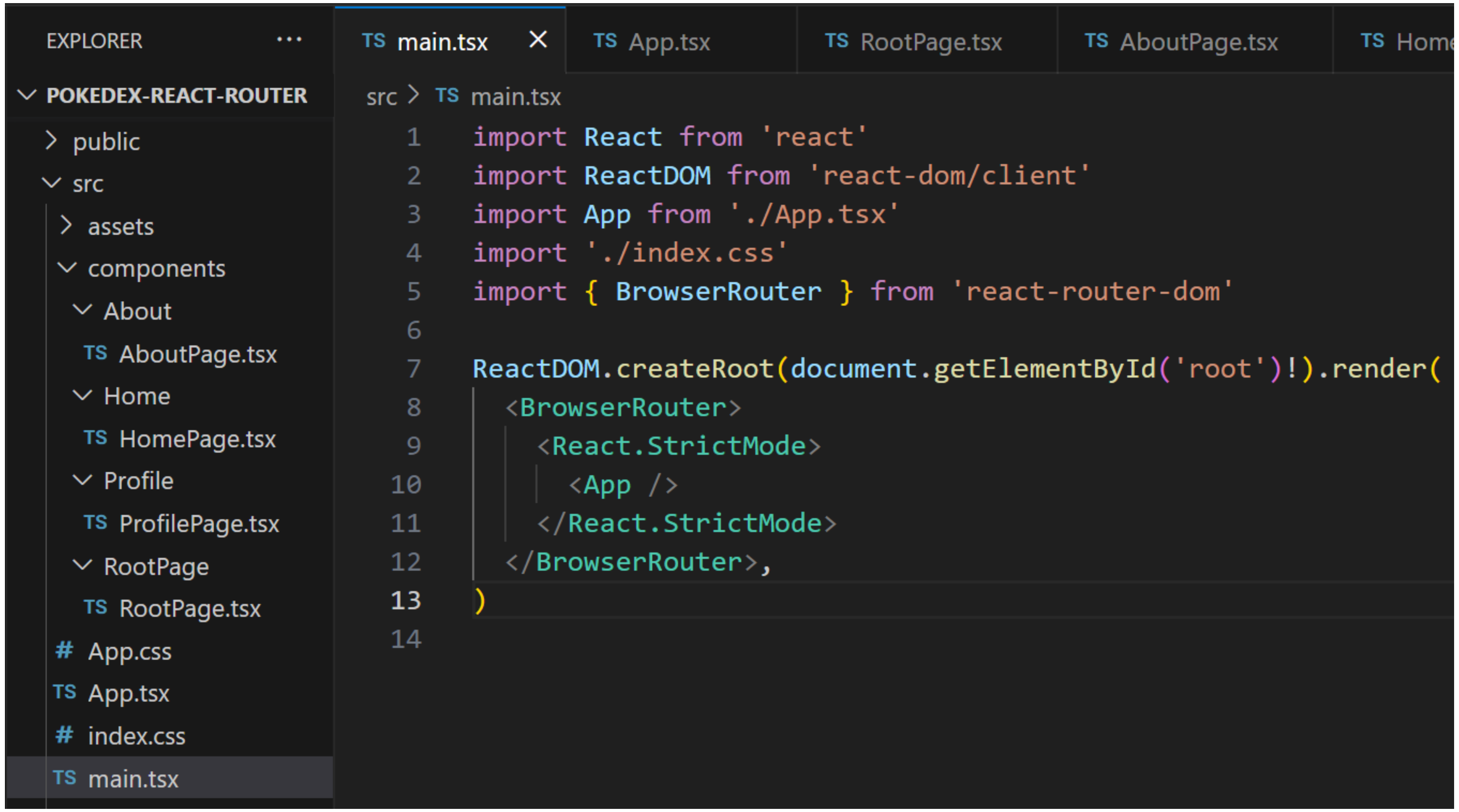
This is the Profile Page Component

Import { BrowserRouter } in main.tsx



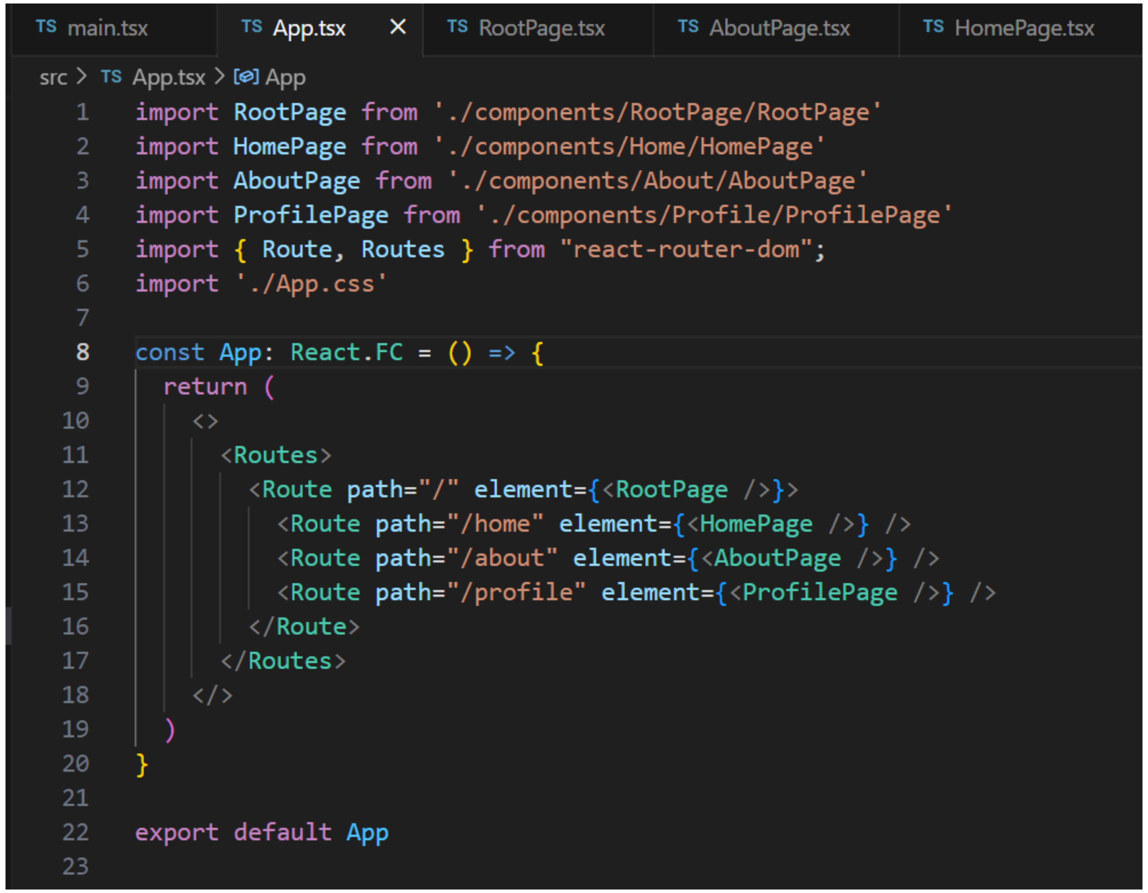
```tsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.tsx'
import './index.css'
import { BrowserRouter } from 'react-router-dom'

ReactDOM.createRoot(document.getElementById('root')!).render(
  <BrowserRouter>
    <React.StrictMode>
      <App />
    </React.StrictMode>
  </BrowserRouter>,
)
```

Import { BrowserRouter } in main.tsx



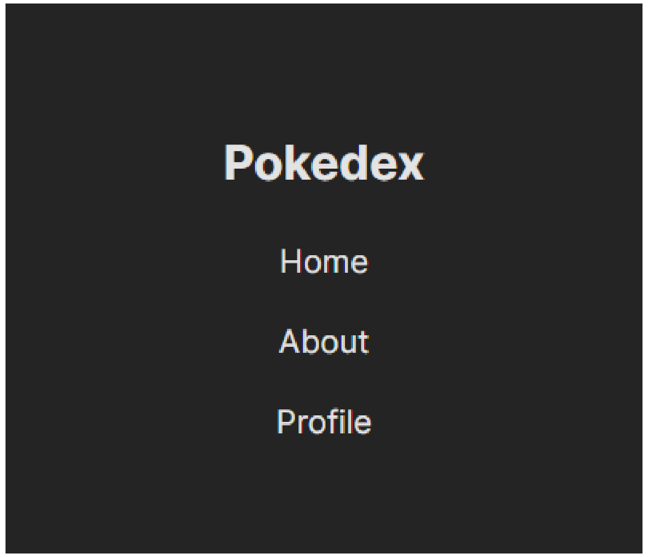
```tsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.tsx'
import './index.css'
import { BrowserRouter } from 'react-router-dom'

ReactDOM.createRoot(document.getElementById('root')!).render(
  <BrowserRouter>
    <React.StrictMode>
      <App />
    </React.StrictMode>
  </BrowserRouter>,
)
```

In you App.tsx import { Routes, Route } from react-router-dom


main.tsx | App.tsx | RootPage.tsx | AboutPage.tsx | HomePage.tsx

src > App.tsx > App

```tsx
import RootPage from './components/RootPage/RootPage'
import HomePage from './components/Home/HomePage'
import AboutPage from './components/About/AboutPage'
import ProfilePage from './components/Profile/ProfilePage'
import { Route, Routes } from "react-router-dom";
import './App.css'

const App: React.FC = () => {
  return (
    <>
      <Routes>
        <Route path="/" element={<RootPage />}>
          <Route path="/home" element={<HomePage />} />
          <Route path="/about" element={<AboutPage />} />
          <Route path="/profile" element={<ProfilePage />} />
        </Route>
      </Routes>
    </>
  )
}

export default App

```


Pokedex

Home

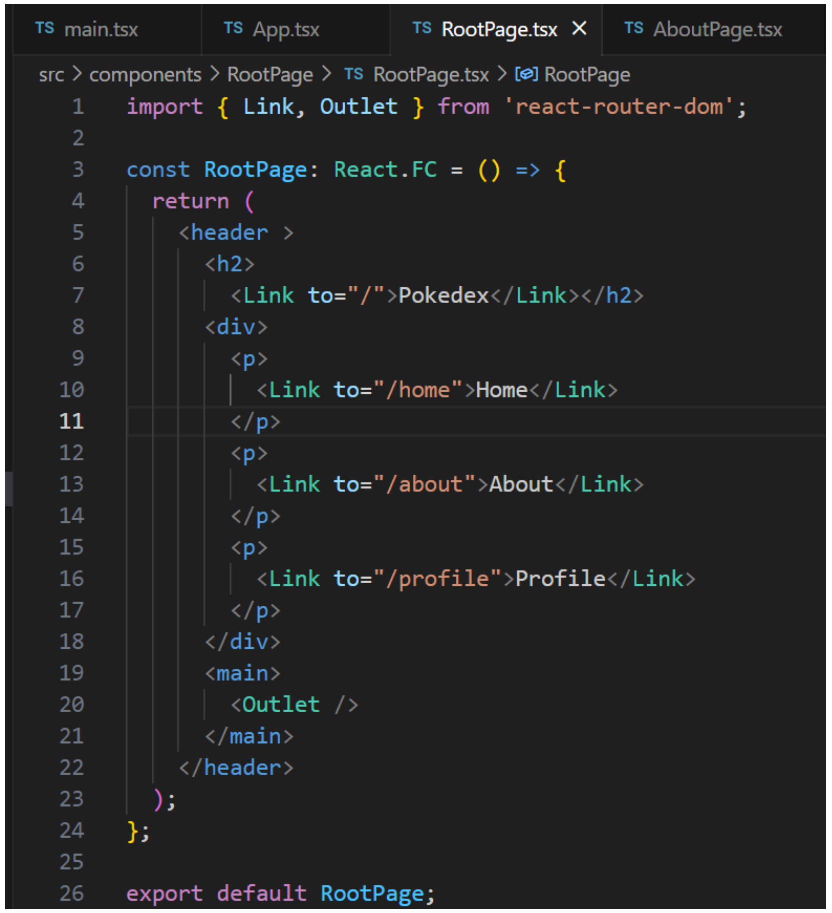
About

Profile

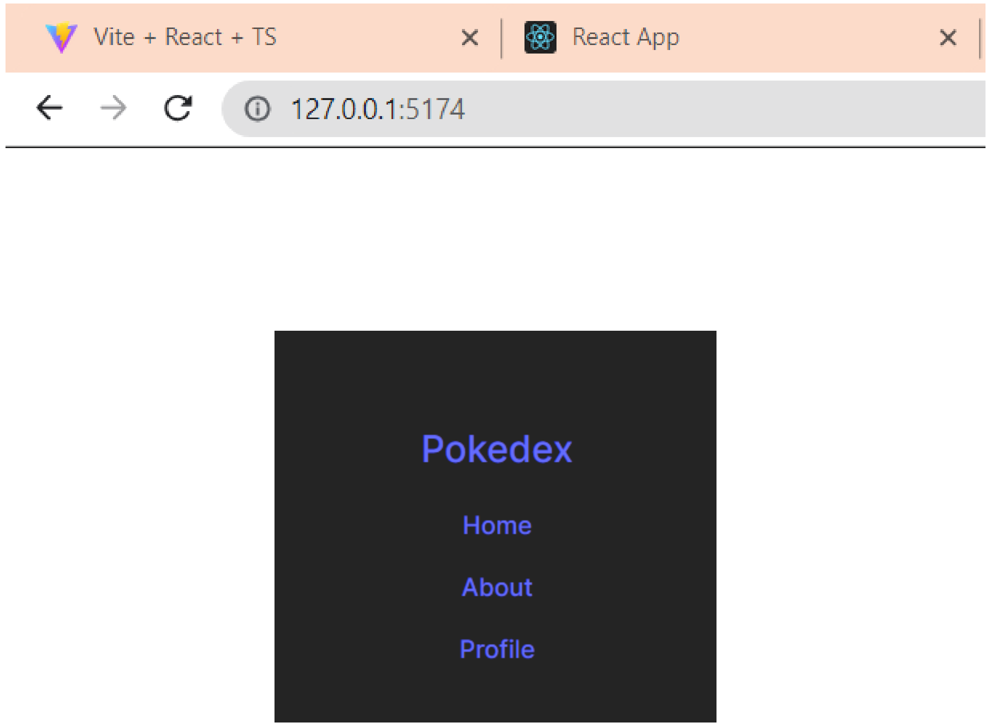
TRAINING THE FUTURE TODAY
www.trainosys.com

HYBRID TECHNOLOGY COURSES

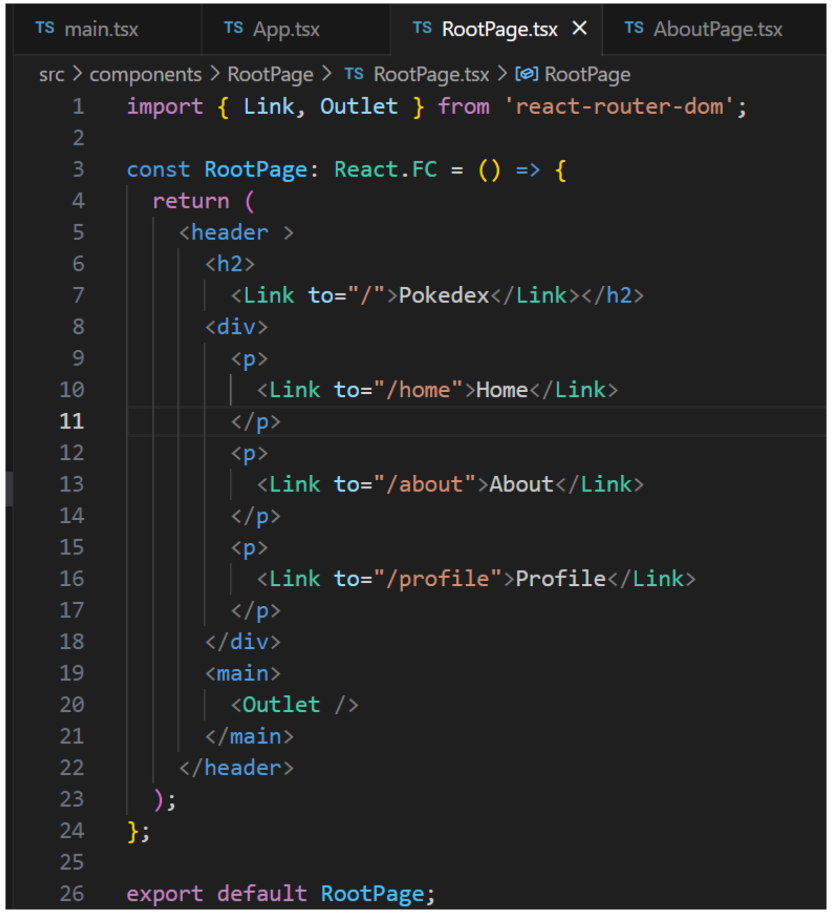
In your <RootPage /> component, import { Link, Outlet } from react-router-dom



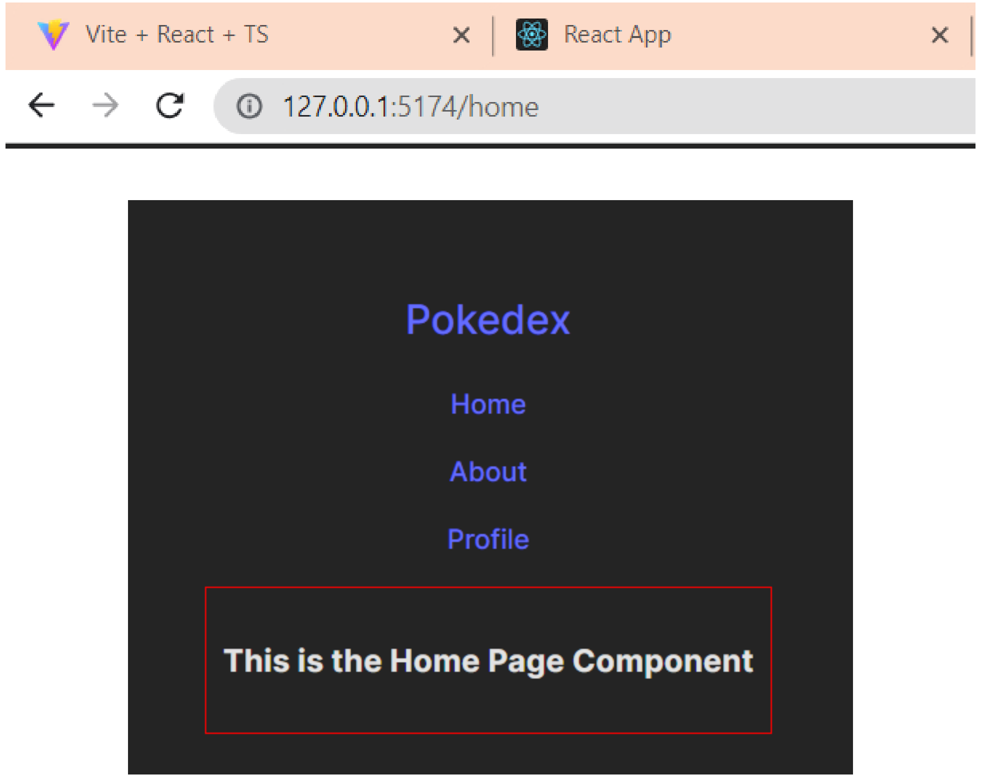
```tsx
import { Link, Outlet } from 'react-router-dom';

const RootPage: React.FC = () => {
  return (
    <header >
      <h2>
        <Link to="/">Pokedex</Link></h2>
      <div>
        <p>
          <Link to="/home">Home</Link>
        </p>
        <p>
          <Link to="/about">About</Link>
        </p>
        <p>
          <Link to="/profile">Profile</Link>
        </p>
      </div>
      <main>
        <Outlet />
      </main>
    </header>
  );
};

export default RootPage;
```



In your <RootPage /> component, import { Link, Outlet } from react-router-dom

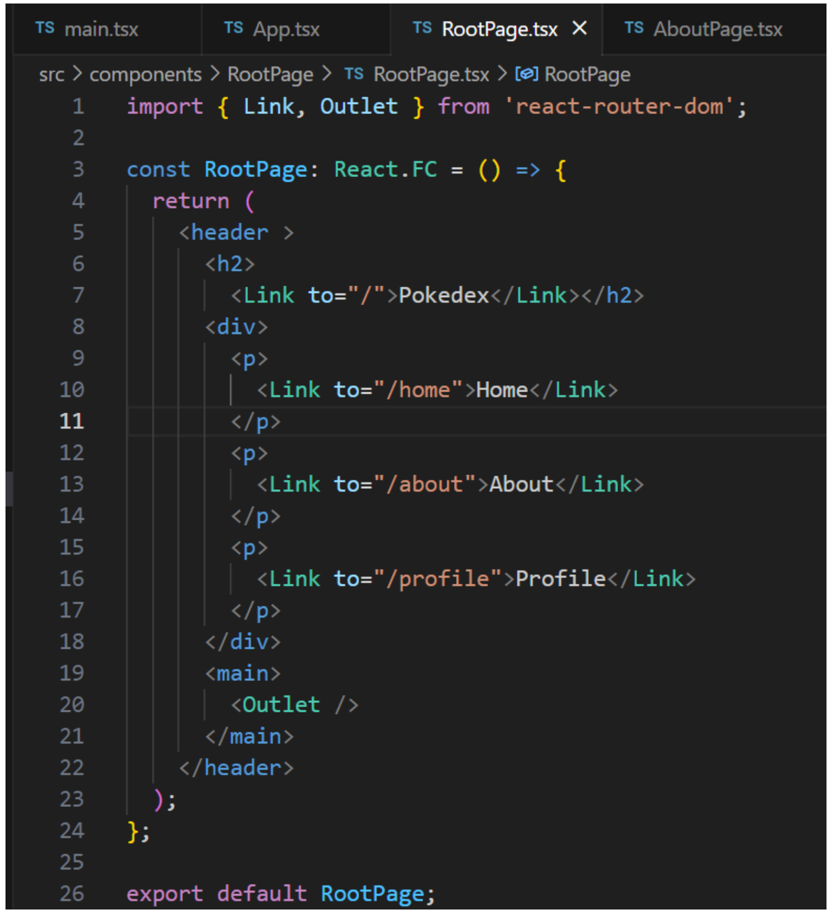


```tsx
import { Link, Outlet } from 'react-router-dom';

const RootPage: React.FC = () => {
  return (
    <header >
      <h2>
        <Link to="/">Pokedex</Link></h2>
      <div>
        <p>
          <Link to="/home">Home</Link>
        </p>
        <p>
          <Link to="/about">About</Link>
        </p>
        <p>
          <Link to="/profile">Profile</Link>
        </p>
      </div>
      <main>
        <Outlet />
      </main>
    </header>
  );
};

export default RootPage;
```



This is the Home Page Component

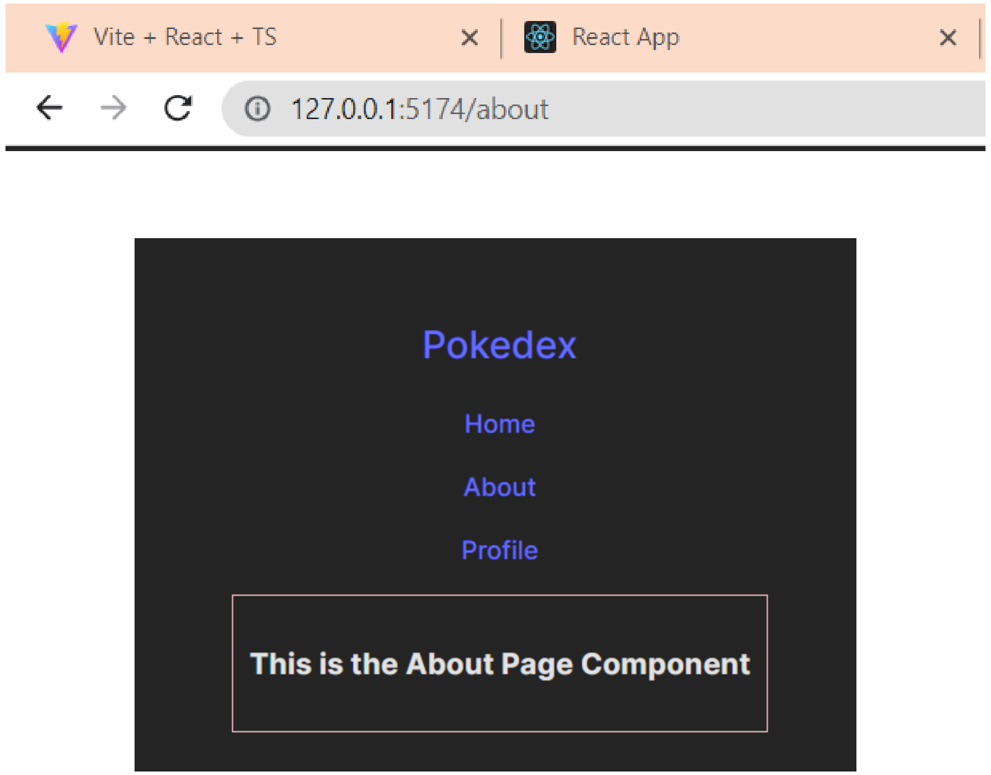
In your <RootPage /> component, import { Link, Outlet } from react-router-dom



main.tsx · App.tsx · RootPage.tsx · AboutPage.tsx

src > components > RootPage > RootPage.tsx > RootPage

```tsx
import { Link, Outlet } from 'react-router-dom';

const RootPage: React.FC = () => {
  return (
    <header >
      <h2>
        <Link to="/">Pokedex</Link></h2>
      <div>
        <p>
          <Link to="/home">Home</Link>
        </p>
        <p>
          <Link to="/about">About</Link>
        </p>
        <p>
          <Link to="/profile">Profile</Link>
        </p>
      </div>
      <main>
        <Outlet />
      </main>
    </header>
  );
};

export default RootPage;
```



