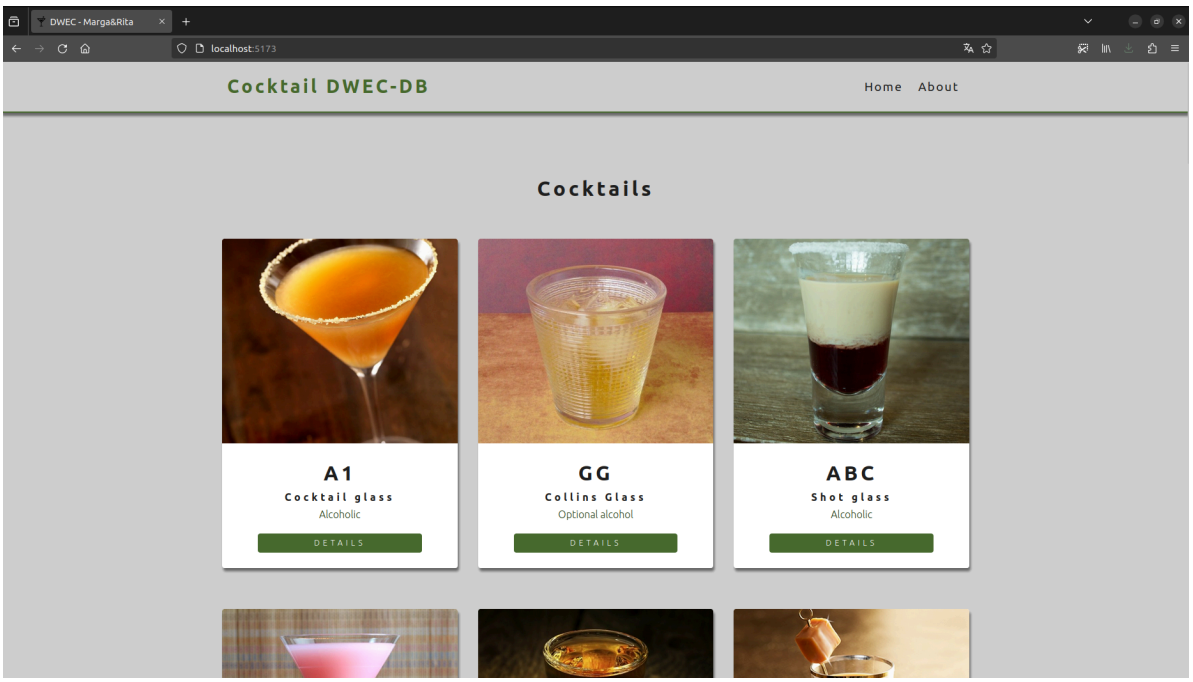
DWEC06 - Tarea 03

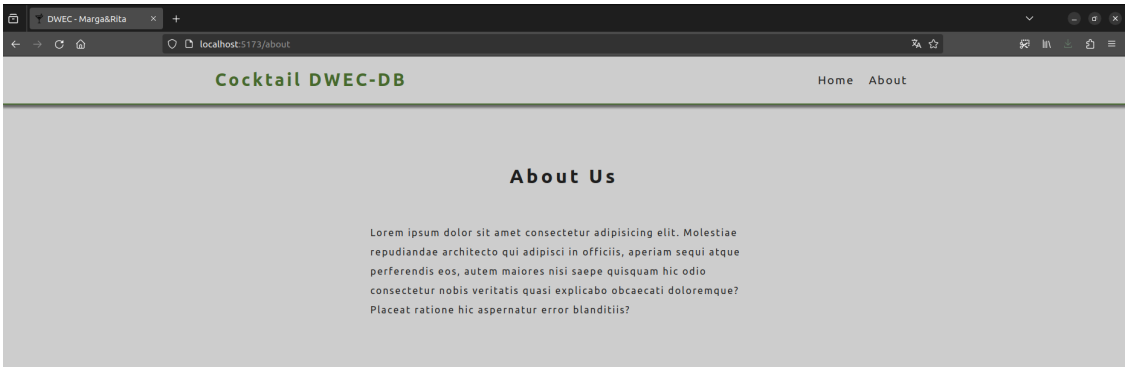
ENUNCIADO DE LA TAREA:

En la plantilla que se os facilita aparece una un documento HTML de ejemplo (después deberás borrar o comentar las líneas que aparecen) para que veais que elementos teneis que crear y con qué clase para que quede correctamente estilado.



