



leonardrichertz

@workspace Generate a login view for the user to enter his email address and his password, which after being submitted makes an HTTP-POST request to our backend API and expects an authentication Token which is to be stored in the local storage. Sanitize the user's input on the client's side.



Workspace

> Used 3 references



Collecting workspace information

To create a login view, we will add a new component `Login.jsx` and handle the form submission to make an HTTP-POST request to the backend API. We will also sanitize the user's input and store the authentication token in local storage.

First, create a new file `src/routes/Login.jsx` :

Next, update your `main.jsx` to include routing and lazy load the `Login` component:

Finally, ensure you have the API URL in your `.env` file:

This setup will create a login view, sanitize user input, make an HTTP-POST request to the backend API, and store the authentication token in local storage.

Similar code found with 1 license type - [View matches](#)





leonardrichertz

@workspace /fix Failed to resolve import "react-router-dom" from "src/main.jsx". Does the file exist?

```
C:/Users/Schola/Desktop/Bachelorarbeit/copilot_generated_code/copilot-react-frontend/src/main.jsx:3:55 17 | import React, { Suspense, lazy } from "react"; 18 | import ReactDOM from "react-dom"; 19 | import { BrowserRouter as Router, Route, Switch } from "react-router-dom"; | ^ 20 | import App from "./App"; 21 | import "./index.css";
```





Workspace


> Used 1 reference

It looks like you are using Vite.js, which requires a different setup for React Router. Let's update the code to use the correct imports and structure for a Vite.js app.

First, install the necessary dependencies:

Next, update your  `main.jsx` to use the correct imports and structure for Vite.js:

Ensure you have the necessary environment variables in your  `.env` file:

Create the `Login.jsx` component in  `Login.jsx` :

This setup should resolve the import issue and correctly structure your Vite.js app with React Router and lazy-loaded components.

Similar code found with 1 license type - [View matches](#)



leonardrichertz

I am using pnpm as a package manager. Please fix the install command for the react-router-dom package



GitHub Copilot

> Used 1 reference

To install the `react-router-dom` package using `pnpm`, you can use the following command:

Next, let's update the code to follow your provided instructions.

1. Install dependencies:

2. **Update** `main.jsx` :

3. **Create** `Login.jsx` :

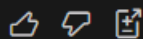
4. Create `src/routes/Loader.jsx`:

5. Create `src/routes/Page.jsx`:

6. Ensure you have the necessary environment variables in your `.env` file:

This setup should follow your instructions and structure the Vite.js app correctly with lazy-loaded components and environment-specific context variables.

Similar code found with 1 license type - [View matches](#)



leonardrichertz

`@workspace` `/fix` SyntaxError: The requested module

`'/node_modules/.vite/deps/react-router-dom.js?v=fb66d042'` does not provide an export named 'Switch' (at main.jsx:3:42) Make use of react browser router to handle the routing in the application.



Workspace

> Used 1 reference

Let's update the code to use the correct imports and structure for Vite.js with React Router v6, which uses `Routes` instead of `Switch`. We'll also follow your instructions to include `loader.jsx` and `page.jsx` files in the `routes` folder, and use lazy loading for the components.

First, install the necessary dependencies:

Next, update your `main.jsx` to use the correct imports and structure for Vite.js:

Ensure you have the necessary environment variables in your `.env` file:

Create the `Login.jsx` component in `Login.jsx` :

Create the `Loader.jsx` component in `src/routes/Loader.jsx`:

Create the `Page.jsx` component in `src/routes/Page.jsx`:

This setup should resolve the import issue and correctly structure your Vite.js app

This setup should resolve the import issue and correctly structure your Vite.js app with React Router v6, lazy-loaded components, and the necessary environment variables.

Similar code found with 1 license type - [View matches](#)



leonardrichertz

I do not need the loader and the page component. Please just give me a login view which allows the user to login. This should be accessible under the path "/login"



GitHub Copilot


> Used 1 reference

Understood. Let's create a simple login view that allows the user to log in and is accessible under the path "/login".

1. **Install dependencies:**

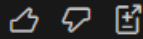
2. **Update**  main.jsx :

3. **Create**  Login.jsx :

4. **Ensure you have the necessary environment variables in your**  .env **file:**


This setup will create a login view, sanitize user input, make an HTTP-POST request to the backend API, and store the authentication token in local storage. The login view will be accessible under the path "/login".

Similar code found with 1 license type - [View matches](#)



leonardrichertz


Make use of Mui Material UI for this page. Use Box, Form, FormLabel and the Button from MUI.

 Login.jsx:34-70




GitHub Copilot

> Used 2 references

Let's update the  Login.jsx component to use Material UI components such as `Box`, `FormControl`, `FormLabel`, and `Button`.

First, install Material UI:


Next, update the  Login.jsx component:

This code uses Material UI components to create a styled login form. The `Box` component is used for layout, `FormControl` and `FormLabel` for form fields, and `Button` for the submit button. The `Typography` component is used for text



leonardrichertz

Create a weather path that renders a page that is only accessible with a valid accessToken. If the user does not have a valid accessToken redirect him to the login page.

 main.jsx:1-19




GitHub Copilot

> Used 2 references

To create a weather path that renders a page only accessible with a valid access token, we need to:


1. Create a `Weather.jsx` component.
2. Implement a private route to check for the access token.
3. Redirect the user to the login page if the token is not present.

First, let's create the `Weather.jsx` component.

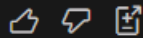
Create  `Weather.jsx` :

Next, let's create a `PrivateRoute` component to handle the authentication check.

Create `src/components/PrivateRoute.jsx` :

Now, update the  `main.jsx` to include the `Weather` route and use the `PrivateRoute` component.

This setup ensures that the `/weather` path is only accessible with a valid access token. If the user does not have a valid token, they will be redirected to the login page.



leonardrichertz

achieve this by implementing a loader function that is executed everytime you access the page. This function should check for the valid accessToken and if none is found redirect to `/login`.

main.jsx:1-19



GitHub Copilot

> Used 2 references

To achieve this, we will implement a loader function that checks for a valid access token and redirects to the login page if none is found. We will use the `useEffect` hook to perform this check when the `Weather` component is accessed.

1. Create `Weather.jsx` component:

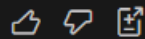
2. Create `Loader.jsx` component:

3. Create `Page.jsx` component:

4. Update `main.jsx` to include the `Weather` route and use the `Loader` component for lazy loading:

5. Ensure you have the necessary environment variables in your `.env` file:

This setup ensures that the `/weather` path is only accessible with a valid access token. If the user does not have a valid token, they will be redirected to the login page. The `Loader` component is used for lazy loading, and the `Page` component is included as per your instructions.



leonardrichertz

This works, but I do want to have a loader function that is executed evrytime the `/weather` route is entered, so I do not have to load page to check whether an `authToken` is present.

`Weather.jsx:1-29`



GitHub Copilot

> Used 2 references

To achieve this, we can use React Router's `loader` function to check for the presence of an `authToken` before rendering the `Weather` component. If

the token is not present, we will redirect the user to the login page.

First, let's create a loader function for the `Weather` route.

Create `src/routes/weatherLoader.js` :

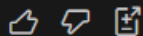
Next, update the `Weather.jsx` component to remove the `useEffect` hook since the loader function will handle the authentication check.

Update `Weather.jsx` :

Finally, update `main.jsx` to use the loader function for the `Weather` route.

Update `main.jsx` :

This setup ensures that the `weatherLoader` function is executed every time the `/weather` route is accessed. If the user does not have a valid `authToken`, they will be redirected to the login page.



leonardrichertz

This does not work, I am able to access the Weather page without having an `authToken` in my `localStorage`.

`weatherLoader.jsx:1-10`



GitHub Copilot

> Used 2 references

To ensure that the loader function works correctly with React Router v6 and Vite.js, we need to make sure that the loader function is properly integrated and executed. Let's update the code to ensure the loader function is correctly used.

