**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

| **PROGRAMA DE FORMACIÓN** | Tecnólogo en análisis y desarrollo de software. |
| --- | --- |

| COMPETENCIA | **220501096**  Construcción del software. | RESULTADOS DE APRENDIZAJE | **20501096-04.** Codificar el software de acuerdo con el diseño establecido |
| --- | --- | --- | --- |

| **NÚMERO DEL COMPONENTE FORMATIVO** | CF031. |
| --- | --- |
| **NOMBRE DEL COMPONENTE FORMATIVO** | DESARROLLO DE FRONT-END CON REACT JS |
| **BREVE DESCRIPCIÓN** | El diseño de un sitio web corresponde a un componente fundamental en el proceso de creación de aplicaciones web, este componente está orientado a la comprensión de los conceptos más importantes que se deben tener en cuenta para la construcción del *Front-End* con la tecnología *React* JS. |
| **PALABRAS CLAVE** | Diseño, estilos, *Html*, J*avaScript*, página web, *React* JS. |

| **ÁREA OCUPACIONAL** | 9 - PROCESAMIENTO, FABRICACIÓN Y ENSAMBLE |
| --- | --- |
| **IDIOMA** | Español |

1. **TABLA DE CONTENIDOS:**

**Introducción**

**1. Fundamentos de REACT JS**

1.1 ¿Qué es *React* JS?

1.2 ¿Qué es JSX?

1.3 Instalando *React*

1.4 Crear un proyecto

1.5 Propiedades y estados de los componentes

1.6 Eventos en *React*

1.7 Listas y *keys*

1.8 Formularios

1.9 Concepto de clases en *React*

1.10 Funciones *Hooks*

* 1. Ciclos de vida de los componentes de *React*
  2. Ejecutar la aplicación

1. **DESARROLLO DE CONTENIDOS:**

**Introducción:**

Se da la bienvenida al componente formativo “Desarrollo de *Front-End* con *React* JS”. Para comenzar el recorrido por el mismo, se debe visitar el recurso didáctico que se muestra a continuación:



1. **Fundamentos de React JS**

React JS, es una nueva tecnología importante para la construcción de *software* en su componente *Front-End,* que requiere tener en cuenta algunas consideraciones, se pueden ver a continuación:



**1.1 ¿Qué es React JS?**

Es una biblioteca JavaScript de código abierto, desarrollada por la multinacional *Facebook* para crear interfaces de usuario, es importante tener en cuenta que una biblioteca o librería, es un conjunto de funcionalidades ya construidas y que son re-utilizables siguiendo su sintaxis. Su principal objetivo además de facilitar el desarrollo de aplicaciones de una sola página, es ser sencillo, declarativo y fácil de combinar.

**1.2 ¿Qué es JSX?**

| Fondo Negro Con Captura De Pantalla De Superposición De Texto | Su nombre viene de la composición de *Java Script* (JS) + XML = (JSX) y corresponde a la sintaxis propuesta en *React* JS para escribir elementos de interfaz web compuestos por *Java Script* + HTML, en donde el HTML sigue las normas de código bien formado XML. |
| --- | --- |

**1.3 Instalando React JS**

Las siguientes instrucciones muestran el proceso de instalación para sistemas operativos *Linux* versión *Ubuntu* 20.04.3 LTS y *Windows,* en ambos casos, en primer lugar se necesita instalar el servidor denominado *Node* y el manejador de paquetes npm para esto se puede empezar haciendo clic en el siguiente link [Node >= 14.0.0 and npm >= 5.6](https://nodejs.org/en/) o puede acceder mediante el enlace <https://nodejs.org/en/> a la fecha la versión *Node* recomendada es 14.1.18 LTS la cual presenta soporte por largo tiempo.

A continuación, seguir los siguientes pasos de Instalación sobre *Linux Ubuntu* 20.04.3 LTS.



Una vez se ha comprobado que la versión instalada es v14.18.1 mediante el comando node –version, continuar con la instalación sobre el sistema operativo Windows, siguiendo los siguientes pasos:



**1.4 Crear un proyecto**

Una vez llevado a cabo el proceso de instalación del paso anterior de *node* y npm, se puede crear el proyecto *React* e ir ejecutando los siguientes comandos:

