



Proyecto “Migas amigas” paso a paso

0.- Descripción.....	1
1.- Creando proyecto.....	2
2.- Instalando componentes.....	3
2.1.- Componentes para habilitar la navegación por url.....	3
2.2.- Componente para habilitar varios idiomas.....	4
3.- Limpiando proyecto.....	4
4.- Ejecutando el proyecto.....	6
5.- Creando estructura de carpetas.....	6
6.- Creando páginas.....	7
7.- Creando navegación.....	9
8.- Creando cabecera y pie de página.....	11
9.- Maquetación.....	13
10.- Añadiendo multilenguaje.....	13
10.1.- Moviendo traducciones a otras carpetas.....	17

0.- Descripción

Se pretende elaborar una interfaz gráfica para el sitio Web de un negocio familiar de panaderos situado en la frontera con Portugal. El nombre del cartel que figura encima de la puerta del local es: **“Migas amigas”**

Después de hablar con los clientes se ha llegado a la conclusión de que su único objetivo es dar a conocer su negocio a través de Internet.

Para ello debemos desarrollar una interfaz que sea sobre todo visual. Por ello nos han pedido que en la página principal pongamos una animación donde con dibujos se vea el proceso de elaboración del pan.

El sitio Web deberá:

- Permitir la lectura del sitio en dos idiomas: Castellano y Portugués.
- Tener tres secciones: panadería, pastelería - bollería y empanadas.
- Mostrar los productos que se elaboran en la panadería: una breve descripción de los ingredientes y el sistema de elaboración acompañados de una fotografía del producto. El cliente nos ha dicho que actualmente tienen un máximo de 10 productos de Panadería, 8 de bollería-pastelería y 4 de empanadas, pero que tienen pensado aumentar hasta 12 los productos de panadería.
- Mostrar la información de contacto con la dirección, el teléfono y un croquis de su ubicación.

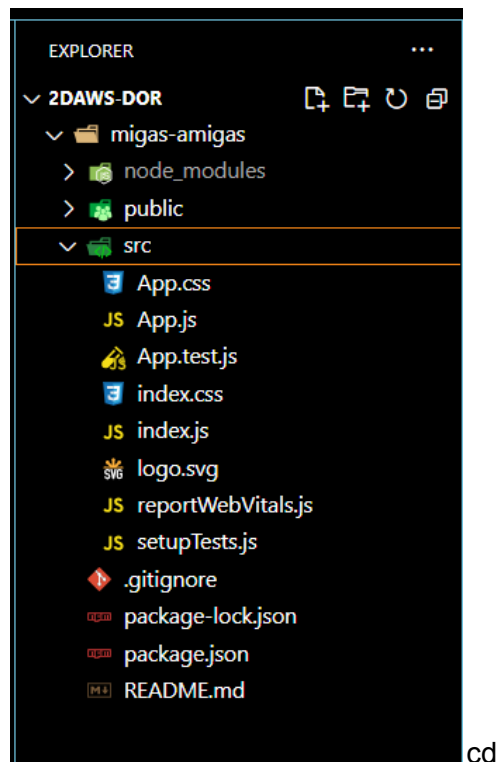


1.- Creando proyecto

Creemos el proyecto “Migas amigas” para ello abrimos un terminal en “Visual Studio Code” (en adelante VSC) y ejecutamos:

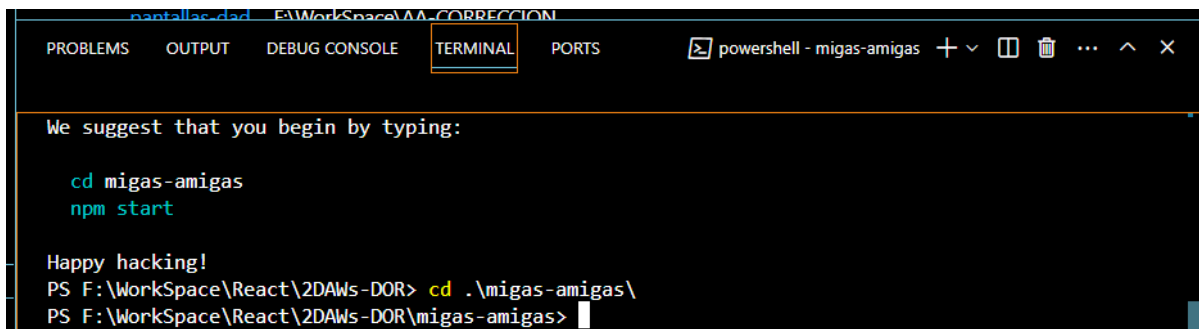
```
npx create-react-app migas-amigas
```