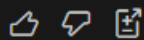
1. Update `weatherLoader.js` to export the loader function correctly:

2. Update `Weather.jsx` to remove the `useEffect` hook since the loader function will handle the authentication check:

3. Update `main.jsx` to use the loader function for the `Weather` route:

4. Ensure you have the necessary environment variables in your `.env` file:


This setup ensures that the `weatherLoader` function is executed every time the `/weather` route is accessed. If the user does not have a valid `authToken`, they will be redirected to the login page. The `weatherLoader` function is lazily imported to follow your instructions.



leonardrichertz

`@workspace` `/fix` have integrated a console log statement. This does not get


logged to the console, indicating that the loader function is indeed not executed on loading the page.


 main.jsx:1-28


Workspace


> Used 2 references

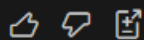
To ensure that the loader function is executed correctly, we need to use the `useLoaderData` hook from `react-router-dom` and ensure that the loader function is properly integrated. Let's update the code to fix this issue.

First, update your  `main.jsx` to correctly use the loader function:

Next, ensure that your  `weatherLoader` function is correctly defined in `src/routes/weatherLoader.js`:

Finally, update your  `Weather.jsx` component to use the `useLoaderData` hook:


This setup ensures that the loader function is executed when the `/weather` route is accessed, and the data returned by the loader function is available in the  `Weather` component using the `useLoaderData` hook.




leonardrichertz

that is not what I want to achieve. I want to run a function everytime the user

that is not what I want to achieve. I want to run a function everytime the user enter the `/weather` route to check whether he is authorized to access the page. I do not want to access the data yet. Make use of React BrowserRouter.

 `main.jsx:1-28`

 **GitHub Copilot**

> Used 2 references

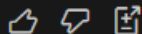
To achieve this, we need to create a function that checks for the presence of an `authToken` in local storage and redirects to the login page if the token is not found. This function should be executed every time the user enters the `/weather` route.

We will use the `useEffect` hook in the `Weather` component to perform this check and redirect if necessary.

1. **Create** `Weather.jsx` **component:**


2. **Update** `main.jsx` **to include the** `Weather` **route:**


This setup ensures that the `useEffect` hook in the `Weather` component is executed every time the `/weather` route is accessed. If the user does not have a valid `authToken`, they will be redirected to the login page.



leonardrichertz

That is not what I am trying to get at. Use `RouterProvider` from `react-router-dom` as a wrapper and provide it with a router using the `createBrowserRouter` function.

 `weatherLoader.jsx:1-11`

 **GitHub Copilot**

> Used 2 references

Understood. Let's update the code to use `RouterProvider` and `createBrowserRouter` from `react-router-dom`. We'll also ensure that the `weatherLoader` function is executed every time the `/weather` route is accessed to check for the presence of an `authToken`.

1. **Update** `weatherLoader.jsx` **to check for the** `authToken` :

2. **Update** `Weather.jsx` **to remove the** `useEffect` **hook since the loader function will handle the authentication check:**

3. **Update** `main.jsx` **to use** `RouterProvider` **and** `createBrowserRouter` :

4. **Ensure you have the necessary environment variables in your** `.env` **file:**

This setup ensures that the `weatherLoader` function is executed every time the `/weather` route is accessed. If the user does not have a valid `authToken`, they will be redirected to the login page. The `RouterProvider` and `createBrowserRouter` are used to manage the routing in the application.

`createBrowserRouter` are used to manage the routing in the application.

Similar code found with 1 license type - [View matches](#)



leonardrichertz

That worked, perfect.

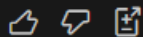
main.jsx:1-31



GitHub Copilot

> Used 2 references

Great! I'm glad it worked for you. If you have any more questions or need further assistance, feel free to ask. Happy coding!





leonardrichertz

Generate a weather page that has a button to get the current location of the user. This location should then be used to make an API get request to our Laravel backend to get the weather data. Expect the data to get returned to the frontend in JSON formatted like this: { "lat":33.44, "lon":-94.04, "timezone":"America/Chicago", "timezone_offset":-18000, "current":{"dt":1684929490, "sunrise":1684926645, "sunset":1684977332, "temp":292.55, "feels_like":292.87, "pressure":1014, "humidity":89, "dew_point":290.69, "uvi":0.16, "clouds":53, "visibility":10000, "wind_speed":3.13, "wind_deg":93, "wind_gust":6.71, "weather":[{"id":803, "main":"Clouds", "description":"broken clouds", "icon":"04d"}]}, "minutely":[{"dt":1684929540, "precipitation":0}, ...], "hourly":[{"dt":1684926000, "temp":292.01, "feels_like":292.33, "pressure":1014,


```
"humidity":91, "dew_point":290.51, "uvi":0, "clouds":54, "visibility":10000,
"wind_speed":2.58, "wind_deg":86, "wind_gust":5.88, "weather":[ { "id":803,
"main":"Clouds", "description":"broken clouds", "icon":"04n" } ], "pop":0.15 }, ... ],
"daily":[ { "dt":1684951200, "sunrise":1684926645, "sunset":1684977332,
"moonrise":1684941060, "moonset":1684905480, "moon_phase":0.16,
"summary":"Expect a day of partly cloudy with rain", "temp":{"day":299.03,
"min":290.69, "max":300.35, "night":291.45, "eve":297.51, "morn":292.55 },
"feels_like":{"day":299.21, "night":291.37, "eve":297.86, "morn":292.87 },
"pressure":1016, "humidity":59, "dew_point":290.48, "wind_speed":3.98,
"wind_deg":76, "wind_gust":8.92, "weather":[ { "id":500, "main":"Rain",
"description":"light rain", "icon":"10d" } ], "clouds":92, "pop":0.47, "rain":0.15,
"uvi":9.23 }, ... ], "alerts": [ { "sender_name": "NWS Philadelphia - Mount Holly (New
Jersey, Delaware, Southeastern Pennsylvania)", "event": "Small Craft Advisory",
"start": 1684952747, "end": 1684988747, "description": "...SMALL CRAFT ADVISORY
REMAINS IN EFFECT FROM 5 PM THIS\nAFTERNOON TO 3 AM EST FRIDAY...\n*
WHAT...North winds 15 to 20 kt with gusts up to 25 kt and seas\n3 to 5 ft
expected.\n* WHERE...Coastal waters from Little Egg Inlet to Great Egg\nInlet NJ
out 20 nm, Coastal waters from Great Egg Inlet to\nCape May NJ out 20 nm and
Coastal waters from Manasquan Inlet\nto Little Egg Inlet NJ out 20 nm.\n*
WHEN...From 5 PM this afternoon to 3 AM EST Friday.\n* IMPACTS...Conditions will
be hazardous to small craft.", "tags": [
```

] Display the current temperature, humidity and the atmospheric pressure in either metric or imperial units.

 Weather.jsx:1-19

 **GitHub Copilot**

> Used 2 references

> Used 2 references

To create a weather page that gets the user's current location and fetches weather data from the Laravel backend, we will:

1. Create a `Weather.jsx` component.
2. Add a button to get the user's current location.
3. Make an API GET request to the Laravel backend using the retrieved location.
4. Display the current temperature, humidity, and atmospheric pressure.

First, let's create the `Weather.jsx` component.

Create `Weather.jsx` :

Update `main.jsx` to include the `Weather` route:

Ensure you have the necessary environment variables in your `.env` file:

This setup ensures that the weather page gets the user's current location, makes an API GET request to the Laravel backend, and displays the current temperature, humidity, and atmospheric pressure. The API request uses a Bearer Token for authentication, which is retrieved from local storage.