Vite + React + TS · React App

127.0.0.1:5174/about

Pokedex

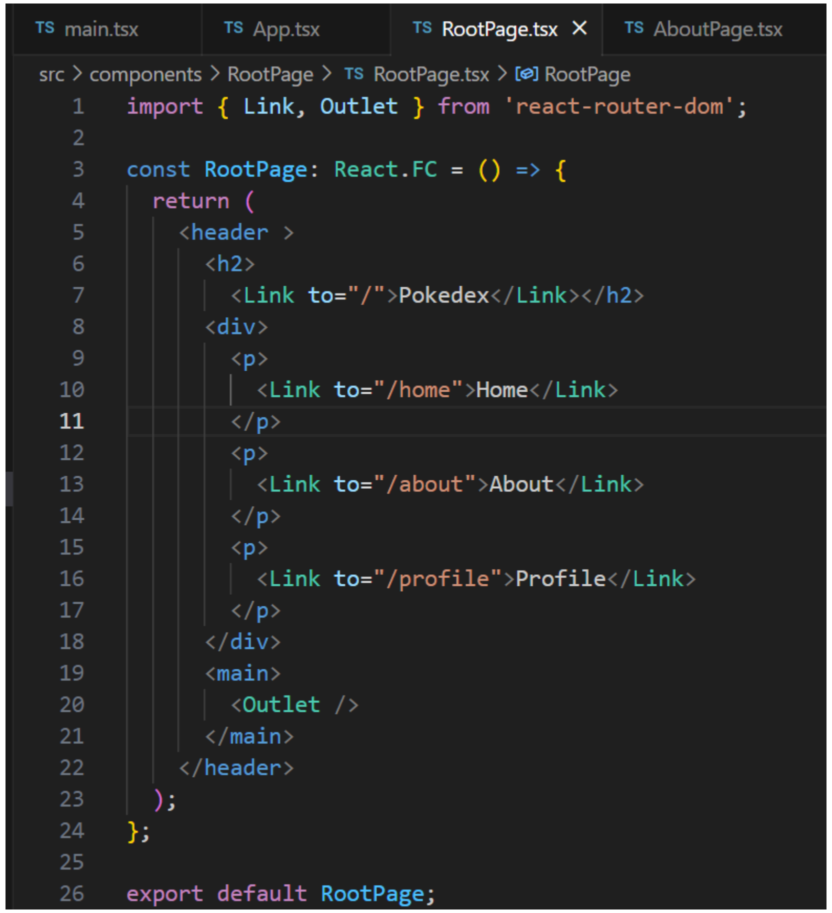
Home

About

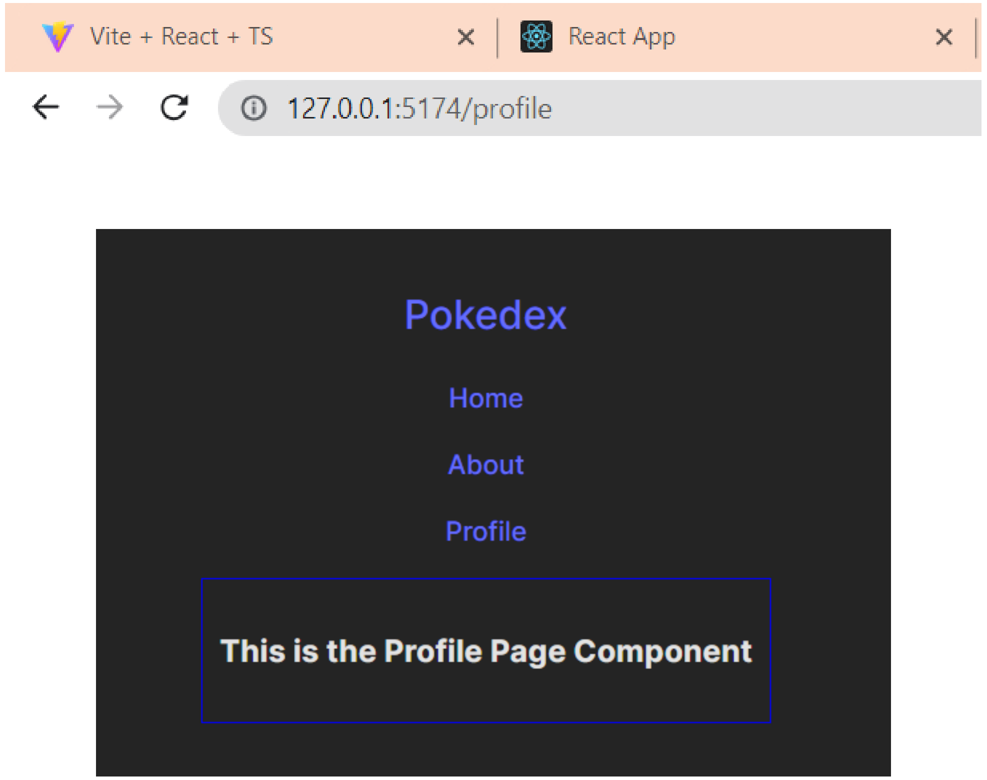
Profile

This is the About Page Component

In your <RootPage /> component, import { Link, Outlet } from react-router-dom



```
import { Link, Outlet } from 'react-router-dom';

const RootPage: React.FC = () => {
  return (
    <header >
      <h2>
        <Link to="/">Pokedex</Link></h2>
      <div>
        <p>
          <Link to="/home">Home</Link>
        </p>
        <p>
          <Link to="/about">About</Link>
        </p>
        <p>
          <Link to="/profile">Profile</Link>
        </p>
      </div>
      <main>
        <Outlet />
      </main>
    </header>
  );
};

export default RootPage;
```



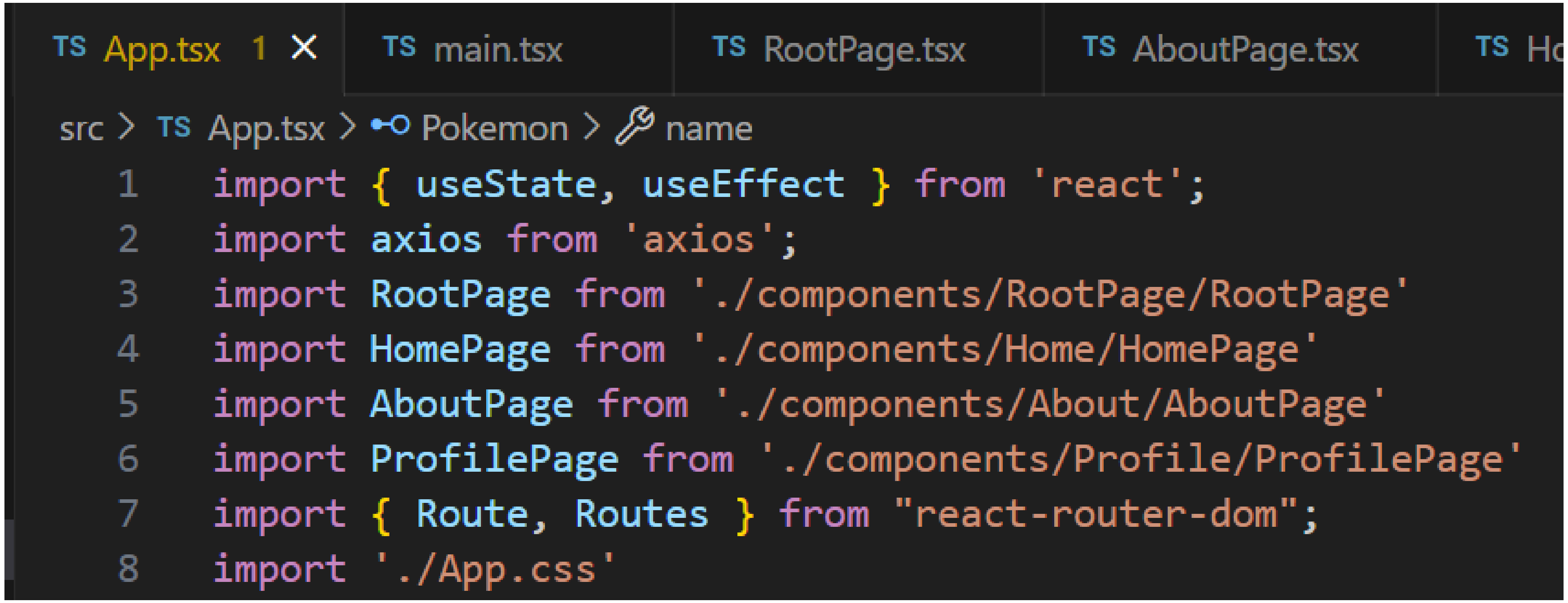
This is the Profile Page Component

TRAINOSYS

FETCHING DATA VIA EXTERNAL API

HYBRID
TECHNOLOGY
COURSES

import { useState, useEffect } from ‘react;
import { axios } from ‘axios’

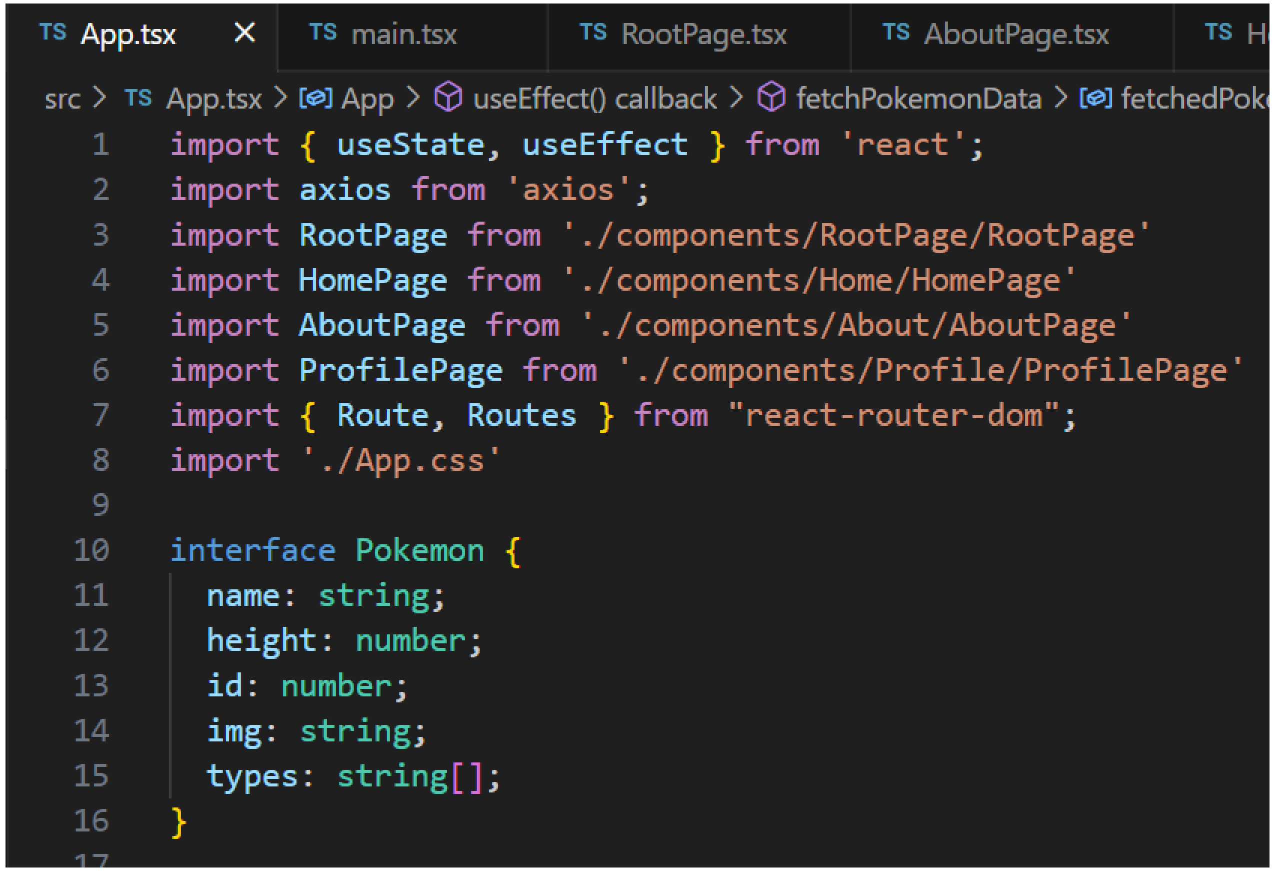


TS App.tsx 1 × | TS main.tsx | TS RootPage.tsx | TS AboutPage.tsx | TS Ho

src > TS App.tsx > Pokemon > name

```tsx
import { useState, useEffect } from 'react';
import axios from 'axios';
import RootPage from './components/RootPage/RootPage'
import HomePage from './components/Home/HomePage'
import AboutPage from './components/About/AboutPage'
import ProfilePage from './components/Profile/ProfilePage'
import { Route, Routes } from "react-router-dom";
import './App.css'
```

Create Interface Pokemon

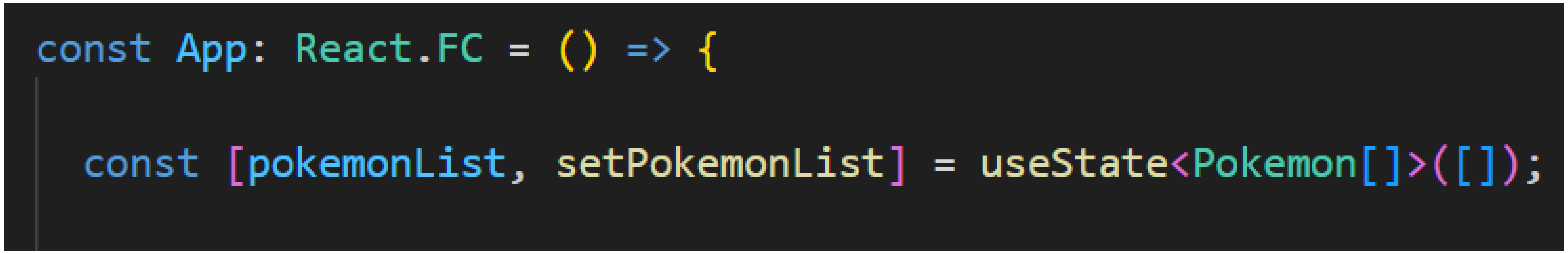


TS App.tsx × TS main.tsx TS RootPage.tsx TS AboutPage.tsx TS H

src > TS App.tsx > App > useEffect() callback > fetchPokemonData > fetchedPok

```
import { useState, useEffect } from 'react';
import axios from 'axios';
import RootPage from './components/RootPage/RootPage'
import HomePage from './components/Home/HomePage'
import AboutPage from './components/About/AboutPage'
import ProfilePage from './components/Profile/ProfilePage'
import { Route, Routes } from "react-router-dom";
import './App.css'

interface Pokemon {
  name: string;
  height: number;
  id: number;
  img: string;
  types: string[];
}
```

useState Hook



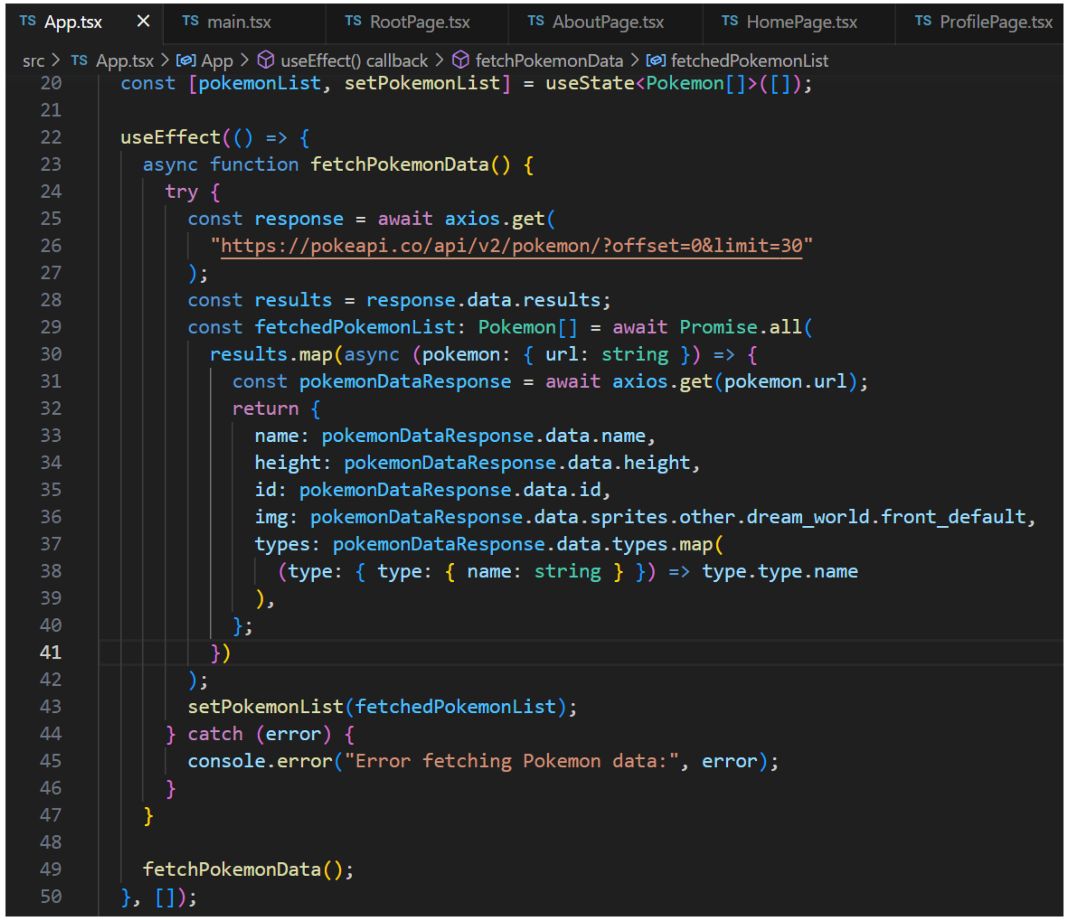
```
const App: React.FC = () => {

  const [pokemonList, setPokemonList] = useState<Pokemon[]>([]);
```

TRAINOSYS

Our Pokemon API

https://pokeapi.co/api/v2/pokemon/?offset=0&limit=30

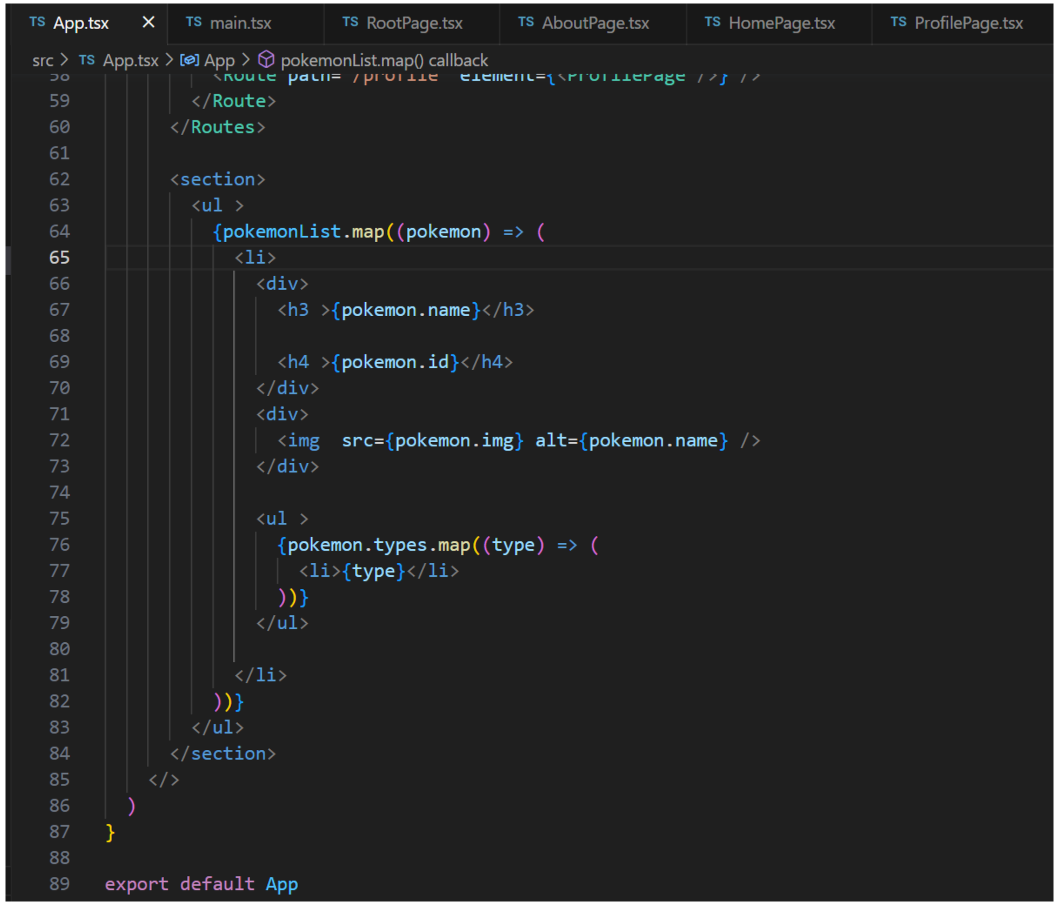
Fetching data via external API using useEffect() hook



App.tsx
main.tsx
RootPage.tsx
AboutPage.tsx
HomePage.tsx
ProfilePage.tsx
src > App.tsx > App > useEffect() callback > fetchPokemonData > fetchedPokemonList

```tsx
  const [pokemonList, setPokemonList] = useState<Pokemon[]>([]);

  useEffect(() => {
    async function fetchPokemonData() {
      try {
        const response = await axios.get(
          "https://pokeapi.co/api/v2/pokemon/?offset=0&limit=30"
        );
        const results = response.data.results;
        const fetchedPokemonList: Pokemon[] = await Promise.all(
          results.map(async (pokemon: { url: string }) => {
            const pokemonDataResponse = await axios.get(pokemon.url);
            return {
              name: pokemonDataResponse.data.name,
              height: pokemonDataResponse.data.height,
              id: pokemonDataResponse.data.id,
              img: pokemonDataResponse.data.sprites.other.dream_world.front_default,
              types: pokemonDataResponse.data.types.map(
                (type: { type: { name: string } }) => type.type.name
              ),
            };
          })
        );
        setPokemonList(fetchedPokemonList);
      } catch (error) {
        console.error("Error fetching Pokemon data:", error);
      }
    }

    fetchPokemonData();
  }, []);
```

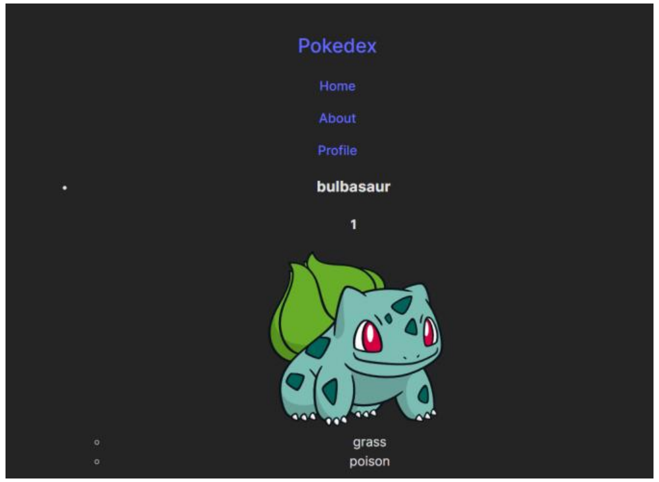
Create your own UI to render the pokemons



