



Universidad Juárez del Estado de Durango

Facultad de Ingeniería Ciencias y Arquitectura

Ingeniería en Tecnologías Computacionales

Programación Web

Reporte de Practica 21

Diego Rea Morales

Ing. Fabian Gallegos Gutiérrez

24 de Octubre del 2025

Índice

Contenido_____3

Conclusiones_____6

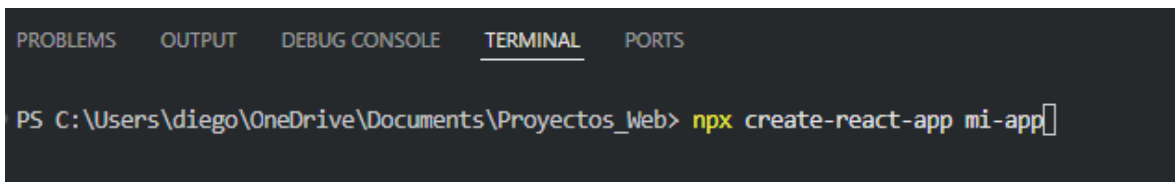
Contenido

La práctica numero 21 tuvo como objetivo, el poder instalar NodeJS y poder crear un proyecto en ReactJS.

NodeJs es un entorno de ejecución de JavaScript, esto nos permite ejecutar código JavaScript del lado del servidor, sin la necesidad de ejecutarlo en el navegador.

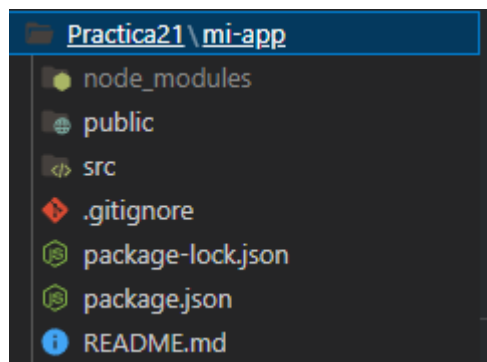
ReactJS es una biblioteca de JavaScript de código abierto para construir interfaces de usuario (UI) interactivas y eficientes, principalmente para aplicaciones web y móviles.

Para poder empezar a crear un proyecto en React, en la terminal de Visual Code ingresamos este comando *npx create-react-app mi-app*.

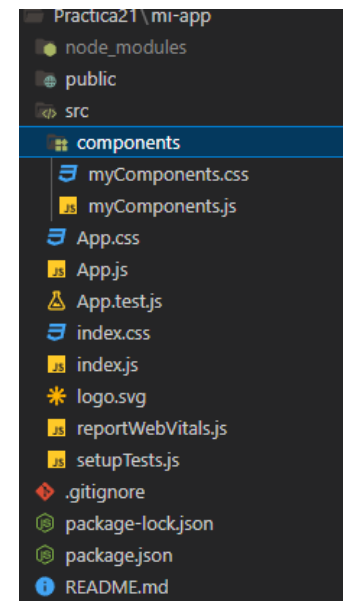


```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\diego\OneDrive\Documents\Proyectos_Web> npx create-react-app mi-app
```

Después de que ya haya instalado todos los paquetes, dentro de la carpeta “src” creamos otra carpeta llamada *components*.



Creamos dos archivos, del mismo nombre *myComponents*, uno con extensión .js y el otro con .css.



