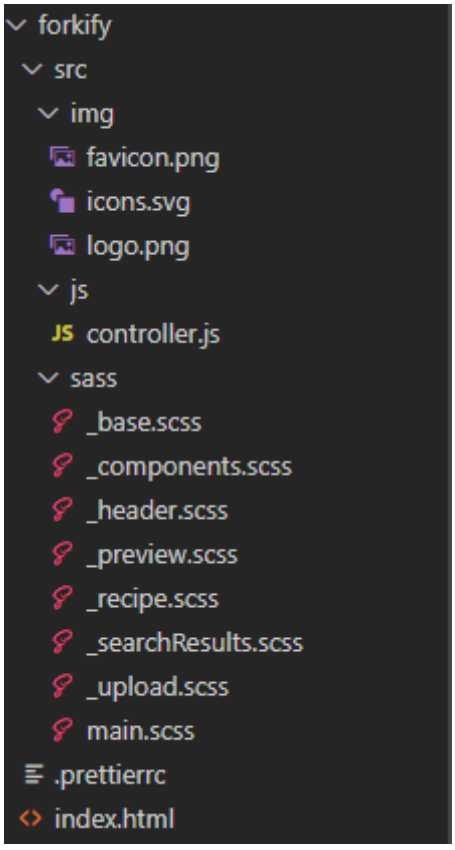
Avance 1

1. Toma la base del proyecto que se incluye en la sección Archivos adjuntos y ábrelo en Visual Studio Code.
2. Una vez que tengas tu código, analiza que tenga todos estos componentes:



3. Para este proyecto se utilizará la herramienta **parcel**, la cual procesa todo el código de la aplicación web y recorre todos los archivos enlazándolos y generando una nueva colección de archivos más apropiados para el navegador.
4. Abre una nueva terminal e inicializa un nuevo proyecto con la instrucción de npm. Utiliza la siguiente configuración:
   a) En nombre del proyecto, coloca forkify si es que no aparece por defecto.
   b) El siguiente parámetro a configurar es el *autor*, coloca tu nombre.
   c) Todas las otras opciones déjalas como están por defecto.

Avance 1

5. Podrás visualizar que se crea un archivo **package.json**.
   a) Dentro de este archivo, modifica la línea 5, en la que aparece la propiedad **main** y el valor **index.js**. Modifícalo para que la propiedad sea source y en el valor coloca **indx.html.**
   b) Dentro de los **scripts**, elimina la propiedad **text** y coloca como propiedad **start** y como valor coloca parcel **index.html**.
   c) Crea una nueva propiedad **build** con el valor **parcel build index.html**.
6. Instala con npm **parcel V2** con todas sus dependencias.
7. Inicializa el parcel con el comando **npm start**. En tu consola se debe desplegar un mensaje similar al siguiente:

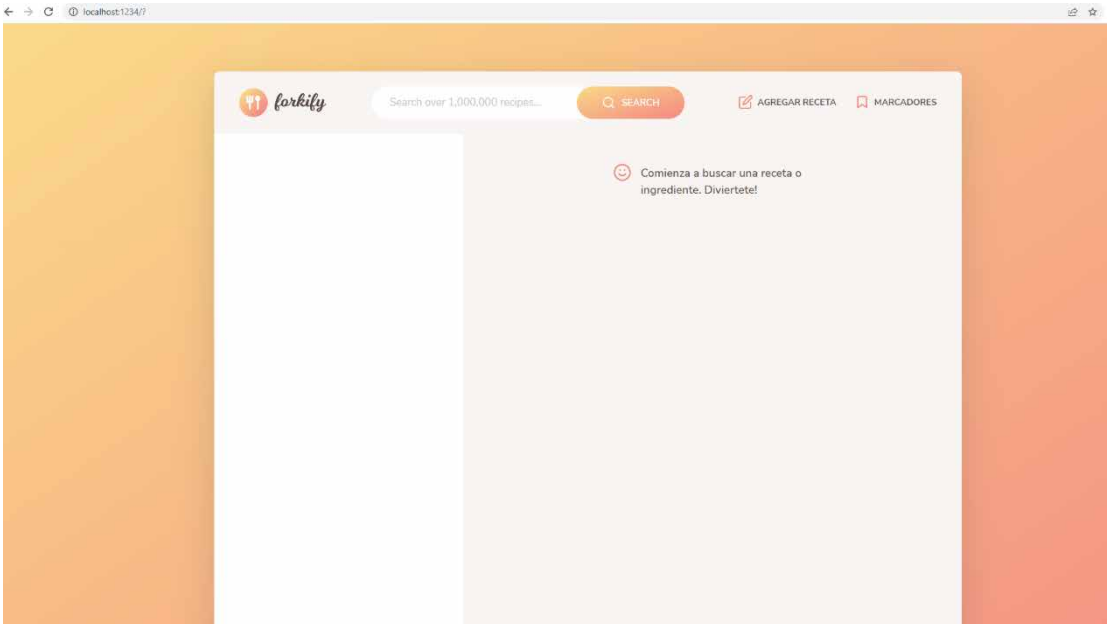
```
root@cdctecmi/#npm start

> forkify@1.0.0 start
> parcel index.html

Server running at http://localhost:1234
✨ Built in 13ms
```