TS App.tsx × TS main.tsx TS RootPage.tsx TS AboutPage.tsx TS HomePage.tsx TS ProfilePage.tsx

src > TS App.tsx > App > pokemonList.map() callback

```
          </Route>
        </Routes>

        <section>
          <ul >
            {pokemonList.map((pokemon) => (
              <li>
                <div>
                  <h3 >{pokemon.name}</h3>

                  <h4 >{pokemon.id}</h4>
                </div>
                <div>
                  <img  src={pokemon.img} alt={pokemon.name} />
                </div>

                <ul >
                  {pokemon.types.map((type) => (
                    <li>{type}</li>
                  ))}
                </ul>

              </li>
            ))}
          </ul>
        </section>
      </>
    )
}

export default App
```



Pokedex

Home

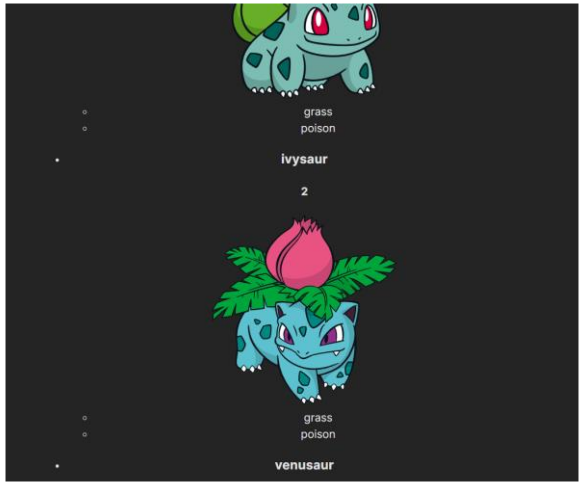
About

Profile

- bulbasaur

  1

  - grass
  - poison



- grass
- poison

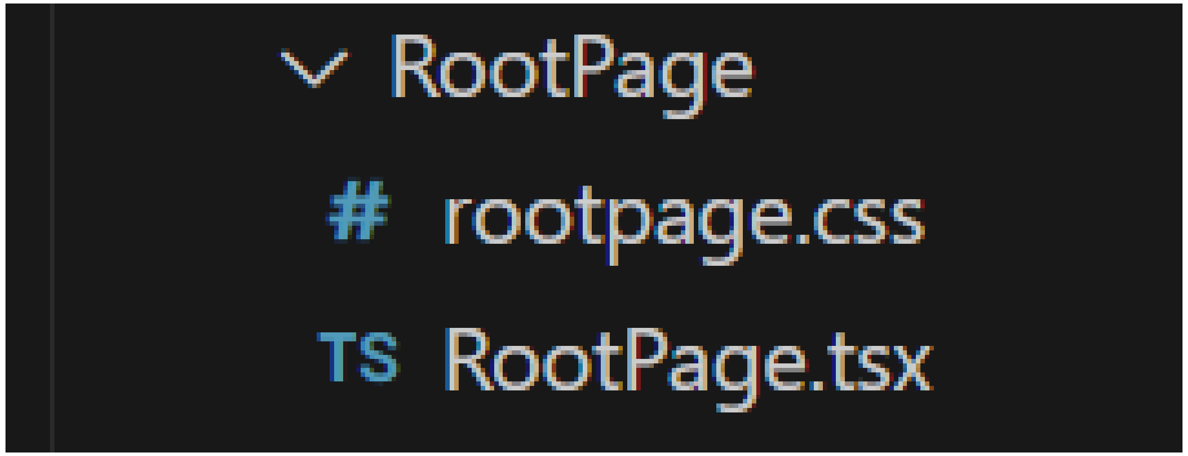
- ivysaur

  2

  - grass
  - poison

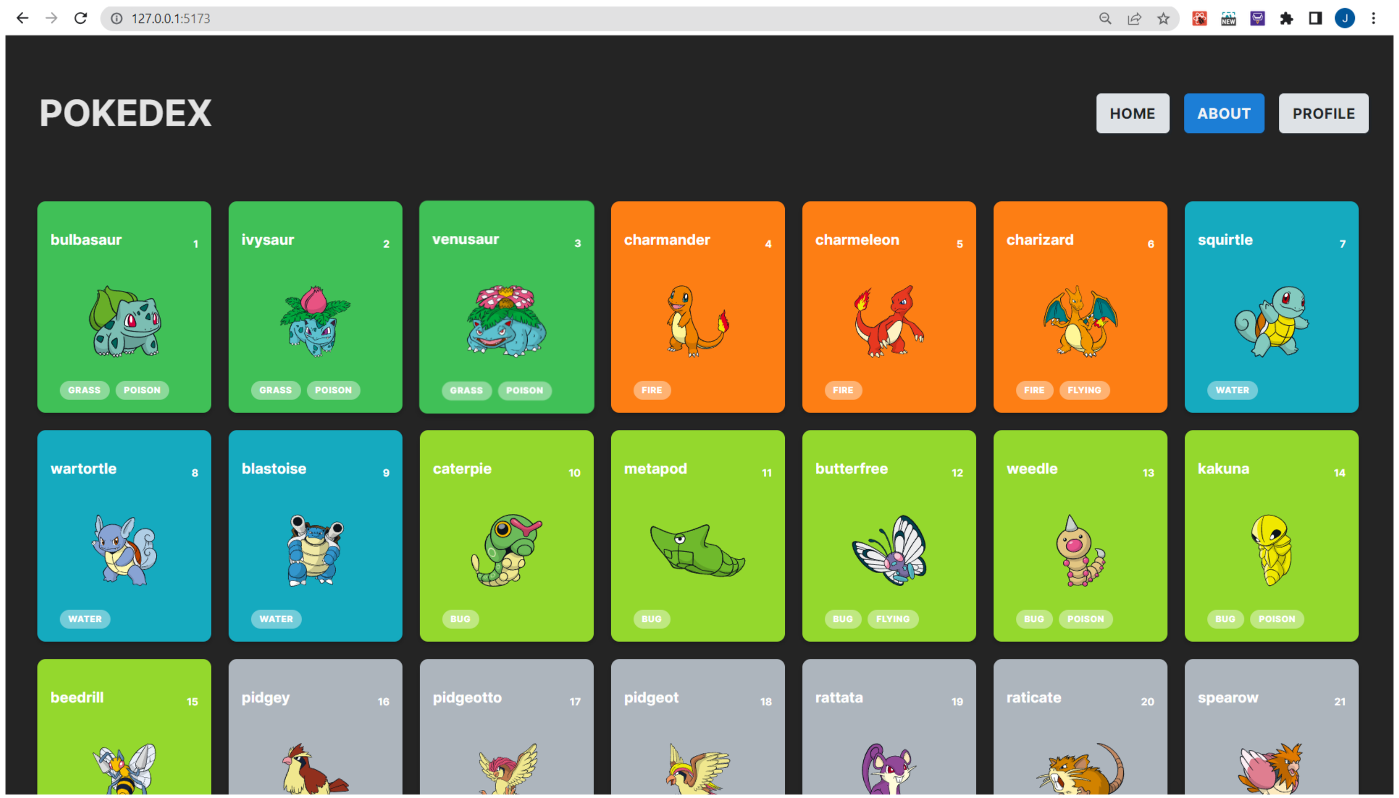
- venusaur

CSS



```
⌄ RootPage
  # rootpage.css
  TS RootPage.tsx
```

Sample UI


127.0.0.1:5173
POKEDEX
HOME
ABOUT
PROFILE
bulbasaur 1 GRASS POISON
ivysaur 2 GRASS POISON
venusaur 3 GRASS POISON
charmander 4 FIRE
charmeleon 5 FIRE
charizard 6 FIRE FLYING
squirtle 7 WATER
wartortle 8 WATER
blastoise 9 WATER
caterpie 10 BUG
metapod 11 BUG
butterfree 12 BUG FLYING
weedle 13 BUG POISON
kakuna 14 BUG POISON
beedrill 15
pidgey 16
pidgeotto 17
pidgeot 18
rattata 19
raticate 20
spearow 21

TRAINOSYS

ACTIVITY 1

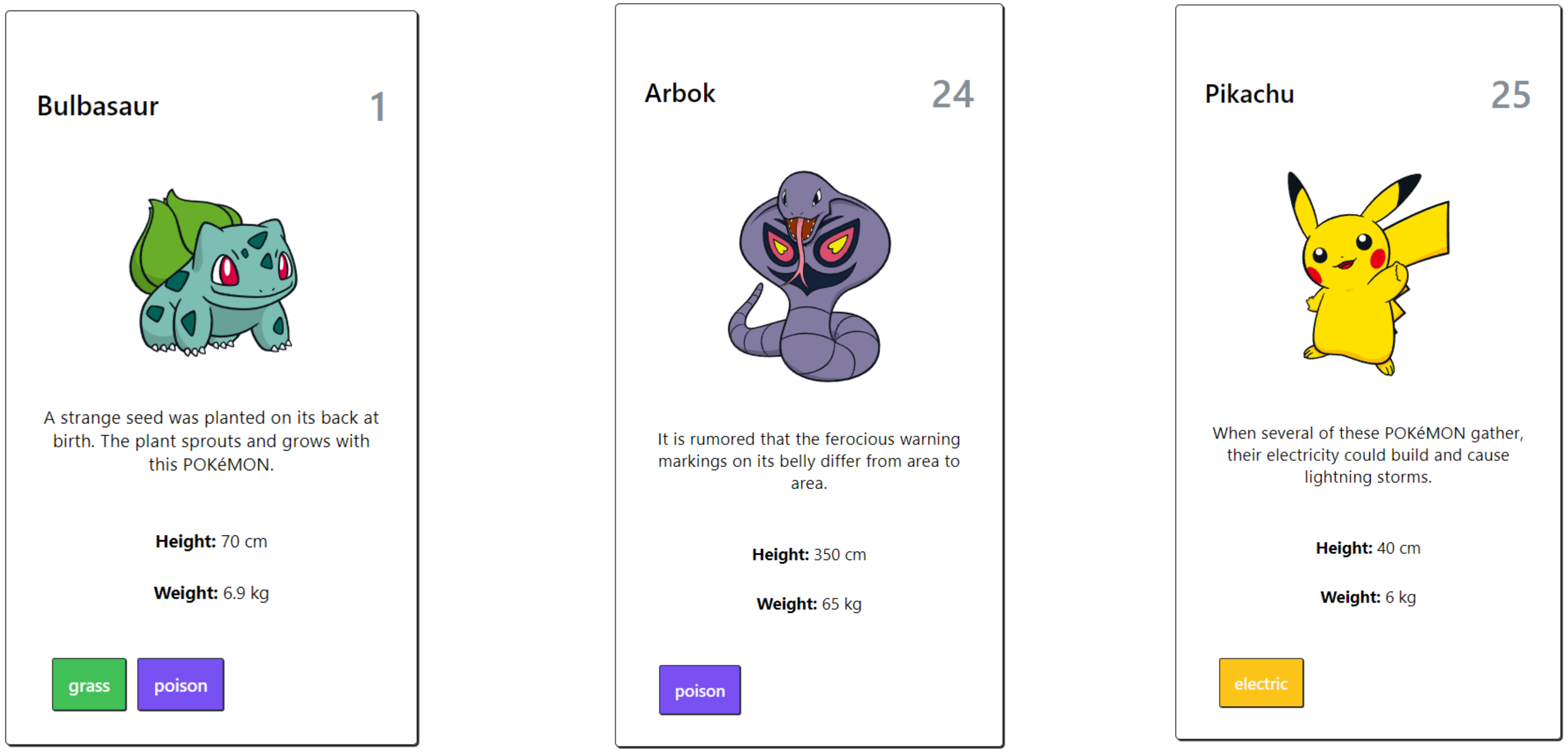
HYBRID
TECHNOLOGY
COURSES

Activity 1

- ❑ Create a new project
- ❑ Project name: rts-d3-act-one

- ❑ Continue the React Router code along and make dynamic routes for each pokemon in the pokedex
- ❑ Create another <Pokemon /> component to render each pokemon

Expected Output



Bulbasaur 1

A strange seed was planted on its back at birth. The plant sprouts and grows with this POKéMON.

Height: 70 cm

Weight: 6.9 kg

grass poison

Arbok 24

It is rumored that the ferocious warning markings on its belly differ from area to area.

Height: 350 cm

Weight: 65 kg

poison

Pikachu 25

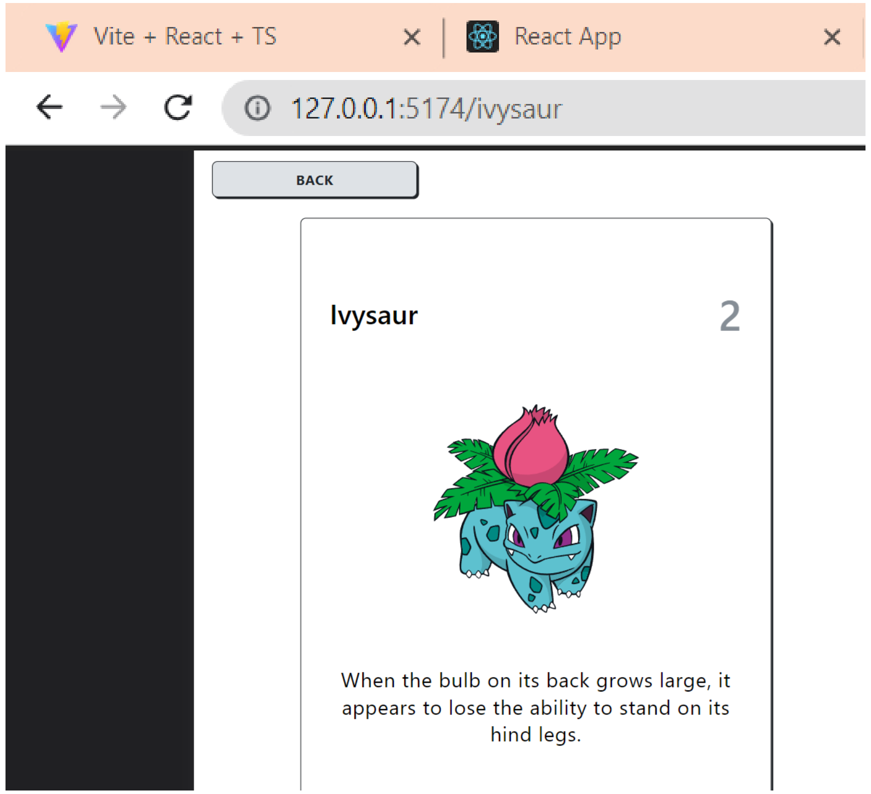
When several of these POKéMON gather, their electricity could build and cause lightning storms.

Height: 40 cm

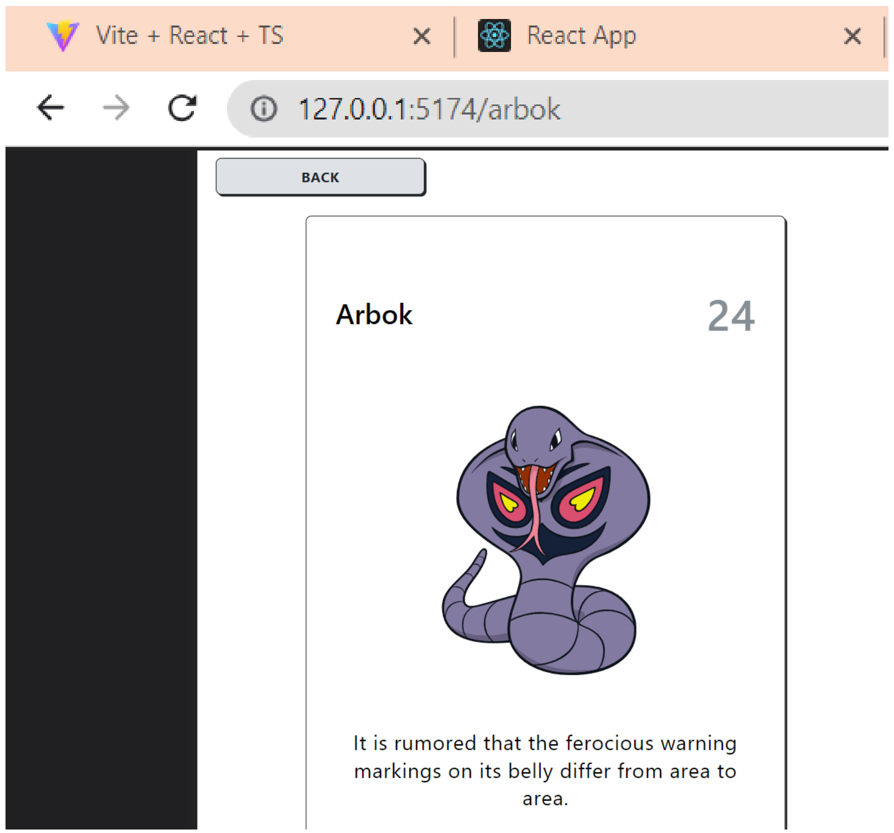
Weight: 6 kg

electric

Expected Output



When the bulb on its back grows large, it appears to lose the ability to stand on its hind legs.

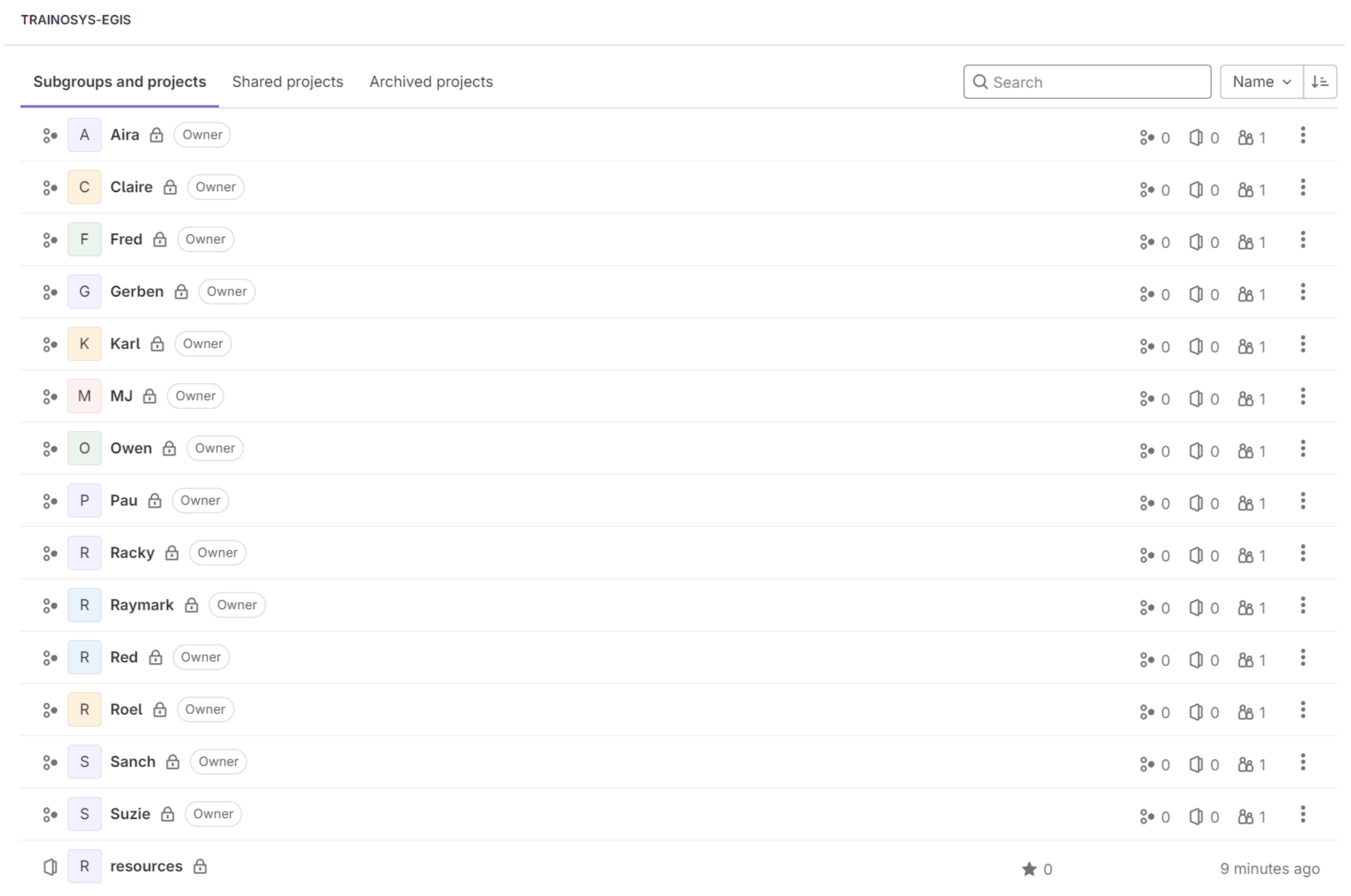


It is rumored that the ferocious warning markings on its belly differ from area to area.