**1.** npx create-react-app my-app

**2.** cd my-app

**3.** npm start

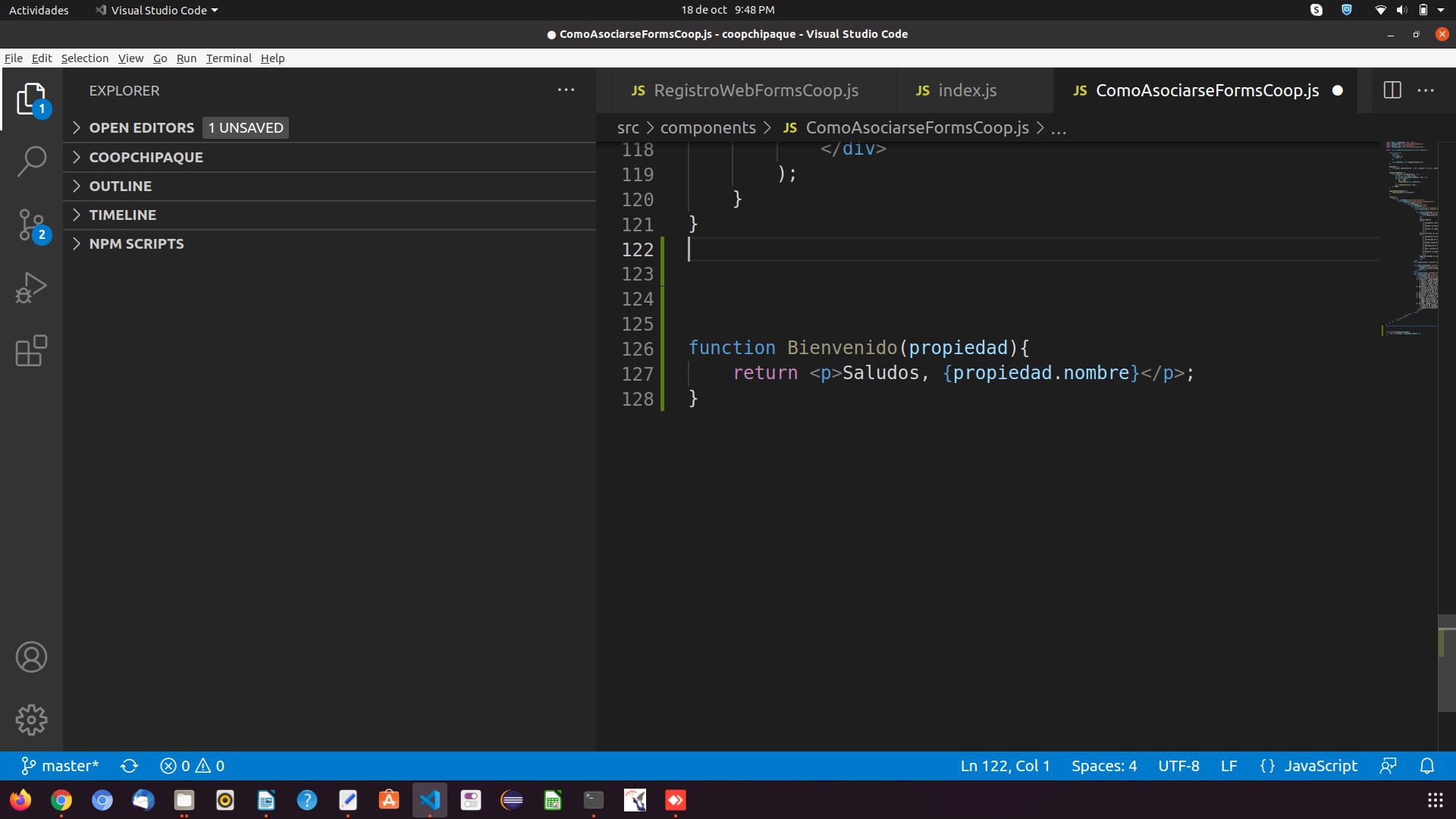
Iniciar a la creación del proyecto, mediante la siguiente secuencia:



**1.5 Propiedades y Estados de los Componentes**

Un componente en *React* hace referencia a la posibilidad de definir de manera independiente una funcionalidad que puede ser re-utilizable dentro del sistema, en esencia estas funcionalidades se escriben a manera de funciones *JavaScript* que reciben unas entradas que se denominan propiedades y devuelven una forma de visualización.



