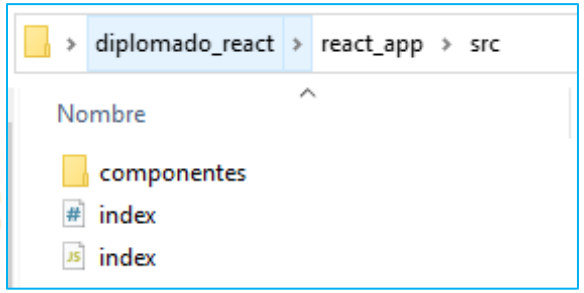
Estilos tradicionales
De CSS dentro de React

Para darle estilos CSS a nuestra aplicación lo que se recomienda es crear un archivo .CSS por cada uno de los componentes que haya en la aplicación, teniendo en cuenta esto, vamos a crear un archivo llamado “index.css" dentro de nuestra carpeta “src"



Luego de esto debemos importar los estilos dentro cada archivo en el que deseemos tenerlos utilizando la declaración "import". En este ejemplo, desde nuestro archivo de “index.js" podemos importar nuestros estilos de la siguiente manera:

```
1   import React, {useState} from 'react';
2   import ReactDOM from 'react-dom/client';
3   import Aula from './componentes/Aula';
4   import './index.css';
```

Estilos tradicionales
De CSS dentro de React

Ahora agregaremos estos estilos a nuestro archivo .css para poder verlos en acción:

```css
* {
    padding: 0;
    margin: 0;
}
body {
    font-size: 15px;
    font-family: Verdana, Arial, Helvetica, sans-serif;
}
#root {
    display: flex;
    align-items: center;
    justify-content: center;
}
.contenedor {
    width: 60%;
    padding: 50px;
    margin: 50px;
    border-radius: 10px;
    -webkit-box-shadow: 0px 15px 10px 5px rgba(0,0,0,0.75);
    -moz-box-shadow: 0px 15px 10px 5px rgba(0,0,0,0.75);
    box-shadow: 0px 15px 10px 5px rgba(0,0,0,0.75);
}
```

Estilos tradicionales
De CSS dentro de React

Y seguidamente, dentro del "index.js" cambiaremos nuestro fragmento por un para poder añadirle una clase utilizando el atributo "className" en lugar de "class" ya que "class" es una palabra reservada en JavaScript y no puedes utilizarla como un atributo en JSX:

```
9   <div className="contenedor">
10    {sesion === true ?
11    <>
12      <Aula/>
13      <button onClick={() => cambiarEstadoSesion(false)}>Cerrar Sesión</button>
14    </>
15    :
16    <>
17      <h1>No has iniciado sesión</h1>
18      <button onClick={() => cambiarEstadoSesion(true)}>Iniciar Sesión</button>
19    </>
20    }
21  </div>
```

Estilos tradicionales
De CSS dentro de React

Cómo resultado final, tendremos en pantalla lo siguiente:

Estilos con Módulos De CSS dentro de React

Los módulos de CSS en React son una forma de modularizar y encapsular estilos dentro de componentes individuales. Con los módulos de CSS, puedes definir estilos específicos para un componente y asegurarte de que no se apliquen a otros componentes.

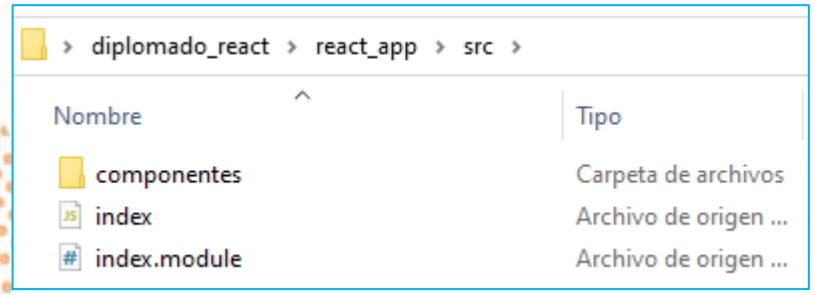
Cuando utilizas módulos de CSS, cada clase de estilo se asigna automáticamente a un nombre único, lo que evita que se produzcan conflictos de nombres de clases. Puedes utilizar estas clases de estilo únicas en tu componente para aplicar estilos específicos.