Observamos que en el Explorer de VSC ya se ha creado la estructura básica:



Entramos al proyecto:

```
cd migas-amigas
```





2.- Instalando componentes

2.1.- Componentes para habilitar la navegación por url

Para implementar la navegación necesitamos un componente Router el `react-router-dom`. Para instalarlo ejecutamos el comando:

NOTA: Asegúrate de estar dentro de la carpeta del proyecto

`npm install react-router-dom`

```
PS F:\Workspace\React\2DAWs-DOR> cd .\migas-amigas\  
PS F:\Workspace\React\2DAWs-DOR\migas-amigas> npm install react-router-dom
```

Podemos comprobar que el fichero `package.json` se ha instalado:

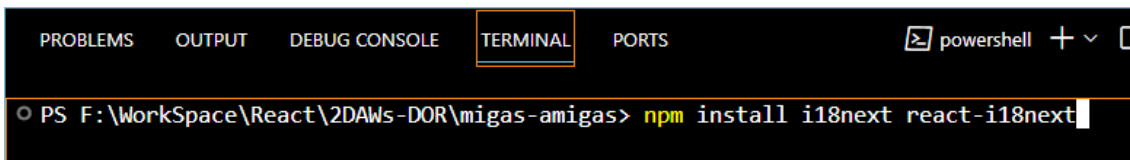
```
{  
  "name": "migas-amigas",  
  "version": "0.1.0",  
  "private": true,  
  "dependencies": {  
    "@testing-library/jest-dom": "^5.17.0",  
    "@testing-library/react": "^13.4.0",  
    "@testing-library/user-event": "^13.5.0",  
    "react": "^18.2.0",  
    "react-dom": "^18.2.0",  
    "react-router-dom": "^6.22.2",  
    "react-scripts": "5.0.1",  
    "web-vitals": "^2.1.4"  
  }  
}
```



2.2.- Componente para habilitar varios idiomas

Vamos a hacer uso de la librería i18n, concretamente la i18next y la de react-i18next. Para ello ejecutamos el siguiente comando en la Terminal de VSC:

```
npm install i18next react-i18next
```



La implementación de la traducción la vemos al final.

3.- Limpiando proyecto

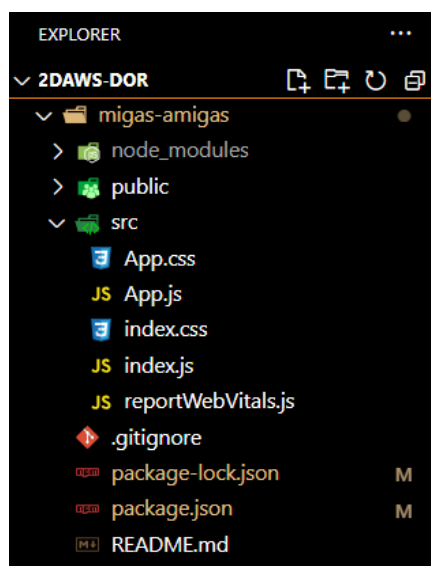
En este proyecto no vamos a hacer uso de Tests así que podemos quitar los ficheros:

- setupTests.js
- App.test.js
- reportWebVitals.js

También quitaremos el logo:

- logo.svg

El proyecto queda así:





Ahora recortaremos los ficheros para eliminar rastros de enlaces a las cosas que hemos eliminado:

Fichero ***index.js***

Eliminamos:

- `import reportWebVitals from './reportWebVitals';`
- `<React.StrictMode>` y `</React.StrictMode>`
- Líneas:
 `// If you want to start measuring performance in your app, pass a function`
 `// to log results (for example: reportWebVitals(console.log))`
 `// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals`
 `reportWebVitals();`