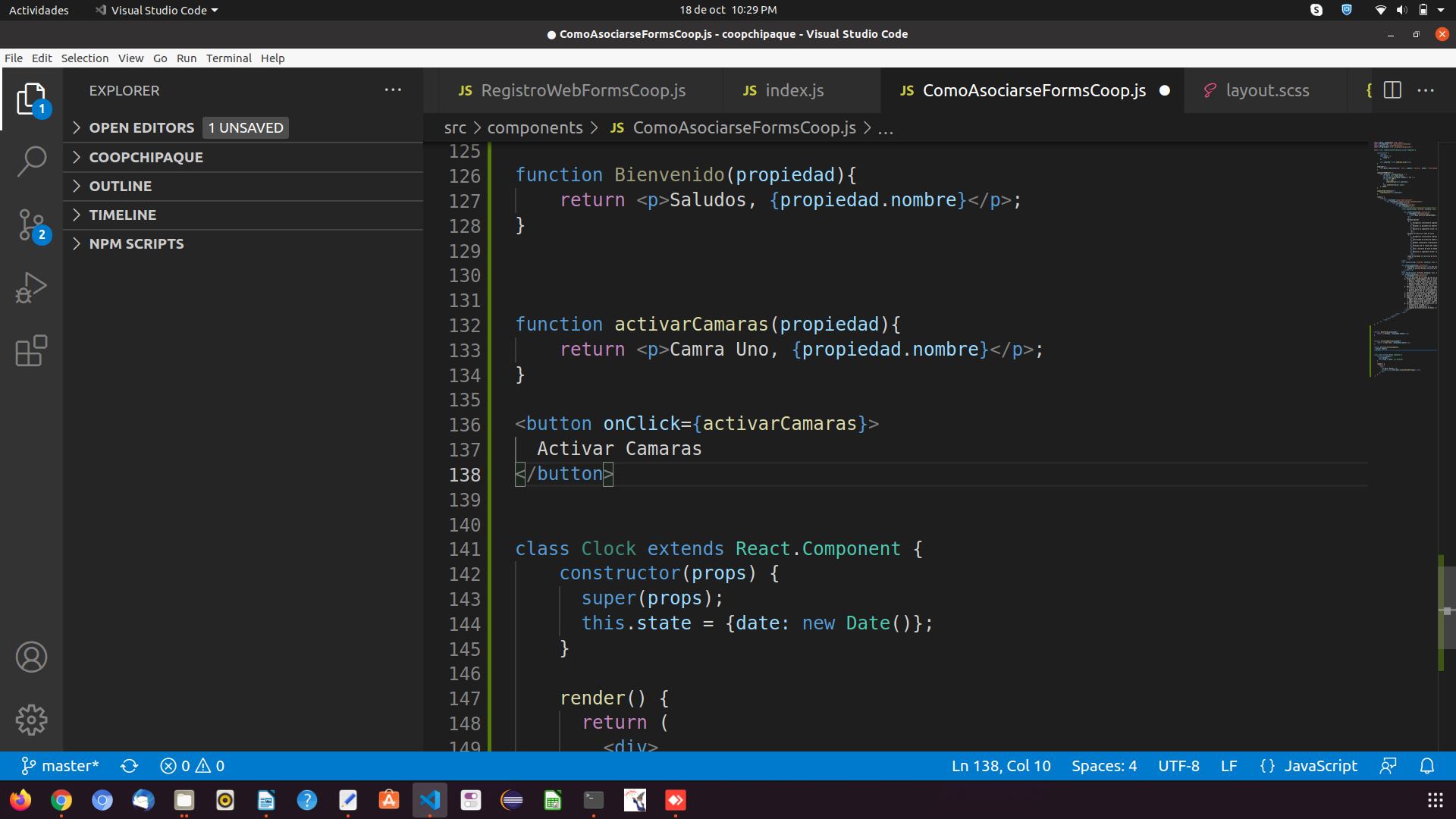


|  | Además, a un componente se le puede aplicar el concepto de estado en donde este, se define como la capacidad de especificar variables que pueden cambiar su valor en el tiempo, convirtiéndose en un comportamiento interno del componente, estos componentes se definen con el nombre de clase con la palabra reservada *class*. |
| --- | --- |

**1.6 Eventos en React**

Los eventos en *React* se manejan semejante a los del DOM se le pasa una función al controlador del evento.





