1. Ejercicio:
2. Investiga la historia de React y menciona dos hitos importantes en su desarrollo.

Historia de React:

React, creado por Facebook, comenzó en 2010 con XHP, que permitió la creación de componentes en PHP. En 2011, Jordan Walke desarrolló FaxJS, un prototipo temprano de React. En 2012, ante la complejidad creciente de gestionar anuncios en Facebook, React fue refinado y adoptado internamente. Ese mismo año, Facebook adquirió Instagram, lo que incrementó la presión para desvincular React de Facebook y hacerlo de código abierto.

En 2013, React fue lanzado públicamente en la JSConf US, con escepticismo inicial. Sin embargo, la comunidad comenzó a adoptarlo gracias a sus ventajas técnicas, como el Virtual DOM, que mejoraba el rendimiento de las aplicaciones. En 2014, React ganó popularidad y se integró con herramientas como React Developer Tools y Atom.

2015 marcó la llegada de React Native, expandiendo React a plataformas móviles, y el lanzamiento de Redux, mejorando la gestión del estado. En 2019, se introdujeron los Hooks, una característica clave que simplificó la lógica en componentes funcionales. Desde entonces, React se ha mantenido estable, con mejoras como React Fiber en 2017, y se ha consolidado como una de las principales bibliotecas de desarrollo web, con una comunidad activa que impulsa su evolución constante.

Dos hitos importantes en la historia de React

- **Introducción del virtual DOM:** En 2013 se introdujo este concepto que permite a React actualizar solo las partes de la interfaz que han cambiado, en lugar de actualizar toda la página. Esto mejora significativamente el rendimiento de las aplicaciones.

- **Creación de los hooks:** React Hooks se introdujo en la versión 16.8 de React en 2019. Los Hooks permiten a los desarrolladores usar estado y otras características de React sin escribir clases. Esta actualización simplificó la forma en que se construyen y gestionan los componentes, y promovió una mayor adopción de funciones en lugar de clases en React.

3. Escribe una breve explicación sobre por qué Facebook decidió crear React.
React fue creado para satisfacer las necesidades de desarrollo de Facebook, que enfrentaba problemas de rendimiento, mantenimiento y escalabilidad a medida que crecía. La gran cantidad de usuarios y las constantes actualizaciones en tiempo real, como comentarios y notificaciones, dificultaban la gestión eficiente de sus aplicaciones web. React surgió como una solución para abordar estos desafíos técnicos, permitiendo a Facebook manejar interfaces de usuario complejas y dinámicas de manera más eficaz.

2. Ejercicio:
4. Menciona tres ventajas de usar React en el desarrollo de aplicaciones web.

- Virtual DOM: React utiliza un Virtual DOM que mejora el rendimiento al minimizar la manipulación directa del DOM real. Esto permite actualizaciones rápidas y eficientes de la interfaz de usuario.

- Componentes reutilizables: React se basa en componentes modulares que pueden ser reutilizados en diferentes partes de la aplicación, lo que facilita el mantenimiento y la escalabilidad del código.
- Actualizaciones fáciles de implementar: React permite adoptar nuevas versiones o características sin necesidad de una reescritura completa de la aplicación, gracias a su estructura modular y la naturaleza de sus actualizaciones.