TRAINOSYS

GITLAB

HYBRID
TECHNOLOGY
COURSES


TRAINOSYS-EGIS

Aira	Owner	0	0	1
Claire	Owner	0	0	1
Fred	Owner	0	0	1
Gerben	Owner	0	0	1
Karl	Owner	0	0	1
MJ	Owner	0	0	1
Owen	Owner	0	0	1
Pau	Owner	0	0	1
Racky	Owner	0	0	1
Raymark	Owner	0	0	1
Red	Owner	0	0	1
Roel	Owner	0	0	1
Sanch	Owner	0	0	1
Suzie	Owner	0	0	1

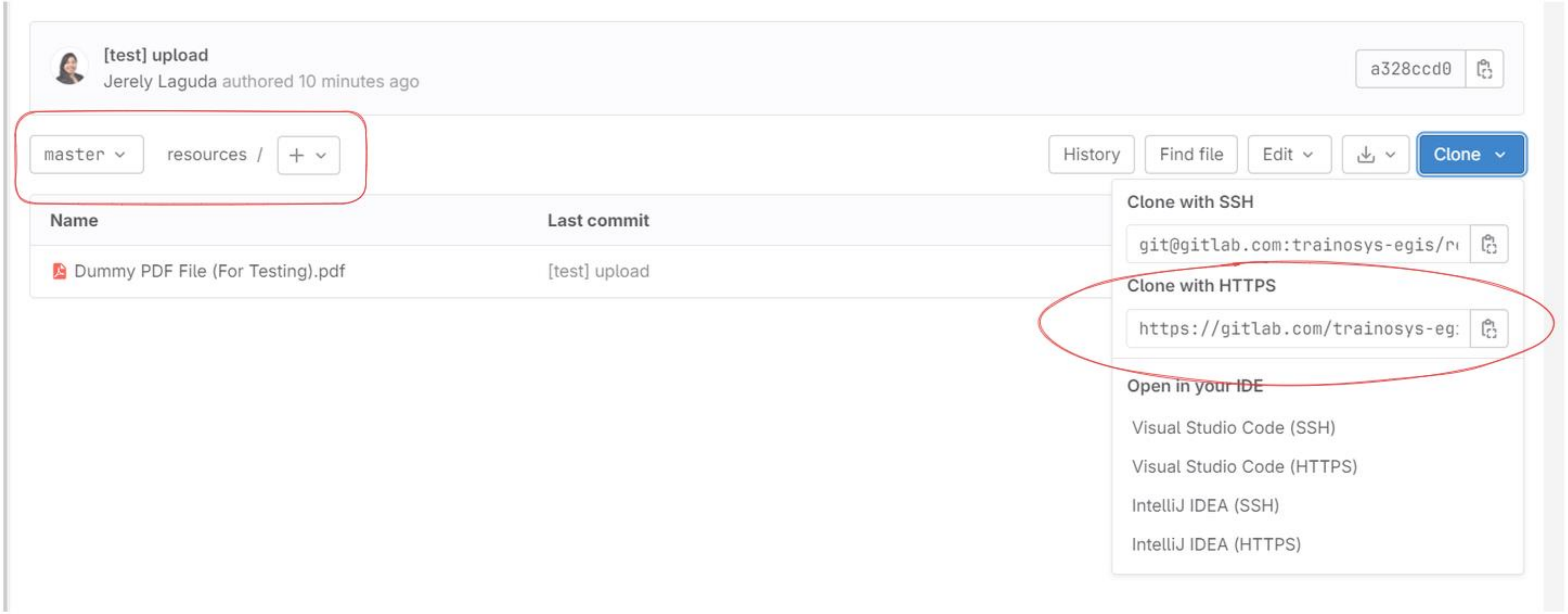
resources ★ 0 9 minutes ago

Upload Activities (new folder)

- ❑ > git clone <repository url>
- ❑ Put all your activities inside
- ❑ In the root folder, open your terminal
- ❑ > git remote -v
- ❑ > git status
- ❑ > git add .
- ❑ > git commit –m "<commit message here>"
- ❑ > git push –set-upstream origin master
- ❑ > git push

Upload Activities (existing folder)

- ❑ Create new folder and put all your activities inside
- ❑ In the root folder, open your terminal
- ❑ > git init
- ❑ > git remote set-url origin <http-url>
- ❑ > git remote -v
- ❑ > git status
- ❑ > git add .
- ❑ > git commit –m “<commit message here>”
- ❑ > git push –set-upstream origin master


[test] upload
Jerely Laguda authored 10 minutes ago
a328ccd0
master
resources /
+
History
Find file
Edit
Clone
Name
Last commit
Dummy PDF File (For Testing).pdf
[test] upload
Clone with SSH
git@gitlab.com:trainosys-egis/r
Clone with HTTPS
https://gitlab.com/trainosys-eg
Open in your IDE
Visual Studio Code (SSH)
Visual Studio Code (HTTPS)
IntelliJ IDEA (SSH)
IntelliJ IDEA (HTTPS)

Download Resources

- ❑ > git clone https://github.com/carpejemm/trainosys-jem.git
- ❑ To download latest files
- ❑ > git pull

TRAINOSYS

CODE ALONG – DAY 2 ACTIVITY ANSWERS

HYBRID TECHNOLOGY COURSES

Generate a new project

```
npm create vite@latest / npm create vite@4.1.0
```

Generate a new project

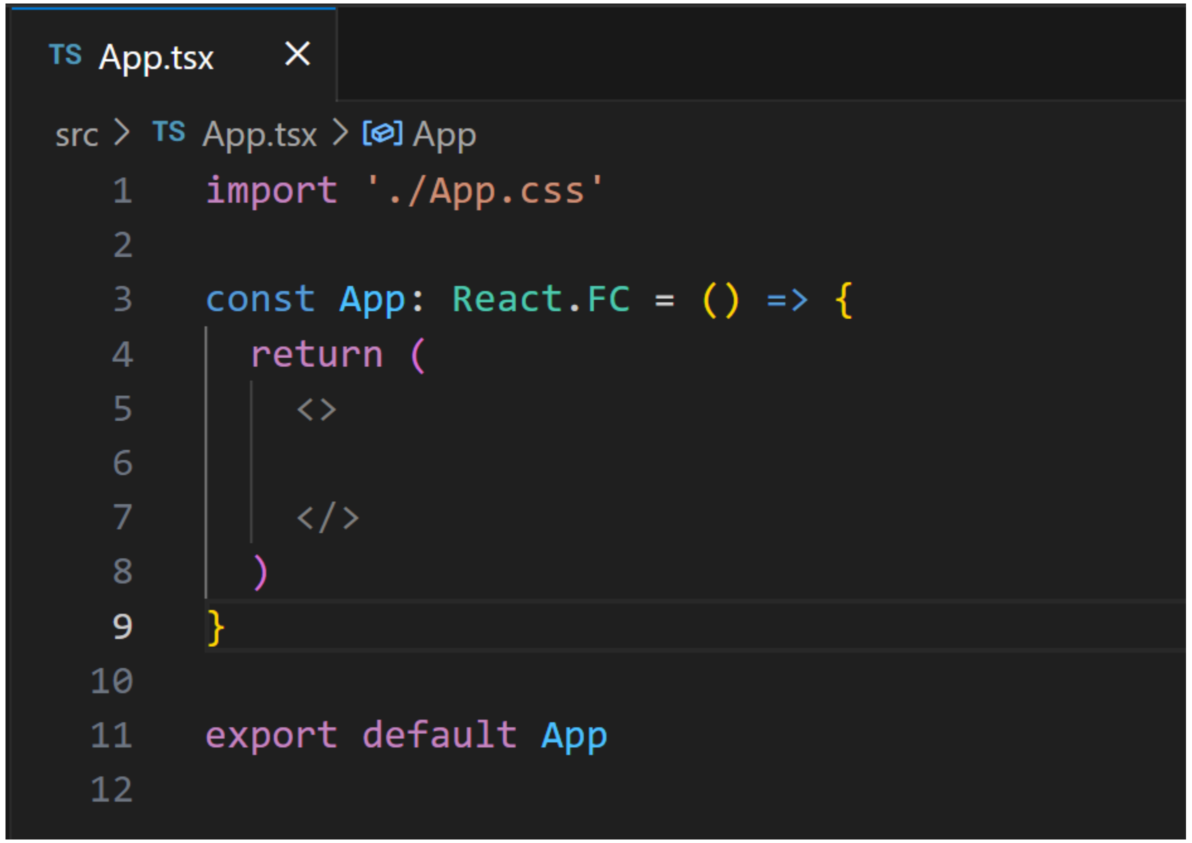
- ❑ Project Name: rts-d2-act-two
- ❑ > React
- ❑ > TypeScript
- ❑ > cd rts-d2-act-two
- ❑ > npm install / npm i
- ❑ > npm run dev

TRAINOSYS

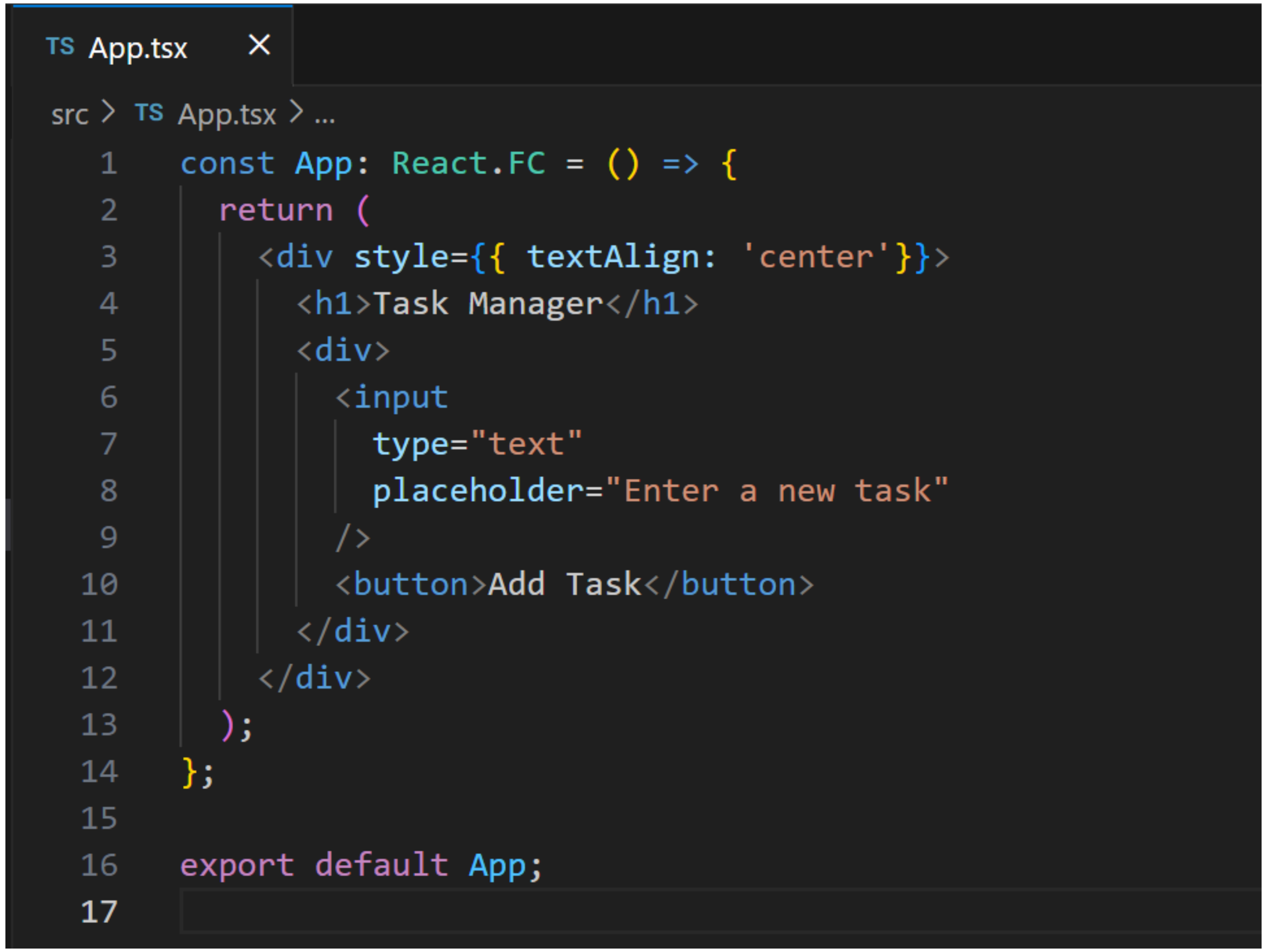
TASK 1:
Create an input form that accepts a string when an "Add Task" button is clicked

HYBRID
TECHNOLOGY
COURSES

Delete jsx in App.tsx and other unnecessary lines of code



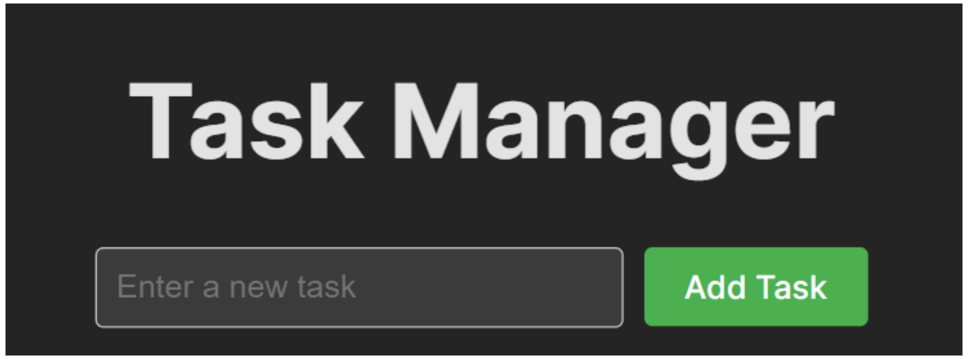
```tsx
import './App.css'

const App: React.FC = () => {
  return (
    <>


    </>
  )
}

export default App

```



TS App.tsx

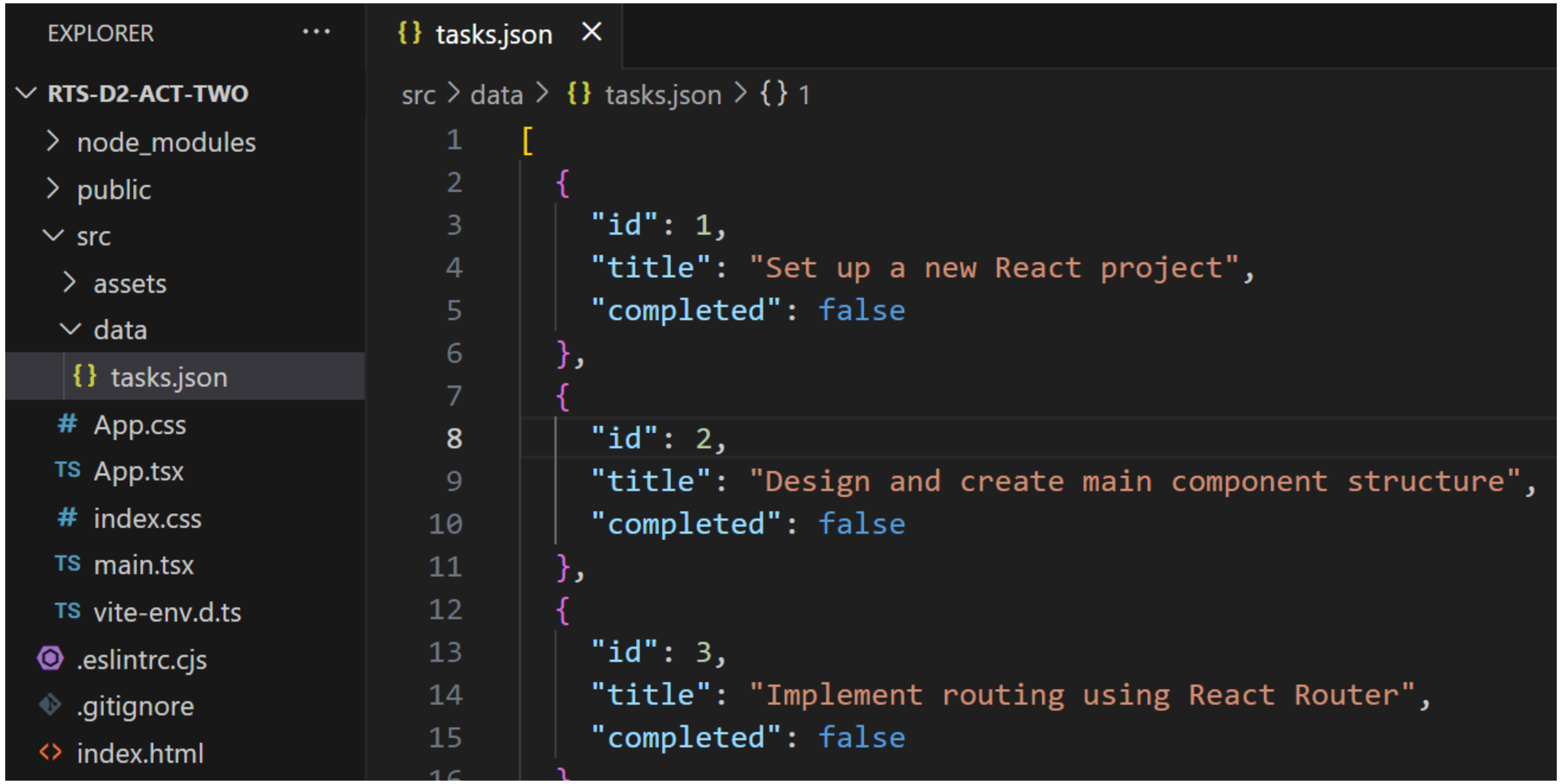
src > TS App.tsx > ...

```tsx
const App: React.FC = () => {
  return (
    <div style={{ textAlign: 'center'}}>
      <h1>Task Manager</h1>
      <div>
        <input
          type="text"
          placeholder="Enter a new task"
        />
        <button>Add Task</button>
      </div>
    </div>
  );
};

export default App;
```



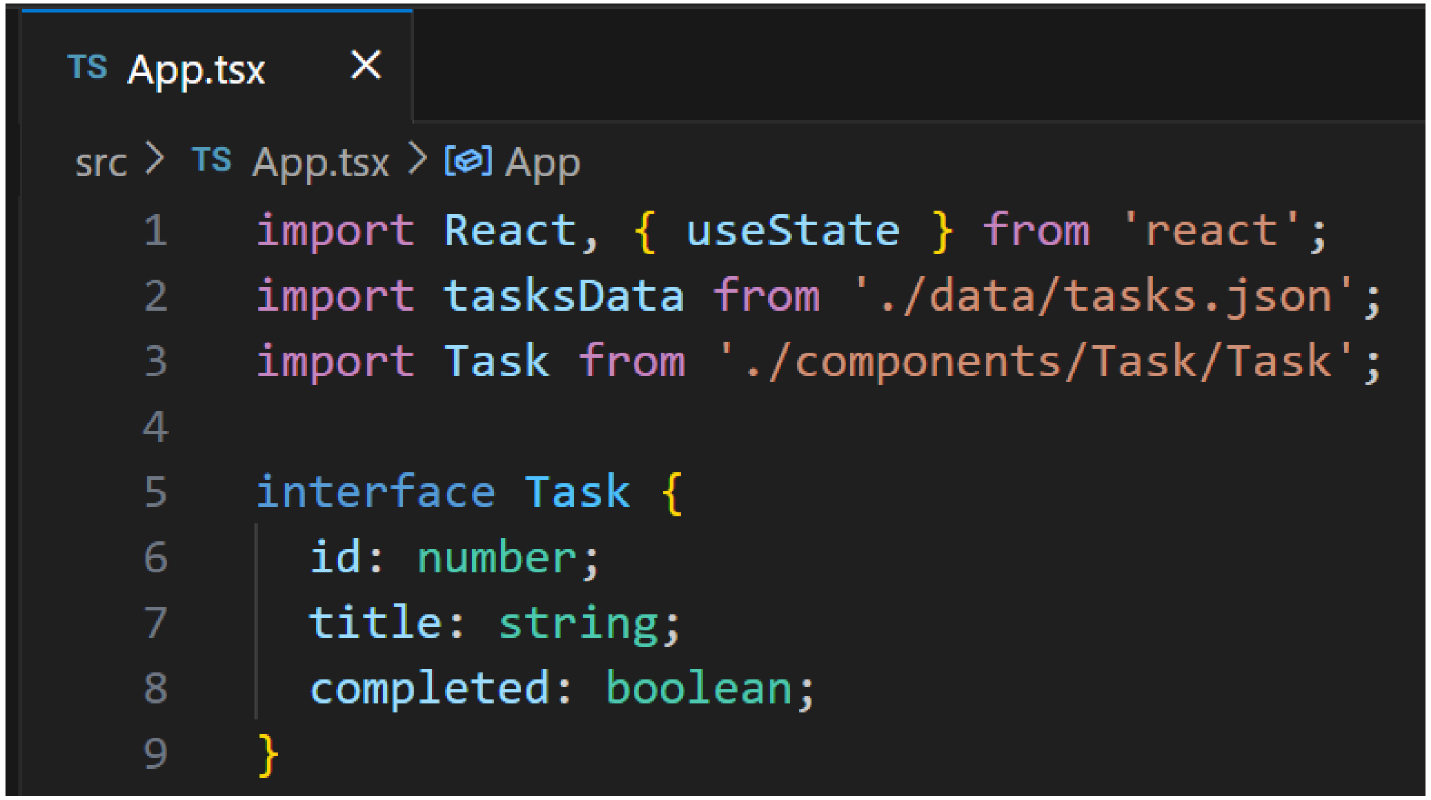
TASK 2:
Create a <Task /> component that accepts task's name as props and display the initial tasks array

Create a JSON for your initial Tasks Data



```json
[
  {
    "id": 1,
    "title": "Set up a new React project",
    "completed": false
  },
  {
    "id": 2,
    "title": "Design and create main component structure",
    "completed": false
  },
  {
    "id": 3,
    "title": "Implement routing using React Router",
    "completed": false
```

Imports – React, Data [Tasks]
Create interface for the Task



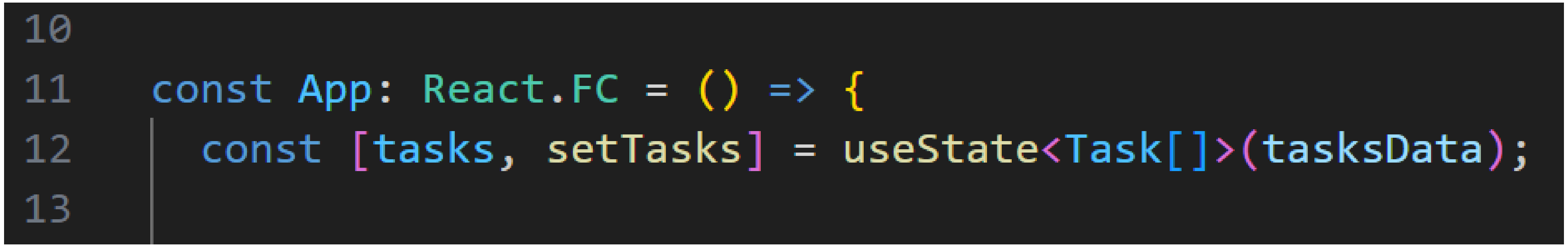
App.tsx

src > App.tsx > App

```
import React, { useState } from 'react';
import tasksData from './data/tasks.json';
import Task from './components/Task/Task';

interface Task {
  id: number;
  title: string;
  completed: boolean;
}
```

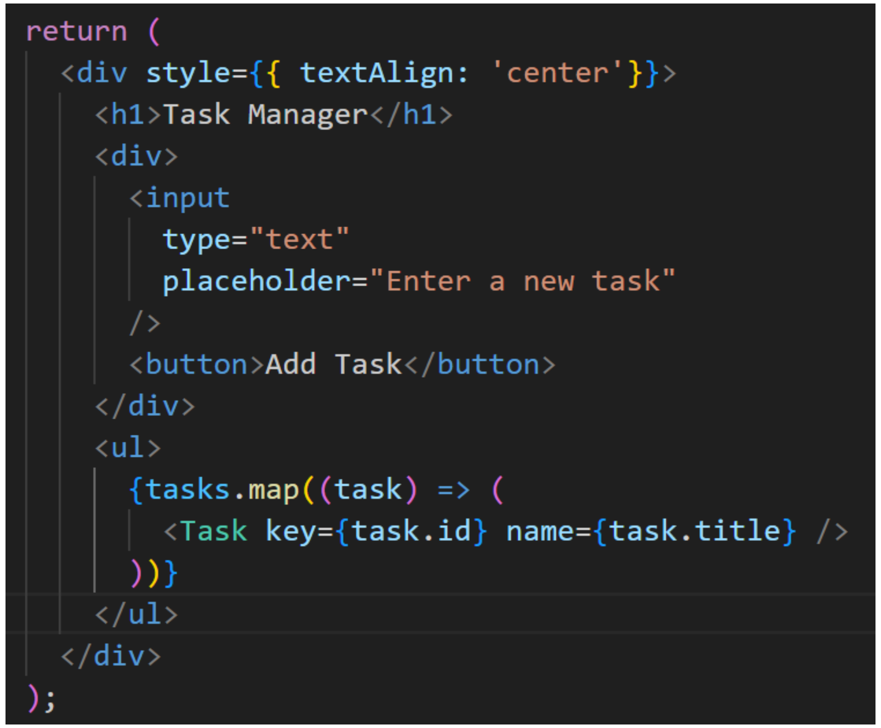
TRAINOSYS

Create a useState Hook for our Tasks



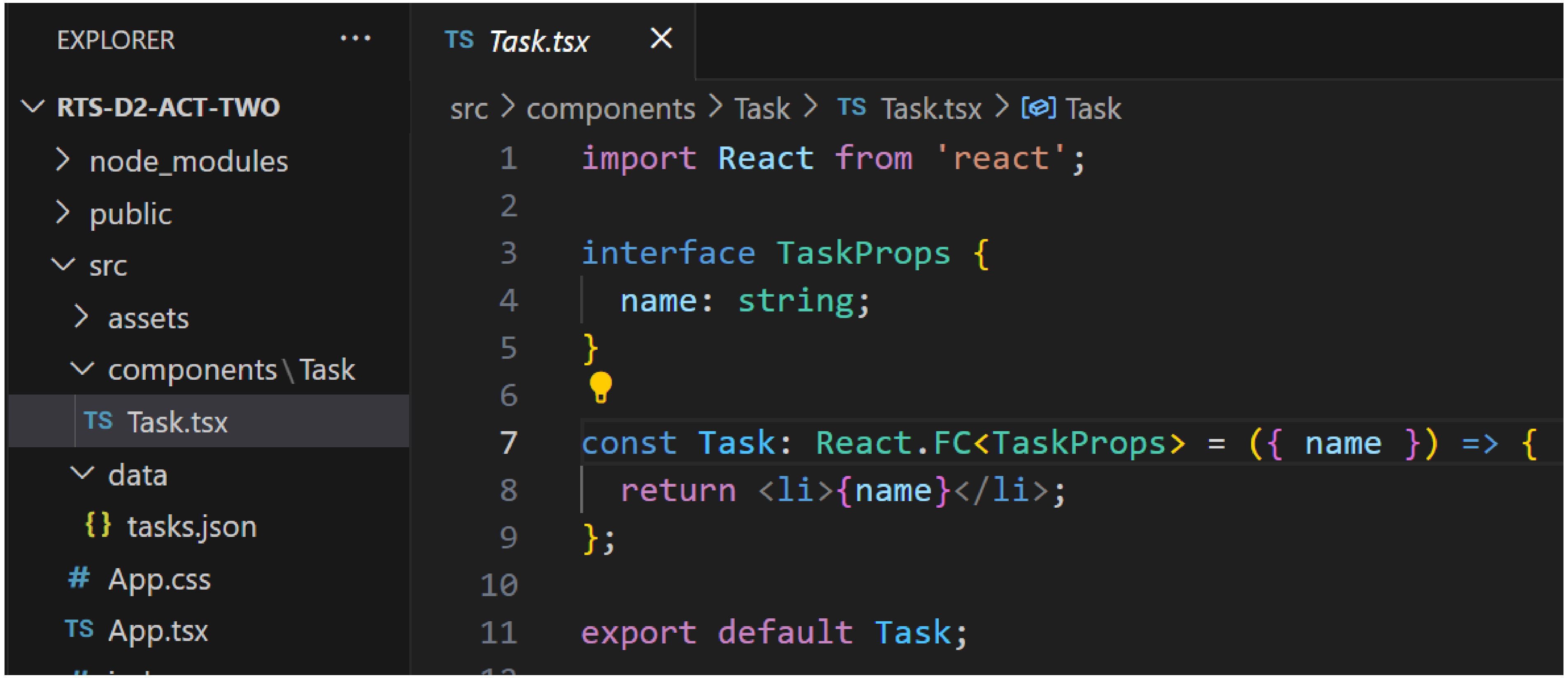
```

const App: React.FC = () => {
  const [tasks, setTasks] = useState<Task[]>(tasksData);