Y acomodamos el código:

```
const container = document.getElementById('root');  
const root = ReactDOM.createRoot(container);  
root.render( <App /> );
```

Finalmente el código queda:

```
import React from 'react';  
import ReactDOM from 'react-dom/client';  
import './index.css';  
import App from './App';  
  
const container = document.getElementById('root');  
const root = ReactDOM.createRoot(container);  
root.render( <App /> );
```

Fichero ***App.js***

Eliminamos la línea: `import logo from './logo.svg';`

Retocamos el código:

```
import './App.css';  
  
function App() {  
  return (  
    <div> <h1>PROYECTO "Migas amigas"</h1> </div>  
  );  
}
```

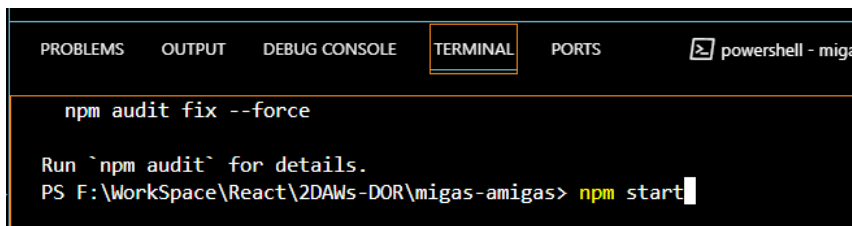


```
);  
}  
export default App;
```

4.- Ejecutando el proyecto

En la terminal ejecutamos

- npm start

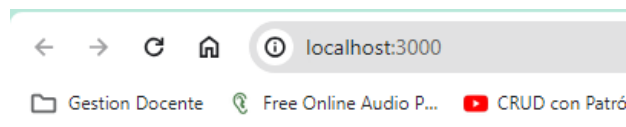


```
npm audit fix --force  
  
Run `npm audit` for details.  
PS F:\WorkSpace\React\2DAWs-DOR\migas-amigas> npm start
```

Si no se nos abre el navegador automáticamente podemos ver que el repor que ha ido saliendo por la terminal nos indica:

http://localhost:3000 la ruta para ver la aplicación.

Y esto es lo que nos debe aparecer:



PROYECTO "Migas amigas"

Justo lo que pusimos en el fichero App.js.

5.- Creando estructura de carpetas

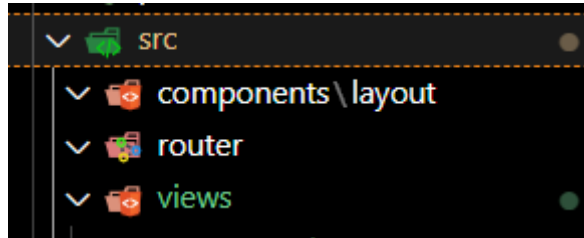
Necesitaremos en principio dos carpetas: una para alojar los componentes y otra para las páginas de navegación. Las llamaremos respectivamente: components, views y router.

Y dentro de componentes una subcarpeta llamada layout para luego guardar ahí el header y footer.



Por **convención** el nombre de las carpetas van en minúscula y el de los componentes en mayúscula.

Dichas carpetas las creamos dentro de src (click derecho sobre “src” y “New Folder”):



6.- Creando páginas

Tendremos las páginas:

Tipo	Descripción	Nombre fichero
Inicio	página principal	Principal.js
Secciones	panadería	Panaderia.js
	pastelería - bollería	PateleriaBolleria.js
	empanadas	Empanadas.js
Contacto	dirección, el teléfono y un croquis de su ubicación	Contacto.js
Error	Error página no encontrada	Error404.js