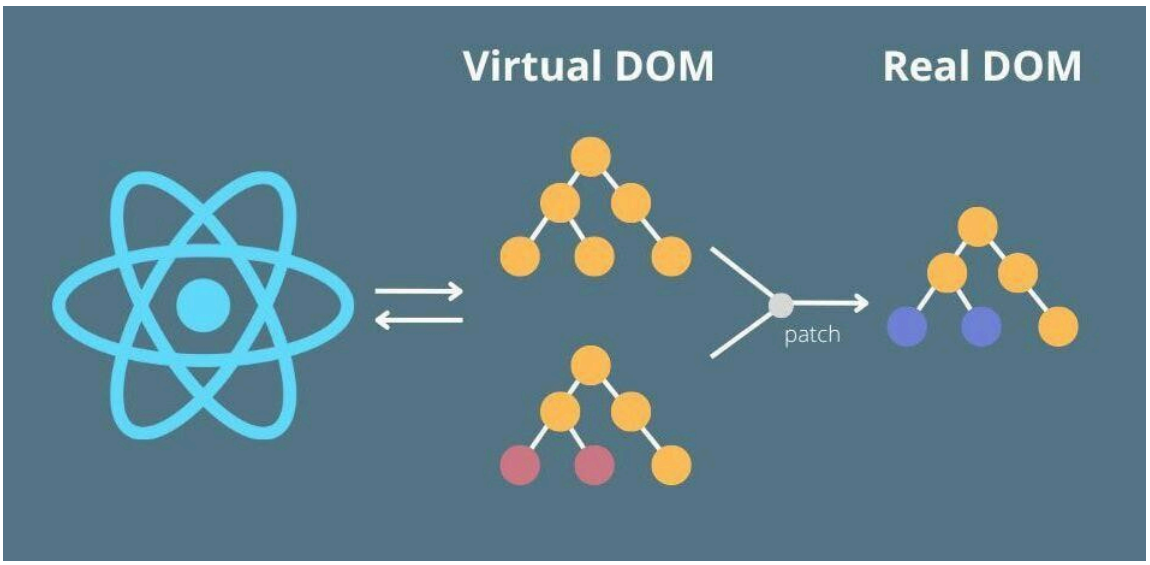
5. Explica cómo el Virtual DOM mejora el rendimiento de una aplicación.

    El Virtual DOM es una de las características clave que hace que React sea tan eficiente y rápido en el desarrollo de aplicaciones web. Básicamente, el Virtual DOM es una representación ligera del DOM real (Document Object Model) que React utiliza para optimizar el rendimiento de las actualizaciones de la interfaz de usuario. En lugar de manipular directamente el DOM real cada vez que hay un cambio en el estado de la aplicación, React primero aplica los cambios en esta versión virtual del DOM.

    Cuando el estado de la aplicación cambia, React crea un nuevo Virtual DOM que refleja esos cambios. Luego, compara esta nueva versión del Virtual DOM con la versión anterior, en un proceso conocido como "diffing". Este proceso permite a React identificar exactamente qué partes del Virtual DOM han cambiado, añadido o eliminado.

    Una vez que React ha determinado las diferencias, actualiza solo las partes específicas del DOM real que han sido modificadas. Esto significa que no se vuelve a renderizar toda la interfaz de usuario, sino solo los elementos que realmente han cambiado. Esta actualización mínima del DOM real reduce significativamente la carga en el navegador, ya que evita manipulaciones innecesarias que pueden ser costosas en términos de rendimiento.

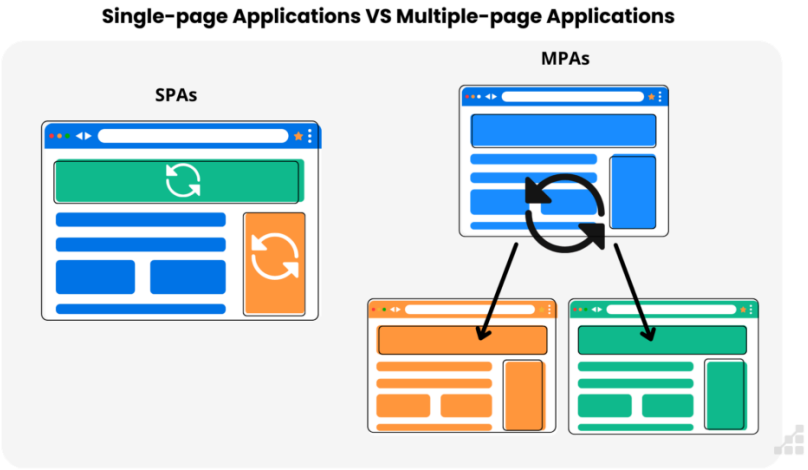
    Al reducir la cantidad de veces que se necesita redibujar y reorganizar elementos en la página (conocido como "repaint" y "reflow"), el Virtual DOM hace que las aplicaciones React sean más rápidas y suaves. Esto es especialmente beneficioso en aplicaciones grandes y complejas, donde las actualizaciones frecuentes podrían causar una disminución notable en la velocidad de la interfaz de usuario si se manipulara el DOM directamente. El Virtual DOM es una herramienta esencial para optimizar el rendimiento en React.