Similar code found with 1 license type - [View matches](#)



leonardrichertz

Unexpected Application Error! Cannot read properties of undefined (reading 'temp') TypeError: Cannot read properties of undefined (reading 'temp') at Weather (<http://localhost:5173/src/routes/Weather.jsx?t=1734971425148:81:29>) at renderWithHooks (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:11596:26) at updateFunctionComponent (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:14630:28) at beginWork (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:15972:22) at beginWork\$1 (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:19806:22) at performUnitOfWork (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:19251:20) at workLoopSync (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:19190:13) at renderRootSync (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:19169:15) at recoverFromConcurrentError (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:18786:28) at performConcurrentWorkOnRoot (http://localhost:5173/node_modules/.vite/deps/chunk-HBJ3AJOL.js?v=214755da:18734:30) 🕒 Hey developer 🙌

You can provide a way better UX than this when your app throws errors by providing your own `ErrorBoundary` or `errorElement` prop on your route. Make sure that the app does not crash this way if there is no data provided.

🔗 `main.jsx:6-34`



GitHub Copilot

> Used 2 references

To handle the case where the weather data might not be available and to prevent the app from crashing, we can add a check to ensure that the data exists before trying to access its properties. Additionally, we can provide a fallback UI for when the data is not available.

Let's update the `Weather.jsx` component to include these checks and provide a better user experience.

Update `Weather.jsx` to handle missing data gracefully:

Update `main.jsx` to include the `weatherLoader` function:

Ensure you have the necessary environment variables in your `.env` file:

This setup ensures that the weather page handles missing data gracefully and provides a better user experience. The `weatherLoader` function is lazily imported, and the `Weather` component includes checks to prevent the app from crashing if the data is not available.


Similar code found with 1 license type - [View matches](#)





leonardrichertz

I still get this error when I click on the `getCurrentLocation` button. Keep in mind that I do not yet have my backend implemented and running. It however should not cause this application error.

 Weather.jsx:55-70



GitHub Copilot

> Used 2 references

To prevent the application from crashing when the backend is not available or when the weather data is not yet fetched, we need to add more robust error handling and ensure that we check for the presence of the `current` property in the `weatherData` object before accessing its properties.

Let's update the `Weather.jsx` component to handle these cases more gracefully.

Update `Weather.jsx` :

Update `main.jsx` to include the `weatherLoader` function:

Ensure you have the necessary environment variables in your `.env` **file:**

This setup ensures that the weather page handles missing data gracefully and provides a better user experience. The `weatherLoader` function is lazily imported

provides a better user experience. The `weatherLoader` function is lazily imported, and the `Weather` component includes checks to prevent the app from crashing if the data is not available.

Similar code found with 1 license type - [View matches](#)



leonardrichertz

The post request is made to this <http://localhost:5173/undefined/login> For which I obviously get a 404 not found.

Login.jsx:12-51



GitHub Copilot

> Used 2 references

It looks like the environment variable for the API URL is not being correctly accessed. Let's ensure that the environment variable is correctly set and accessed in the code.

1. Ensure you have the necessary environment variables in your `.env` file:

Create a `.env` file in the root of your project if it doesn't already exist and add the following line:

Replace `http://your-backend-api-url` with the actual URL of your Laravel backend.

2. Update the `Login.jsx` component to correctly access the environment variable:

3. Ensure you have the necessary environment variables in your `.env` file:

This setup ensures that the POST request is made to the correct URL specified in the environment variable.



leonardrichertz

After successfully logging in, I want to navigate to `/weather`. Also make sure the login component is the default route `/`

`Login.jsx:18-50`



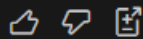
GitHub Copilot

> Used 2 references

To navigate to the `/weather` route after a successful login and make the login component the default route, we need to use the `useNavigate` hook from

This setup ensures that after a successful login, the user is redirected to the `/weather` route. The login component is set as the default route (`/`).

Similar code found with 1 license type - [View matches](#)



leonardrichertz

@workspace remove the login path and make sure that the login page visible on the `/` route is centered.

main.jsx:1-34

Workspace

> Used 6 references

✓ Collecting workspace information

To remove the `/login` path and ensure that the login page visible on the `/` route is centered, we need to update the routing configuration and add some styles to center the login page.

First, update your main.jsx):

Next, update your Login.jsx) to ensure the login page is centered:

This setup removes the `/login` path and ensures that the login page visible on the `/` route is centered.

Similar code found with 2 license types - [View matches](#)



leonardrichertz

@workspace /fix login component is on the left side of the screen. Please fix this. I want to have it centered both horizontally and vertically

index.css:1-41



Workspace

> Used 2 references

To center the login component both horizontally and vertically, we need to update the CSS styles. We can achieve this by modifying the `body` styles in `# index.css` and ensuring the login component's container takes full height and uses flexbox for centering.

First, update your `# index.css` :

Next, update your `Login.jsx` component to ensure it uses the full height and centers its content:

This setup ensures that the login component is centered both horizontally and

vertically on the screen. The `body` styles in `# index.css` use flexbox to center the content, and the `Login` component's container is styled to take full width and center its content.

Similar code found with 2 license types - [View matches](#)



leonardrichertz

Add a loading button to the submit button while the data is being sent to the backend. Also disable it while submitting the form

Login.jsx:61-95



GitHub Copilot

> Used 2 references

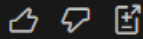
To add a loading indicator to the submit button and disable it while the form is being submitted, we can use the `LoadingButton` component from Material-UI. We will also manage the loading state using a state variable.

First, install the `@mui/lab` package which contains the `LoadingButton` component:

Next, update the `Login.jsx` component to include the `LoadingButton` and manage the loading state:

This setup adds a `LoadingButton` from Material-UI that shows a loading indicator and disables the button while the form is being submitted. The `loading` state is managed using a state variable.

Similar code found with 2 license types - [View matches](#)



leonardrichertz

Include a Mui Button group that allows the user to switch units. Make the request to weather to accomodate this backend end : <?php

```
namespace App\Http\Controllers;
```

```
use Illuminate\Http\Request; use Illuminate\Support\Facades\Http;
```

```
class WeatherController extends Controller { public function getWeather(Request $request) { $request->validate([ 'lat' => 'required|numeric', 'lon' => 'required|numeric', 'units' => 'nullable|string|in:standard,metric,imperial', ]);
```

```
}
```

Weather.jsx:1-39



GitHub Copilot

> Used 2 references

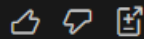
To include a MUI Button group that allows the user to switch units and make the request to the backend to accommodate the units, we need to update the

Weather.jsx component.

1. Update `Weather.jsx` to include the Button group and handle unit switching:

2. Ensure you have the necessary environment variables in your `.env` file:

This setup includes a MUI Button group that allows the user to switch units between "standard", "metric", and "imperial". The `getWeatherData` function is updated to include the `units` parameter in the request to the backend. The `handleUnitChange` function updates the units state and fetches the weather data again if it is already available.



leonardrichertz

I do not want to have a standard button. The user should only be able to choose between imperial and metric. Initialize to metric. And also only make the request if a location is present.

`Weather.jsx:43-84`



GitHub Copilot

> Used 2 references

Understood. Let's update the `Weather.jsx` component to initialize the units

to "metric" and provide buttons to switch between "metric" and "imperial". We will also ensure that the request is only made if a location is present.

Update `Weather.jsx` **to remove the "standard" button and initialize to "metric":**

This setup initializes the units to "metric" and provides buttons to switch between "metric" and "imperial". The request to fetch weather data is only made if a location is present. The "standard" button has been removed as per your instructions.



leonardrichertz

Also display the correct units in the view

`Weather.jsx:54-93`



GitHub Copilot

> Used 2 references

To display the correct units in the view and ensure the code follows your instructions, we will update the `Weather.jsx` component to include the units in the displayed data. We will also ensure that the Bearer Token authentication is used for all API requests.


Update `Weather.jsx` **to display the correct units and use Bearer Token authentication:**

This setup ensures that the correct units are displayed in the view based on the selected unit system (metric or imperial). The `getTemperatureUnit` function returns the appropriate unit symbol for the temperature. The Bearer Token authentication is used for all API requests to the backend.



leonardrichertz

Only include celcius and fahrenheit as units, since we always make the request with eithe rmetric or imperial.

 Weather.jsx:54-93



GitHub Copilot

> Used 2 references

Understood. Let's update the `Weather.jsx` component to only include Celsius and Fahrenheit as units, since we always make the request with either "metric" or "imperial".