En estos archivos pegaremos el contenido de la diapositiva.

```
myComponents.js X
Practica21 > mi-app > src > components > myComponents.js > ...
1 import React, { useState } from "react"; // Importa React y el hook useState desde React.
2 import "../myComponents.css"; // Importa un archivo CSS para estilizar el componente.
3 export default MyComponent; // Exporta el componente para su uso en otras partes de la aplicación.
4 function MyComponent() {
5   const [imageUrl, setImageUrl] = useState(
6     "https://images.pexels.com/photos/2234685/pexels-photo-2234685.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dp=1"
7   ); // Inicializa un estado para la URL de la imagen.
8   const [hue, setHue] = useState(0); // Inicializa un estado para el valor de rotación de matiz (hue).
9   const [opacity, setOpacity] = useState(100); // Inicializa un estado para el valor de opacidad.
10  const [blur, setBlur] = useState(0); // Inicializa un estado para el valor de desenfoque (blur).
11  const [brightness, setBrightness] = useState(100); // Inicializa un estado para el valor de brillo.
12  const [sepia, setSepia] = useState(0); // Inicializa un estado para el valor de sepia.
13  const handleImageChange = () => {
14    // Obtiene el elemento de imagen por su ID.
15    // Aplica un filtro CSS en función de los valores de estado actuales.
16    const image = document.getElementById("color-change-image");
17    image.style.filter = `hue-rotate(${hue}deg) opacity(${opacity}%) blur(${blur}px) brightness(${brightness}%) sepia(${sepia}%)`;
18  };
19  return (
20    <div>
21      <h1>Image Filter App</h1>
22      <input
23        type="text"
24        placeholder="Image URL"
25        value={imageUrl}
26        onChange={(e) => setImageUrl(e.target.value)}
27      />
28      <div>
29        <h2>Image Filter App</h2>
30        <input
31          type="text"
32          placeholder="Image URL"
33          value={imageUrl}
34          onChange={(e) => setImageUrl(e.target.value)}
35        />
36        <button onClick={handleImageChange}>Apply Filters</button>
37      </div>
38      <div>
39        <label>Hue Rotation</label>
40        <input
41          type="range"
42          min="0"
43          max="360"
44          value={hue}
45          onChange={(e) => setHue(e.target.value)}
46        />
47        <label>Opacity</label>
48        <input
49          type="range"
50          min="0"
51          max="100"
52          value={opacity}
53          onChange={(e) => setOpacity(e.target.value)}
54        />
55        <label>Blur</label>
56        <input
57          type="range"
58          min="0"
59          max="10"
60          value={blur}
61          onChange={(e) => setBlur(e.target.value)}
62        />
63        <label>Brightness</label>
64        <input
65          type="range"
66          min="0"
67          max="200"
68          value={brightness}
69          onChange={(e) => setBrightness(e.target.value)}
70        />
71        <label>Sepia</label>
72        <input
73          type="range"
74          min="0"
75          max="100"
76          value={sepia}
77          onChange={(e) => setSepia(e.target.value)}
78        />
79      </div>
80      <img
81        id="color-change-image"
82        src={imageUrl}
83        alt="Image to Apply Filters"
84      />
85    </div>
86  );
87
88 }
```

```
myComponents.js X
Practica21 > mi-app > src > components > myComponents.js > ...
4 function MyComponent() {
43   onChange={(e) => setHue(e.target.value)}
44 }
45 <label>Opacity</label>
46 <input
47   type="range"
48   min="0"
49   max="100"
50   value={opacity}
51   onChange={(e) => setOpacity(e.target.value)}
52 />
53 <label>Blur</label>
54 <input
55   type="range"
56   min="0"
57   max="10"
58   value={blur}
59   onChange={(e) => setBlur(e.target.value)}
60 />
61 <label>Brightness</label>
62 <input
63   type="range"
64   min="0"
65   max="200"
66   value={brightness}
67   onChange={(e) => setBrightness(e.target.value)}
68 />
69 <label>Sepia</label>
70 <input
71   type="range"
72   min="0"
73   max="100"
74   value={sepia}
75   onChange={(e) => setSepia(e.target.value)}
76 />
77 </div>
78 <img
79   id="color-change-image"
80   src={imageUrl}
81   alt="Image to Apply Filters"
82 />
83 </div>
84 </div>
85 </div>
86
87 );
88 }
```

```
myComponents.css X
Practica21 > mi-app > src > components > myComponents.css > ...
1 label {
2   display: block;
3   margin-top: 10px;
4   color: #3B3B3B;
5 }
6 img {
7   max-width: 100%;
8   height: auto;
9   filter: none;
10  border: 1px solid #00FAFF;
11 }
12
13 .App {
14   text-align: center;
15   font-family: "Roboto", sans-serif;
16   background-color: #FEEFDD; /* Dark background */
17   color: #3B3B3B; /* Neon blue text color */
18   padding: 20px;
19 }
20 h1 {
21   color: #3B3B3B;
22   text-transform: uppercase;
23 }
24 input[type="text"] {
25   padding: 10px;
26   font-size: 16px;
27   background-color: #C97C5D;
28   color: #3B3B3B;
29   border: 1px solid #C97C5D;
30   width: 100%;
31   margin-bottom: 10px;
32 }
33 button {
34   padding: 10px 20px;
35   font-size: 16px;
36   background-color: #FFFB8E;
37   color: #3B3B3B;
38   border: none;
39   border-radius: 5px;
40   cursor: pointer;
41   transition: background-color 0.3s;
42 }
```

```
43 button:hover {
44   background-color: #FF6B00; /* Neon orange on hover */
45 }
46 input[type="range"] {
47   width: 100%;
48   height: 6px;
49   appearance: none; /* elimina el estilo por defecto */
50   background: #C97C5D; /* color de la pista */
51   border-radius: 5px;
52   outline: none; /* Neon blue */
53 }
54 input[type="range"]::-webkit-slider-runnable-track {
55   background: #C97C5D; /* color de la pista */
56   height: 6px;
57   border-radius: 5px;
58 }
59 input[type="range"]::-webkit-slider-thumb {
60   appearance: none;
61   width: 20px;
62   height: 20px;
63   background: #3B3B3B;
64   border-radius: 50%;
65   cursor: pointer;
66   margin-top: -7px;
67   transition: background 0.2s;
68 }
69 input[type="range"]::-webkit-slider-thumb:hover {
70   background: #F5C396;
71 }
```

