



Universidad Mariano Galvez de Guatemala

Nombre: Lesly Maribel Garcia Gonzalez

Carnet: 9941-16-15599

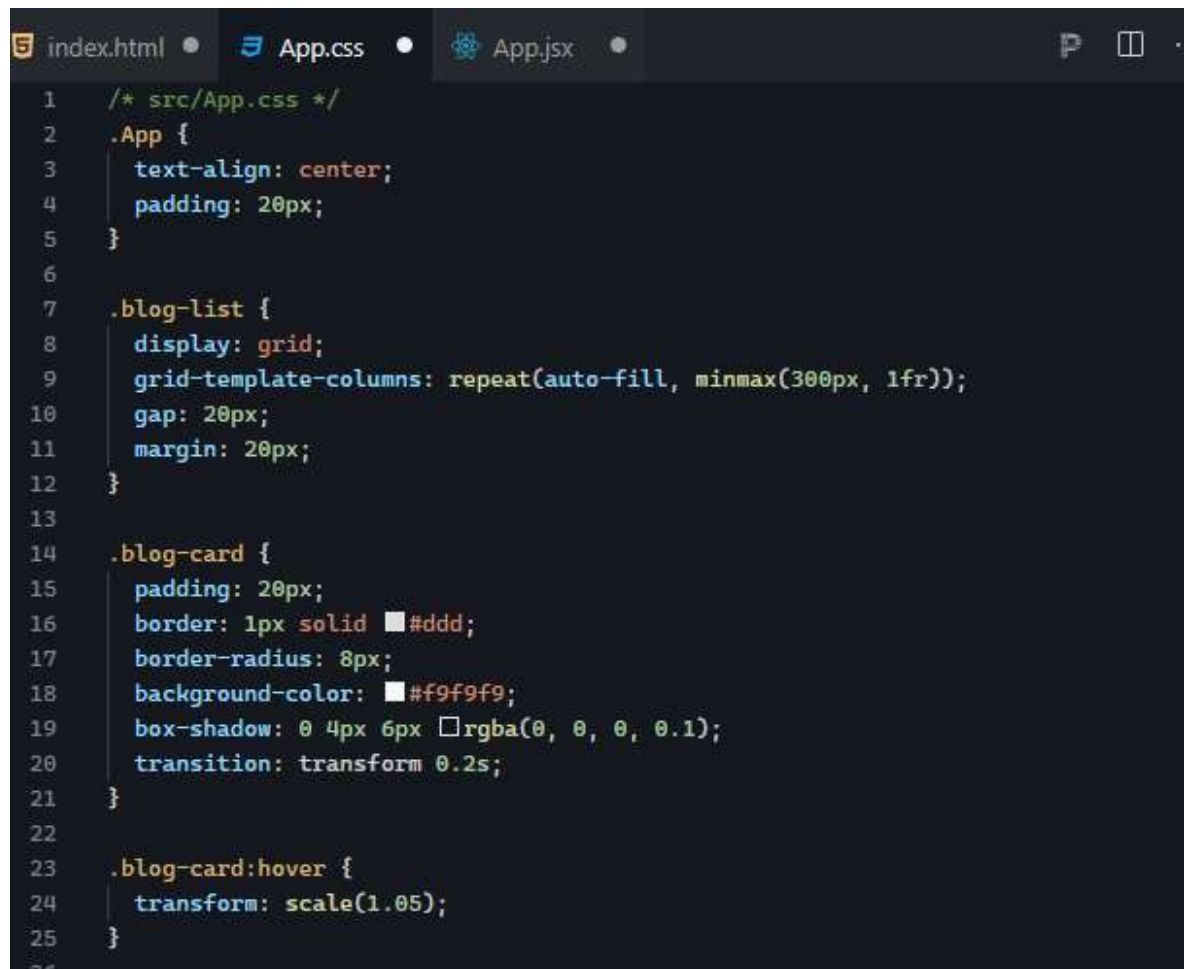
Segundo Parcial - II Serie

Resumen de lo Realizado:

1. Configuración inicial: Se inició un proyecto de React utilizando Vite como entorno de desarrollo.
2. Consumo de la API: Usé el hook `useEffect` para hacer una petición con `fetch` al endpoint <https://api.vercel.app/blog> y así obtener los datos del blog.
3. Componentes: La aplicación se estructuró en dos componentes principales, `BlogCard` y `BlogList`, para gestionar el contenido de los blogs.
4. Estilo: Se aplicaron estilos CSS para dar formato y mejorar el aspecto visual de la lista de blogs, para lograr un diseño atractivo.
5. Funcionamiento final: La aplicación muestra los títulos, descripciones y enlaces de las entradas del blog obtenidas desde el endpoint, permitiendo al usuario acceder a más detalles con un clic. Este ejercicio muestra como consumir APIs en React y estructurar una aplicación simple con componentes reutilizables y estilización básica.