Update `Weather.jsx` **to include only Celsius and Fahrenheit units:**

This setup ensures that the user can only choose between Celsius and Fahrenheit units, and the correct units are displayed in the view. The request to fetch weather data is made with either "metric" or "imperial" units.



leonardrichertz

Make the weather view look more structured. Right now the buttons all stick together. Separate them a bit and also display an icon for the weather condition based on the current.weather.id: { "lat": 49.9768, "lon": 6.5209, "timezone": "Europe/Berlin", "timezone_offset": 3600, "current": { "dt": 1735463469, "sunrise": 1735457535, "sunset": 1735486779, "temp": -1.44, "feels_like": -4.8, "pressure": 1031, "humidity": 95, "dew_point": -2.05, "uvi": 0.28, "clouds": 100, "visibility": 400, "wind_speed": 2.57, "wind_deg": 240, "weather": [{ "id": 741, "main": "Fog", "description": "fog", "icon": "50d" }] }, "minutely": [{ "dt": 1735463520, "precipitation": 0 }, { "dt": 1735463580, "precipitation": 0 }, { "dt": 1735463640, "precipitation": 0 }, { "dt": 1735463700, "precipitation": 0 }, { "dt": 1735463760, "precipitation": 0 }, { "dt": 1735463820, "precipitation": 0 }, { "dt": 1735463880, "precipitation": 0 }, { "dt": 1735463940, "precipitation": 0 }, { "dt": 1735464000, "precipitation": 0 }, { "dt": 1735464060, "precipitation": 0 }, { "dt": 1735464120, "precipitation": 0 }, { "dt": 1735464180, "precipitation": 0 }, { "dt": 1735464240, "precipitation": 0 }, { "dt": 1735464300, "precipitation": 0 }, { "dt": 1735464360, "precipitation": 0 }, { "dt": 1735464420, "precipitation": 0 }, { "dt": 1735464480, "precipitation": 0 }, { "dt": 1735464540, "precipitation": 0 }, { "dt": 1735464600, "precipitation": 0 }, { "dt": 1735464660, "precipitation": 0 }, { "dt": 1735464720, "precipitation": 0 }, { "dt": 1735464780, "precipitation": 0 }, { "dt": 1735464840, "precipitation": 0 }, { "dt": 1735464900, "precipitation": 0 }, { "dt": 1735464960, "precipitation": 0 }, { "dt": 1735465020, "precipitation": 0 }, { "dt": 1735465080, "precipitation": 0 }, { "dt": 1735465140, "precipitation": 0 }, { "dt": 1735465200, "precipitation": 0 }, { "dt": 1735465260, "precipitation": 0 }, { "dt": 1735465320, "precipitation": 0 }, { "dt": 1735465380, "precipitation": 0 }, { "dt": 1735465440, "precipitation": 0 }, { "dt": 1735465500, "precipitation": 0 }, { "dt": 1735465560, "precipitation": 0 }, { "dt": 1735465620, "precipitation": 0 }, { "dt": 1735465680, "precipitation": 0 }, { "dt": 1735465740, "precipitation": 0 }, { "dt": 1735465800, "precipitation": 0 }, { "dt": 1735465860, "precipitation": 0 }, { "dt": 1735465920, "precipitation": 0 }, { "dt": 1735465980, "precipitation": 0 }, { "dt": 1735466040, "precipitation": 0 }, { "dt": 1735466100, "precipitation": 0 }, { "dt": 1735466160,

```
precipitation": 0 }, { "dt": 1735465500, "precipitation": 0 }, { "dt": 1735465560,
"precipitation": 0 }, { "dt": 1735465620, "precipitation": 0 }, { "dt": 1735465680,
"precipitation": 0 }, { "dt": 1735465740, "precipitation": 0 }, { "dt": 1735465800,
"precipitation": 0 }, { "dt": 1735465860, "precipitation": 0 }, { "dt": 1735465920,
"precipitation": 0 }, { "dt": 1735465980, "precipitation": 0 }, { "dt": 1735466040,
"precipitation": 0 }, { "dt": 1735466100, "precipitation": 0 }, { "dt": 1735466160,
"precipitation": 0 }, { "dt": 1735466220, "precipitation": 0 }, { "dt": 1735466280,
"precipitation": 0 }, { "dt": 1735466340, "precipitation": 0 }, { "dt": 1735466400,
"precipitation": 0 }, { "dt": 1735466460, "precipitation": 0 }, { "dt": 1735466520,
"precipitation": 0 }, { "dt": 1735466580, "precipitation": 0 }, { "dt": 1735466640,
"precipitation": 0 }, { "dt": 1735466700, "precipitation": 0 }, { "dt": 1735466760,
"precipitation": 0 }, { "dt": 1735466820, "precipitation": 0 }, { "dt": 1735466880,
"precipitation": 0 }, { "dt": 1735466940, "precipitation": 0 }, { "dt": 1735467000,
"precipitation": 0 }, { "dt": 1735467060, "precipitation": 0 } ], "hourly": [ { "dt":
1735462800, "temp": -1.44, "feels_like": -1.44, "pressure": 1031, "humidity": 95,
"dew_point": -2.05, "uvi": 0.28, "clouds": 100, "visibility": 10000, "wind_speed": 1.29,
"wind_deg": 216, "wind_gust": 1.55, "weather": [ { "id": 804, "main": "Clouds",
"description": "overcast clouds", "icon": "04d" } ], "pop": 0 }, { "dt": 1735466400,
"temp": -0.92, "feels_like": -2.96, "pressure": 1031, "humidity": 90, "dew_point":
-2.18, "uvi": 0.55, "clouds": 83, "visibility": 10000, "wind_speed": 1.6, "wind_deg":
220, "wind_gust": 2.32, "weather": [ { "id": 803, "main": "Clouds", "description":
"broken clouds", "icon": "04d" } ], "pop": 0 }, { "dt": 1735470000, "temp": 0,
"feels_like": -2.85, "pressure": 1031, "humidity": 87, "dew_point": -1.68, "uvi": 0.75,
"clouds": 67, "visibility": 10000, "wind_speed": 2.34, "wind_deg": 227, "wind_gust":
3.28, "weather": [ { "id": 803, "main": "Clouds", "description": "broken clouds",
"icon": "04d" } ], "pop": 0 }, { "dt": 1735473600, "temp": 1.09, "feels_like": -2.02,
"pressure": 1031, "humidity": 86, "dew_point": -0.87, "uvi": 0.76, "clouds": 52,
"visibility": 10000, "wind_speed": 2.81, "wind_deg": 237, "wind_gust": 3.76,
"weather": [ { "id": 803, "main": "Clouds", "description": "broken clouds", "icon":
"04d" } ], "pop": 0 }, { "dt": 1735477200, "temp": 2.07, "feels_like": -0.86, "pressure":
1031, "humidity": 84, "dew_point": -0.3, "uvi": 0.57, "clouds": 83, "visibility": 10000,
"wind_speed": 2.82, "wind_deg": 246, "wind_gust": 3.52, "weather": [ { "id": 803
```



```
"wind_speed": 2.82, "wind_deg": 246, "wind_gust": 3.53, "weather": [ { "id": 803,
"main": "Clouds", "description": "broken clouds", "icon": "04d" } ], "pop": 0 }, { "dt":
1735480800, "temp": 2.77, "feels_like": 0.22, "pressure": 1030, "humidity": 84,
"dew_point": 0.27, "uvi": 0.3, "clouds": 86, "visibility": 10000, "wind_speed": 2.54,
"wind_deg": 242, "wind_gust": 3.51, "weather": [ { "id": 804, "main": "Clouds",
"description": "overcast clouds", "icon": "04d" } ], "pop": 0 }, { "dt": 1735484400,
"temp": 2.46, "feels_like": 0.18, "pressure": 1031, "humidity": 84, "dew_point": 0.09,
"uvi": 0, "clouds": 90, "visibility": 10000, "wind_speed": 2.21, "wind_deg": 248,
"wind_gust": 2.82, "weather": [ { "id": 804, "main": "Clouds", "description": "overcast
clouds", "icon": "04d" } ], "pop": 0 }, { "dt": 1735488000, "temp": 1.75, "feels_like":
-0.45, "pressure": 1031, "humidity": 87, "dew_point": -0.19, "uvi": 0, "clouds": 93,
"visibility": 10000, "wind_speed": 2.03, "wind_deg": 217, "wind_gust": 2.06,
"weather": [ { "id": 804, "main": "Clouds", "description": "overcast clouds", "icon":
"04n" } ], "pop": 0 }, { "dt": 1735491600, "temp": 0.48, "feels_like": -1.67, "pressure":
1031, "humidity": 91, "dew_point": -0.83, "uvi": 0, "clouds": 87, "visibility": 10000,
"wind_speed": 1.82, "wind_deg": 214, "wind_gust": 1.77, "weather": [ { "id": 804,
"main": "Clouds", "description": "overcast clouds", "icon": "04n" } ], "pop": 0 }, { "dt":
1735495200, "temp": -0.45, "feels_like": -0.45, "pressure": 1032, "humidity": 92,
"dew_point": -1.55, "uvi": 0, "clouds": 81, "visibility": 10000, "wind_speed": 1.3,
"wind_deg": 234, "wind_gust": 1.3, "weather": [ { "id": 803, "main": "Clouds",
"description": "broken clouds", "icon": "04n" } ], "pop": 0 }, { "dt": 1735498800,
"temp": -0.65, "feels_like": -0.65, "pressure": 1032, "humidity": 93, "dew_point":
-1.68, "uvi": 0, "clouds": 19, "visibility": 10000, "wind_speed": 1.2, "wind_deg": 216,
"wind_gust": 1.15, "weather": [ { "id": 801, "main": "Clouds", "description": "few
clouds", "icon": "02n" } ], "pop": 0 }, { "dt": 1735502400, "temp": -0.88, "feels_like":
-0.88, "pressure": 1032, "humidity": 94, "dew_point": -1.78, "uvi": 0, "clouds": 27,
"visibility": 10000, "wind_speed": 1.23, "wind_deg": 219, "wind_gust": 1.19,
"weather": [ { "id": 802, "main": "Clouds", "description": "scattered clouds", "icon":
"03n" } ], "pop": 0 }, { "dt": 1735506000, "temp": -0.97, "feels_like": -0.97, "pressure":
1033, "humidity": 94, "dew_point": -1.87, "uvi": 0, "clouds": 21, "visibility": 10000,
"wind_speed": 1.03, "wind_deg": 221, "wind_gust": 0.97, "weather": [ { "id": 801,
```

```
1033, "humidity": 94, "dew_point": -1.87, "uvi": 0, "clouds": 21, "visibility": 10000,
"wind_speed": 1.03, "wind_deg": 221, "wind_gust": 0.97, "weather": [ { "id": 801,
"main": "Clouds", "description": "few clouds", "icon": "02n" } ], "pop": 0 }, { "dt":
1735509600, "temp": -1.06, "feels_like": -1.06, "pressure": 1033, "humidity": 94,
"dew_point": -1.93, "uvi": 0, "clouds": 18, "visibility": 10000, "wind_speed": 1.28,
"wind_deg": 213, "wind_gust": 1.16, "weather": [ { "id": 801, "main": "Clouds",
"description": "few clouds", "icon": "02n" } ], "pop": 0 }, { "dt": 1735513200, "temp":
-1.19, "feels_like": -1.19, "pressure": 1033, "humidity": 94, "dew_point": -2.04, "uvi":
0, "clouds": 21, "visibility": 10000, "wind_speed": 0.94, "wind_deg": 229,
"wind_gust": 0.86, "weather": [ { "id": 801, "main": "Clouds", "description": "few
clouds", "icon": "02n" } ], "pop": 0 }, { "dt": 1735516800, "temp": -1.25, "feels_like":
-1.25, "pressure": 1033, "humidity": 94, "dew_point": -2.11, "uvi": 0, "clouds": 20,
"visibility": 10000, "wind_speed": 0.97, "wind_deg": 221, "wind_gust": 0.95,
"weather": [ { "id": 801, "main": "Clouds", "description": "few clouds", "icon": "02n" }
], "pop": 0 }, { "dt": 1735520400, "temp": -1.31, "feels_like": -1.31, "pressure": 1033,
"humidity": 94, "dew_point": -2.16, "uvi": 0, "clouds": 2, "visibility": 10000,
"wind_speed": 0.77, "wind_deg": 226, "wind_gust": 0.76, "weather": [ { "id": 800,
"main": "Clear", "description": "clear sky", "icon": "01n" } ], "pop": 0 }, { "dt":
1735524000, "temp": -1.32, "feels_like": -1.32, "pressure": 1033, "humidity": 94,
"dew_point": -2.18, "uvi": 0, "clouds": 1, "visibility": 10000, "wind_speed": 1.14,
"wind_deg": 211, "wind_gust": 1.13, "weather": [ { "id": 800, "main": "Clear",
"description": "clear sky", "icon": "01n" } ], "pop": 0 }, { "dt": 1735527600, "temp":
-1.37, "feels_like": -1.37, "pressure": 1032, "humidity": 94, "dew_point": -2.21, "uvi":
0, "clouds": 1, "visibility": 10000, "wind_speed": 1.04, "wind_deg": 213, "wind_gust":
0.96, "weather": [ { "id": 800, "main": "Clear", "description": "clear sky", "icon": "01n"
} ], "pop": 0 }, { "dt": 1735531200, "temp": -1.36, "feels_like": -1.36, "pressure": 1032,
"humidity": 94, "dew_point": -2.21, "uvi": 0, "clouds": 1, "visibility": 10000,
"wind_speed": 0.8, "wind_deg": 221, "wind_gust": 0.82, "weather": [ { "id": 800,
"main": "Clear", "description": "clear sky", "icon": "01n" } ], "pop": 0 }, { "dt":
1735534800, "temp": -1.33, "feels_like": -1.33, "pressure": 1032, "humidity": 94,
"dew_point": -2.2, "uvi": 0, "clouds": 0, "visibility": 10000, "wind_speed": 0.77
```