**1.7 Listas y keys**

Las listas hacen referencia a un arreglo de datos, que bien puede ser un arreglo de objetos o de datos simples con carácteres, números, etc. Como toda lista en programación, *React* proporciona un método para extraer estos datos y mostrarlos en la página en la cual se está trabajando, a continuación se presenta cómo se hace:

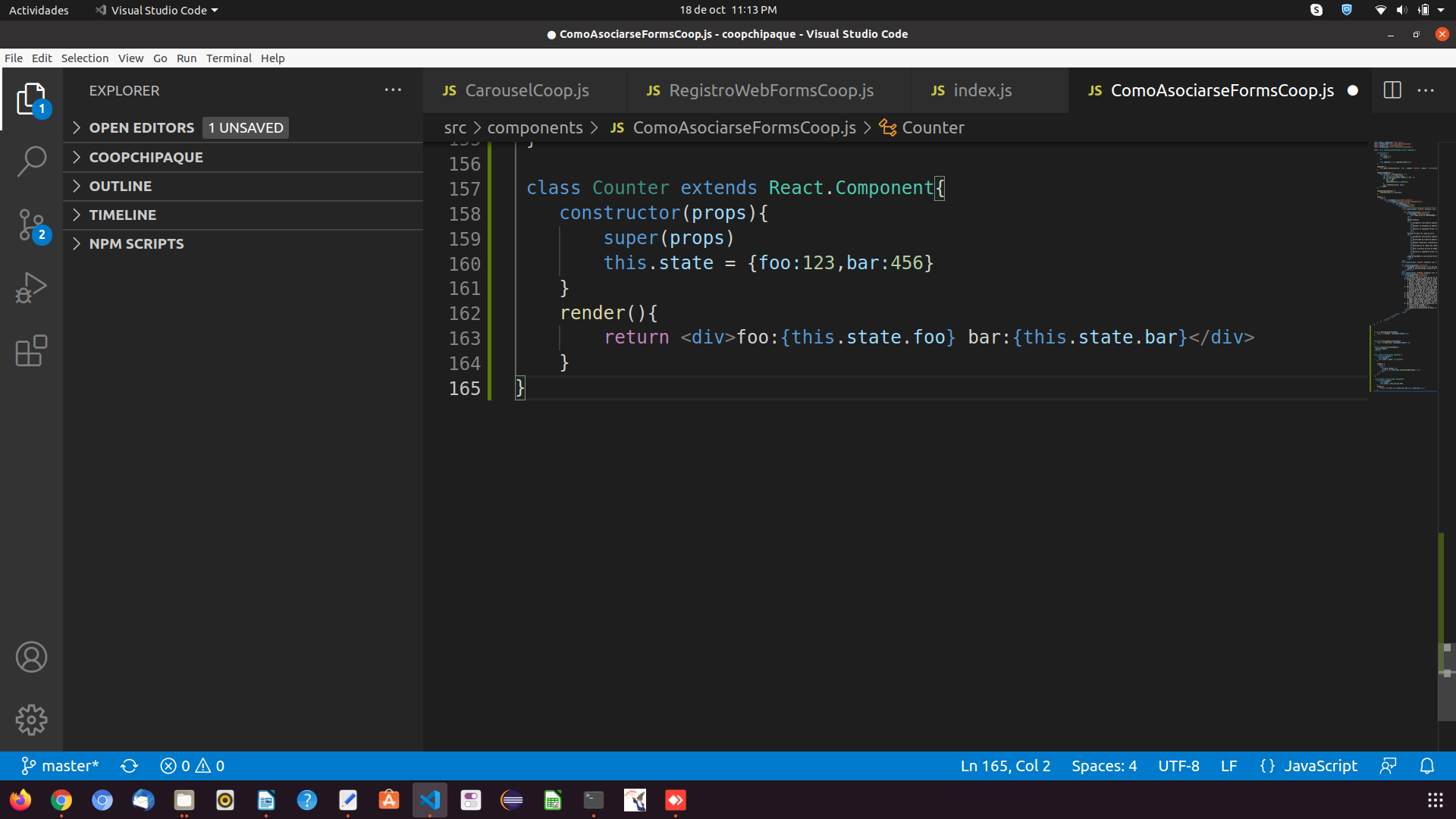


* 1. **Formularios**

|  | En *React* el envío de formularios se comporta como en HTML, solo que aquí, el ingreso y cambio de valores en las entradas se deben manejar con componentes controlados por medio de estados. |
| --- | --- |

**1.9 Concepto de clases en *React***

Las clases son otra forma de definir un componente en *React* se caracterizan por tener propiedades y a diferencia de las funciones pueden definir estados propios sin utilizar el concepto de *Hook*; dentro de los componentes de clase existe una función constructora que se utiliza para definir el estado inicial del componente. Es muy importante llamar a la función super() dentro del constructor para que los parámetros funcionen adecuadamente, como puede verse en la siguiente imagen:





Explorar el siguiente recurso y conocer los aspectos más relevantes relacionados con el concepto de clases en *React*:



**1.10 Funciones *Hooks***

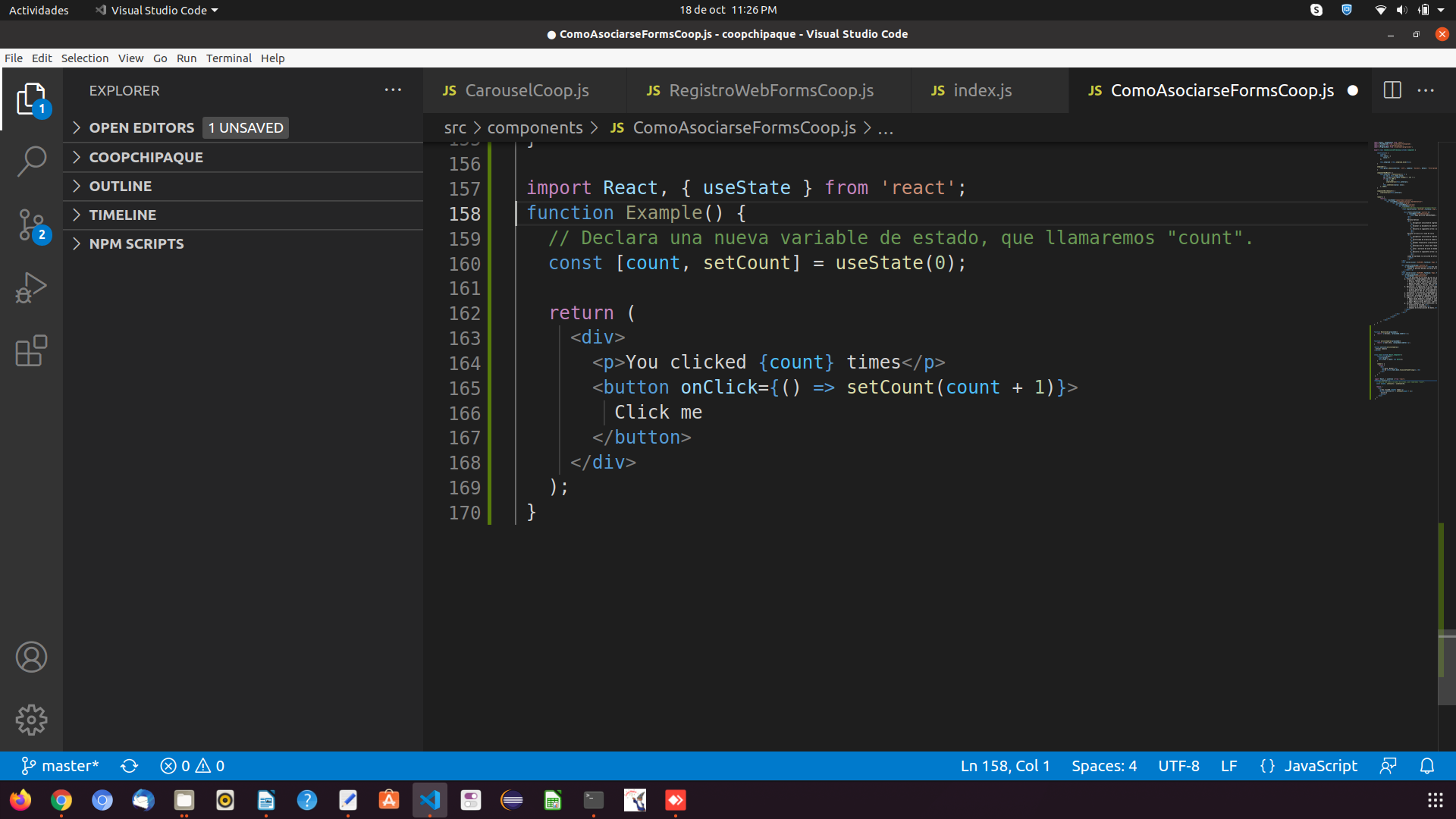
Es el concepto que permite darle a los componentes definidos por medio de funciones manejo de variables de estado locales, así como lo hacen las clases.