Estilos CSS

Se añadió estilos CSS para que el diseño coincida con el que se solicitó.

A screenshot of a code editor with three tabs: 'index.html', 'App.css', and 'App.jsx'. The 'App.css' tab is active, showing CSS code. The code defines styles for a root element '.App', a grid container '.blog-list', individual cards '.blog-card', and a hover state for the cards. The '.App' selector sets text-align to center and padding to 20px. The '.blog-list' selector uses a grid layout with columns that repeat based on available space, with a gap of 20px and a margin of 20px. The '.blog-card' selector has a padding of 20px, a 1px solid #ddd border, an 8px border-radius, a #f9f9f9 background color, a box-shadow, and a 0.2s transition for transform. The ':hover' state for '.blog-card' scales the element by 1.05.

```
1  /* src/App.css */
2  .App {
3    text-align: center;
4    padding: 20px;
5  }
6
7  .blog-list {
8    display: grid;
9    grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
10   gap: 20px;
11   margin: 20px;
12 }
13
14 .blog-card {
15   padding: 20px;
16   border: 1px solid #ddd;
17   border-radius: 8px;
18   background-color: #f9f9f9;
19   box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
20   transition: transform 0.2s;
21 }
22
23 .blog-card:hover {
24   transform: scale(1.05);
25 }
26
```

```

27 .blog-card h2 {
28   font-size: 1.5rem;
29   margin-bottom: 10px;
30 }
31
32 .blog-card p {
33   font-size: 1rem;
34   color: #555;
35 }
36
37 .blog-card a {
38   display: inline-block;
39   margin-top: 10px;
40   padding: 10px;
41   background-color: #007bff;
42   color: #fff;
43   text-decoration: none;
44   border-radius: 5px;
45 }
46
47 .blog-card a:hover {
48   background-color: #0056b3;
49 }
50
51

```

Consumir la API

Para consumir el enlace <https://api.vercel.app/blog>, usamos el hook **useEffect** en React para hacer la petición y **useState** para manejar los datos.

```

index.html • App.css • App.jsx •
1 // src/App.jsx
2 import React, { useEffect, useState } from 'react';
3 import BlogList from '../components/BlogList';
4
5 function App() {
6   const [blogs, setBlogs] = useState([]);
7
8   useEffect(() => {
9     fetch('https://api.vercel.app/blogLinks')
10      .then((response) => response.json())
11      .then((data) => setBlogs(data))
12      .catch((error) => console.error('Error fetching data:', error));
13   }, []);
14
15   return (
16     <div className="App">
17       <h1>Blog Posts</h1>
18       <BlogList blogs={blogs} />
19     </div>
20   );
21 }
22
23 export default App;
24

```

Crear los Componentes de Blog:

- **BlogCard**: Este componente mostrará la información de un solo blog post.
- **BlogList**: Este componente renderiza una lista de BlogCard utilizando los datos traídos desde la API.

```
index.html • App.css • App.jsx •  
25 // src/components/BlogCard.jsx  
26 import React from 'react';  
27  
28 const BlogCard = ({ title, description, link }) => {  
29   return (  
30     <div className="blog-card">  
31       <h2>{title}</h2>  
32       <p>{description}</p>  
33       <a href={link} target="_blank" rel="noopener noreferrer">Read More</a>  
34     </div>  
35   );  
36 };  
37  
38 export default BlogCard;  
39 // src/components/BlogList.jsx  
40 import React from 'react';  
41 import BlogCard from './BlogCard';  
42  
43 const BlogList = ({ blogs }) => {  
44   return (  
45     <div className="blog-list">  
46       {blogs.length > 0 ? (  
47         blogs.map((blog) => (  
48           <BlogCard  
49             key={blog.id}  
50             title={blog.title}
```

```
51         description={blog.description}
52         link={blog.link}
53     />
54     ))
55     ) : (
56     <p>No blogs available</p>
57     )}
58 </div>
59 );
60 };
61
62 export default BlogList;
63 import './App.css';
64
```

Terminal

```
11:35:36 [vite] page reload index.html (x5)
11:35:36 [vite] page reload index.html (x6)
█
```

Ejecutar el Proyecto

Una vez que todo está implementado, puedes ejecutar el proyecto localmente usando:

```
index.html • App.css • App.jsx
```

```
1  npm create vite@latest my-blog-app --template react
2  cd my-blog-app
3  npm install
4  npm run dev
5
6
```