Para usar estos módulos, es necesario tener en cuenta lo siguiente:

1. Asegúrate de que tu proyecto de React esté configurado para admitir módulos de CSS. Esto se hace automáticamente si has creado tu proyecto mediante `create-react-app`.

Estilos con Módulos
De CSS dentro de React

2. Nombra a tu archivo CSS utilizando la extensión `.module.css` en lugar de `.css`. Por ejemplo, en lugar de `estilos.css`, utiliza `estilos.module.css`:



3. En tu archivo de React, importa el archivo de estilo utilizando una importación de JavaScript:

```
1  import React, {useState} from 'react';
2  import ReactDOM from 'react-dom/client';
3  import Aula from './componentes/Aula';
4  import styles from './index.module.css';
```

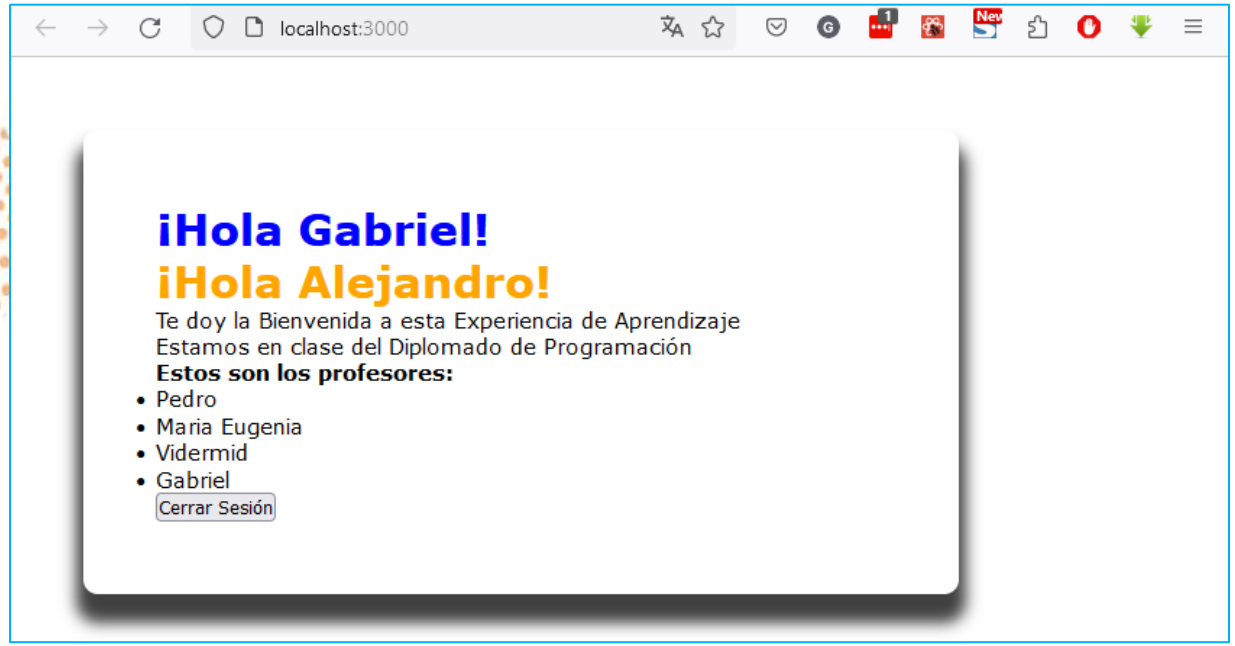
Estilos con Módulos
De CSS dentro de React

4. Dentro de tu componente, utiliza las clases de estilo exportadas como propiedades del objeto `styles`:

```
9     <div className={styles.contenedor}>
10      {sesion === true ?
11      <>
12        <Aula/>
13        <button onClick={() => cambiarEstadoSesion(false)}>Cerrar Sesión</button>
14      </>
15      :
16      <>
17        <h1>No has iniciado sesión</h1>
18        <button onClick={() => cambiarEstadoSesion(true)}>Iniciar Sesión</button>
19      </>
20      }
21    </div>
```