3. Ejercicio:

6. Define qué es una Single Page Application (SPA).

Es una aplicación web que se carga en una sola página donde todo el HTML, CSS y JS se carga al inicio para gestionar el contenido y la interacción del usuario sin necesidad de recargar la página completa desde el servidor. En lugar de cargar nuevas páginas desde el servidor para cada interacción, una SPA actualiza el contenido dinámicamente en la misma página, brindando una experiencia más fluida de navegación para el usuario



7. Explica cómo React facilita la creación de una SPA. Proporciona un ejemplo de cómo un componente de React puede actualizar la interfaz sin recargar la página.

React permite construir la interfaz de usuario en forma de componentes modulares y reutilizables. Cada componente puede manejar su propio estado y lógica, lo que facilita la construcción de interfaces interactivas sin necesidad de recargar la página. Además, React utiliza un Virtual DOM para realizar actualizaciones eficientes. En lugar de actualizar el DOM real directamente, React realiza cambios en un Virtual DOM y luego compara estos cambios con el DOM real. Solo se actualizan las partes del DOM que realmente han cambiado, lo que mejora el rendimiento.

Ejemplo:

home.js

```
import React from 'react';

function Home() {
    return (
        <div>
            <h2>Home Page</h2>
            <p>Bienvenido al Home Page!</p>
        </div>
    );
}

export default Home;

export default App;
```

about.js

```
import React from 'react';

function About() {
    return (
        <div>
            <h2>About Page</h2>
            <p>Learn more about us on this page.</p>
        </div>
    );
}

export default About;
```

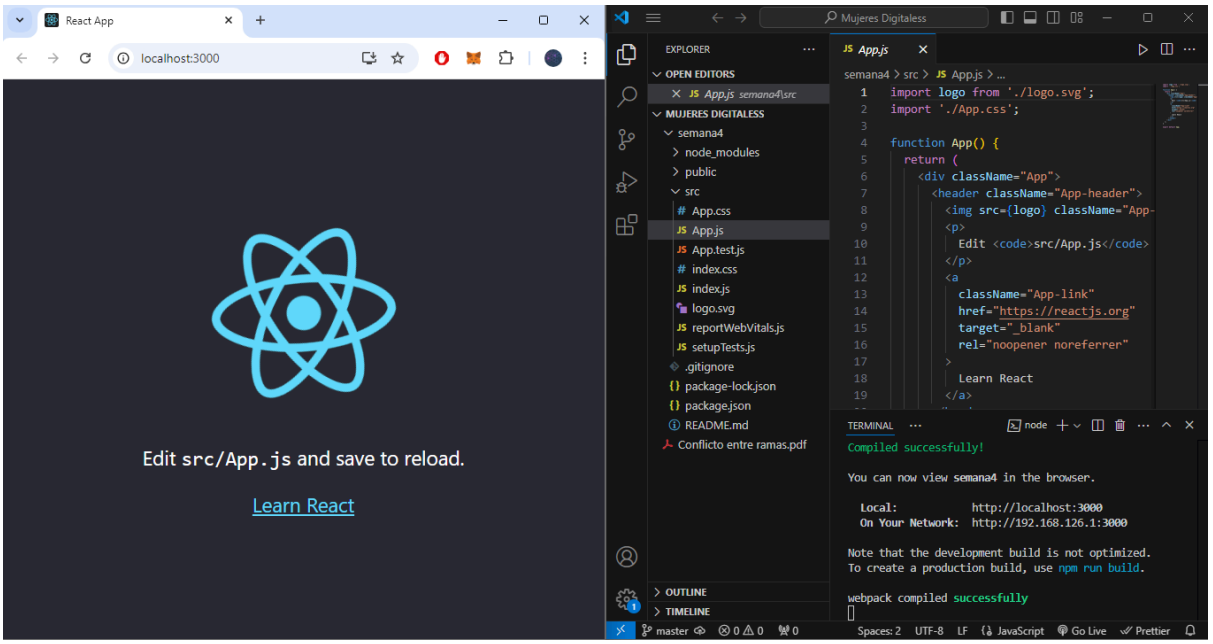
App.js

```
import React, {useState} from 'react';
import Home from './Home';
import About from './About';

function App() {
    const [currentPage, setCurrentPage] = useState('home');

    return (
        <div>
            <nav>
                <button onClick={() =>
setCurrentPage('home')}>Home</button>
                <button onClick={() =>
setCurrentPage('about')}>About</button>
            </nav>
            <main>
```

```
                {currentPage === 'home' ? <Home /> : <About />}
            </main>
        </div>
    );
}

export default App;
```

4. Ejercicio:

8. Crea un nuevo proyecto React utilizando Create React App.
9. Inicia el servidor de desarrollo y comparte una captura de pantalla de tu proyecto en ejecución.