```

Pass Tasks[] in the <Task /> component

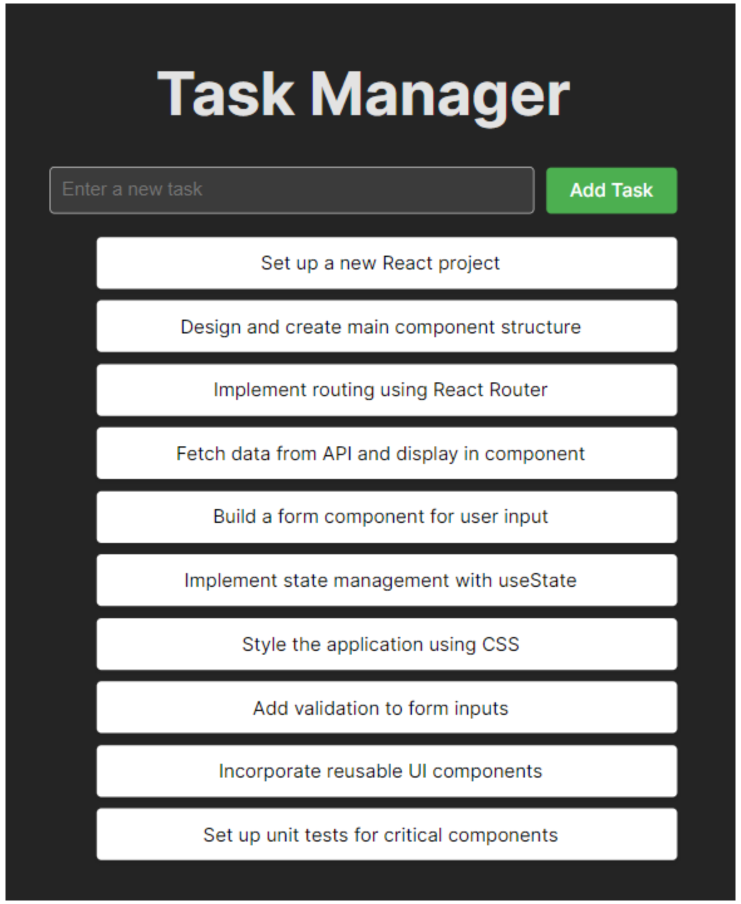


```
return (
  <div style={{ textAlign: 'center'}}>
    <h1>Task Manager</h1>
    <div>
      <input
        type="text"
        placeholder="Enter a new task"
      />
      <button>Add Task</button>
    </div>
    <ul>
      {tasks.map((task) => (
        <Task key={task.id} name={task.title} />
      ))}
    </ul>
  </div>
);
```

Create components folder and Task folder inside and <Task /> component



```tsx
import React from 'react';

interface TaskProps {
  name: string;
}

const Task: React.FC<TaskProps> = ({ name }) => {
  return <li>{name}</li>;
};

export default Task;
```



Task Manager

Enter a new task

Add Task

Set up a new React project

Design and create main component structure

Implement routing using React Router

Fetch data from API and display in component

Build a form component for user input

Implement state management with useState

Style the application using CSS

Add validation to form inputs

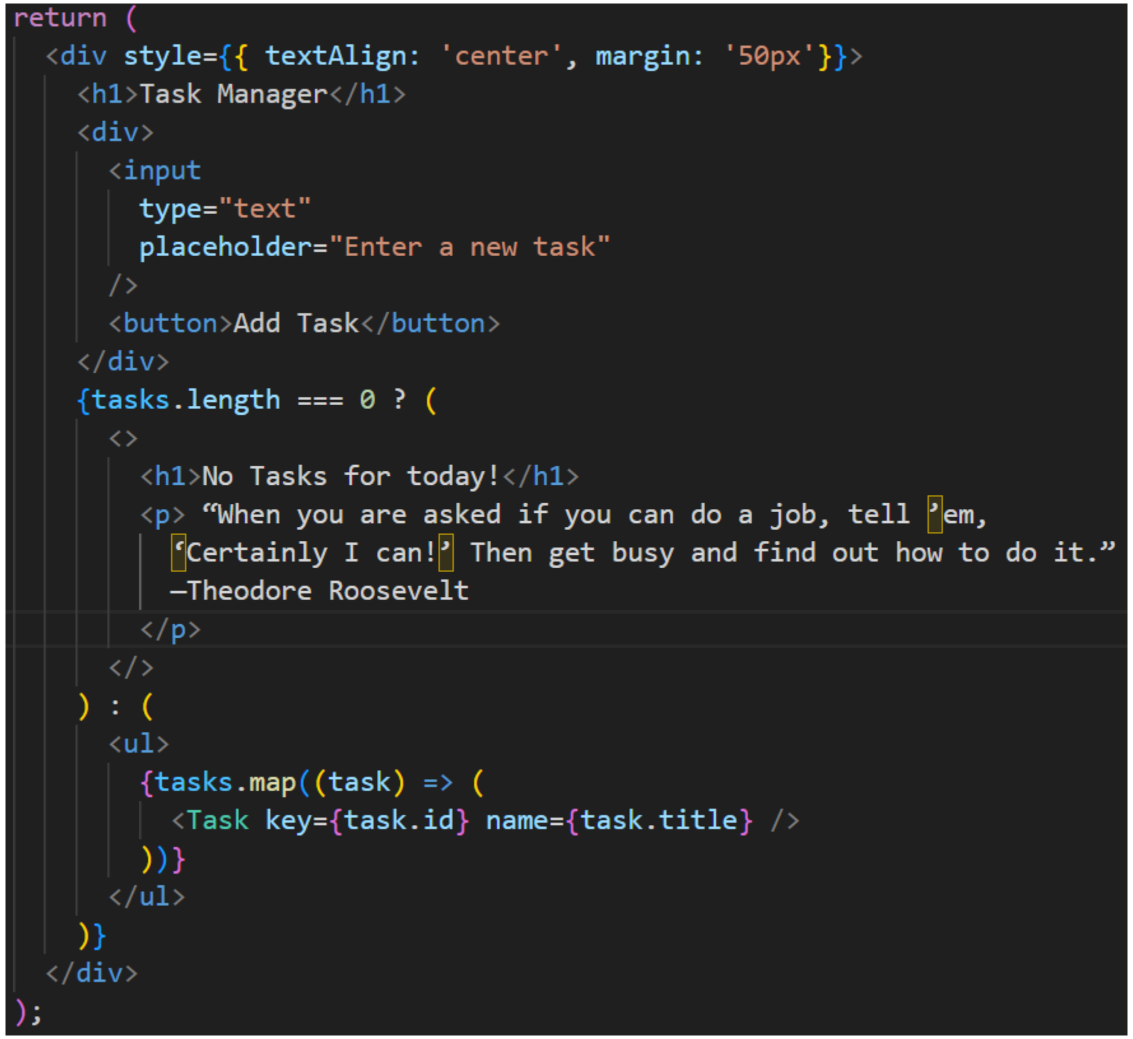
Incorporate reusable UI components

Set up unit tests for critical components

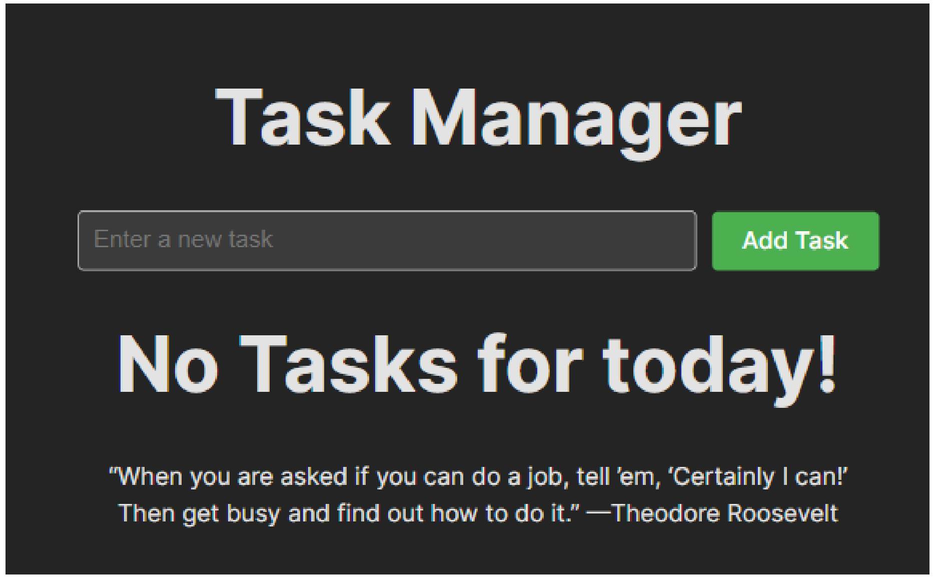
TRAINOSYS

TASK 3: Conditionally render the component if the Tasks array is empty

HYBRID TECHNOLOGY COURSES



```
return (
  <div style={{ textAlign: 'center', margin: '50px'}}>
    <h1>Task Manager</h1>
    <div>
      <input
        type="text"
        placeholder="Enter a new task"
      />
      <button>Add Task</button>
    </div>
    {tasks.length === 0 ? (
      <>
        <h1>No Tasks for today!</h1>
        <p> “When you are asked if you can do a job, tell ’em,
          ‘Certainly I can!’ Then get busy and find out how to do it.”
          —Theodore Roosevelt
        </p>
      </>
    ) : (
      <ul>
        {tasks.map((task) => (
          <Task key={task.id} name={task.title} />
        ))}
      </ul>
    )}
  </div>
);
```



Task Manager

Enter a new task | Add Task

No Tasks for today!

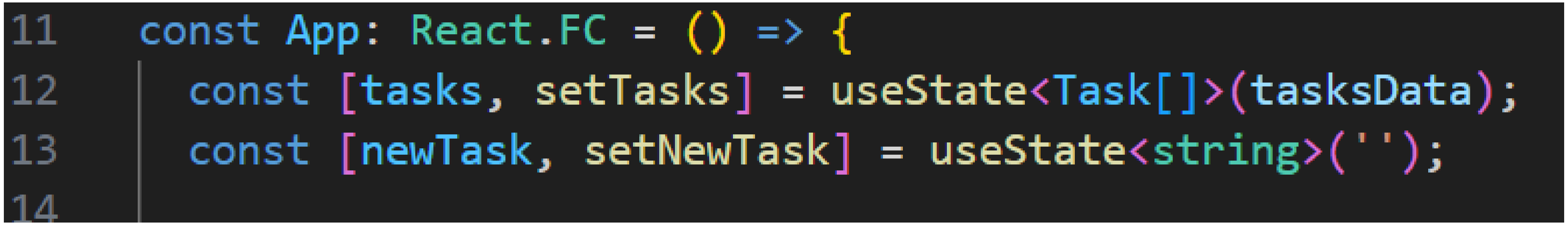
“When you are asked if you can do a job, tell ’em, ‘Certainly I can!’ Then get busy and find out how to do it.” —Theodore Roosevelt

TRAINOSYS

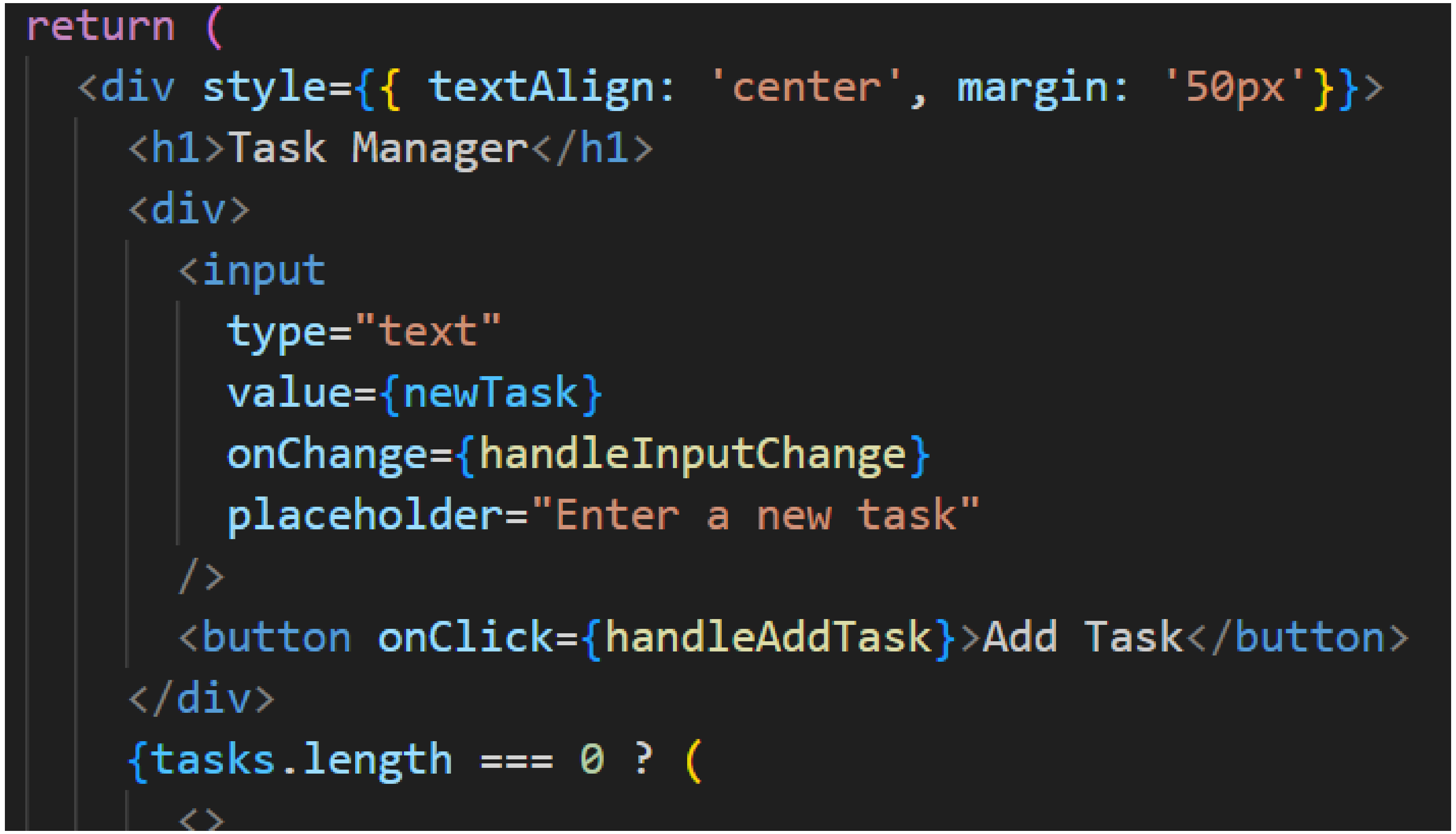
LET'S MAKE IT WORK!

HYBRID
TECHNOLOGY
COURSES

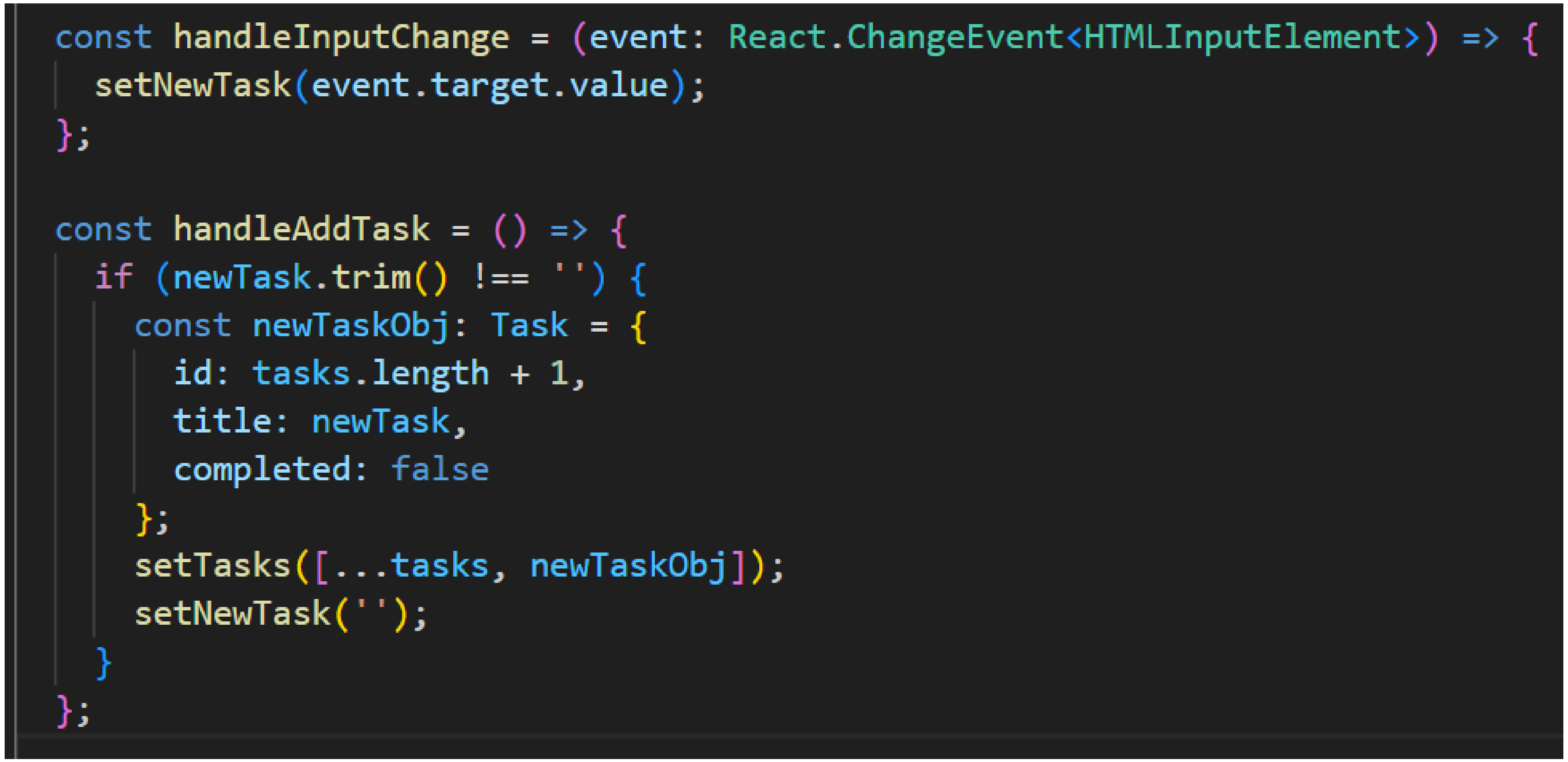
TRAINOSYS



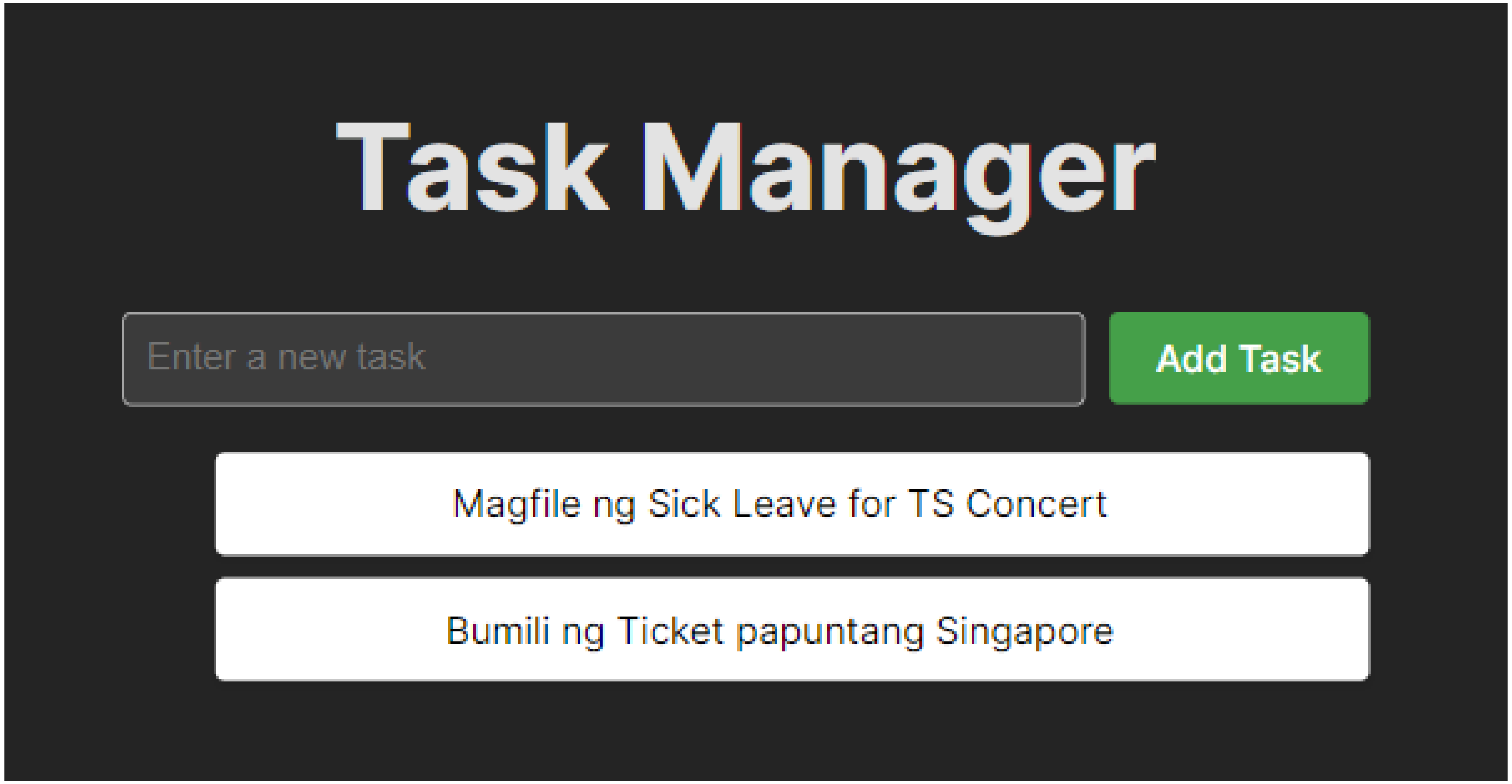
```
const App: React.FC = () => {
  const [tasks, setTasks] = useState<Task[]>(tasksData);
  const [newTask, setNewTask] = useState<string>('');

```



```
return (
  <div style={{ textAlign: 'center', margin: '50px'}}>
    <h1>Task Manager</h1>
    <div>
      <input
        type="text"
        value={newTask}
        onChange={handleInputChange}
        placeholder="Enter a new task"
      />
      <button onClick={handleAddTask}>Add Task</button>
    </div>
    {tasks.length === 0 ? (
      <>
```



```
const handleInputChange = (event: React.ChangeEvent<HTMLInputElement>) => {
  setNewTask(event.target.value);
};

const handleAddTask = () => {
  if (newTask.trim() !== '') {
    const newTaskObj: Task = {
      id: tasks.length + 1,
      title: newTask,
      completed: false
    };
    setTasks([...tasks, newTaskObj]);
    setNewTask('');
  }
};
```