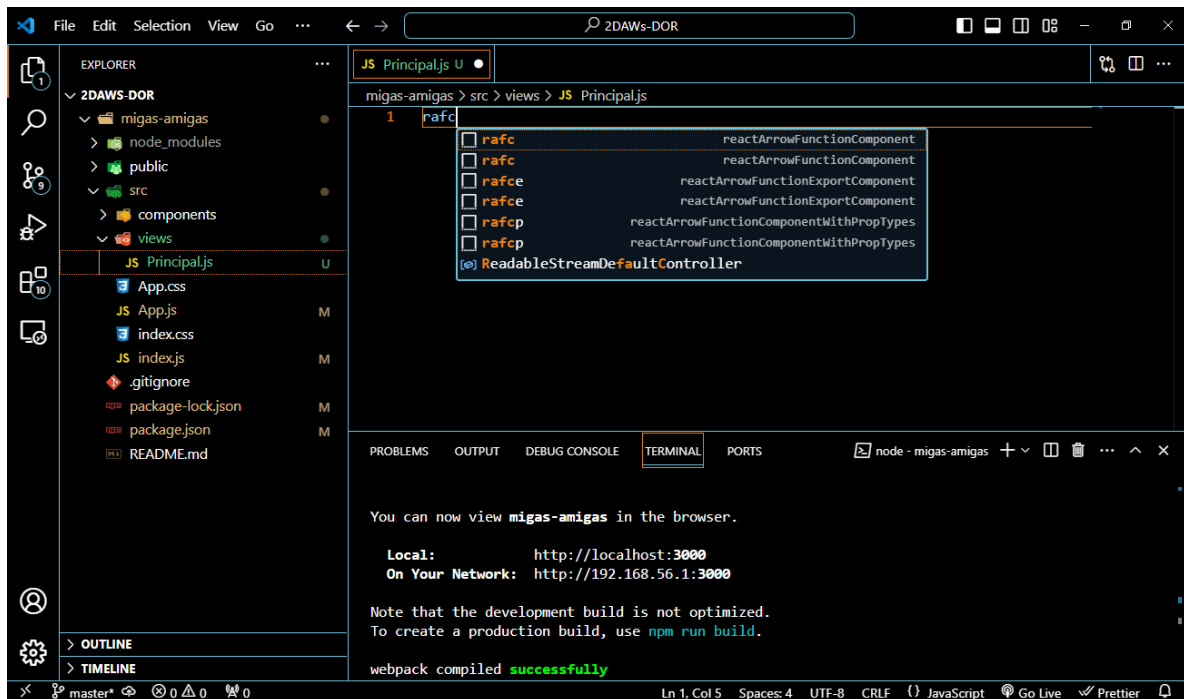
NOTA: para REACT las vistas son también componentes.

Añadimos, como buena práctica, una página de Error en caso de que el usuario use una url inapropiada.

Crearemos dentro de la carpeta views un fichero js con el nombre de cada página:

Podemos hacer uso de los snippets instalados con los componentes de VSC.

Si escribimos **rafc** elegimos el primero para crear la estructura de un componente:



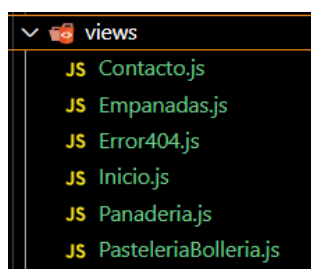
En medio del <div> añadimos un <h1> y dentro “Esta es la página principal”:

```
import React from 'react'
const Principal = () => {
  return (
    <div>
      <h1>Esta es la página principal</h1>
    </div>
  )
}
export default Principal
```

Ahora tú

Crea el resto de vistas tú.

Debe quedar:





7.- Creando navegación

Para crear la navegación haremos uso de la librería react-router-dom ya instalada previamente.

Primeramente creemos dentro de la carpeta src una nueva llamada router y dentro el fichero MisRutas.js. Debe quedar algo como:

:

Importamos los elementos con los que vamos a trabajar:

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

Explicación

BrowserRouter → Crea el contexto de navegación. Todo lo escrito en su interior se repetirá en todas las vistas (páginas). Así que, es buen lugar para poner el **header** y el **nav** (la barra de navegación).

Routes → Crea el contexto para añadir las rutas de navegación.