Estilos con Módulos
De CSS dentro de React

De esta forma ya estamos aplicando estilos a través de Módulos de CSS, pero en el caso de este ejemplo, nuestro ID "root" perdió los estilos debido a que ese ID no se encuentra en nuestro "index.js" si no en el "index.html", por ello no pudimos aplicarle los estilos de módulo:



Por lo cuál en este caso puntual, lo ideal sería trabajar con los estilos tradicionales que aprendimos anteriormente

Estilos en React con StyledComponents

Los Styled Components son una biblioteca de React que te permite escribir CSS en tus componentes utilizando sintaxis de JavaScript.

Esto hace que tus estilos estén más cerca de tus componentes, lo que facilita el mantenimiento y la reutilización del código.

Los estilos definidos con Styled Components se generan como clases CSS únicas y se insertan en el `<head>` del documento una vez que el componente se renderiza. Esto garantiza que los estilos sean aplicados de manera aislada y no interfieran con otros componentes.

Estilos en React con StyledComponents

A continuación veremos cómo usar StyledComponents:

1. Primero, debes instalar los Styled Components en tu proyecto de React. Puedes hacerlo ejecutando el siguiente comando en tu terminal:

   - npm install styled-components

   En caso de que la consola esté ejecutando la aplicación, debes primero presionar los comandos Ctrl + C para finalizar esa acción y luego si poder ejecutar la instalación de Styled Components

2. Importa el módulo `styled` de los Styled Components en el archivo donde quieras usarlos. En nuestro caso lo haremos dentro del componente "aula.js" de la siguiente manera:

```
3   import styled from 'styled-components';
```

Estilos en React con StyledComponents

3. Crea tus estilos personalizados utilizando la sintaxis de los Styled Components. Puedes crear un componente estilizado utilizando la función `styled` seguida del nombre del elemento HTML que deseas estilizar. En este caso crearemos un Styled Component para los párrafos dentro del archivo del componente de Aula:

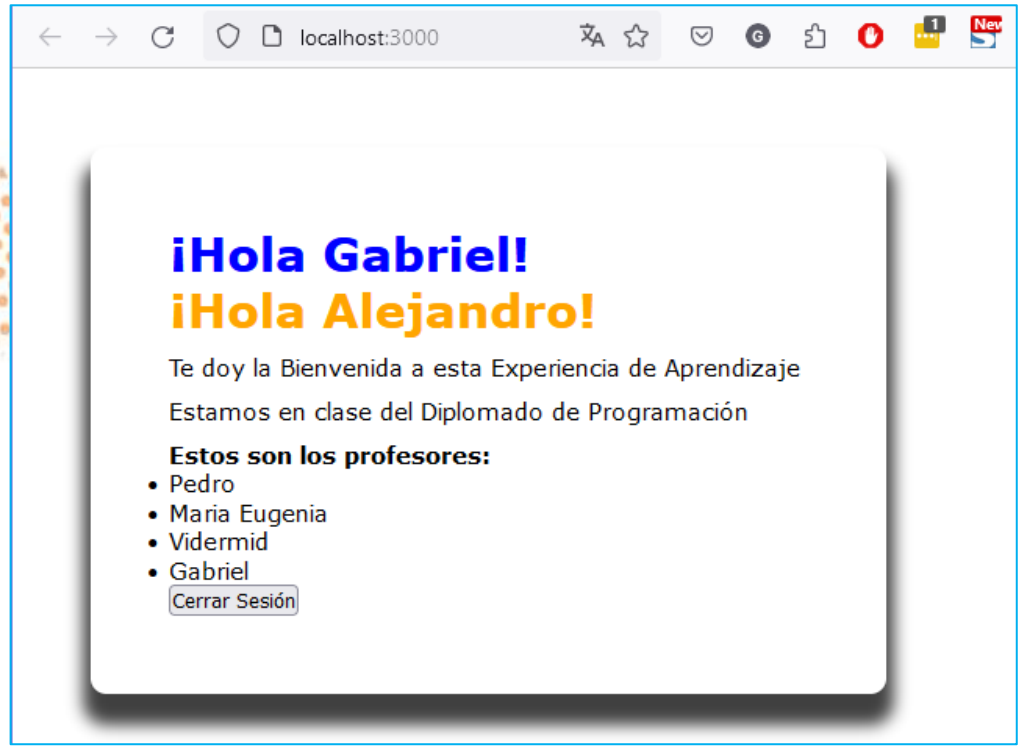
```
25  const Parrafo = styled.p`
26    margin: 10px 0;
27  `;
28
29  export default Aula;
```