Task Manager

Enter a new task

Magfile ng Sick Leave for TS Concert

Bumili ng Ticket papuntang Singapore

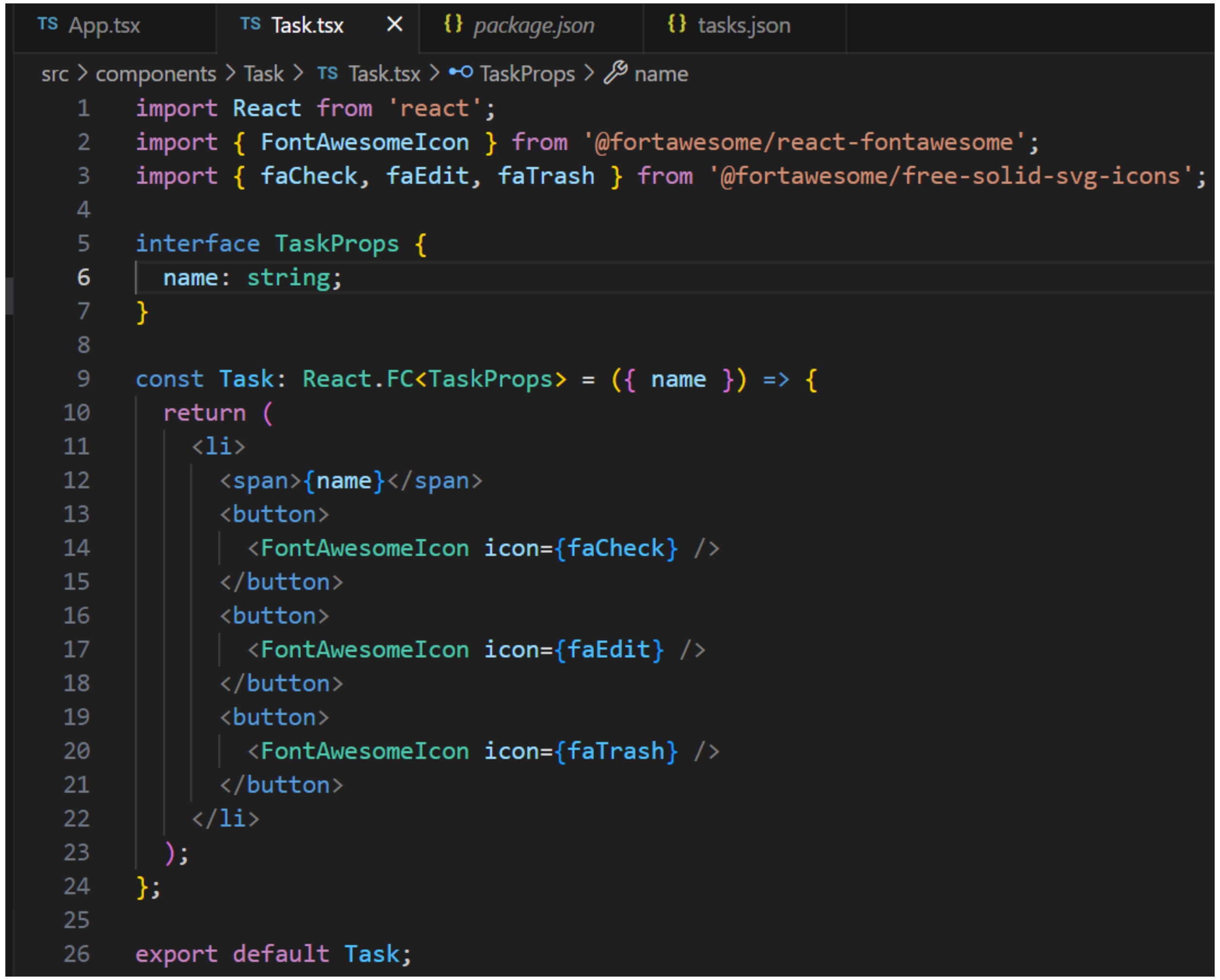
TRAINOSYS

TASK 4:
Add a done, edit and delete button / icon in the <Task /> component

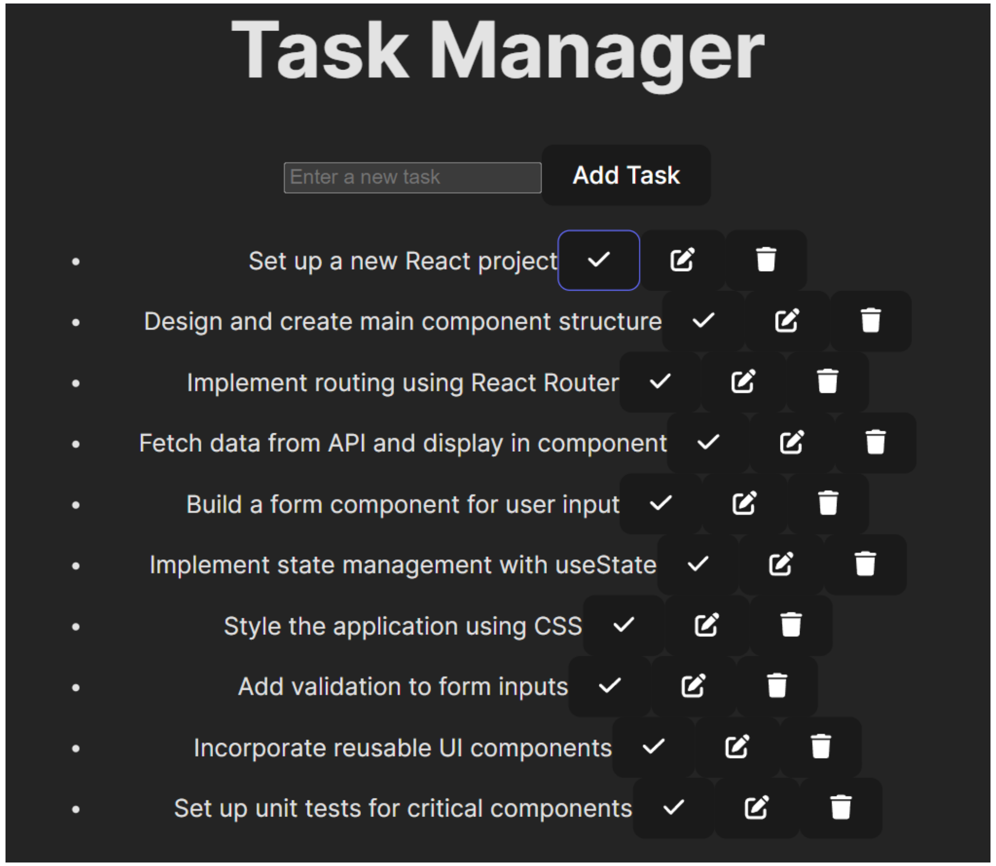
HYBRID
TECHNOLOGY
COURSES

Install some packages

```
npm install @fortawesome/fontawesome-svg-core
@fortawesome/free-solid-svg-icons @fortawesome/react-
fontawesome
```



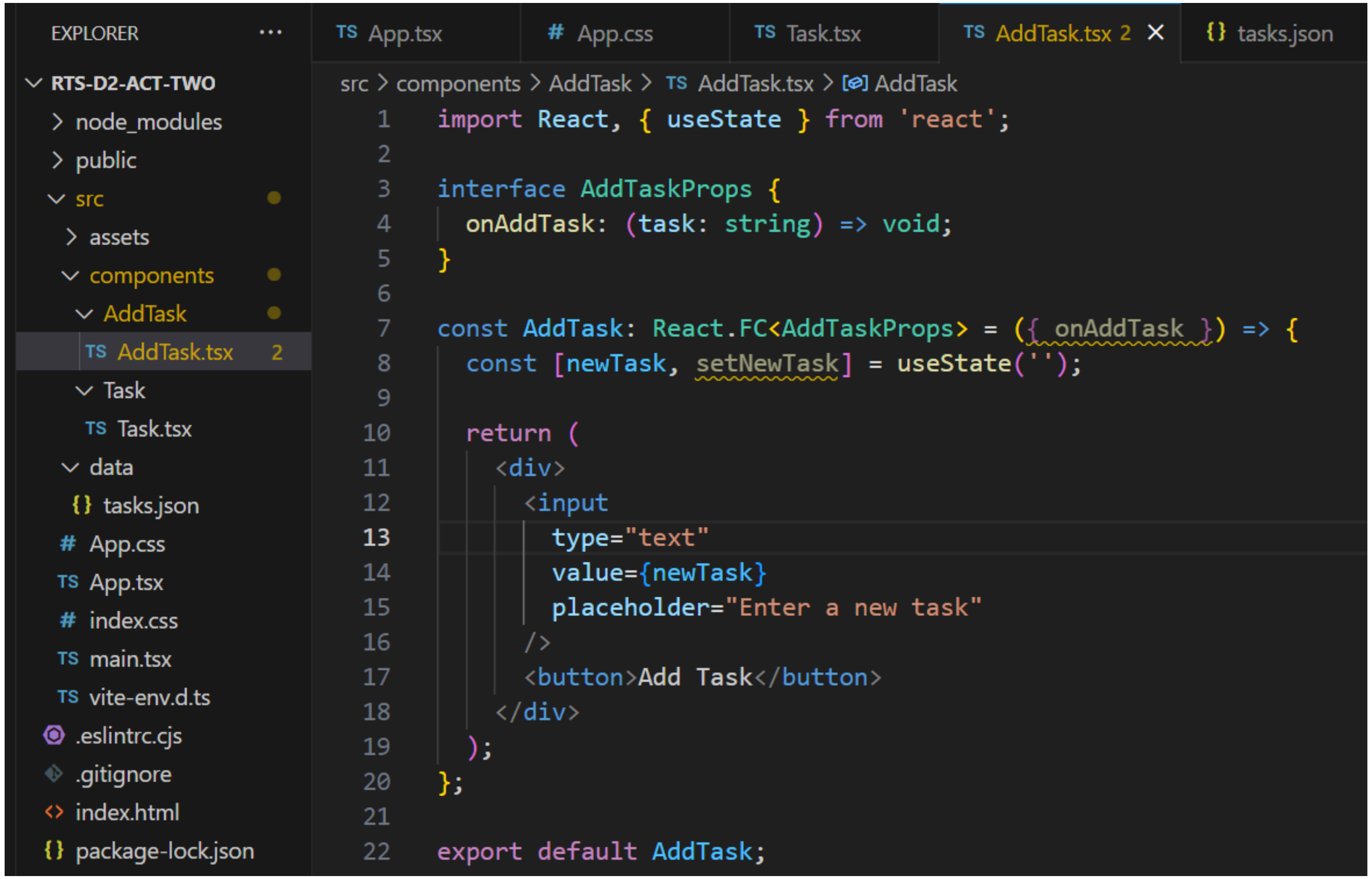
```tsx
import React from 'react';
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
import { faCheck, faEdit, faTrash } from '@fortawesome/free-solid-svg-icons';

interface TaskProps {
  name: string;
}

const Task: React.FC<TaskProps> = ({ name }) => {
  return (
    <li>
      <span>{name}</span>
      <button>
        <FontAwesomeIcon icon={faCheck} />
      </button>
      <button>
        <FontAwesomeIcon icon={faEdit} />
      </button>
      <button>
        <FontAwesomeIcon icon={faTrash} />
      </button>
    </li>
  );
};

export default Task;
```



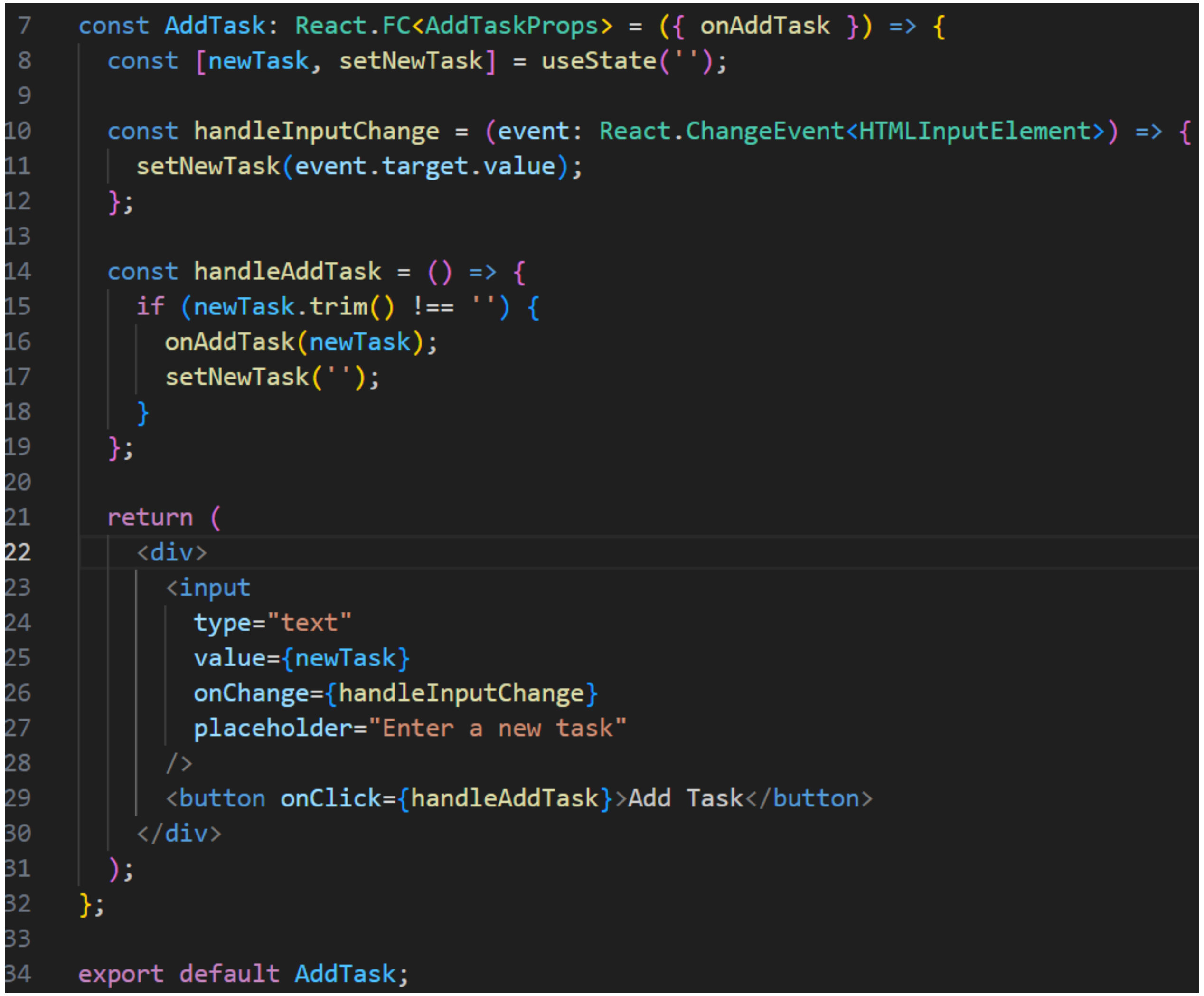
Task Manager

- Set up a new React project
- Design and create main component structure
- Implement routing using React Router
- Fetch data from API and display in component
- Build a form component for user input
- Implement state management with useState
- Style the application using CSS
- Add validation to form inputs
- Incorporate reusable UI components
- Set up unit tests for critical components

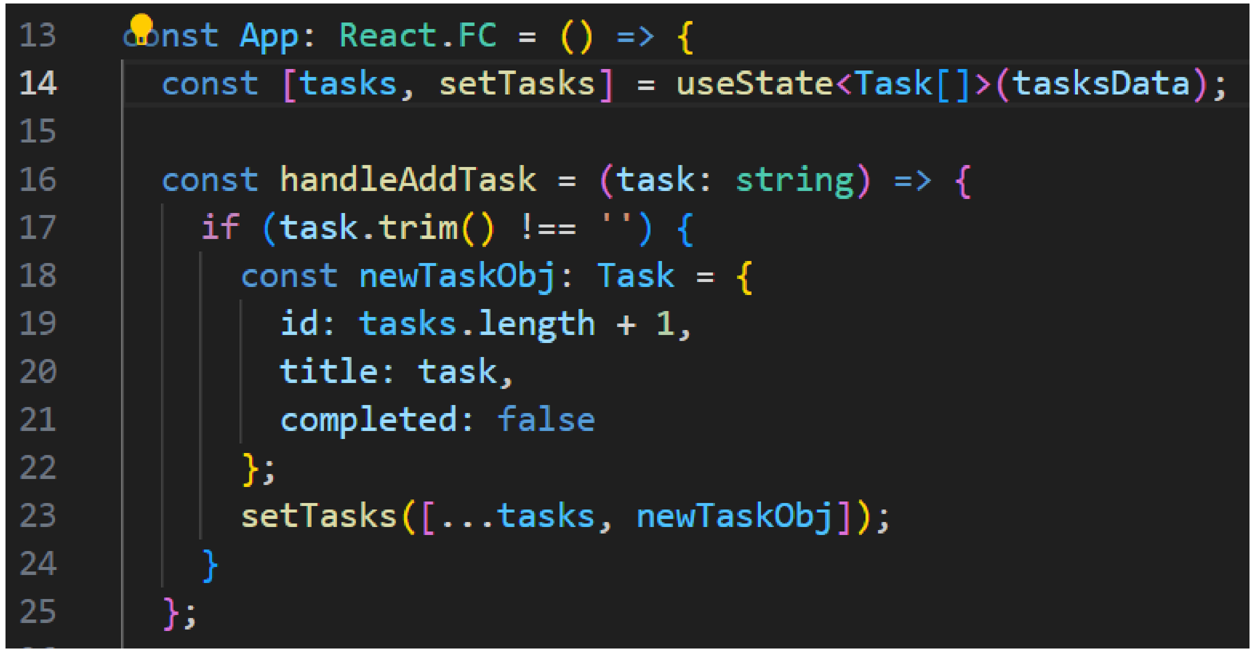
TASK 5:
Migrate the input form in the <AddTask /> component



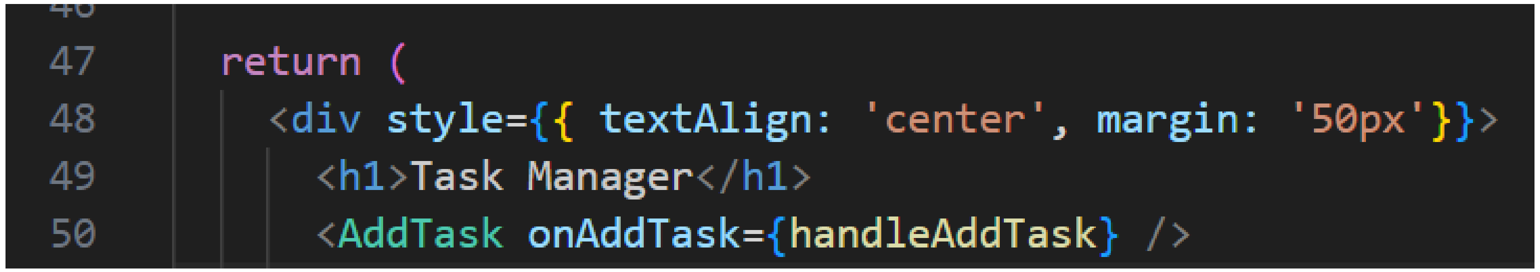
```tsx
import React, { useState } from 'react';

interface AddTaskProps {
  onAddTask: (task: string) => void;
}

const AddTask: React.FC<AddTaskProps> = ({ onAddTask }) => {
  const [newTask, setNewTask] = useState('');

  return (
    <div>
      <input
        type="text"
        value={newTask}
        placeholder="Enter a new task"
      />
      <button>Add Task</button>
    </div>
  );
};

export default AddTask;
```



```
const AddTask: React.FC<AddTaskProps> = ({ onAddTask }) => {
  const [newTask, setNewTask] = useState('');

  const handleInputChange = (event: React.ChangeEvent<HTMLInputElement>) => {
    setNewTask(event.target.value);
  };

  const handleAddTask = () => {
    if (newTask.trim() !== '') {
      onAddTask(newTask);
      setNewTask('');
    }
  };

  return (
    <div>
      <input
        type="text"
        value={newTask}
        onChange={handleInputChange}
        placeholder="Enter a new task"
      />
      <button onClick={handleAddTask}>Add Task</button>
    </div>
  );
};

export default AddTask;
```



```
const App: React.FC = () => {
  const [tasks, setTasks] = useState<Task[]>(tasksData);

  const handleAddTask = (task: string) => {
    if (task.trim() !== '') {
      const newTaskObj: Task = {
        id: tasks.length + 1,
        title: task,
        completed: false
      };
      setTasks([...tasks, newTaskObj]);
    }
  };
```

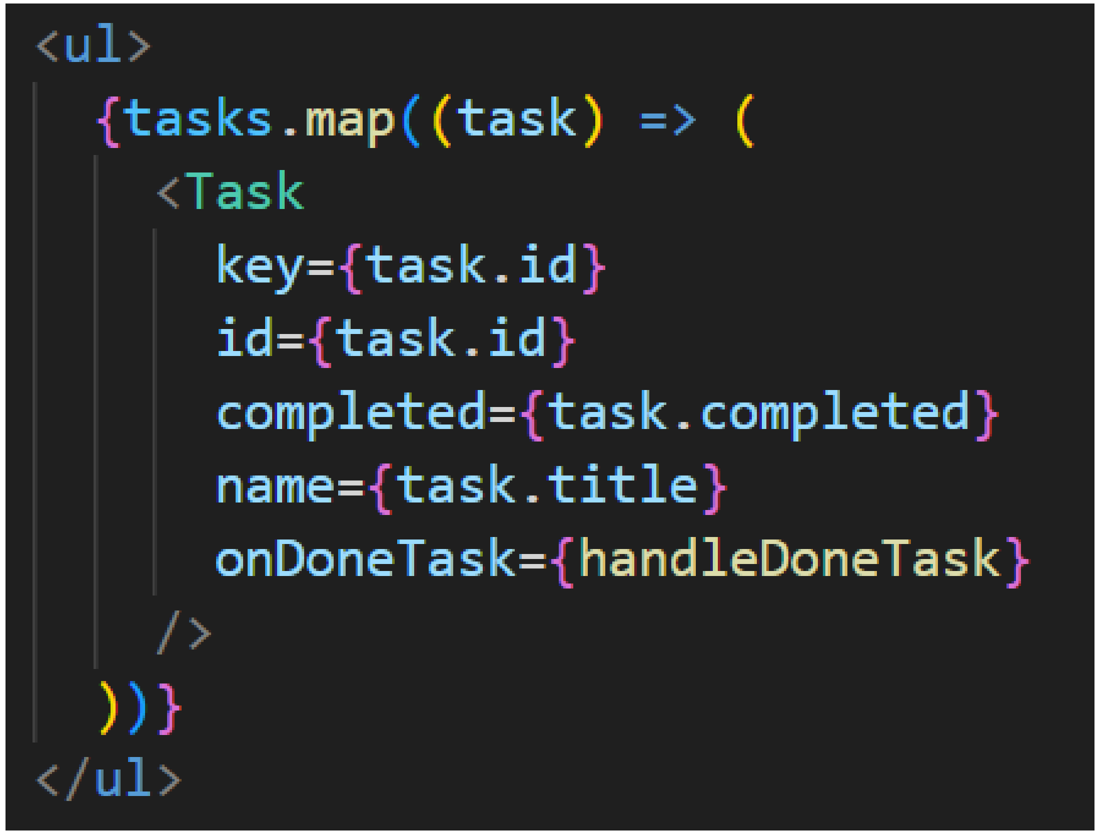


```

  return (
    <div style={{ textAlign: 'center', margin: '50px'}}>
      <h1>Task Manager</h1>
      <AddTask onAddTask={handleAddTask} />
```