09d 311 Drizzle drizzle rain 09d 312 Drizzle heavy intensity drizzle rain 09d 313 Drizzle shower rain and drizzle 09d 314 Drizzle heavy shower rain and drizzle 09d 321 Drizzle shower drizzle 09d Group 5xx: Rain 500 Rain light rain 10d 501 Rain moderate rain 10d 502 Rain heavy intensity rain 10d 503 Rain very heavy rain 10d 504 Rain extreme rain 10d 511 Rain freezing rain 13d 520 Rain light intensity shower rain 09d 521 Rain shower rain 09d 522 Rain heavy intensity shower rain 09d 531 Rain ragged shower rain 09d Group 6xx: Snow 600 Snow light snow 13d 601 Snow snow 13d 602 Snow heavy snow 13d 611 Snow sleet 13d 612 Snow light shower sleet 13d 613 Snow shower sleet 13d 615 Snow light rain and snow 13d 616 Snow rain and snow 13d 620 Snow light shower snow 13d 621 Snow shower snow 13d 622 Snow heavy shower snow 13d Group 7xx: Atmosphere 701 Mist mist 50d 711 Smoke smoke 50d 721 Haze haze 50d 731 Dust sand/dust whirls 50d 741 Fog fog 50d 751 Sand sand 50d 761 Dust dust 50d 762 Ash volcanic ash 50d 771 Squall squalls 50d 781 Tornado tornado 50d Group 800: Clear 800 Clear clear sky 01d 01n Group 80x: Clouds 801 Clouds few clouds: 11-25% 02d 02n 802 Clouds scattered clouds: 25-50% 03d 03n 803 Clouds broken clouds: 51-84% 04d 04n 804 Clouds overcast clouds: 85-100% 04d 04n Use Mui icons to display it visually



leonardrichter

Make the weather view look more structured. Right now the buttons all stick together. Separate them a bit and also display an icon for the weather condition

based on the current.weather.id: { "lat": 49.9768, "lon": 6.5209, "timezone":

"Europe/Berlin", "timezone_offset": 3600, "current": { "dt": 1735463469, "sunrise":

1735457535, "sunset": 1735486779, "temp": -1.44, "feels_like": -4.8, "pressure":

1031, "humidity": 95, "dew_point": -2.05, "uvi": 0.28, "clouds": 100, "visibility": 400,

"wind_speed": 2.57, "wind_deg": 240, "weather": [{ "id": 741, "main": "Fog",

"description": "fog", "icon": "50d" }] }, "minutely": [{ "dt": 1735463520,

"precipitation": 0 }, { "dt": 1735463580, "precipitation": 0 }, { "dt": 1735463640,

"precipitation": 0 }, { "dt": 1735463700, "precipitation": 0 }, { "dt": 1735463760,

"precipitation": 0 }, { "dt": 1735463820, "precipitation": 0 }, { "dt": 1735463880,

"precipitation": 0 }, { "dt": 1735463940, "precipitation": 0 }, { "dt": 1735464000,

"precipitation": 0 }, { "dt": 1735464060, "precipitation": 0 }, { "dt": 1735464120,

"precipitation": 0 }, { "dt": 1735464180, "precipitation": 0 }, { "dt": 1735464240,

"precipitation": 0 }, { "dt": 1735464300, "precipitation": 0 }, { "dt": 1735464360,

"precipitation": 0 }, { "dt": 1735464420, "precipitation": 0 }, { "dt": 1735464480,

"precipitation": 0 }, { "dt": 1735464540, "precipitation": 0 }, { "dt": 1735464600,

"precipitation": 0 }, { "dt": 1735464660, "precipitation": 0 }, { "dt": 1735464720,

"precipitation": 0 }, { "dt": 1735464780, "precipitation": 0 }, { "dt": 1735464840,

"precipitation": 0 }, { "dt": 1735464900, "precipitation": 0 }, { "dt": 1735464960,

"precipitation": 0 }, { "dt": 1735465020, "precipitation": 0 }, { "dt": 1735465080,

"precipitation": 0 }, { "dt": 1735465140, "precipitation": 0 }, { "dt": 1735465200,

"precipitation": 0 }, { "dt": 1735465260, "precipitation": 0 }, { "dt": 1735465320,

"precipitation": 0 }, { "dt": 1735465380, "precipitation": 0 }, { "dt": 1735465440,

"precipitation": 0 }, { "dt": 1735465500, "precipitation": 0 }, { "dt": 1735465560,

"precipitation": 0 }, { "dt": 1735465620, "precipitation": 0 }, { "dt": 1735465680,

"precipitation": 0 }, { "dt": 1735465740, "precipitation": 0 }, { "dt": 1735465800,

"precipitation": 0 }, { "dt": 1735465860, "precipitation": 0 }, { "dt": 1735465920,

"precipitation": 0 }, { "dt": 1735465980, "precipitation": 0 }, { "dt": 1735466040,


"precipitation": 0 }, { "dt": 1735466100, "precipitation": 0 }, { "dt": 1735466160,

Ask Copilot



leonardrichertz

Make the weather view look more structured. Right now the buttons all stick together. Separate them a bit and also display an icon for the weather condition based on the `current.weather.id`: These are the codes: 200-232: Thunderstorm, 300-321: Drizzle: 500-531: Rain: 600-622: snow: 800 clear: 801-804: clouds, 701-781: something else like tornado or volcanic ash. Use Mui icons to display it visually

 Weather.jsx:1-39




GitHub Copilot

> Used 2 references

To make the weather view more structured and display an icon for the weather condition based on the `current.weather.id`, we will use MUI icons and add some spacing between the buttons. We will also ensure the weather data is displayed in a more organized manner.

First, install the necessary MUI icons package:

Next, update the  Weather.jsx component:

This setup includes:


- MUI icons for different weather conditions.
- Spacing between the buttons using the `sx` prop.
- A more structured layout for displaying the weather data, including an icon

- A more structured layout for displaying the weather data, including an icon and description for the current weather condition.



leonardrichertz


Also adjust the wind speed based on the units


 Weather.jsx:120-142

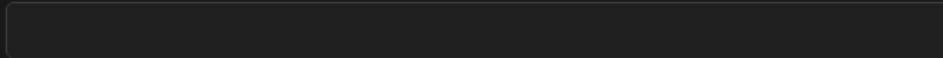


GitHub Copilot


> Used 2 references

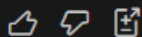
To adjust the wind speed based on the units and ensure the code follows your instructions, we will update the  Weather.jsx component to include the correct wind speed unit and adjust the wind speed value accordingly.

Update  Weather.jsx **to adjust the wind speed based on the units:**



This setup includes:


- Adjusting the wind speed based on the selected units (metric or imperial).
- Adding spacing between the buttons using the `sx` prop.
- Displaying an icon for the weather condition based on the  `current.weather.id`.
- Using MUI icons to visually represent different weather conditions.





leonardrichertz


@workspace create a separate component that display all the weather warnings in a list. And implement them in then Weather component. This is how the alerts look like: [{ "sender_name": "Deutscher Wetterdienst", "event": "frost", "start": 1735401600, "end": 1735470000, "description": "There is a risk of frost (level 1 of 2).\nMinimum temperature: -2 - -5 °C; local minimum: in valleys and dips > -7 °C", "tags": ["Extreme low temperature"] }, { "sender_name": "Deutscher Wetterdienst", "event": "icy surfaces", "start": 1735462800, "end": 1735470000, "description": "There is a slight risk of icy surfaces (level 1 of 4).", "tags": ["Snow/Ice"] }, { "sender_name": "Deutscher Wetterdienst", "event": "frost", "start": 1735470000, "end": 1735488000, "description": "There is a risk of frost (level 1 of 2).\nMinimum temperature: 0 - -3 °C", "tags": ["Extreme low temperature"] }, { "sender_name": "Deutscher Wetterdienst", "event": "fog", "start": 1735461660, "end": 1735470000, "description": "There is a risk of fog (level 1 of 1).\nVisibility: < 150 m", "tags": ["Fog"] }]

 Weather.jsx:111-142

Workspace

> Used 4 references

✓ Collecting workspace information

Let's create a separate component to display the weather warnings in a list and then integrate it into the  Weather component.

First, create a new file `src/routes/WeatherWarnings.jsx`:

Next, update your `Weather.jsx` component to use the `WeatherWarnings` component:

This setup creates a separate `WeatherWarnings` component to display weather warnings in a list and integrates it into the `Weather` component. The `WeatherWarnings` component takes an `alerts` prop and displays each alert with its details.



leonardrichertz

Please change the buttons, they do not line up and there is no space between them. Make use of flexbox please

`Weather.jsx:46-81`



GitHub Copilot

> Used 2 references

To ensure the buttons line up properly and have space between them, we can use Flexbox. We will update the `Weather.jsx` component to use Flexbox for the button layout.

Update `Weather.jsx` **to use Flexbox for button layout:**

In this update:

- We use a `Box` component with `display: 'flex'` and `justifyContent: 'space-between'` to ensure the buttons are properly aligned and spaced.
- The `ButtonGroup` component is used to group the unit change buttons together.



leonardrichertz

[@workspace](#) [/new](#) Build me a new component that display the forecast data using chart.js. Make use of this for the temperature and humidity forecast and use the daily forecasts that look like this: [{ "dt": 1735470000, "sunrise": 1735457535, "sunset": 1735486779, "moonrise": 1735454700, "moonset": 1735479840, "moon_phase": 0.95, "summary": "There will be partly cloudy today", "temp": { "day": -0.72, "min": -1.46, "max": 2.46, "night": -1.06, "eve": 0.48, "morn": -1.16 }, "feels_like": { "day": -3.7, "night": -1.06, "eve": -1.67, "morn": -1.16 }, "pressure": 1031, "humidity": 91, "dew_point": -1.85, "wind_speed": 2.82, "wind_deg": 246, "wind_gust": 3.76, "weather": [{ "id": 803, "main": "Clouds", "description": "broken clouds", "icon": "04d" }], "clouds": 83, "pop": 0, "uvi": 0.76 }, { "dt": 1735556400, "sunrise": 1735543939, "sunset": 1735573232, "moonrise": 1735544760, "moonset": 1735569600, "moon_phase": 0, "summary": "Expect a day of partly cloudy with clear spells", "temp": { "day": 2.59, "min": -1.37, "max": 3.72, "night": -0.87, "eve": -0.04, "morn": -1.33 }, "feels_like": { "day": 1.3, "night": -0.87, "eve": -0.04, "morn": -1.33 }, "pressure": 1032, "humidity": 81, "dew_point": -0.34, "wind_speed": 1.55, "wind_deg": 212, "wind_gust": 2.22, "weather": [{ "id": 800, "main": "Clear", "description": "clear sky", "icon": "01d" }], "clouds": 8, "pop": 0, "uvi": 0.67 }, { "dt": 1735642800, "sunrise": 1735630339, "sunset": 1735659688, "moonrise": 1735634160, "moonset": 1735660200, "moon_phase": 0.02,

Weather.jsx:109-142

Workspace

> Used 2 references

Sure, here's a proposed directory structure for a Vite.js frontend app:

```


vite-frontend-app
├── src
│   ├── components
│   │   ├── Weather.jsx
│   │   └── ForecastChart.jsx
│   ├── App.jsx
│   └── main.jsx
├── styles
│   └── # main.css
├── public
│   └── index.html
├── .env
├── package.json
├── pnpm-lock.yaml
└── README.md
```


Create Workspace...



leonardrichertz

Only show me the forecast component and the weather component

 ForecastChart.jsx:34-67

 **GitHub Copilot**

> Used 2 references

Sure, here are the `ForecastChart` and `Weather` components:

`ForecastChart.jsx` :

`Weather.jsx`:

In this setup:

- The `ForecastChart` component uses `chart.js` to display temperature and humidity forecasts.
- The `Weather` component fetches weather data and displays current weather, warnings, and forecast charts.
- Components are lazily loaded using `React.lazy` and `Suspense` for better performance.


Similar code found with 2 license types - [View matches](#)



leonardrichertz


C:\Users\Aster\Desktop\React\leahertz\copilot-generated\code\copilot-generated


```
C:\Users\Schola\Desktop\Bachelorarbeit\copilot_generated_code\copilot-react-frontend\src\routes\Weather.jsx: Identifier 'ForecastChart' has already been declared. (14:6) 17 | const [weatherData, setWeatherData] = useState(null); C:/Users/Schola/Desktop/Bachelorarbeit/copilot_generated_code/copilot-react-frontend/src/routes/Weather.jsx:14:6 12 | 13 | const WeatherWarnings = lazy(() => import("./WeatherWarnings")); 14 | const ForecastChart = lazy(() => import("./ForecastChart")); | ^ 15 |
```

 Weather.jsx:1-36


GitHub Copilot

> Used 2 references

It looks like there is a duplicate import for  ForecastChart . Let's remove the redundant import and ensure the code follows the provided instructions.

Update  Weather.jsx :

In this update:

- Removed the redundant import of  ForecastChart .
- Ensured the code follows the provided instructions for using lazy loading and Bearer Token authentication.



leonardrichertz

[plugin:vite:import-analysis] Failed to resolve import "react-chartjs-2" from "src/routes/ForecastChart.jsx". Does the file exist?

C:/Users/Schola/Desktop/Bachelorarbeit/copilot_generated_code/copilot-react-

ForecastChart.jsx:30

GitHub Copilot

> Used 2 references

It looks like the `react-chartjs-2` package is not installed. You can install it using `pnpm`. Additionally, ensure that `chart.js` is also installed as it is a peer dependency of `react-chartjs-2`.

First, install the required packages:

Next, ensure your `ForecastChart.jsx` component is correctly set up:

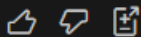
`ForecastChart.jsx` :

Ensure your `Weather.jsx` component is correctly set up to use lazy loading:

`Weather.jsx`:

This setup ensures that the `react-chartjs-2` and `chart.js` packages are installed and used correctly in the `ForecastChart` component. The `Weather` component uses lazy loading for `WeatherWarnings` and `ForecastChart` components.


Similar code found with 2 license types - [View matches](#)





leonardrichertz

Make 2 separate forecast charts so the scales are correct. Just build a forecast component that receives the data as props and define the data in the weather component

 ForecastChart.jsx:27-69



GitHub Copilot

> Used 2 references

Sure, let's create two separate forecast charts for temperature and humidity to ensure the scales are correct. We'll build a `ForecastChart` component that receives the data as props and define the data in the `Weather` component.

Update `ForecastChart.jsx` :

Update `Weather.jsx`:


In this setup:

- The `ForecastChart` component is designed to receive `labels`, `data`, `label`, `borderColor`, and `backgroundColor` as props.
- The `Weather` component defines the data for temperature and humidity forecasts and passes them to the `ForecastChart` component.
- Two separate `ForecastChart` components are used to display temperature and humidity forecasts with correct scales.



leonardrichertz

Store the location in the state and only render the save button when a location is present


 Weather.jsx:28-69



GitHub Copilot

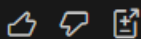
> Used 2 references

To store the location in the state and only render the save button when a location is present, we need to update the `Weather` component. We'll add a state variable for the location and conditionally render the save button based on whether the location is present.

Update  Weather.jsx :

In this update:


- Added a state variable `location` to store the location.
- Conditionally rendered the "Save Location" button based on whether the location is present.
- Updated the `saveLocation` function to be asynchronous and handle the API request properly.





leonardrichertz

```
const saveLocation = async (location) => { try { const authToken =
localStorage.getItem("authToken"); await
axios.post(`${import.meta.env.VITE_API_URL}/location`, location, { headers: {
Authorization: `Bearer ${authToken}`, }, }); } catch (err) { setError("Failed to save
location"); } }; Show a toast success message if the saving was successful and also
if there was an error, show a toast error message
```

 Weather.jsx:57-68



GitHub Copilot

> Used 2 references

To show toast messages for success and error scenarios, you can use a library like `react-toastify`. First, install `react-toastify` using `pnpm`:

Next, update your `Weather.jsx` component to use `react-toastify` for displaying toast messages.

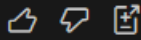
Update `Weather.jsx` :

In this update:

- Installed `react-toastify` and imported it into the `Weather.jsx` component.
- Used `toast.success` to show a success message when the location is saved successfully.
- Used `toast.error` to show an error message when there is a failure in saving

the location.

- Added `<ToastContainer />` to the component to render the toast notifications.



leonardrichertz

Create a component that lists all the saved locations for the user and allows the user to select that location, which set the location to that location and then a way to delete this location with its id. These are the backend routes:

```
Route::get('/locations', [LocationController::class, 'getUserLocations']);
```

```
Route::delete('/location/{id}', [LocationController::class, 'deleteLocation']);
```

Weather.jsx:122-163




GitHub Copilot

> Used 2 references

To create a component that lists all the saved locations for the user, allows the user to select a location, and delete a location by its ID, we need to follow these steps:

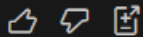
1. Create a new component `SavedLocations.jsx`.
2. Fetch the saved locations from the backend.
3. Allow the user to select a location, which sets the location.
4. Allow the user to delete a location by its ID.