Cuando ya tengamos todo ahora podemos ejecutar el programa para mostrarlo en pantalla, para esto, primero nos posicionamos en la carpeta de mi-app, para ello utilizamos el comando `cd Practica21/mi-app`.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
● PS C:\Users\diego\OneDrive\Documents\Proyectos_Web> cd Practica21/mi-app
○ PS C:\Users\diego\OneDrive\Documents\Proyectos_Web\Practica21\mi-app> |
```

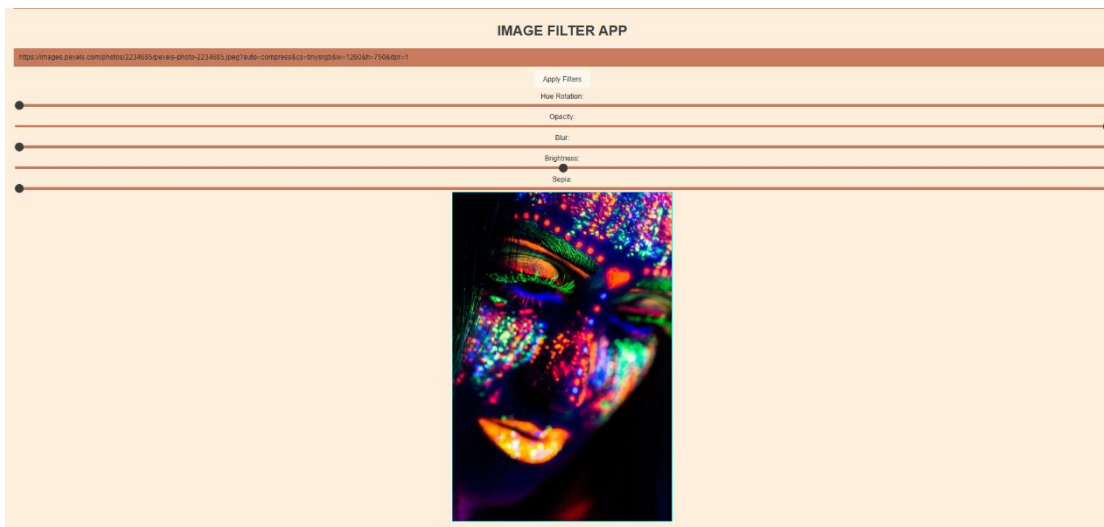
Ya que estemos en la carpeta de mi-app, podemos ejecutar el siguiente comando `npm start dev`. Este comando nos lanzara al entorno de ejecución en un servidor local.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\Users\diego\OneDrive\Documents\Proyectos_Web> cd Practica21/mi-app
PS C:\Users\diego\OneDrive\Documents\Proyectos_Web\Practica21\mi-app> npm start dev

> mi-app@0.1.0 start
> react-scripts start dev

(node:20692) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:20692) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
Starting the development server...
compiled with warnings.
```

Después de que hallamos ejecutado el comando, en pantalla se mostrará la pagina web corriendo.



Conclusiones

La práctica número 21 permitió comprender el proceso de instalación y configuración de NodeJS, así como la creación de un proyecto básico con ReactJS. A través de esta actividad se pudo conocer la estructura de un proyecto en React, la forma en que se organizan los componentes y cómo se ejecuta una aplicación en un servidor local. Además, se reforzó la importancia de NodeJS como entorno necesario para el desarrollo moderno con JavaScript y de ReactJS como una herramienta eficiente para construir interfaces dinámicas e interactivas en aplicaciones web.