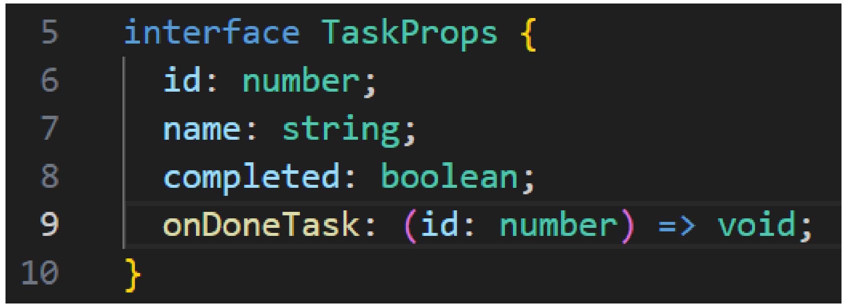
TRAINOSYS

TASK 6:
Make the done button / icon work by creating a doneTask function

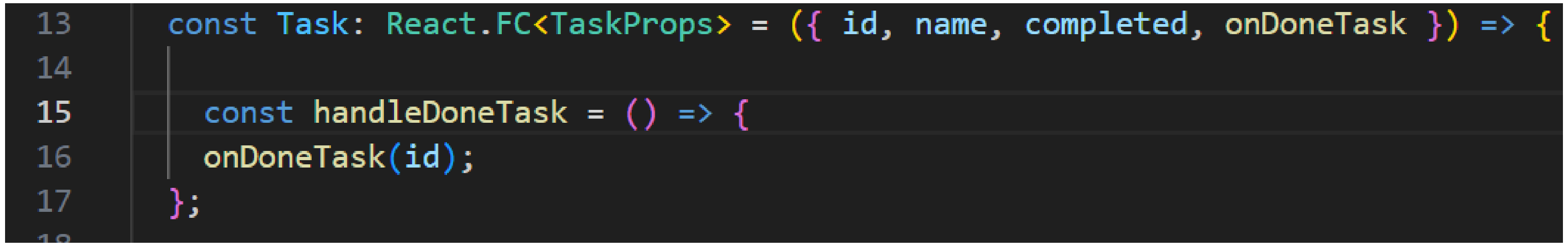
HYBRID TECHNOLOGY COURSES



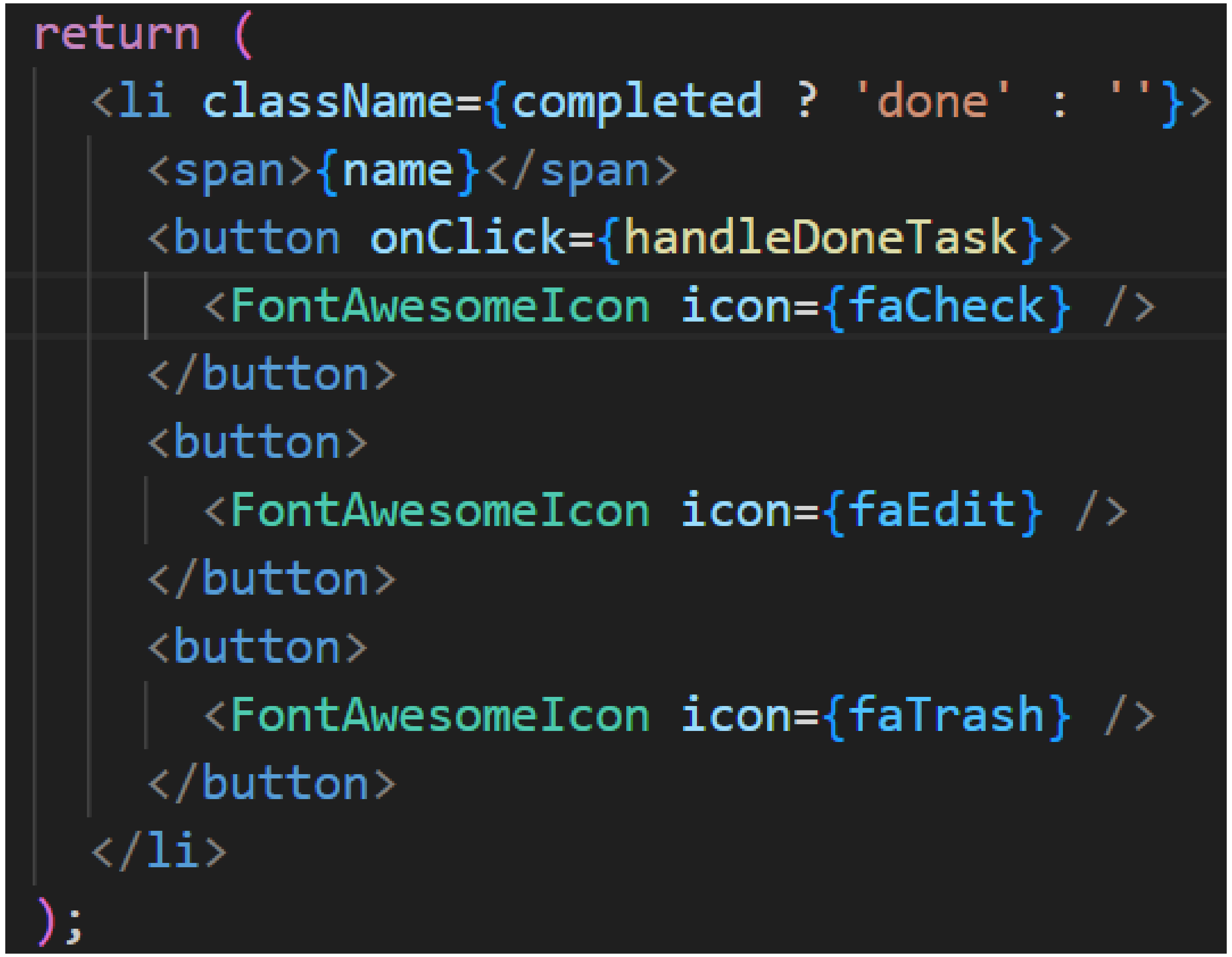
```
<ul>
  {tasks.map((task) => (
    <Task
      key={task.id}
      id={task.id}
      completed={task.completed}
      name={task.title}
      onDoneTask={handleDoneTask}
    />
  ))}
</ul>
```



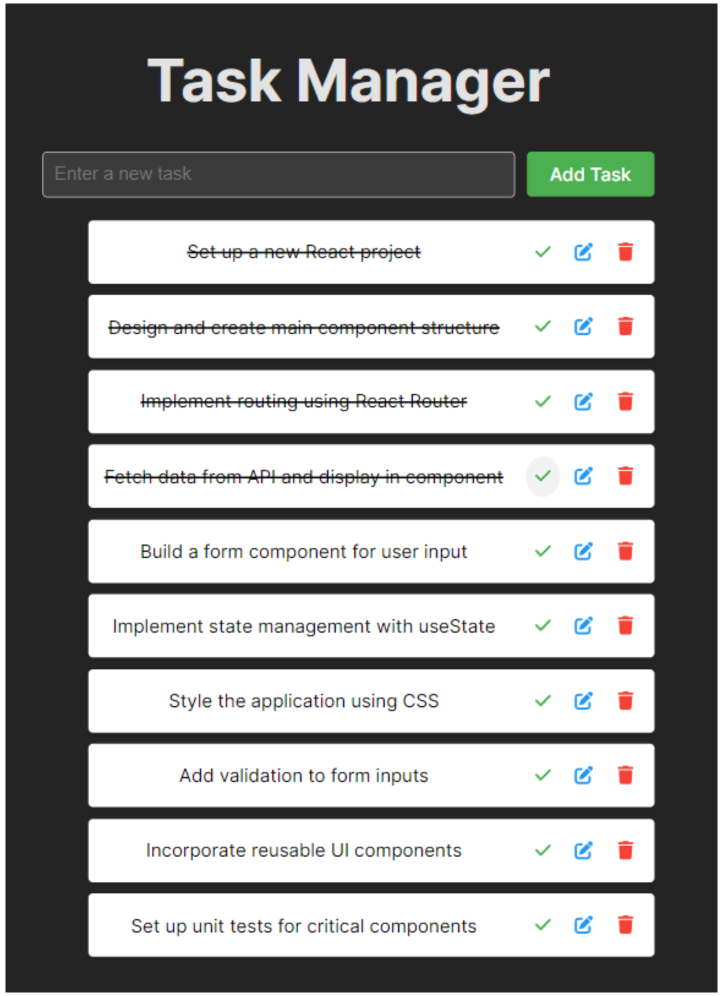
```
interface TaskProps {
  id: number;
  name: string;
  completed: boolean;
  onDoneTask: (id: number) => void;
}
```



```
const Task: React.FC<TaskProps> = ({ id, name, completed, onDoneTask }) => {

  const handleDoneTask = () => {
  onDoneTask(id);
};
```



```
return (
  <li className={completed ? 'done' : ''}>
    <span>{name}</span>
    <button onClick={handleDoneTask}>
      <FontAwesomeIcon icon={faCheck} />
    </button>
    <button>
      <FontAwesomeIcon icon={faEdit} />
    </button>
    <button>
      <FontAwesomeIcon icon={faTrash} />
    </button>
  </li>
);
```



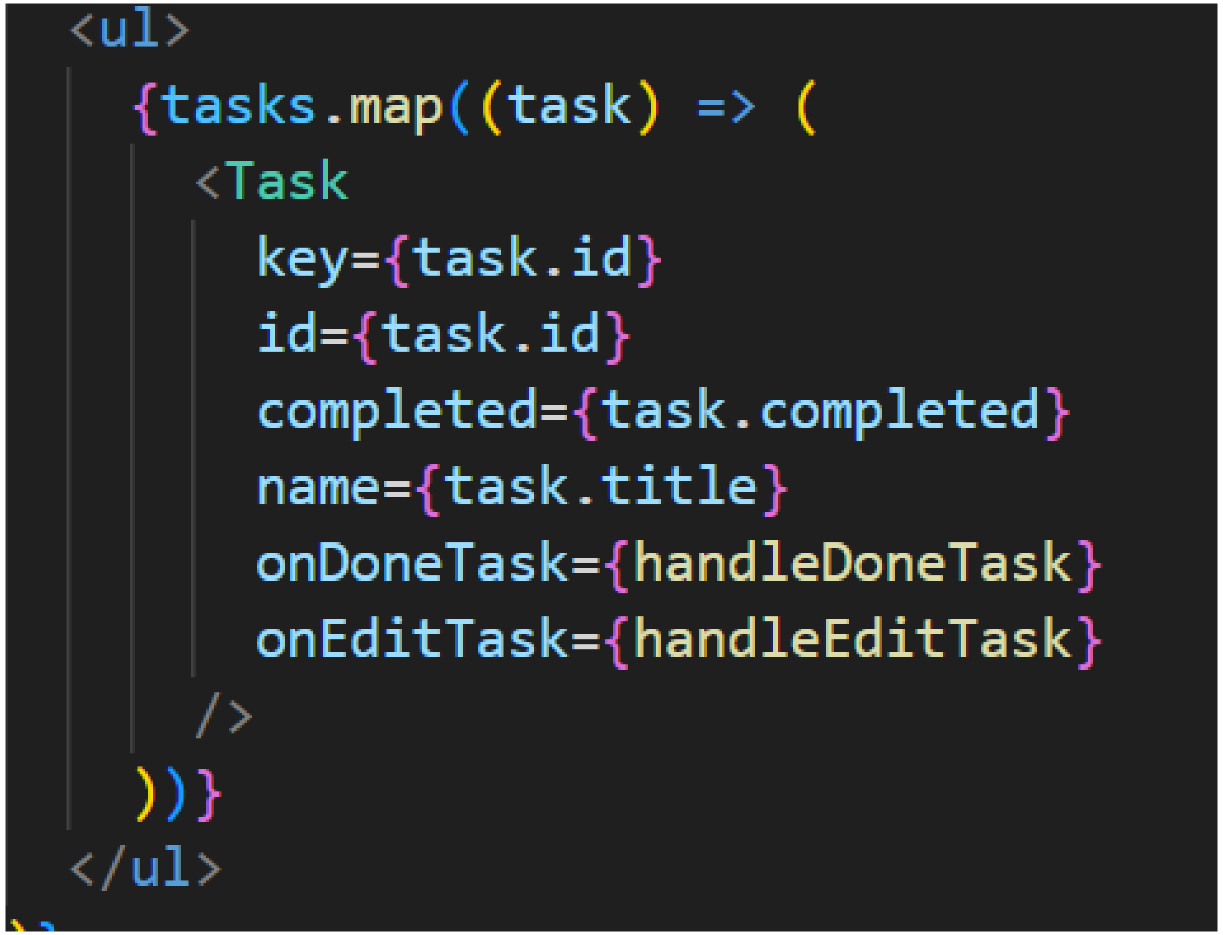
Task Manager

~~Set up a new React project~~ ✓
~~Design and create main component structure~~ ✓
~~Implement routing using React Router~~ ✓
~~Fetch data from API and display in component~~ ✓
Build a form component for user input
Implement state management with useState
Style the application using CSS
Add validation to form inputs
Incorporate reusable UI components
Set up unit tests for critical components

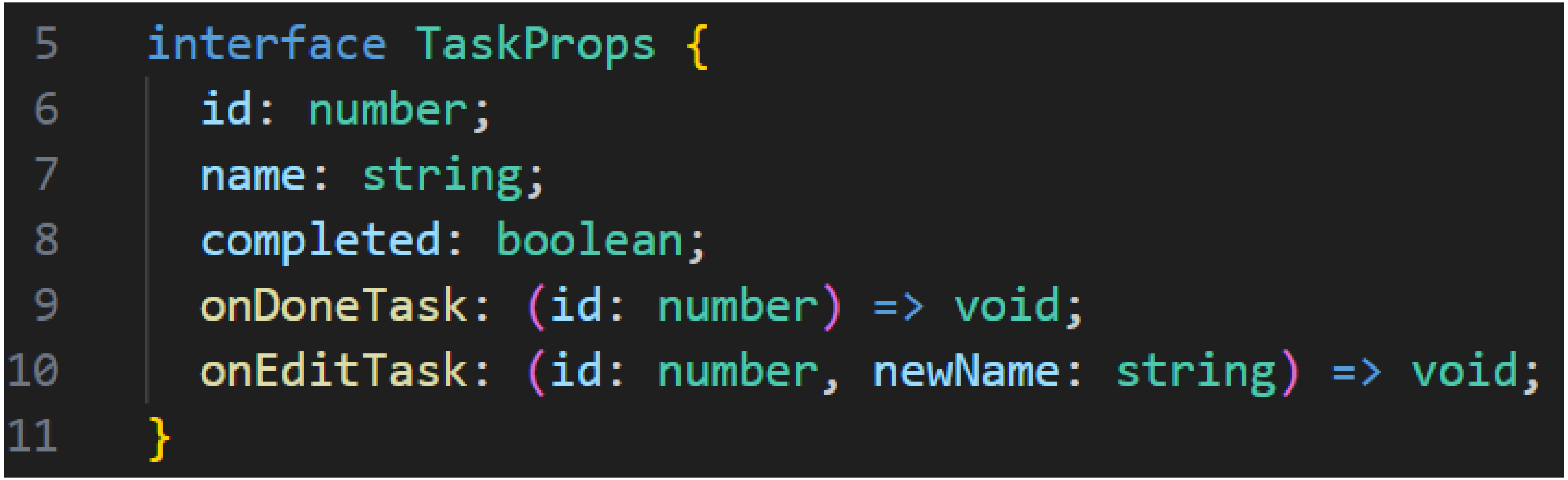
TRAINOSYS

TASK 7:
Make the edit button / icon work by creating an editTask function

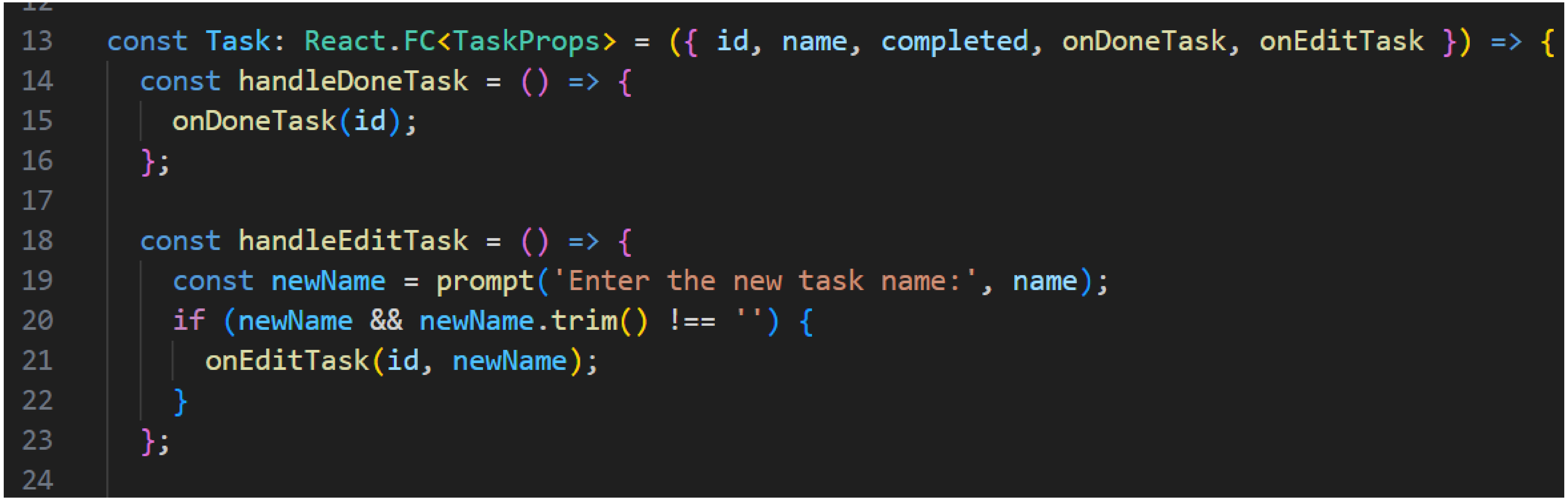
HYBRID
TECHNOLOGY
COURSES



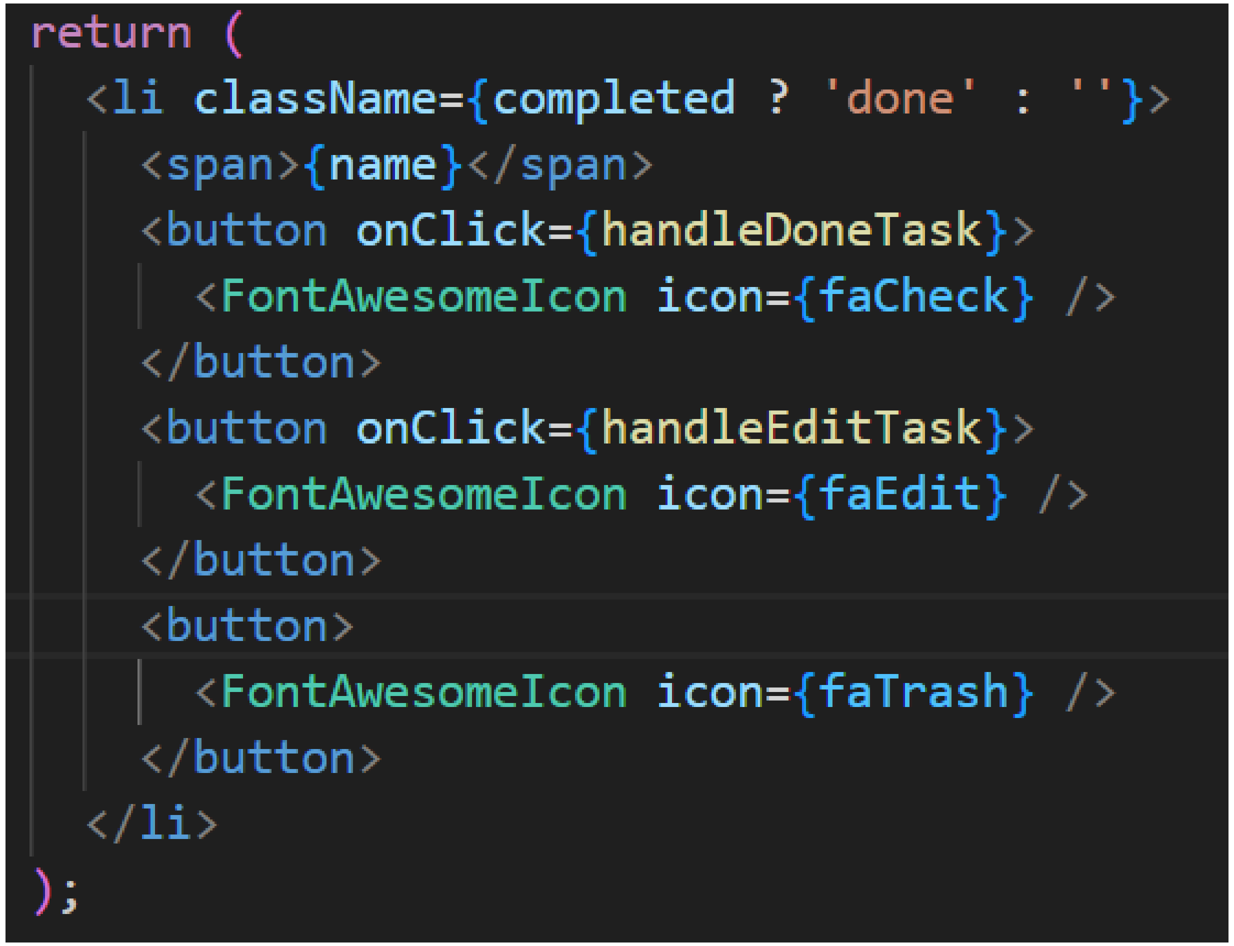
```
<ul>
  {tasks.map((task) => (
    <Task
      key={task.id}
      id={task.id}
      completed={task.completed}
      name={task.title}
      onDoneTask={handleDoneTask}
      onEditTask={handleEditTask}
    />
  ))}
</ul>
```



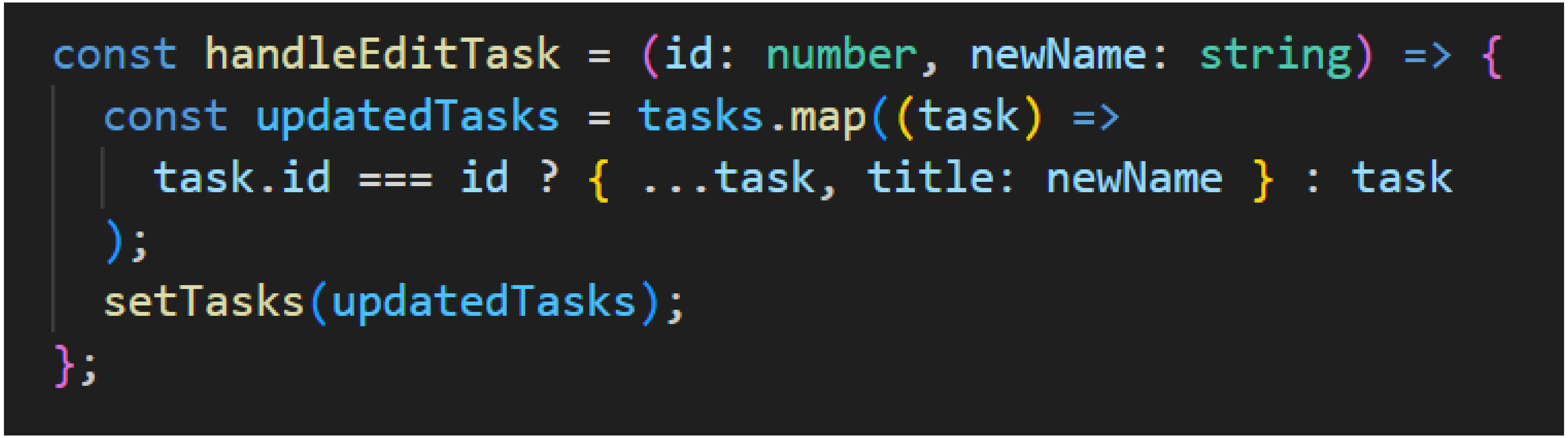
```
interface TaskProps {
  id: number;
  name: string;
  completed: boolean;
  onDoneTask: (id: number) => void;
  onEditTask: (id: number, newName: string) => void;
}
```