8. Cuando ejecutas esta instrucción, observa que también deben aparecer las siguientes líneas en tu archivo json:

```
"devDependencies": {
   "@parcel/transformer-sass": "^2.8.0",
   "parcel": "^2.8.0"
 }
```

9. En caso de que no aparezcan, analiza que no se te haya presentado ningún error en la terminal.
10. Valida también que hayan aparecido en tu explorador de archivos los directorios **node_modules y dist**.
11. Valida que en la carpeta **dist** se encuentren los archivos listos para ser enviados al navegador.
12. En el archivo **index.html**, en la instrucción que manda a llamar al controller.js, coloca en la referencia que es una llamada al controller de tipo módulo.
13. Valida que tu proyecto esté funcionando, siguiendo el link generado http://localhost:1234.
14. Por ahora tu aplicación debe tener el siguiente aspecto:

Avance 1



15. Para validar que puedas obtener información de la API, realiza la siguiente prueba:
    a. Crea una función asíncrona llamada showRecipe. En el try, crea una constante resp, iguálala a un await para que espere el resultado de la función de búsqueda (fetch), la cual devolverá una promesa, y pásale como parámetro la siguiente URL válida para la API: https://forkify-api.herokuapp.com/api/v2/recipes/5ed6604591c37cdc054bc886.
    b. Una vez que tengas el resultado, es necesario convertirlo a JSON. Para ello, declara una constante data, que será igual a resp, utilizando el método json() (no olvides utilizar el await).
    c. Envía a la consola las constantes resp y data.
    d. En caso de error, vas a enviar en una alerta el error generado.
    e. Invoca a la función showRecipe.
    f. ¿Cuál es el contenido de resp?
    g. ¿Cuál es el contenido de data?
    h. ¿Qué sucede si le pasas esta URL a la función?
    https://forkify-api.herokuapp.com/api/v2/recipes/5ed6604591c37cdc054bc886zzz
    i. ¿Es la misma respuesta?
    j. Para poder visualizar los datos que se necesitan desplegar en la pantalla, crea la variable recipe del tipo objeto e iguálala a data.data.

Avance 1

k. Ahora desestructura recipe de la siguiente manera:

```
recipe = {
id: recipe.id,
title: recipe.title,
publisher: recipe.publisher,
sourceUrl: recipe.source_url,
image: recipe.image_url,
servings: recipe.servings,
cookTime: recipe.cooking_time,
ingredients: recipe.ingredients,
};
```

l. Imprime en la consola el contenido.

16. Como puedes notar, no se visualiza el contenido de la consulta en la página. Para hacerlo, ve al archivo **index.html** y copia desde la línea 161 a 253 al archivo **controller.js** dentro de la función en la variable **markup**. Utiliza **template string**.