Los *Hooks* son funciones *JavaScript* e imponen dos reglas:







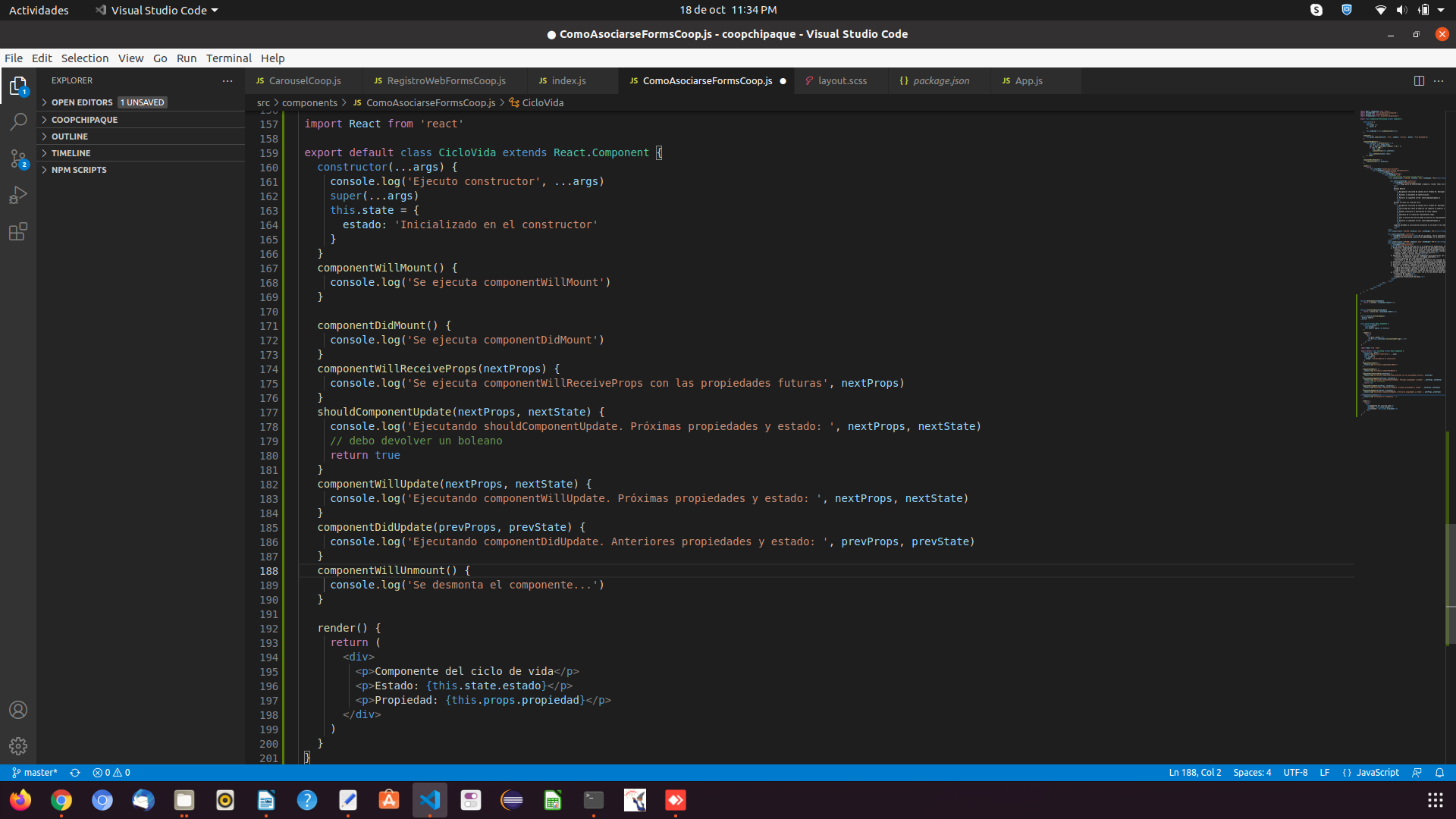


**1.11 Ciclos de Vida de los Componentes de React**

Hace referencia a los métodos que se pueden utilizar por defecto (es decir hacen parte de la librería de *React*) cuando se ejecuta o se renderiza en la línea de tiempo de ejecución del programa un componente estos métodos son:

| *componentWillMount()* | *componentDidMount()* | *componentWillReceiveProps(nextProps)* |
| --- | --- | --- |
| *shouldComponentUpdate(nextProps* | *nextState)* | *componentWillUpdate(nextProps* |
| *nextState)* | *componentDidUpdate(prevProps, prevState)* | *componentWillUnmount()* |

Se puede observar en la siguiente imagen cómo quedará el archivo:





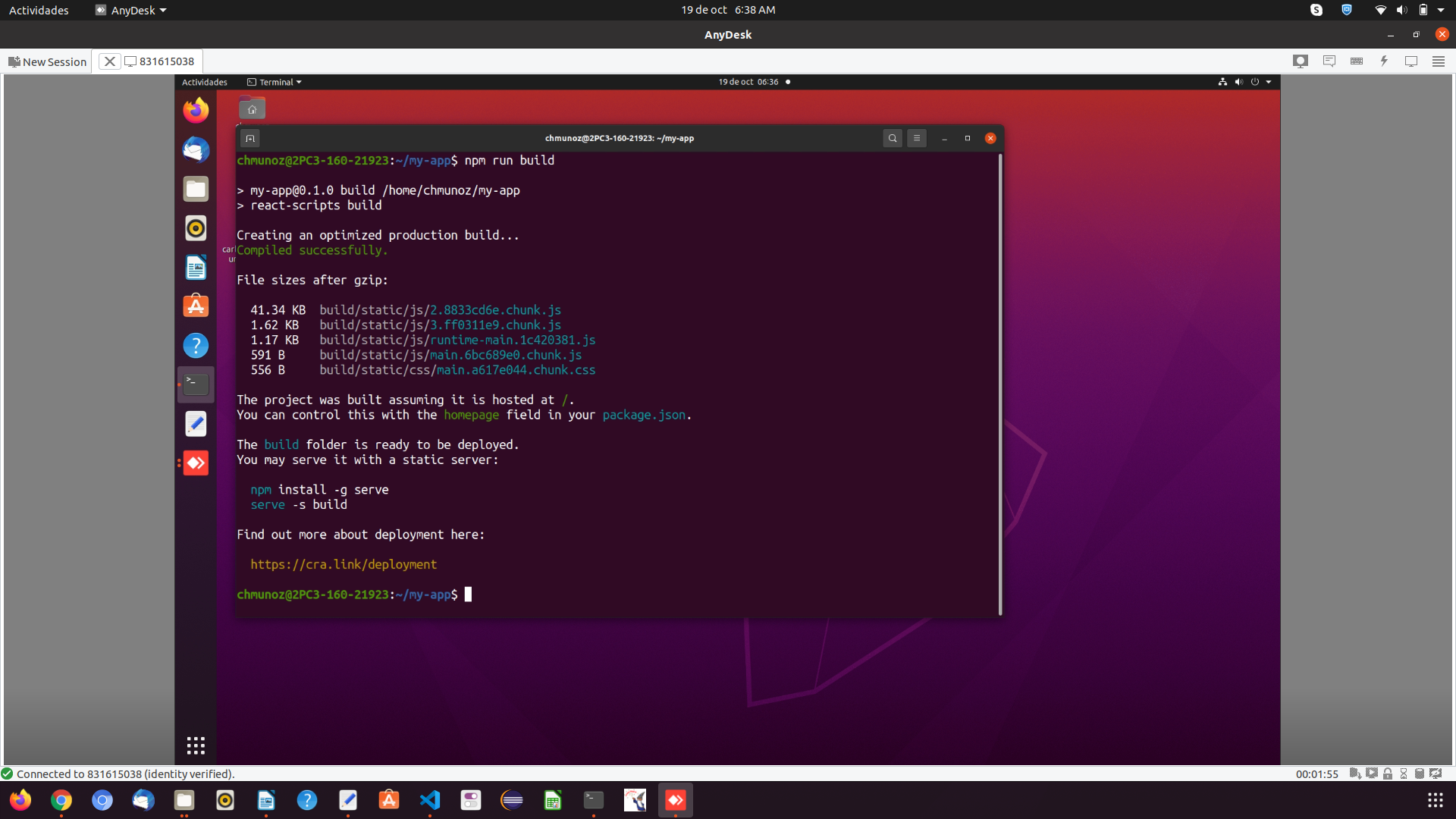


Las utilidades del método de ciclo de vida de *React*, pueden ser detalladas a continuación:

| **Métodos de ciclo de vida de React** | |
| --- | --- |
| **Nombre del Método** | **Utilidad** |
| *componentWillMount()* | Se ejecuta justo antes del primer pintado del componente. |
| *componentDidMount()* | Se ejecuta después del primer pintado del componente. |
| *componentWillReceiveProps(nextProps)* | Se invoca cuando las propiedades se van a actualizar. |
| *shouldComponentUpdate(nextProps, nextState)* | Se invocará cuando se producen cambios de propiedades o cambios de estado. |
| *componentWillUpdate(nextProps, nextState)* | Este se invocará justo antes de que el componente vaya a actualizar su vista. |
| *componentDidUpdate(prevProps, prevState)* | Este se invocará justo después de que el componente vaya a actualizar su vista. |
| *componentWillUnmount()* | Se ejecuta en el momento que el componente se va a retirar del DOM. |

**1.12 Ejecutar la aplicación**

Para ejecutar una aplicación *React JS*, se usa el comando *npm start* y para generar la versión de producción se debe ejecutar el siguiente comando *npm run build*.



Este comando genera la carpeta *build* con la versión compilada y minificada del proyecto *React* SPA *(Single Page Aplication),* a continuación, se observará cómo debe quedar la organización y registro de éste proceso.

build

├── asset-manifest.json

├── favicon.ico

├── index.html

├── logo192.png

├── logo512.png

├── manifest.json

├── robots.txt

└── static

├── css

│ ├── main.a617e044.chunk.css

│ └── main.a617e044.chunk.css.map

├── js

│ ├── 2.8833cd6e.chunk.js

│ ├── 2.8833cd6e.chunk.js.LICENSE.txt

│ ├── 2.8833cd6e.chunk.js.map

│ ├── 3.ff0311e9.chunk.js

│ ├── 3.ff0311e9.chunk.js.map

│ ├── main.6bc689e0.chunk.js

│ ├── main.6bc689e0.chunk.js.map

│ ├── runtime-main.1c420381.js

│ └── runtime-main.1c420381.js.map

└── media

└── logo.6ce24c58.svg

1. **ACTIVIDADES DIDÁCTICAS (OPCIONALES SI SON SUGERIDAS)**

| **DESCRIPCIÓN DE ACTIVIDAD DIDÁCTICA** | |
| --- | --- |
| **Nombre de la Actividad** | **Conceptos de diseño de interfaces web con *React* JS.** |
| **Objetivo de la actividad** | **Afianzar algunos de los conceptos más importantes de *React* JS.** |
| **Tipo de actividad sugerida** | **Selección múltiple.** |
| **Archivo de la actividad**  **(Anexo donde se describe la actividad propuesta)** | **Anexo Actividad didáctica 1** |

1. **MATERIAL COMPLEMENTARIO:**

| **Tema** | **Referencia APA del Material** | **Tipo de material**  **(Video, capítulo de libro, artículo, otro)** | **Enlace del Recurso o**  **Archivo del documento o material** |
| --- | --- | --- | --- |
| Instalando *React* | Editor HTML gratuito en línea, limpiador y convertidor. (2021, 27 de octubre). [Página web]. https://htmled.it/or/ | Página web. | https://htmled.it/or/ |
| Instalando *React* | Cubic Factory. (2021, 27 de octubre). [Página web].  https://www.cubicfactory.com/jseditor/ | Página web. | https://www.cubicfactory.com/jseditor/ |
| Propiedades y estados de los componentes | CodeSandbox: Online Code Editor and IDE for Rapid Web Development. (2021, 27 de octubre). [Página web]. https://codesandbox.io | Página web. | https://codesandbox.io |
| Crear un proyecto | React Playground. (2021, 27 de octubre). [Página web]. <https://playcode.io/react/> | Página web. | <https://playcode.io/react/> |

1. **GLOSARIO:**

| **TÉRMINO** | **SIGNIFICADO** |
| --- | --- |
| CSS3 | Hojas de estilo en cascada versión 3. |
| DOM | Modelo de objetos de un documento HTML |
| ES6 | Abreviación del estándar ECMA SCRIPT. Versión 6. |
| FRONTEND | Se refiere a la capa de presentación de una aplicación web. |
| HTML | Lenguaje de marcado hipertexto. |
| JAVASCRIPT | Lenguaje de programación. |
| JSX | Sintaxis propuesta en *React* JS para escribir elementos de interfaz web. |
| NODE.js | Es un servidor de aplicaciones. |
| NPM | Es un manejador de paquetes de sistema operativo. |
| SPA | Tipo de aplicación Web. |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Node. (2021). Node JS Server. [*https://nodejs.org/en/*](https://nodejs.org/en/)

React JS. (2021). React – A JavaScript library for building user interfaces. Biblioteca de React. <https://reactjs.org/>

1. **CONTROL DEL DOCUMENTO**

|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** |
| --- | --- | --- | --- | --- |
| **Autor (es)** | Carlos Hernán Muñoz Carvajal | Experto Temático | Regional-Cauca Centro de teleinformática y producción industrial. | Septiembre de 2021 |
| María Inés Machado López | Diseñadora Instruccional | Regional Distrito Capital. Centro de Diseño y Metrología | Octubre de 2021 |
| Jhon Jairo Rodríguez Pérez | Diseñador y evaluador instruccional | Regional Distrito Capital. Centro para la Industria de la Comunicación Gráfica | Diciembre de 2021 |

1. **CONTROL DE CAMBIOS**

|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** | **Razón del Cambio** |
| --- | --- | --- | --- | --- | --- |
| **Autor (es)** |  |  |  |  |  |