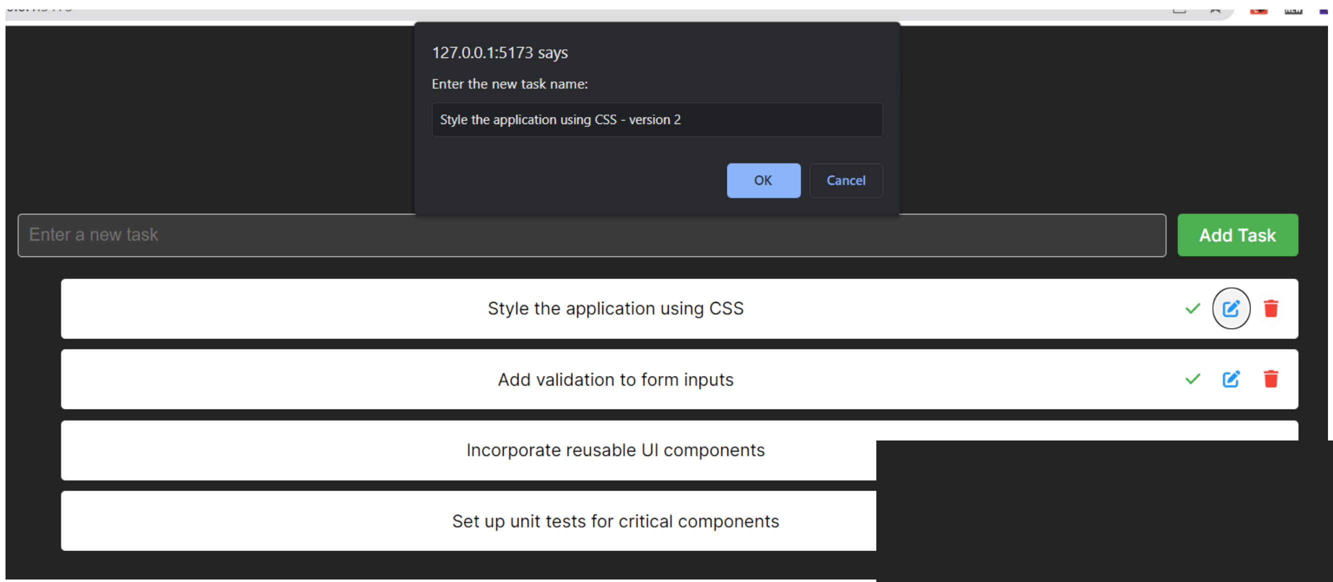
```
const Task: React.FC<TaskProps> = ({ id, name, completed, onDoneTask, onEditTask }) => {
  const handleDoneTask = () => {
    onDoneTask(id);
  };

  const handleEditTask = () => {
    const newName = prompt('Enter the new task name:', name);
    if (newName && newName.trim() !== '') {
      onEditTask(id, newName);
    }
  };

```



```
return (
  <li className={completed ? 'done' : ''}>
    <span>{name}</span>
    <button onClick={handleDoneTask}>
      <FontAwesomeIcon icon={faCheck} />
    </button>
    <button onClick={handleEditTask}>
      <FontAwesomeIcon icon={faEdit} />
    </button>
    <button>
      <FontAwesomeIcon icon={faTrash} />
    </button>
  </li>
);
```



```
const handleEditTask = (id: number, newName: string) => {
  const updatedTasks = tasks.map((task) =>
    task.id === id ? { ...task, title: newName } : task
  );
  setTasks(updatedTasks);
};
```


127.0.0.1:5173 says

Enter the new task name:

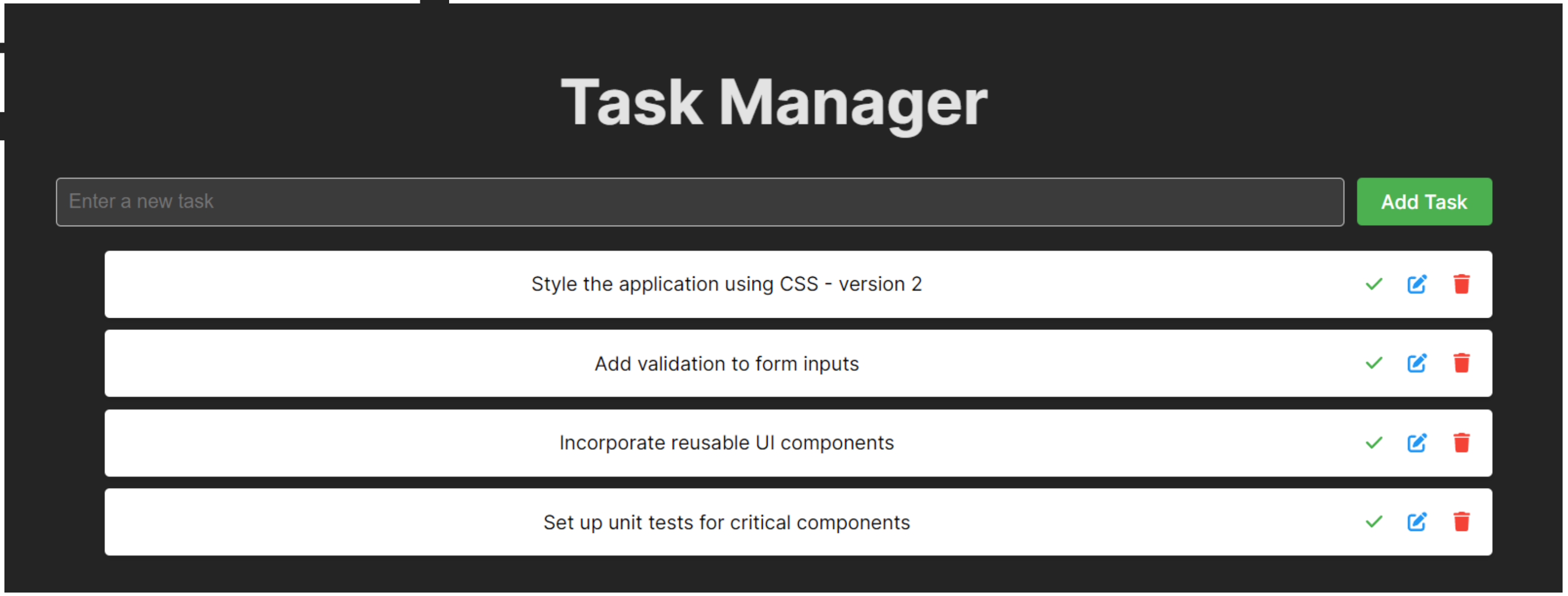
Style the application using CSS - version 2

Style the application using CSS

Add validation to form inputs

Incorporate reusable UI components

Set up unit tests for critical components


Task Manager

Style the application using CSS - version 2

Add validation to form inputs

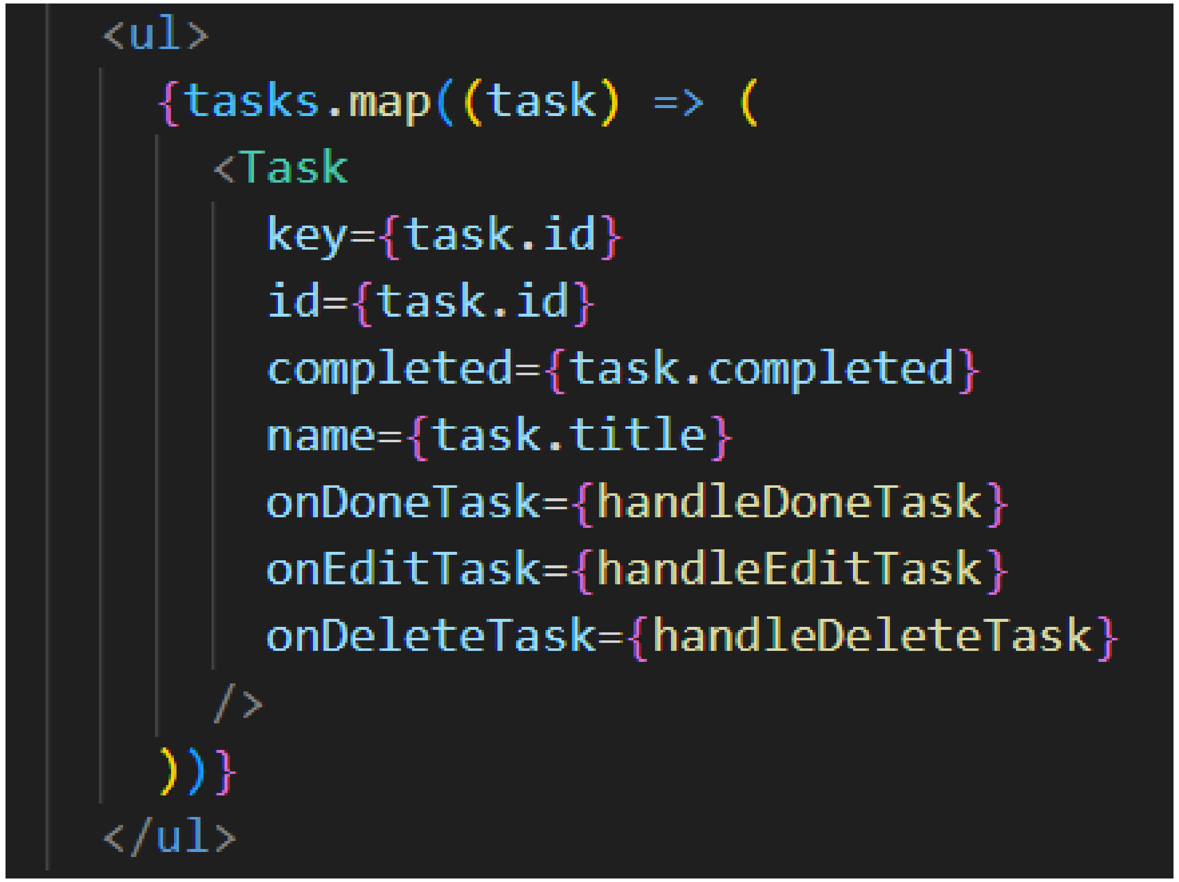
Incorporate reusable UI components

Set up unit tests for critical components

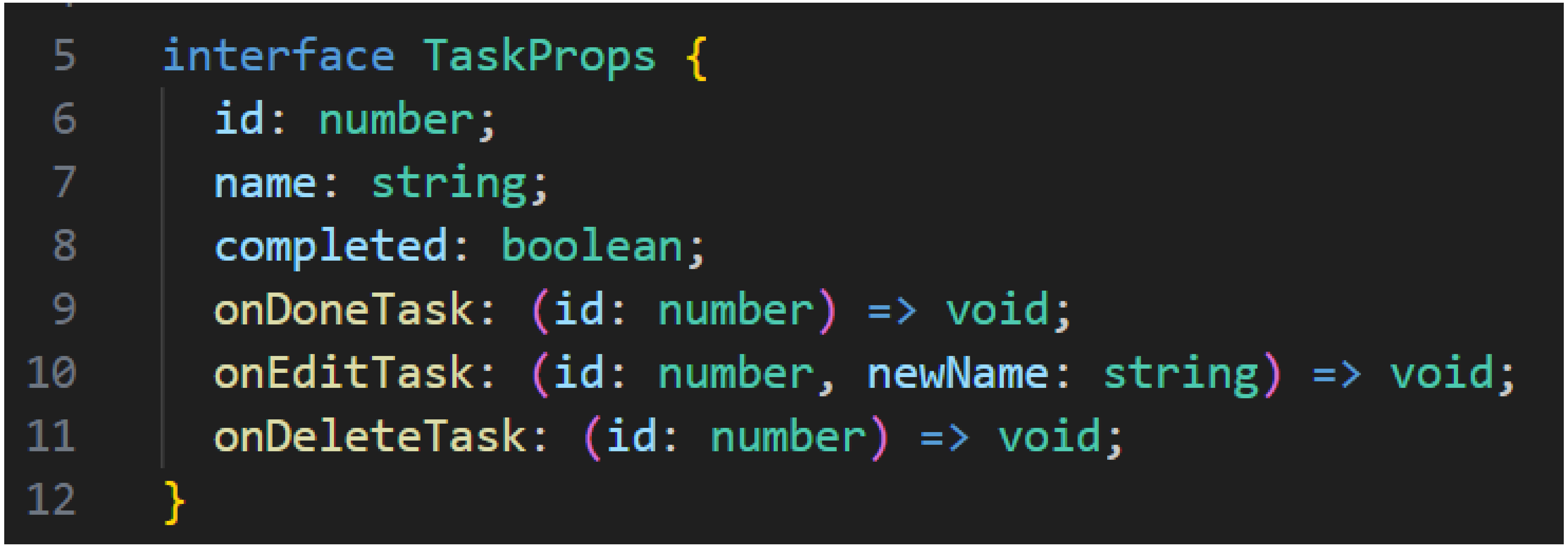
TRAINOSYS

TASK 8:
Make the delete button / icon work by creating a deleteTask function

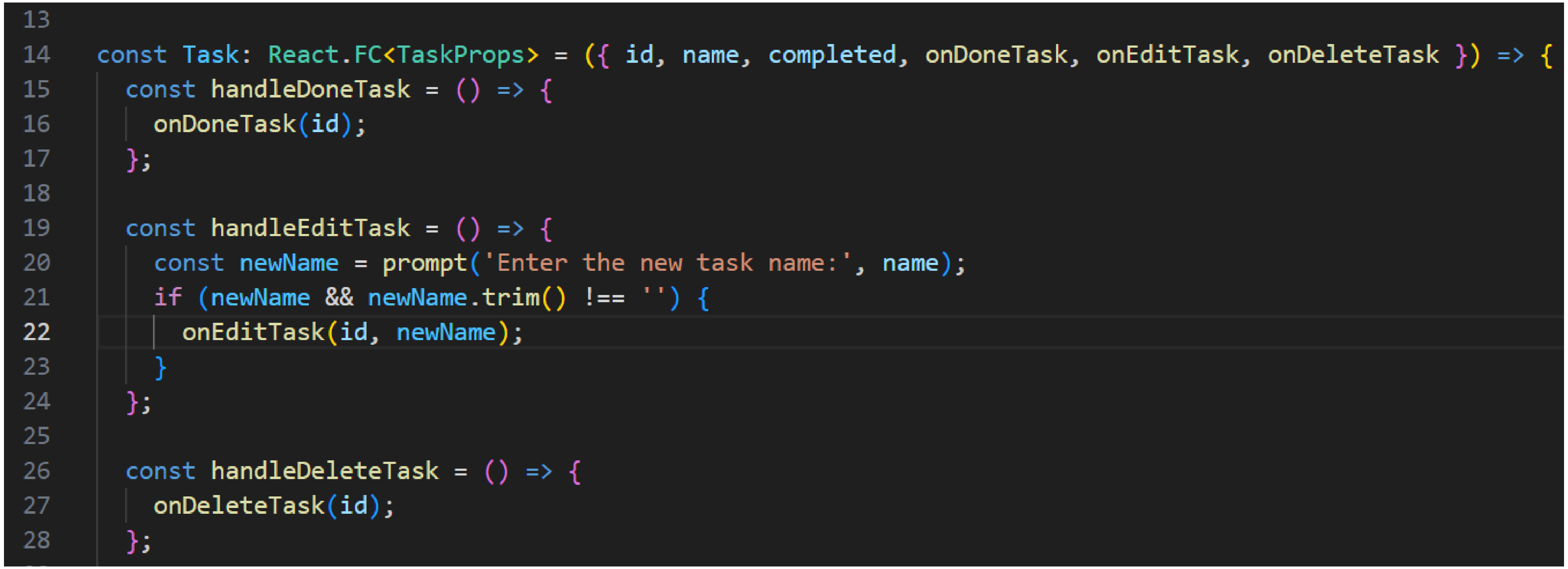
HYBRID
TECHNOLOGY
COURSES



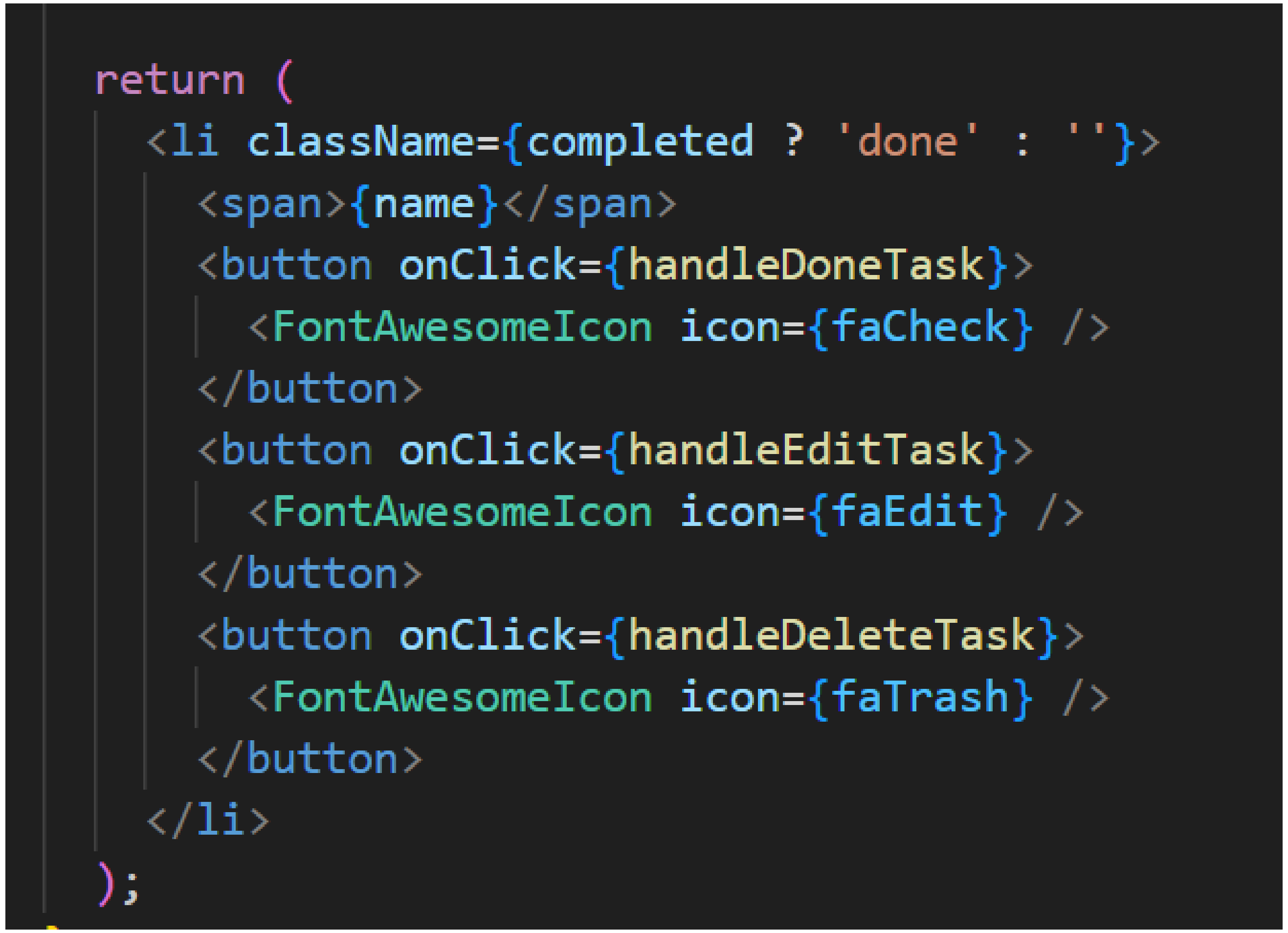
```
<ul>
  {tasks.map((task) => (
    <Task
      key={task.id}
      id={task.id}
      completed={task.completed}
      name={task.title}
      onDoneTask={handleDoneTask}
      onEditTask={handleEditTask}
      onDeleteTask={handleDeleteTask}
    />
  ))}
</ul>
```



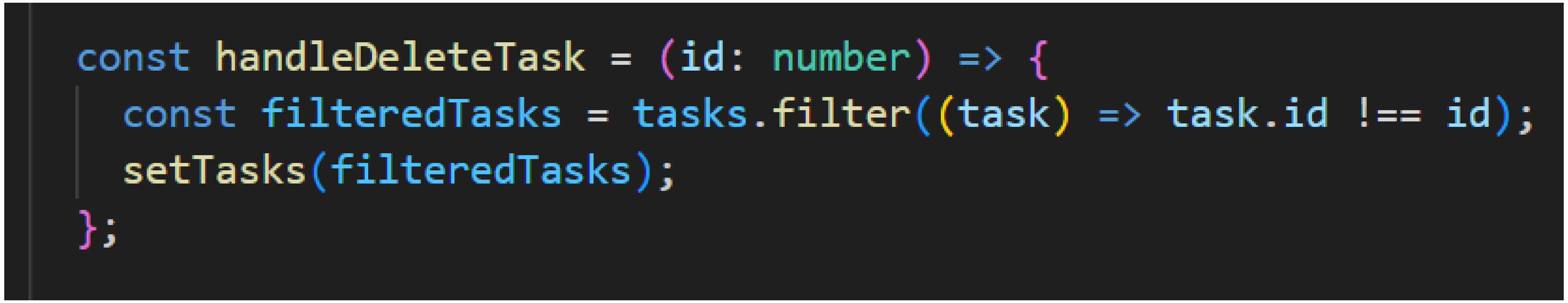
```
interface TaskProps {
  id: number;
  name: string;
  completed: boolean;
  onDoneTask: (id: number) => void;
  onEditTask: (id: number, newName: string) => void;
  onDeleteTask: (id: number) => void;
}
```



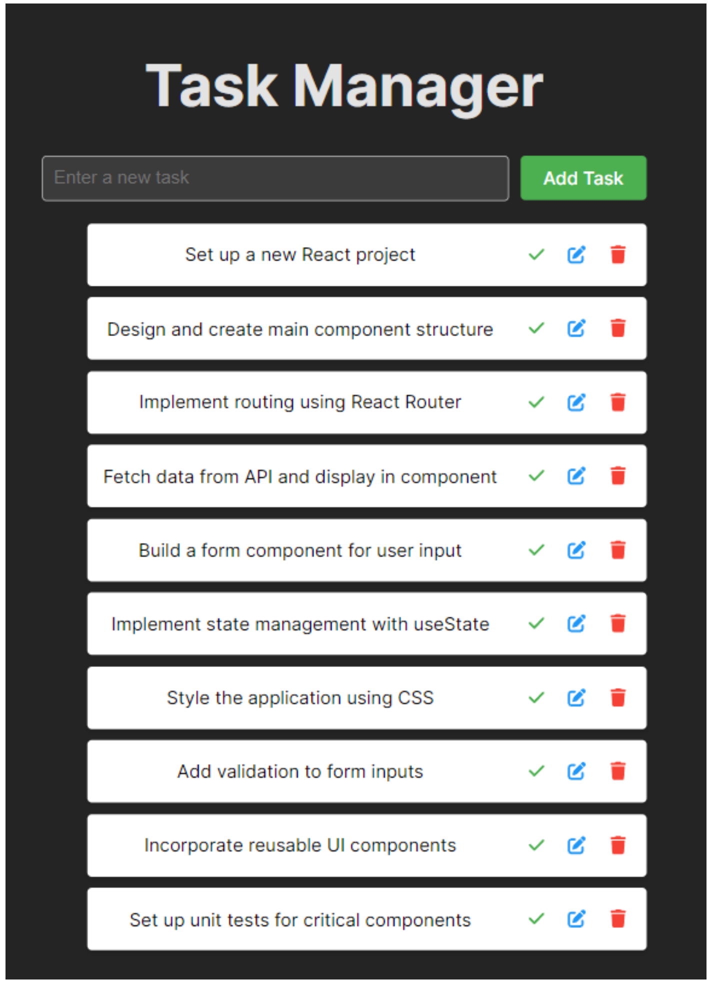
```

const Task: React.FC<TaskProps> = ({ id, name, completed, onDoneTask, onEditTask, onDeleteTask }) => {
  const handleDoneTask = () => {
    onDoneTask(id);
  };

  const handleEditTask = () => {
    const newName = prompt('Enter the new task name:', name);
    if (newName && newName.trim() !== '') {
      onEditTask(id, newName);
    }
  };

  const handleDeleteTask = () => {
    onDeleteTask(id);
  };
```



```
return (
  <li className={completed ? 'done' : ''}>
    <span>{name}</span>
    <button onClick={handleDoneTask}>
      <FontAwesomeIcon icon={faCheck} />
    </button>
    <button onClick={handleEditTask}>
      <FontAwesomeIcon icon={faEdit} />
    </button>
    <button onClick={handleDeleteTask}>
      <FontAwesomeIcon icon={faTrash} />
    </button>
  </li>
);
```



```
const handleDeleteTask = (id: number) => {
  const filteredTasks = tasks.filter((task) => task.id !== id);
  setTasks(filteredTasks);
};
```



Task Manager

Enter a new task

Add Task

- Set up a new React project
- Design and create main component structure
- Implement routing using React Router
- Fetch data from API and display in component
- Build a form component for user input
- Implement state management with useState
- Style the application using CSS
- Add validation to form inputs
- Incorporate reusable UI components
- Set up unit tests for critical components

TRAINOSYS

Activity 2

- ❑ npm create vite@latest / npm create vite@4.1.0
- ❑ Project name: rts-d3-act-two

- ❑ Create a React Router for the Task Manager
- ❑ Make dynamic routes for each Task in the Task List [id]
- ❑ Create another <SingleTask /> component to render each Task

TRAINOSYS
Training the Future Today

Reach Us!

Visit Us

12th/F The Trade & Financial Tower Unit 1206
32nd Street & 7th Avenue Bonifacio Global City,
Taguig 1634 Philippines

Email Us

inquiry@trainosys.com

Browse Our Website

www.trainosys.com

TRAINOSYS

Training the Future Today