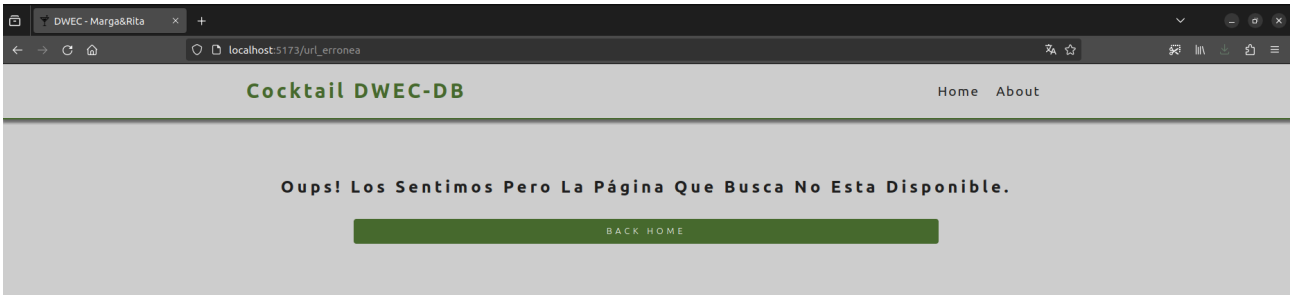
Vamos a realizar una página donde podamos ver la carta de cocktails que disponemos en nuestro nuevo local de moda, el **Marga&Rita**. Esta página será una SPA en la que tendremos que implementar unas rutas concretas:

- La página inicial siempre será la de **HOME**. Aquí podremos ver todos los cocktails en formato reducido.
- Tenemos una página donde veremos información acerca de nosotros (**ABOUT**), los nuevos dueños del local (podéis poner lo que queráis, en el ejemplo hay texto “lorem ipsum”).

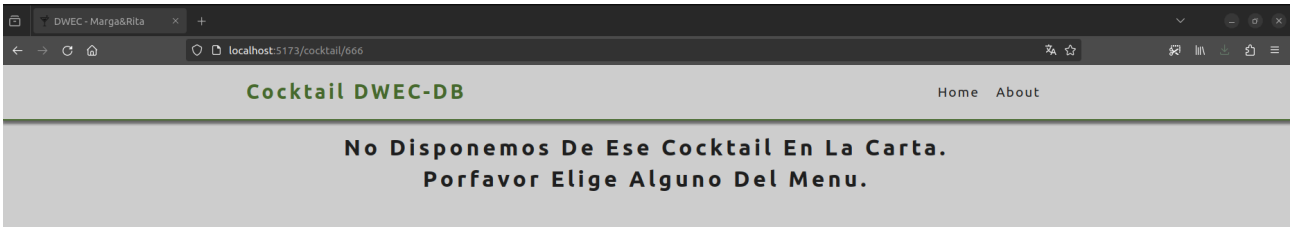


- Cuando pulsemos en el enlace para ir a detalles de los cocktails, iremos a una página donde se verán más en detalle los ingredientes y las instrucciones. La página se llamará **COCKTAIL** pero tendrá que ser dinámica ya que cada cocktail tendrá su página de detalles.

Si por el un casual, algún usuario cambia la URL a mano y no coincide con ninguna de nuestras rutas/páginas se mostrará un mensaje como el siguiente con un enlace para volver a la página de **HOME**.



Lo mismo va ocurrir cuando intentemos acceder a algún cocktail que pongamos manualmente en la URL.



ACLARACIONES EL ENUNCIADO:

- Cuando os descarguéis la plantilla, ejecutar “npm i” para instalar las dependencias del proyecto.