17. Sustituye los valores fijos (datos duros) por los valores que contienen el objeto recipe, por ejemplo, el source de las imágenes en la línea:

```
<img src="src/img/test-1.jpg" alt="Tomato" class="recipe__img" />
sustitúyelo por:
<img src="${recipe.image}" alt="Tomato" class="recipe__img" />

La línea:
<span>Pasta with tomato cream sauce</span>
Sustitúyela por:
<span>${recipe.title}</span>

La línea:
<span class="recipe__info-data recipe__info-data--minutes">45</span>
Sustitúyela por:
<span class="recipe__info-data recipe__info-data--minutes">${recipe.cookTime}</span>

La línea:
<span class="recipe__info-data recipe__info-data--people">4</span>
Sustitúyela por:
<span class="recipe__info-data recipe__info-data--people">${recipe.servings}</span>

La línea:
    <span class="recipe__publisher">The Pioneer Woman</span>. Please check out
    Sustitúyela por:
<span class="recipe__publisher">${recipe.publisher}</span>. Please check out

La línea:
href="http://thepioneerwoman.com/cooking/pasta-with-tomato-cream-sauce/"
Sustitúyela por:
href="${recipe. sourceUrl}"
```

Avance 1

18. Como puedes observar, aún se visualiza el mensaje inicial. Para quitarlo, utiliza el método innerHTML sobre recipeContainer y asígnale una cadena vacía.
19. Para visualizar el nuevo contenido en la página, utiliza el método insertAdjacentHTML en el recipeContainer y pásale como parámetros afterbegin y la variable markup.
20. Para visualizar los ingredientes:
    a. Recorre el arreglo **recipe.ingredients** y aplícale la función map para que se puedan visualizar cada uno de los elementos. Pásale como argumentos al **map** la función flecha img que envuelve al elemento de la lista con clase **recipe_ingredient**.
    b. Como el resultado se va a mostrar en el documento html, es necesario aplicar la función **join**.
    c. Tu código debería quedar similar al siguiente:

```
${recipe.ingredients
            .map(ing => {
            return `
            <li class="recipe__ingredient">
              <svg class="recipe__icon">
                <use href="src/img/icons.svg#icon-check"></use>
              </svg>
              <div class="recipe__quantity">${ing.quantity}</div>
              <div class="recipe__description">
                <span class="recipe__unit">${ing.unit}</span>
                ${ing.description}
              </div>
            </li>
            `;
          }).join('')}
          </div>
```

    d) Elimina todos los elementos con clase **recipe__ingredient** del documento.
21. Como puedes notar, no se visualizan correctamente los íconos. En la parte superior del archivo **controller.js**, importa los íconos desde el archivo **../img/icons.svg**.
22. Sustituye en todos los lugares que aparezca **src/img/icons.cvg** por **${icons}**.
23. En la página terminada se ve un spinner que gira en lo que se carga la imagen de una receta. Para crearlo, toma el div con la clase **spinner** del archivo **index.html** y llévalo al archivo **controller.js:**
    a. Crea una función que se llame **renderSpinner** que recibirá un elemento padre (**parentEl**). Dentro del cuerpo de la función, crea la variable markup y asígnale el último código copiado (aplica el punto 22).
    b. Utiliza el método **insertAdjacentHTML** al parámetro **parentEl**. Pasa como parámetro **'afterbegin'** y la variable markup.
    c. Antes de esta última línea, borra el contenido del elemento padre con innerHTML, asignándole una cadena vacía.
    d. Llama a la función **renderSpinner** desde el cuerpo de la función **showRecipe**. Se recomienda que lo pongas inmediatamente después del try y pásale como parámetro **recipeContainer**.

Avance 1

Consideraciones

Instructor:
En esta semana es importante validar que el aprendedor sepa utilizar NPM y la terminal de Visual Studio Code. Además, es esencial que sepa analizar el contenido de los archivos y modificar el archivo JSON.
Se sugiere que se fomente el uso de GIT y que el aprendedor pueda manipularlo.

Aprendedor:
Asegúrate de tener NPM en tu computadora para que puedas realizar las actividades del proyecto.
Una vez que tengas todos tus cambios realizados, recuerda subirlos a tu repositorio GIT.