Create `SavedLocations.jsx`:

Update  Weather.jsx **to use** **SavedLocations**:


In this setup:

- The **SavedLocations** component fetches and displays the user's saved locations, allows selecting a location to set it, and deleting a location by its ID.
- The **Weather** component uses the **SavedLocations** component and handles selecting and deleting locations.
- Toast notifications are used to show success and error messages.



leonardrichertz


[{ "id": 1, "user_id": 1, "name": "", "latitude": 49.9768236, "longitude": 6.5209355, "created_at": "2024-12-29T11:10:21.000000Z", "updated_at": "2024-12-29T11:10:21.000000Z" }]These are the locations. Please display the data correctly. Do not include the name


 SavedLocations.jsx:23-62



GitHub Copilot

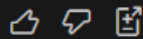
> Used 2 references

To display the saved locations correctly without including the name, we need to update the  SavedLocations.jsx component to use the **latitude** and **longitude** fields for display.

Update  SavedLocations.jsx :

In this update:

- The `SavedLocations` component fetches and displays the user's saved locations using the `latitude` and `longitude` fields.
- The `name` field is excluded from the display.
- The component allows selecting a location to set it and deleting a location by its ID.
- Toast notifications are used to show success and error messages.



leonardrichertz

pass the setLocation function to the savedlocazions component to then be able to set the location. import React, { useState, useEffect } from "react"; import axios from "axios"; import { Box, List, ListItem, ListItemText, IconButton } from "@mui/material"; import DeleteIcon from "@mui/icons-material/Delete"; import { toast } from "react-toastify";

```
const SavedLocations = ({ onSelectLocation }) => { const [locations, setLocations] = useState([]); const [error, setError] = useState("");
```

```
useEffect(() => { const fetchLocations = async () => { try { const authToken = localStorage.getItem("authToken"); const response = await axios.get(`${import.meta.env.VITE_API_URL}/locations`, { headers: { Authorization: Bearer ${authToken}, }, }); setLocations(response.data); } catch (err) { setError("Failed to fetch locations"); } };
```

```
}, []);
```


```


const handleDeleteLocation = async (id) => { try { const authToken =
localStorage.getItem("authToken"); await
axios.delete(`${import.meta.env.VITE_API_URL}/location/${id}`, { headers: {
Authorization: `Bearer ${authToken}`, }, }); setLocations(locations.filter((location)
=> location.id !== id)); toast.success("Location deleted successfully!"); } catch (err)
{ toast.error("Failed to delete location"); } };

return ( <Box> <h2>Saved Locations</h2> {error && <p>{error}</p>} <List>
{locations.map((location) => ( <ListItem key={location.id} sx={{ display: "flex",
justifyContent: "space-between" }} > <ListItemText primary={ Lat:
`${location.latitude}`, Lon: `${location.longitude} } onClick={() =>
onSelectLocation(location)} /> <IconButton edge="end" aria-label="delete"
onClick={() => handleDeleteLocation(location.id)} > <DeleteIcon />
</IconButton> </ListItem> )} } </List> </Box> ); };

```

export default SavedLocations;

 Weather.jsx:124-165

 **GitHub Copilot**

> Used 2 references

To pass the `setLocation` function to the `SavedLocations` component and allow it to set the location, we need to update both the `Weather.jsx` and `SavedLocations.jsx` components.

Update `SavedLocations.jsx` :

Update `Weather.jsx` :

In this update:

- The `SavedLocations` component now receives the `setLocation` function as a prop and uses it to set the location when a location is selected.
- The `Weather` component passes the `setLocation` function to the `SavedLocations` component.



leonardrichertz

Make sure that the location state is updated when I create a location, so it is directly visible without having to reload the page.

```
import React, { useState,
useEffect } from "react"; import axios from "axios"; import { Box, List, ListItem,
ListItemText, IconButton, Button, } from "@mui/material"; import DeleteIcon from
"@mui/icons-material/Delete"; import { toast } from "react-toastify";
```

```
const SavedLocations = ({ onSelectLocation, setLocation }) => { const [locations,
setLocations] = useState([]); const [error, setError] = useState("");
```

```
useEffect(() => { const fetchLocations = async () => { try { const authToken =
localStorage.getItem("authToken"); const response = await axios.get(
`${import.meta.env.VITE_API_URL}/locations`, { headers: { Authorization:
`Bearer ${authToken}`, }, }); setLocations(response.data); } catch (err) {
setError("Failed to fetch locations"); } };
```


```
}, []);
```


```
const handleDeleteLocation = async (id) => { try { const authToken =
localStorage.getItem("authToken"); await
axios.delete(`${import.meta.env.VITE_API_URL}/location/${id}`, { headers: {
```

```
axios.delete(`${import.meta.env.VITE_API_URL}/location/${id}`, { headers: {
  Authorization: `Bearer ${authToken}`, }, }); setLocations(locations.filter((location)
=> location.id !== id)); toast.success("Location deleted successfully!"); } catch (err)
{ toast.error("Failed to delete location"); } };
```

```
return ( <Box> <h2>Saved Locations</h2> <List> {locations.map((location) => (
  <ListItem key={location.id} sx={{ display: "flex", justifyContent: "space-between" }}
  onClick={() => { onSelectLocation(location); setLocation({ lat: location.lat, lon:
    location.lon }); }} > <ListItemText primary={Lat: ${location.lat}, Lon:
    ${location.lon}} /> <Button onClick={setLocation(location)}>Select</Button>
    <IconButton edge="end" aria-label="delete" onClick={() =>
      handleDeleteLocation(location.id)} > <DeleteIcon /> </IconButton> </ListItem>
    )} </List> </Box> );
```

export default SavedLocations;

 Weather.jsx:126-167

 **GitHub Copilot**

> Used 2 references

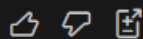
To ensure that the location state is updated when a location is created and is directly visible without having to reload the page, we need to update both the `Weather` and `SavedLocations` components. Specifically, we will update the `saveLocation` function to update the locations state and pass the updated locations to the `SavedLocations` component.

Update `Weather.jsx` :

Update `SavedLocations.jsx` :

In this update:


- The `Weather` component maintains the `locations` state and passes it along with `setLocations` to the `SavedLocations` component.
- The `SavedLocations` component updates the `locations` state when a new location is added or deleted, ensuring the UI reflects the changes immediately.



leonardrichertz

```
{ "message": "Too Many Attempts.", "exception":  
"Illuminate\\Http\\Exceptions\\ThrottleRequestsException", "file":  
"C:\\Users\\Schola\\Desktop\\Bachelorarbeit\\copilot_generated_code\\copilot-laravel-  
backend\\bachelor-copilot-  
backend\\vendor\\laravel\\framework\\src\\Illuminate\\Routing\\Middleware\\ThrottleRe  
quests.php", "line": 246, "trace": [ { "file":  
"C:\\Users\\Schola\\Desktop\\Bachelorarbeit\\copilot_generated_code\\copilot-laravel-  
backend\\bachelor-copilot-  
backend\\vendor\\laravel\\framework\\src\\Illuminate\\Routing\\Middleware\\ThrottleRe  
quests.php", "line": 156, "function": "buildException", "class":  
"Illuminate\\Routing\\Middleware\\ThrottleRequests", "type": "->" }, { "file":  
"C:\\Users\\Schola\\Desktop\\Bachelorarbeit\\copilot_generated_code\\copilot-laravel-  
backend\\bachelor-copilot-  
backend\\vendor\\laravel\\framework\\src\\Illuminate\\Routing\\Middleware\\ThrottleRe  
quests.php", "line": 93, "function": "handleRequest", "class":  
"Illuminate\\Routing\\Middleware\\ThrottleRequests", "type": "->" }, { "file":  
"C:\\Users\\Schola\\Desktop\\Bachelorarbeit\\copilot_generated_code\\copilot-laravel-
```


```
line": 17, "function": "handleRequest", "class": "Illuminate\\Foundation\\Application", "type": "->" } ]  
} Display the error message in toast messages, when an API request fails.
```

 Weather.jsx:20-59

GitHub Copilot

> Used 2 references

To display error messages in toast notifications when an API request fails, we can use `react-toastify`. We will update the `Weather` component to show toast error messages for API request failures.

Update  Weather.jsx :

In this update:

- Added `toast.error` to display error messages when an API request fails.
- Updated the `getWeatherData`, `handleGetLocation`, and `saveLocation` functions to show toast error messages when they encounter an error.