- Podéis utilizar y guiaros con las estructura de ficheros e *imports* que utiliza un proyecto de vite de prueba.

- Los nombres de los componentes los decidís vosotros mismos.

- Disponemos de 2 URLs de la API para obtención de datos:

    - *https://www.thecocktaildb.com/api/json/v1/1/search.php?s=*
    - *https://www.thecocktaildb.com/api/json/v1/1/lookup.php?i=**id_cocktail***

  En la primera no poner nada más detrás, nos sirve para buscar por string que coincida con los nombres de cocktails pero no vamos a usarlo, al hacerlo así, obtenemos todos los disponibles.

- En los ejemplo de los HTML solo teneis que fijaros en las clases, los enlaces los tendréis que hacer con ReactRouter.

- Necesitaremos en total 4 páginas:
    - HOME, para ver toda la lista de cocktails.
    - ABOUT, para ver información sobre los dueños y demás.
    - ERROR, para mostrar cuando no se encuentra la página solicitada.
    - COCTAIL, para ver la información detallada de cada cocktail.

- Necesitaremos en total 4 componentes:
    - NAVBAR, la barra de navegación para el título y los enlaces a diferentes páginas. Estará visible siempre en cualquier página en la que estemos.
    - COCKTAIL-LIST, para ver/mostrar todos los cocktails en formato pequeño.
    - COCKTAIL, se utilizará para crear cada una de las miniaturas de los cocktails.
    - LOADING, este componente se visualiza mientras estemos pidiendo los datos de los cocktails. Mientras se esté haciendo la petición de datos a la API (muy poco tiempo) se visualiza un div que tiene una clase especial que renderiza un pequeño efecto. El código para ese componente es este:

```
const Loading = () => {
    return (
        <div className="loader">
        </div>
    )
}

export default Loading
```

ACLARACIONES SOBRE LA ENTREGA DE LA TAREA:

- Para la entrega de esta tarea tendréis que subir el proyecto a alguna plataforma (github, gitlab, …) para que yo pueda descargarla ya que por tema del tamaño del proyecto no podremos subirlo a la plataforma moodle. Nombrar al repositorio con la nomenclatura habitual.

  Apellido1_Apellido2_Nombre_DWEC06_Tarea03

- Utilizar los comentarios de código para realizar breves explicaciones. En caso de necesitar/querer explicar más en detalle el ejercicio utilizar la plantilla e ir adjuntando capturas de pantalla del código y las explicaciones y aclaraciones necesarias .

- Criterios de evaluación relacionados con la tarea:

RA-5	CE	a) Se han reconocido las posibilidades del lenguaje de marcas relativas a la captura de los eventos producidos.
	CE	b) Se han identificado las características del lenguaje de programación relativas a la gestión de los eventos.
	CE	c) Se han diferenciado los tipos de eventos que se pueden manejar.
	CE	d) Se ha creado un código que capture y utilice eventos.
	CE	h) Se ha probado y documentado el código.
RA-6	CE	a) Se ha reconocido el modelo de objetos del documento de una página web.
	CE	b) Se han identificado los objetos del modelo, sus propiedades y métodos.
	CE	c) Se ha creado y verificado un código que acceda a la estructura del documento.
	CE	d) Se han creado nuevos elementos de la estructura y modificado elementos ya existentes.
	CE	e) Se han asociado acciones a los eventos del modelo.
	CE	g) Se han programado aplicaciones web de forma que funcionen en navegadores con diferentes implementaciones del modelo.
	CE	h) Se han independizado las tres facetas (contenido, aspecto y comportamiento), en aplicaciones web.
RA-7	CE	c) Se han utilizado los objetos relacionados.
	CE	d) Se han identificado sus propiedades y sus métodos.

	CE	e) Se ha utilizado comunicación asíncrona en la actualización dinámica del documento web.
	CE	f) Se han utilizado distintos formatos en el envío y recepción de información.
	CE	h) Se han clasificado y analizado librerías que faciliten la incorporación de las tecnologías de actualización dinámica a la programación de páginas web.
	CE	i) Se han creado y depurado programas que utilicen estas librerías.