**Tailwind setup voor Vite + React:** [**https://tailwindcss.com/docs/guides/vite**](https://tailwindcss.com/docs/guides/vite)

1. Maak React project aan met Vite.

npm install -D tailwindcss postcss autoprefixer

1. Draai het volgende in je client folder:
2. Verander default tailwind config:

/\*\* @type {import('tailwindcss').Config} \*/

export default {

content: ["./index.html", "./src/\*\*/\*.{js,ts,jsx,tsx}"],

theme: {

extend: {

***fontFamily: {***

***sans: ["Roboto", "sans-serif"],***

***},***

***gridTemplateColumns: {***

***"70/30": "70% 28%",***

***},***

},

},

plugins: [],

};

1. Verander de index.css config naar

@tailwind base;

@tailwind components;

@tailwind utilities;

1. Verander de VSCode settings (voor alle projecten) door “ctrl + shift + p” en zoek naar “open user settings”, zoek vervolgens naar “unkown **at rules**” kies hier bij CSS naar ignore
2. Verander fontFamily extensie als je een eigen font wilt gebruiken met specifieke naam
3. Verander grid template of verwijder het als je er geen gebruik van wilt maken.

Extra uitleg voor tailwind config

fontFamily:

In de extend file is deze naam alleen de configuratie naam, de echte prefix die je gebruikt in praktijk in je HTML of JSX is ‘font-(iets zoals sans).

gridTemplateColumns:

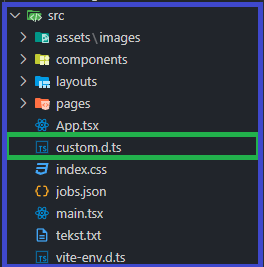
We gebruiken in de extend file bij de naam 70/30 quotes om aan te geven dat het een string hoort te zijn i.v.m. speciale karakters. Bij sans had je ook “sans” kunnen doen maar dat is onnodig

“naam”: “waarde”

**TypeScript fouten die je kunt tegenkomen**

* *Importeren van SVG’s geeft type fouten* ***fix***:

Maak in je src folder een custom.d.ts file met hetvolgende als code:



declare module "\*.svg" {

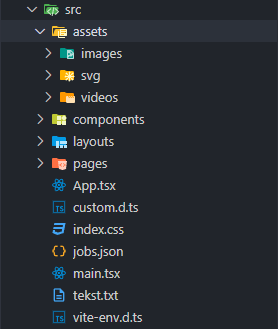
const content: string;

export default content;

}

**Bestand structuur voor React**

In je source folder kan je een erg recht toe rechtaan structuur maken zoals hier:



In **assets** kan je media bestanden bewaren en opdelen in folders als je dat wilt.

In **components** plaats je alle pagina componenten.

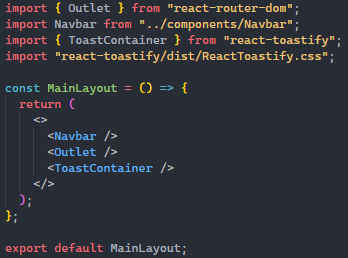
In **layouts** plaats je bepaalde layouts die je wilt hergebruiken (zie meer hieronder).

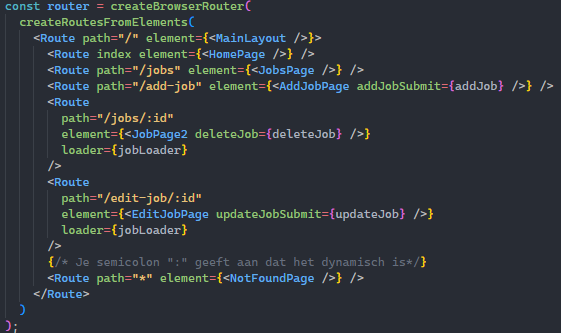
In **pages** bewaar je alle pagina’s waar later kunnen worden bezocht.

Elke pagina in dit website voorbeeld wilt gebruik maken van een MainLayout (in app.tsx hieronder zie je dat).

Maar dat de gegevens die staan genest in de MainLayout bij het app.tsx bestand (hieronder) betekent nog niet dat ze toegankelijk zijn, daarvoor moet je de outlet gebruiken (je mag maar 1 ding returnen).

toast.succes(“tekst”) triggerd de toast container die aboslute is gepositioneerd.





**Interessante dingen**

npm i react-toastify: <https://www.npmjs.com/package/react-toastify>, <https://fkhadra.github.io/react-toastify/introduction/>

import { toast } from "react-toastify";

toast.success("Goed gedaan");

import { ToastContainer } from "react-toastify";

import "react-toastify/dist/ReactToastify.css";

In MainLayout  <ToastContainer />

npm i react-icons: <https://www.npmjs.com/package/react-icons>, <https://react-icons.github.io/react-icons/>

import { FaMapMarker } from "react-icons/fa";

<div className="text-orange-700 mb-3">

              <FaMapMarker className="inline text-lg mb-1 mr-1" />

              {job.location}

            </div>

npm i react-spinners: <https://www.npmjs.com/package/react-spinners>, <https://www.davidhu.io/react-spinners/>

import { ClipLoader } from "react-spinners";

{Maak spinner component}

import Spinner from "../components/Spinner";

return loading ? (

    <Spinner />

  ) : (

    <>

      <h1>{job.title}</h1>

    </>

  );

npm i react-router-dom: <https://www.npmjs.com/package/react-router-dom>, <https://reactrouter.com/en/main/start/tutorial>

import {

  Route,

  createBrowserRouter,

  createRoutesFromElements,

  RouterProvider,

} from "react-router-dom";

import { useParams, useLoaderData, useNavigate } from "react-router-dom";

import { Link } from "react-router-dom";

import { NavLink } from "react-router-dom"; // Geeft een isActive class mee voor dynamische kleuring van bijvoorbeeld navigatie.

import { useState, useEffect } from "react";

import { ReactNode } from "react"; // Types voor TypeScript als je children zoals JSX in een component hebt genest die zoals <Card> <p>Hello </p> </Card>, p wordt meegegeven.

**MOCKEND**: npm i -D json-server: <https://www.npmjs.com/package/json-server>, bij package.json in scripts voeg toe: “server”: “json-server –watch src/jobs.json –port 8000” draai met npm run server.