4. Una vez que hayas creado tu componente estilizado, puedes usarlo en tu aplicación de React como cualquier otro componente. Simplemente llámalo como si fuera un componente normal, y se aplicarán los estilos definidos. En este caso, lo haremos así en lugar de la etiqueta tradicional de los párrafos <p>:

```
12      <Parrafo>Te doy la Bienvenida a esta Experiencia de Aprendizaje</Parrafo>
13      {clase && <Parrafo>Estamos en clase del {clase}</Parrafo>}
```

Estilos en React con StyledComponents

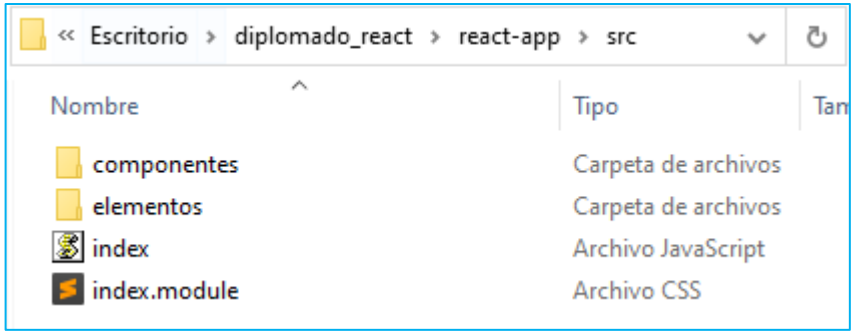
De esta forma ya hemos aplicado estilos con StyledComponents y el resultado final es el siguiente:



StyledComponents desde archivos separados

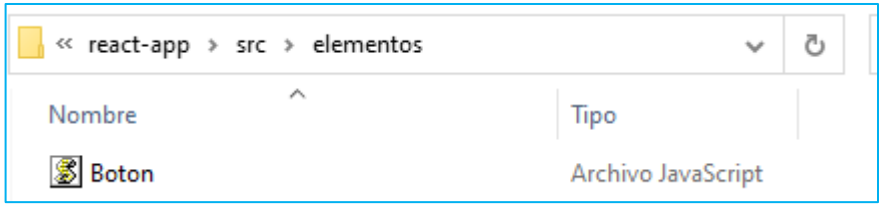
Crear Componentes de Estilos en archivos separados también es una posibilidad y nos da cómo ventaja el hecho de poder usarlos en diferentes secciones de nuestra app, no solamente dentro del archivo dónde se crearon.

Para aprender cómo hacerlo de esta forma, haremos un componente de estilos de botón y para ello, lo primero que haremos es crear una nueva carpeta dentro de “src” llamada “elementos”:



StyledComponents desde archivos separados

Seguidamente dentro de la carpeta de elementos crearemos el archivo “Boton.js”:



Luego en el archivo “Boton.js" vamos a importar StyledComponents en la línea 1, colocamos nuestro componente de estilos y luego lo exportamos:

```
1   import styled from "styled-components";
2
3   const Boton = styled.button`
4       color: #FFFFFF;
5       background: #FF6D00;
6       display: inline-block;
7       margin-top: 10px;
8       padding: 10px;
9       border: none;
10      border-radius: 5px;
11      font-weight: bold;
12      font-family: Verdana, Arial, Helvetica, sans-serif;
13      cursor: pointer;
14  `;
15
16  export default Boton;
```