Route → Crea la ruta:

path → es la ruta que añadiremos al final de la url para navegar a dicha vista.

element → carga la vista (componente vista)

Tendremos que importar al fichero cada uno de los componentes de la vista:

El código quedará como:

```
import React from 'react'
```



```
import { Routes, Route, BrowserRouter } from "react-router-dom";
import Inicio from "../views/Inicio";
import Panaderia from "../views/Panaderia";
import PasteleriaBolleria from "../views/pasteleriabolleria";
import Empanadas from "../views/Empanadas";
import Contacto from "../views/Contacto";

const MisRutas = () => {
  return (
    <BrowserRouter>
      { /* Header y Navegacion */ }
      <h1>Cabecera</h1>
      <hr />
      { /* Contenido Central */ }
      <Routes>
        <Route path="/" element={<Inicio />} />
        <Route path="/inicio" element={<Inicio />} />
        <Route path="/panaderia" element={<Panaderia />} />
        <Route path="/pasteleriabolleria"
element={<PasteleriaBolleria />} />
        <Route path="/empanadas" element={<Empanadas />} />
        <Route path="/contacto" element={<Contacto />} />
        <Route path="*" element={<Error404 />} />
      </Routes>
      { /* Footer */ }
      <hr />
      <h1>Pié de página</h1>
    );
  }
export default MisRutas
```

Ahora vamos a añadir el componente MisRutas.js a App.js para que realmente se produzca la magia:

```
import './App.css';
import MisRutas from './router/MisRutas';

function App() {
  return (
    <div>
      <MisRutas />
    </div>
  );
}
```

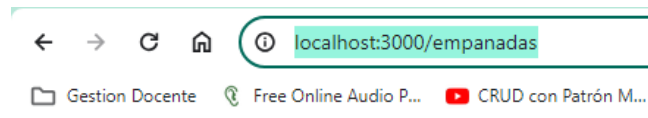


```
</div>

);
}

export default App;
```

Con ello ya podrás navegar por la url. Por ejemplo a “empanadas”



Cabecera

Esta es la página empanadas

Pié de página

Observamos que al navegar sólo se modifica lo que está dentro de Routes. La cabecera y pié de página se mantienen intactos. Es decir, el navegador solo actualiza la parte de Router lo que implica menos tráfico en la red y mayor rapidez.

8.- Creando cabecera y pié de página

Dentro de la carpeta components/layout creamos los componentes HeaderNav.js y Footer.js

Sí hemos llamado HeaderNav porque aprovecharemos a crear una barra de navegación.

HeaderNav.js

```
import React from 'react'

const HeaderNav = () => {
  return (
    <div>
      <h1>Cabecera y barra de navegacion</h1>
    </div>
  )
}
```



```
}  
  
export default HeaderNav
```

Footer.js

```
import React from 'react'  
  
const Footer = () => {  
  return (  
    <div>  
      <h1>Pié de página</h1>  
    </div>  
  )  
}  
  
export default Footer
```

Y retocamos el fichero **MisRutas.js** para añadir los componentes:

```
import React from 'react'  
import { Routes, Route, BrowserRouter, NavLink } from  
"react-router-dom";  
import Inicio from "../views/Inicio";  
import Panaderia from "../views/Panaderia";  
import PasteleriaBolleria from "../views/PasteleriaBolleria";  
import Empanadas from "../views/Empanadas";  
import Contacto from "../views/Contacto";  
import Error404 from "../views/Error404";  
import HeaderNav from "../components/layout/HeaderNav";  
import Footer from "../components/layout/Footer";  
  
const MisRutas = () => {  
  return (  
    <BrowserRouter>  
      { /* Header y Navegacion */ }  
      <HeaderNav />  
      <hr />  
      { /* Contenido Central */ }
```



```
<Routes>
  <Route path="/" element={<Inicio />} />
  <Route path="/inicio" element={<Inicio />} />
  <Route path="/panaderia" element={<Panaderia />} />
  <Route path="/pasteleriabolleria"
element={<PasteleriaBolleria />} />
  <Route path="/empanadas" element={<Empanadas />} />
  <Route path="/contacto" element={<Contacto />} />
  <Route path="*" element={<Error404 />} />
</Routes>

  {/* Footer */}
  <hr />
  <Footer />
</BrowserRouter>
);
}
export default MisRutas
```

9.- Maquetación

Ahora solo queda dar forma con HTML y CSS cada página.

Es recomendable siempre usar un grid, bien de BootsTrap, Flex o React-Bootstrap (frameworks de CSS) para colocar los componentes y hacerlo responsivo, pero esto será tema de otra unidad. Por ahora crea el esqueleto con HTML haciendo uso de las etiquetas: div, section y ul.

En cuanto al dar un aspecto agradable (fuente, tamaño y color de letras, color de fondo, etc) haz uso de algún framework CSS de los nombrados, o si usas clases propias añádele en el fichero index.css.

10.- Añadiendo multilenguaje

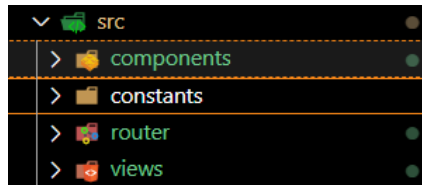
Tal como nos piden los requisitos debemos poder traducir la aplicación a portugués y español.

Necesitamos unas librería: i18next react-i18next que al comienzo de la creación del proyecto las instalamos.

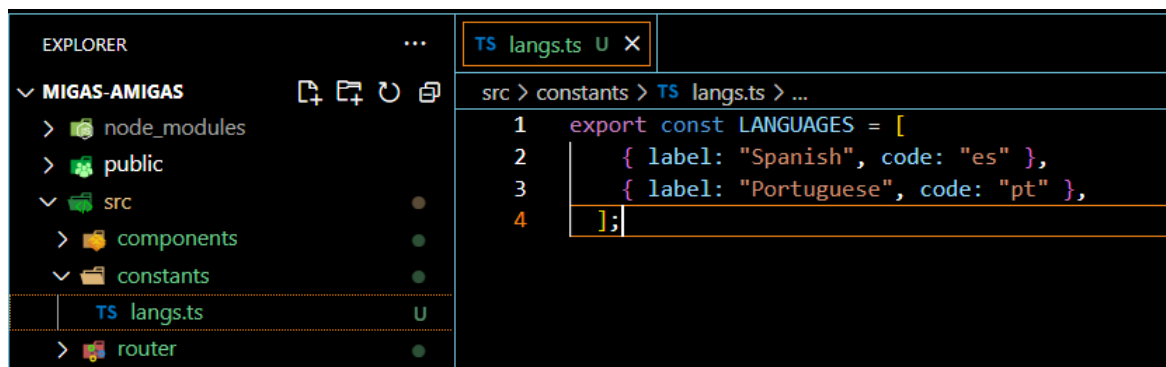


Ahora seguimos los siguientes pasos:

1.- Creamos la carpeta constants dentro de src:



2.- Creamos dentro el fichero lang.ts y dentro definimos los idiomas en que implementaremos la página web.



3.- Creamos en src el fichero i18n.ts con el contenido:

```
import i18n from "i18next";  
import { initReactI18next } from "react-i18next";  
  
i18n  
  .use(initReactI18next)  
  .init({  
    lng: "es",  
    fallbackLng: "es",  
    interpolation: {  
      escapeValue: false,  
    },  
    resources: {  
      pt: {  
        translation: {  
  
        }  
      },  
    },  
  });
```



```
es: {  
  translation: {  
  
  }  
},},  
));  
  
export default i18n;
```

Es en este fichero donde se hacen las traducciones.

4.- Añadimos en index.js el:

```
import './i18n.ts';
```

5.- Insertando el selector de idiomas en el HeaderNav.

```
import React from 'react'  
import { useTranslation } from 'react-i18next';  
import { LANGUAGES } from '../constants/lang.ts';  
  
const HeaderNav = () => {  
  
  const { i18n, t } = useTranslation();  
  
  return (  
    <div>  
      <h1>Cabecera y barra de navegacion</h1>  
  
      <select defaultValue={i18n.language}  
onChange={onChangeLang}>  
        {LANGUAGES.map(({ code, label }) => (  
          <option key={code} value={code}>  
            {label}  
          </option>  
        ))}  
      </select>  
    </div>  
  )  
}  
  
export default HeaderNav
```



6.- Realizar las traducciones.

Empecemos por el texto de la cabecera del fichero HeaderNav.js.

El texto dentro de h1 lo sustituimos por un identificador tal como sigue:

```
return (  
  <div>  
    <h1>{t("titleHeader")}</h1>
```

Y en el fichero i18n.ts añadimos:

```
pt: {  
  translation: {  
    titleHeader: 'Cabeçalho e barra de navegação',  
  },  
},  
es: {  
  translation: {  
    titleHeader: 'Cabecera y barra de navegacion',  
  },  
},},
```

El código completo es:

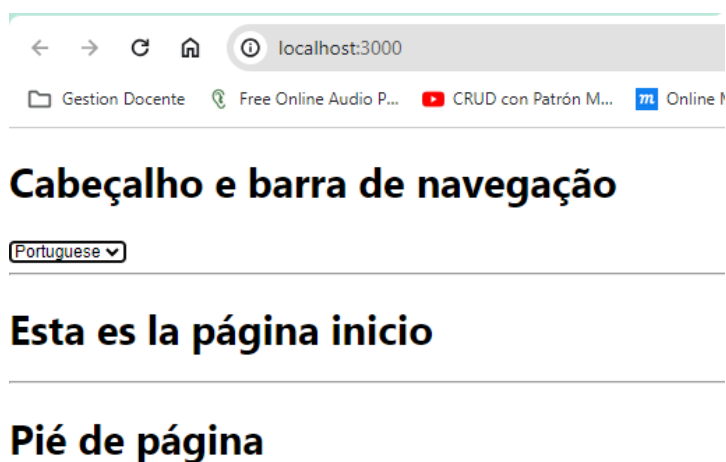
```
import React from 'react';  
import { useTranslation } from "react-i18next";  
import { LANGUAGES } from "../../constants/index.ts";  
  
const HeaderNav = () => {  
  
  const { i18n, t } = useTranslation();  
  
  const onChangeLang = (e: React.ChangeEvent<HTMLSelectElement>) => {  
    const lang_code = e.target.value;  
    i18n.changeLanguage(lang_code);  
  };  
  
  return (  
    <div>  
      <h1>{t("titleHeader")}</h1>
```




```
<select defaultValue={i18n.language} onChange={onChangeLang}>
  {LANGUAGES.map(({ code, label }) => (
    <option key={code} value={code}>
      {label}
    </option>
  ))}
</select>
</div>
)
}

export default HeaderNav
```

7.- Comprobamos la traducción ejecutando el proyecto (npm start) y ver en el navegador:



Ahora tú

Traduce el resto de páginas.

10.1.- Moviendo traducciones a otras carpetas

A.- Cargar librería i18next-http-backend

Detenemos el proyecto y cargamos la librería i18next-http-backend. Para ello desde la terminal, dentro de la carpeta del proyecto, ejecutamos:

```
npm install i18next-http-backend
```



B.- Creando estructura de carpetas

Ahora vamos a crear dentro de la carpeta public una carpeta i18n.

Y dentro vamos a ir creando archivos .json que serán nombrados según sea su traducción, por ejemplo. El archivo es.json será para las traducciones en Español, el archivo pt.json sera solo para las traducciones en portugués.



C.- Movemos los contenidos

Movemos cada contenido del objeto translation del archivo i18n.ts a su archivo JSON correspondiente.

Por ejemplo el archivo es.json.

```
{
  navInicio: 'Inicio',
  navEmpanadas: 'Empanadas',
  navPanaderia: 'Panadería',
  navPasteleriaBolleria: 'Pastelería y Bollería',
  navContacto: 'Contacto',
  titleHeader: 'Cabecera y barra de navegacion',
  titleEmpanadas: 'Esta es la página de empanadas',
  titleHome: 'Esta es la página de inicio',
  titlePanaderia: 'Esta es la página de panadería',
  titlePiePagina: 'Título pie de página',
  titleError404: 'Error 404',
  titlePagNoEncon: 'Página no encontrada',
}
```

D.- Modificando fichero i18n.ts

- Añadimos el import de la nueva librería:
`import i18nBackend from "i18next-http-backend";`
- Añadimos el nuevo use de la nueva librería:
`use(i18nBackend)`

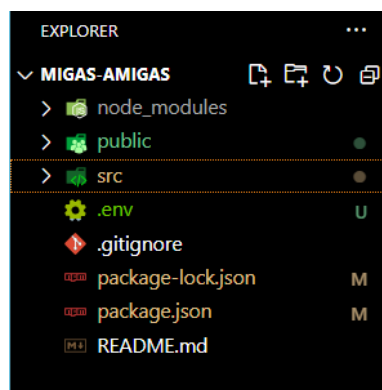


- Añadimos un backend

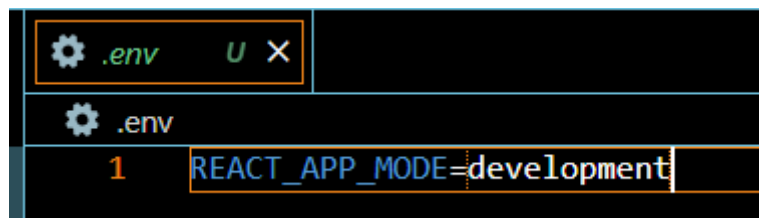
```
backend: {  
  loadPath: `${getCurrentHost}/public/i18n/${lng}.json`,  
},
```

Pero hay un paso más que hacer, si notas en la propiedad `loadPath`, el host es <http://localhost:5550> (utilizo el puerto de “Go Live” de VSC) y cuando lo suba a producción, no funcionarían las traducciones por lo cual debemos validar si estamos en modo desarrollo o no, para poder agregar el host correcto.

Pero antes crearemos el fichero `.env` en la raíz del proyecto:



Y en el añadimos la línea:



Y el código de `i18n.ts` queda como:

```
import i18n from "i18next";  
import i18nBackend from "i18next-http-backend";  
import { initReactI18next } from "react-i18next";  
  
const getCurrentHost = process.env.REACT_APP_MODE === 'development' ?  
'http://localhost:5500' : 'LINK TO PROD'  
  
console.log("REACT_APP_MODE: " + process.env.REACT_APP_MODE);  
console.log("CurrentHost: " + getCurrentHost);
```



```
i18n
  .use(i18nBackend)
  .use(initReactI18next)
  .init({
    lng: "es",
    fallbackLng: "es",
    interpolation: {
      escapeValue: false,
    },
    backend: {
      loadPath: `${getCurrentHost}/public/i18n/{{lng}}.json`,
    },
  });

export default i18n;
```

NOTA:

Puesto que estamos haciendo uso de un backend simulado tenemos que arrancar el servidor de VSC, el “Go Live” para ejecutar correctamente el proyecto.

Como se puede producir un retraso en la carga de los datos de traducción, en local no se nota pero en la realidad puede que sí. Deberemos poner un Suspense de React que envuelva a BrowserRouter. De manera que MisRutas.js queda como:

```
import { React, Suspense } from "react";
import { Routes, Route, BrowserRouter } from "react-router-dom";
import Inicio from "../views/Inicio";
import Panaderia from "../views/Panaderia";
import PasteleriaBolleria from "../views/PasteleriaBolleria";
import Empanadas from "../views/Empanadas";
import Contacto from "../views/Contacto";
import Error404 from "../views/Error404";
import HeaderNav from "../components/layout/HeaderNav.tsx";
import Footer from "../components/layout/Footer";

const MisRutas = () => {
  return (
    <Suspense fallback="loading">
      <BrowserRouter>
        { /* Header y Navegacion */ }
      </BrowserRouter>
    </Suspense>
  );
};
```



```
<HeaderNav />
<hr />

{/* Contenido Central */}

<Routes>
  <Route path="/" element={<Inicio />} />
  <Route path="/inicio" element={<Inicio />} />
  <Route path="/panaderia" element={<Panaderia />} />
  <Route path="/pasteleriabolleria"
element={<PasteleriaBolleria />} />
  <Route path="/empanadas" element={<Empanadas />} />
  <Route path="/contacto" element={<Contacto />} />
  <Route path="*" element={<Error404 />} />
</Routes>

{/* Footer */}
<hr />
<Footer />
</BrowserRouter>
</Suspense>
);
};

export default MisRutas;
```