5. Ejercicio:

10. Explica brevemente el propósito de las carpetas src y public en un proyecto React.
public: Contiene archivos estáticos que se sirven directamente al navegador, contiene archivos públicos que no serán procesados
src: Contiene todo el código fuente de la app

6. Ejercicio:

11. Explica cómo JSX se diferencia del HTML tradicional.
La diferencia principal entre JSX y HTML tradicional es que JSX es una extensión de JavaScript que permite escribir estructuras similares a HTML dentro de archivos JavaScript, integrando lógica y estructura en un solo lugar. En contraste, HTML tradicional es un lenguaje de marcado independiente usado para definir la estructura de páginas web, que no puede incluir directamente lógica de JavaScript y se ejecuta directamente en el navegador sin necesidad de compilación. Además, JSX requiere que las etiquetas sean cerradas y usa camelCase para los atributos, mientras que HTML usa minúsculas para los atributos y no siempre requiere el cierre de etiquetas.

7. Ejercicio:

12. Define los roles principales en un equipo SCRUM.

Los roles principales en un equipo SCRUM son:

- **Product Owner**: Responsable de definir la visión del producto y gestionar el Product Backlog, priorizando las características y asegurando que el equipo trabaje en las tareas más valiosas.
- **Scrum Master**: Facilita el proceso SCRUM, eliminando obstáculos, asegurando que las prácticas SCRUM se sigan correctamente, y ayudando al equipo a mejorar continuamente. Organiza y facilita las ceremonias de Scrum, como las Daily, Sprint Planning, Sprint Review, y Sprint Retrospective, asegurando que sean productivas y enfocadas.
- **Development Team**: Un grupo de profesionales multidisciplinarios que son responsables de entregar el producto incrementable al final de cada Sprint. Trabajan de manera colaborativa y auto-organizada para cumplir con los objetivos del Sprint.

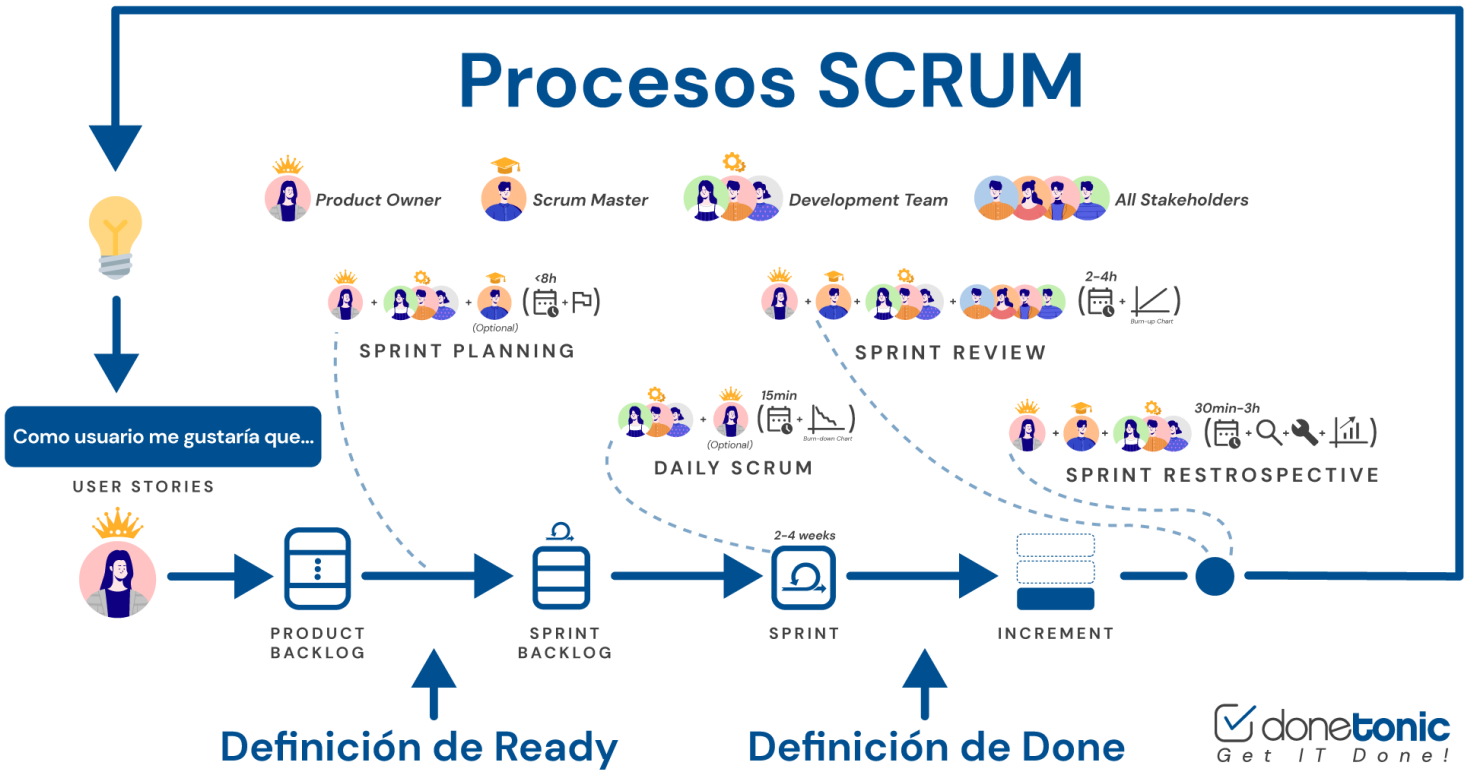
13. Explica qué es un sprint y cómo se planifica.

Un Sprint es un ciclo de trabajo en Scrum, generalmente de 1 a 4 semanas, durante el cual el equipo desarrolla un incremento del producto que es potencialmente entregable. Es un período de tiempo fijo en el que se realizan todas las actividades necesarias para alcanzar un conjunto de objetivos definidos previamente.

Para planifircar un sprint se hace un Sprint Planning, es la reunión al inicio del Sprint donde el equipo define qué se va a hacer durante el Sprint, en esta reunión se hace:

- **Definir el objetivo del Sprint**: El Product Owner presenta el Product Backlog, que es una lista priorizada de tareas. El equipo colabora con el Product Owner para definir un Objetivo del Sprint (Sprint Goal), que es la meta que se quiere alcanzar al final del Sprint.
- **Seleccionar las historias de usuario o tareas**: El equipo selecciona del Product Backlog los elementos que pueden completar dentro del Sprint, basándose en su capacidad y el objetivo del Sprint.
- **Crear el Sprint Backlog**: Los elementos seleccionados se descomponen en tareas más pequeñas y específicas, que se organizan en el Sprint Backlog, el plan detallado para el Sprint.
- **Estimar el trabajo**: El equipo estima el esfuerzo necesario para completar cada tarea, utilizando técnicas como puntos de historia o estimación en horas, para asegurarse de que todo el trabajo sea realista dentro del Sprint.

Además se realizan a diario unas reuniones llamadas Daily para revisar el progreso y ajustar el plan si es necesario. Al final del Sprint, se realiza una Sprint Review para mostrar el trabajo completado y una Sprint Retrospective para reflexionar sobre cómo mejorar en el siguiente Sprint.


Procesos SCRUM
Product Owner
Scrum Master
Development Team
All Stakeholders
<8h
(Optional)
SPRINT PLANNING
2-4h
SPRINT REVIEW
15min
(Optional)
DAILY SCRUM
30min-3h
SPRINT RESTROSPECTIVE
Como usuario me gustaría que...
USER STORIES
2-4 weeks
PRODUCT BACKLOG
SPRINT BACKLOG
SPRINT
INCREMENT
Definición de Ready
Definición de Done
donetonic
Get IT Done!