StyledComponents desde archivos separados

Ahora debemos importar nuestro botón dentro del archivo que queremos usarlo, en este caso lo haremos dentro del "index.js", es como importar un componente:

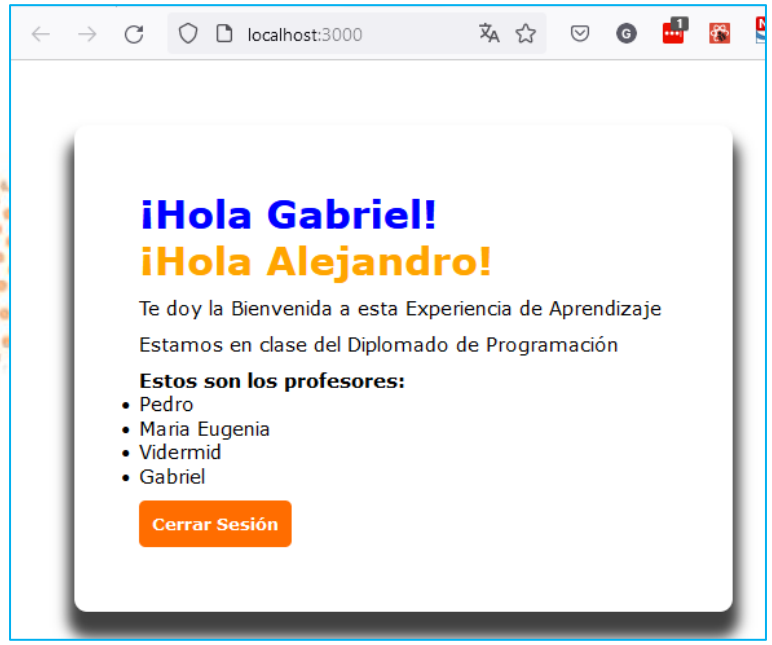
```
1   import React, {useState} from 'react';
2   import ReactDOM from 'react-dom/client';
3   import Aula from './componentes/Aula';
4   import styles from './index.module.css';
5   import Boton from './elementos/Boton';
```

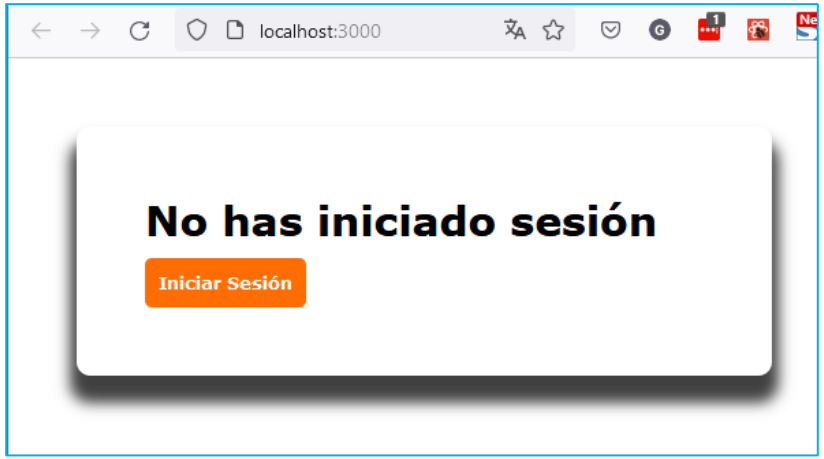
Para continuar, debemos reemplazar los botones <button> tradicionales de HTML por nuestro componente <Boton>:

```
12      <>
13        <Aula/>
14        <Boton onClick={() => cambiarEstadoSesion(false)}>Cerrar Sesión</Boton>
15      </>
16      :
17      <>
18        <h1>No has iniciado sesión</h1>
19        <Boton onClick={() => cambiarEstadoSesion(true)}>Iniciar Sesión</Boton>
20      </>
```

StyledComponents desde archivos separados

De esta forma, ya tendremos en pantalla nuestros botones estilizados gracias a StyledComponents:



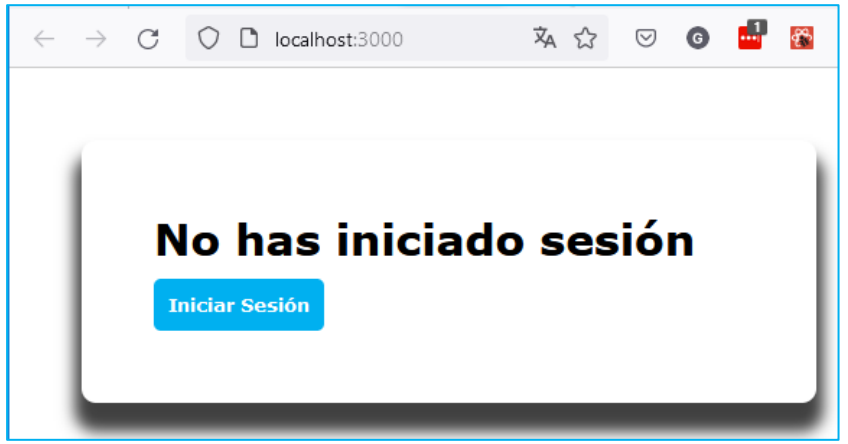


Sin embargo, aún podemos agregar más estilos y para eso vamos a aprender como usar las pseudoclases y pseudoelementos de estilos con StyledComponents

StyledComponents desde archivos separados

Para ello agregaremos un efecto al pasar el cursor del mouse en nuestro botón y para hacerlo debemos ingresar a nuestro componente de estilos del botón y dentro del mismo colocar un ampersand (&) seguido de 2 puntos (:) y el pseudoelemento/pseudoclase que necesitamos, que en este caso será hover:

```
13      cursor: pointer;
14
15      &:hover{
16          color: #FFFFFF;
17          background: #00B0F0;
18      }
19  `;
20
21  export default Boton;
```



De esta forma ya nuestro botón tiene los efectos deseados gracias a StyledComponents.

Propiedades de StyledComponents

Las propiedades de los StyledComponents en React son argumentos que puedes pasar a un componente personalizado creado con StyledComponents. Estas propiedades se utilizan para ajustar y personalizar el estilo del componente. Puedes definir tus propias propiedades y luego acceder a ellas dentro del styled-component para aplicar estilos dinámicos basados en esas propiedades.

Por ejemplo, supongamos que tienes un componente personalizado llamado `Button` creado con StyledComponents. Puedes definir propiedades como `color`, `size`, `background`, etc., y luego usar esas propiedades dentro del styled-component para aplicar diferentes estilos según los valores de las propiedades proporcionadas.

Propiedades de StyledComponents

Las propiedades de los StyledComponents en React son argumentos que puedes pasar a un componente personalizado creado con StyledComponents. Estas propiedades se utilizan para ajustar y personalizar el estilo del componente. Puedes definir tus propias propiedades y luego acceder a ellas dentro del styled-component para aplicar estilos dinámicos basados en esas propiedades.

Por ejemplo, supongamos que tienes un componente personalizado llamado `Button` creado con StyledComponents. Puedes definir propiedades como `color`, `size`, `background`, etc., y luego usar esas propiedades dentro del styled-component para aplicar diferentes estilos según los valores de las propiedades proporcionadas.

Propiedades de StyledComponents

Para implementarlas, vamos a colocar en nuestros botones propiedades diferentes. En el index.js dentro del botón de cierre de sesión colocaremos la propiedad “rojo”:

```
14          <Boton rojo onClick={() => cambiarEstadoSesion(false)}>Cerrar Sesión</Boton>
```

Luego dentro de nuestro componente de estilos “Boton.js”, agregaremos la importación de css a la importación de styled:

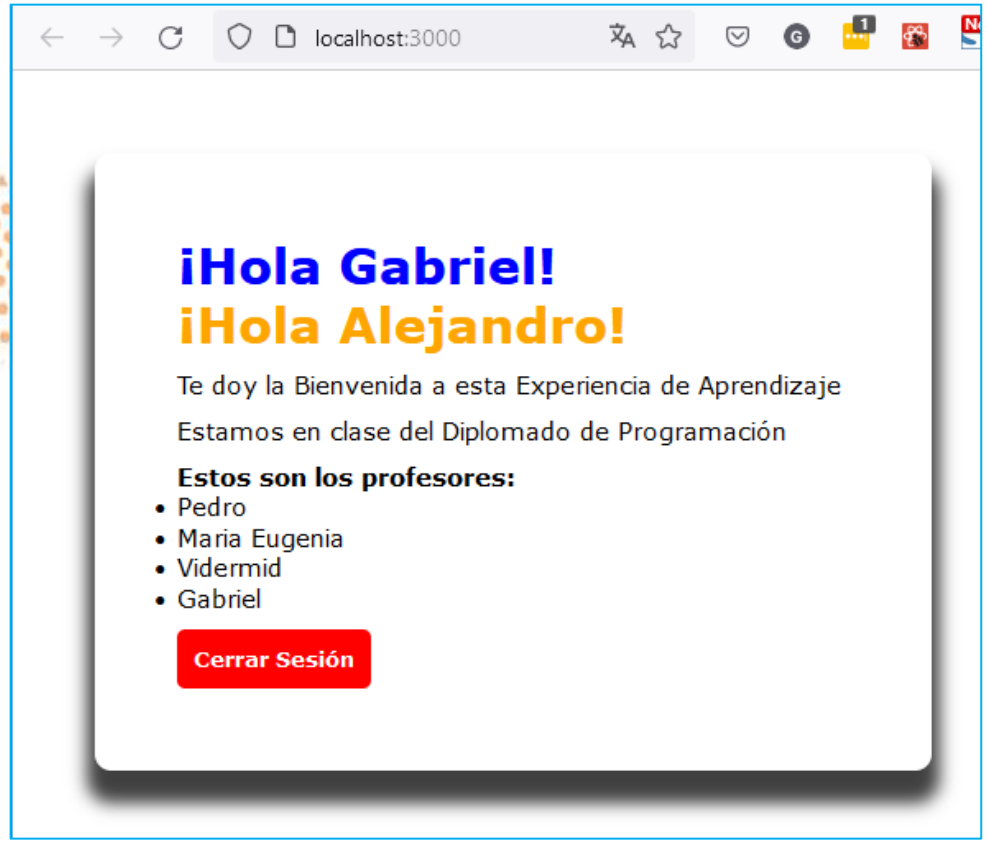
```
1   import styled, {css} from "styled-components";
```

Seguidamente, dentro de los estilos del componente, accederemos a las propiedades del mismo, especificamos a cuál propiedad queremos acceder, que en este caso será “rojo” y colocamos color de fondo rojo y color de texto blanco en caso de que la misma exista:

```
20      ${props => props.rojo && css`
21          color: #FFFFFF;
22          background: red;
23      `}
24  `;
25
26  export default Boton;
```

Propiedades de StyledComponents

De esta forma obtenemos como resultado, que nuestro botón de cierre de sesión, ahora es de color rojo:



Propiedades de StyledComponents

Ahora en el index.js dentro del botón de inicio de sesión colocaremos la propiedad "verde" y la propiedad "anchocompleto":

```
19    <Boton verde anchocompleto onClick={() => cambiarEstadoSesion(true)}>Iniciar Sesión</Boton>
```

Seguidamente, dentro de los estilos del componente, colocaremos un width dinámico, por lo cuál, su valor dependerá de si existe la propiedad "anchocompleto" o no. En caso de existir, el valor del width será de "100%", de lo contrario, su valor será automático:

```
13    cursor: pointer;
14    width: ${props => props.anchocompleto ? "100%" : "auto"};
```

Adicionalmente, dentro de los estilos del componente, accederemos a las propiedades del mismo, especificamos a cuál propiedad queremos acceder, que en este caso será "verde" y colocamos color de fondo verde y color de texto blanco en caso de que la misma exista:

```
26    ${props => props.verde && css`
27        color: #FFFFFF;
28        background: green;
29    `}
```

Propiedades de StyledComponents

De esta forma obtenemos como resultado, que nuestro botón de inicio de sesión, ahora es de color verde y abarca el ancho completo del elemento que lo contiene: