



El futuro digital
es de todos

MinTIC



CICLO IV:

Desarrollo de Aplicaciones Web

Mision
TIC2022



El futuro digital
es de todos

MinTIC



UNIVERSIDAD
DEL NORTE

Vigilada Mineducación

Registro Discord – Aplicaciones Web

Mision
TIC2022



Tripulante,

Para este ciclo te queremos contar que estaremos utilizando una nueva herramienta como canal oficial con tus tutores, docentes y equipo trabajo para que las interacciones sean más eficientes.

LA PLATAFORMA QUE ESTAREMOS USANDO ES



DISCORD



¿Qué es Discord?

Un servicio de mensajería instantánea que cuenta con canales de texto o voz, y videos. Combina las mejores características de los programas más utilizados como Skype y Slack, con una interfaz fácil de usar

DISPONIBLE PARA:



iOS



Puedes acceder a ella desde la web o descargarla en tu computador o dispositivos móviles.



¿Cómo puedo unirme al servidor de MisiónTIC Uninorte?



1. Ingresa a la página principal: <https://discord.com/>
2. Registra una cuenta con el correo uninorte usando el nombre de usuario de la Universidad. Por ejemplo: si tu correo es manuelp@uninorte.edu.co tu usuario es manuelp.
3. Activa tu cuenta por medio del mensaje que te llegará a tu correo electrónico.



¿Cómo puedo unirme al servidor de MisiónTIC Uninorte?

Una vez tu cuenta esté activa:

- a) Puedes ir a la sesión de agregar servidor dentro de discord y colocar el link de invitación del servidor de la Universidad.
- b) O puedes simplemente dar clic al link y aceptar la invitación al servidor.

Link de invitación servidor Ciclo 4a- Web: <https://discord.gg/xUMHXJmcnH>



El futuro digital
es de todos

MinTIC

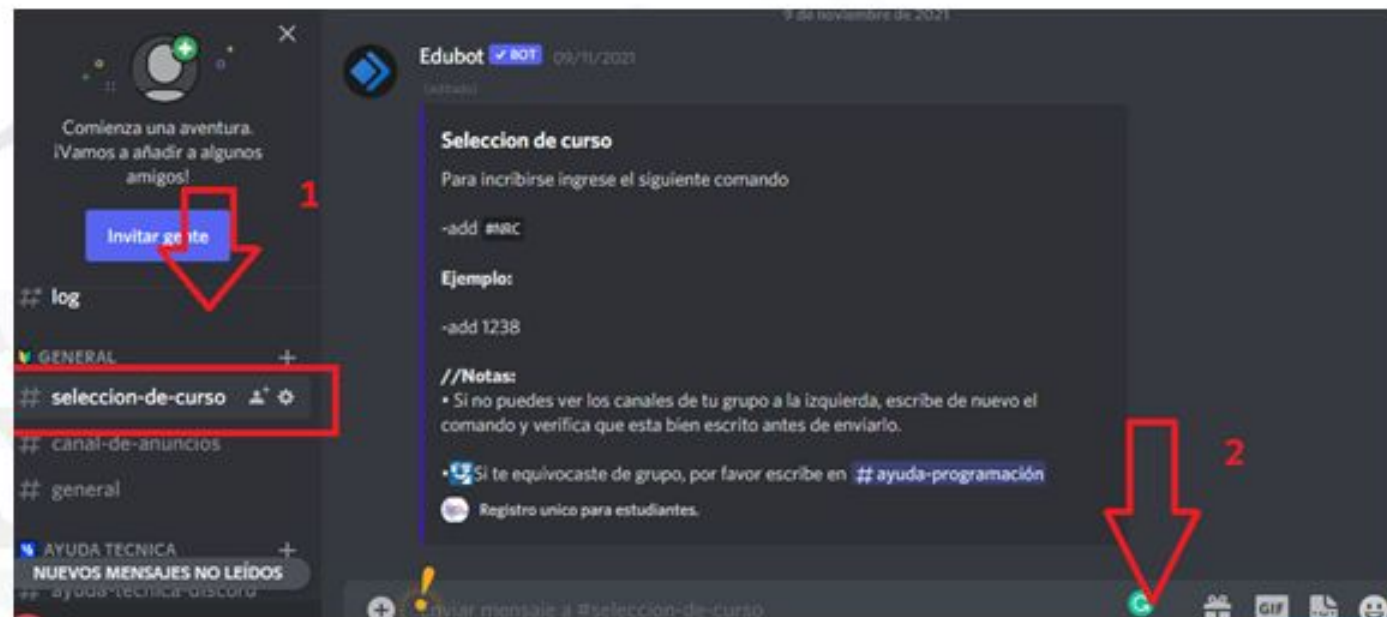
¿Cómo puedo unirme al servidor de MisiónTIC Uninorte?



El último paso es dirigirte al canal #selección de cursos.

Una vez allí vas a escribir la información que te pide la plataforma:

Movil: -add NRC



Realizado este
paso,
automáticamente
el servidor te
agregará a tu
grupo
correspondiente.

Misión
TIC2022



NOTA:

Si no sabes cuál es tu NRC puedes dirigirte a la plataforma de Misión TIC Uninorte y revisar el título de tu curso:

DESARROLLO_DE_APLICACIONES_WEB_8888001231IST1047_01

Contenido Calendario Debates Libro de calificaciones Mensajes Estadísticas

**LOS 4 NÚMEROS ALLÍ SELECCIONADOS
SON EL NRC DEL CURSO.**



En ese sentido, veamos un ejemplo de cómo debería escribir el comando si haces parte del **NRC 1273** y el **equipo 8**:

-add 12 31



IMPORTANTE: DEBES LLEVAR A CABO ESTE SEGUNDO PASO PARA QUE TU REGISTRO SE COMPLETE. VERIFICA QUE TU NOMBRE **CAMBIA DE COLOR** PARA CONFIRMAR QUE YA ESTÁS ESTAS INSCRITO DE FORMA EXITOSA EN TU GRUPO.



El futuro digital
es de todos

MinTIC



Vigilada Mineducación

Sesión 06:

Desarrollo de Aplicaciones Web

Desarrollo de Front-End web con React





Objetivos de la sesión

Al finalizar esta sesión estarás en capacidad de:

1. Instalar el framework de React en un ambiente de desarrollo.
2. Crear un proyecto básico de React.



React

- React es una librería de Javascript para crear interfaces de usuario declarativas, basadas en componentes altamente reutilizables.
- Es una librería open-source, creada y mantenida por el equipo de Facebook.
- Actualmente se considera la librería de desarrollo Front-End más utilizada.
- Nos permite crear Single Page Applications (SPA), lo cual consiste en una página web que es construida y modificada por el cliente, es decir, las vistas son generadas por el cliente y no por el servidor.
- Dentro de sus implementaciones más populares nos encontramos con:
 - Create React App (CRA)
 - Gatsby
 - Next.js





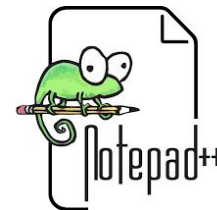
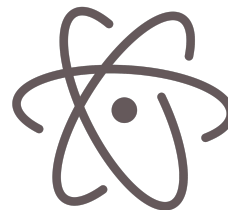
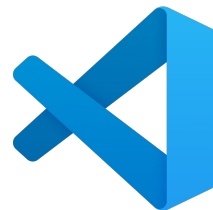
React - Instalación con CRA

- Para crear un proyecto o una web app con CRA deberemos seguir los siguientes pasos:
- Descargar e instalar la versión Long Term Support (LTS) de [Node.js](#).
- Ejecutaremos el comando `npx create-react-app <nombre de la app> --use-npm` en nuestra terminal.
 - **npx**: Comando global de *npm* (controlador de paquetes de Node.js), usado para ejecutar scripts configurados globalmente en el repositorio de node.
 - **use-npm**: Opción utilizada para especificar que el controlador de paquetes que utilizaremos sea efectivamente npm, ya que si llegamos a tener otro controlador de paquetes instalado, como *yarn*, este tomaría precedencia.



React - Instalación con CRA

- Una vez ejecutado el comando este creara un folder con el nombre de **<nombre de la app>** desde el directorio en el que nos encontremos desde la terminal.
- Podemos abrir nuestro editor de texto directamente con el folder de nuestro proyecto y nos encontraremos con la siguiente estructura de carpetas.





React - Estructura

- Estructura de carpetas

```
| -- <nombre de la app>/  
| | -- README.md  
| | -- node_modules/  
| | -- package.json  
| | -- public/  
| | | -- index.html  
| | | -- favicon.ico  
| | -- src/  
| | | -- App.css  
| | | -- App.js  
| | | -- App.test.js  
| | | -- index.css  
| | | -- index.js  
| | | -- logo.svg
```

- Para que el proyecto se compile, estos archivos deben existir con nombres exactos:
 - public/index.html es la plantilla de la página.
 - src/index.js es el punto de entrada de JavaScript.
- Se puede eliminar o renombrar los demás archivos.



React - Estructura

- Tomando en cuenta nuestra estructura de carpetas a partir de src:

```
| -- src/  
| | -- App.css  
| | -- App.js  
| | -- App.test.js  
| | -- index.css  
| | -- index.js  
| `-- logo.svg
```

- Es importante darnos cuenta que tenemos la libertad de ordenar nuestras carpetas a nuestra disposición.
- Esto podría prestarse para generar una estructura de carpetas desordenadas que nos cueste escalar y mantener nuestro código.



React - Estructura

- Es considerado como buena práctica tener las siguientes carpetas:

```
| -- src/  
| | -- /components  
| | -- /context  
| | -- /hooks  
| | -- /pages  
| | -- /services  
| | -- /store  
| | -- App.css  
| | -- App.css  
| | -- App.js  
| | -- App.test.js  
| | -- index.css  
| | -- index.js  
| `-- logo.svg
```

- **components:** Aquí residirán la mayoría de nuestros componentes.
- **context:** Aquí ubicamos nuestros contextos de context API.
- **hooks:** Aquí definiremos hooks personalizados.
- **pages:** Aquí definiremos las vistas de nuestras rutas en la web app.
- **services:** Aquí configuraremos servicios de terceros como API's, Back-Ends, sockets, entre otros.
- **store:** Aquí configuraremos nuestro almacenamiento con redux API.



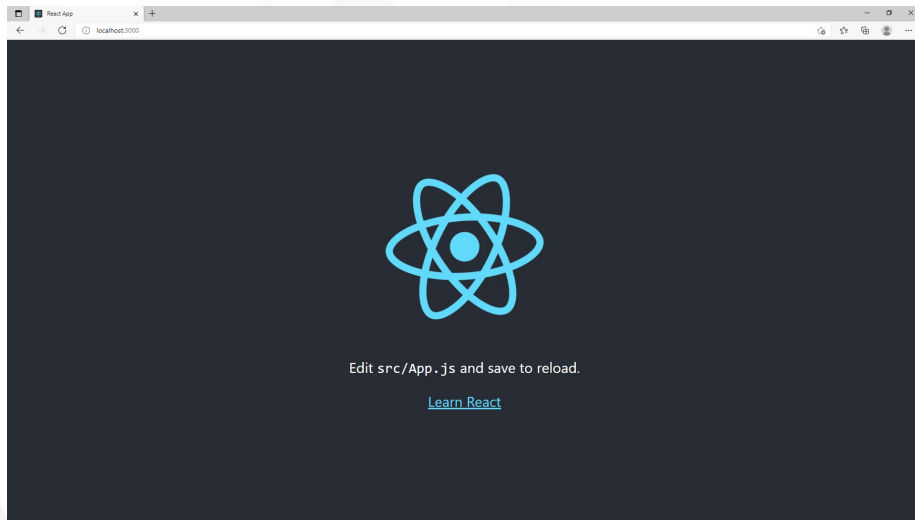
React - Ejecución de Aplicaciones

- Analizando nuestro archivo *package.json*, nos encontramos con los siguientes comandos bajo la configuración *scripts*:
 - **npm start**: Ejecuta nuestra aplicación en modo de desarrollo.
 - **npm test**: Ejecuta las pruebas unitarias de nuestra aplicación.
 - **npm run build**: Compila nuestra aplicación.
 - **npm run eject**: Hace visible la configuración webpack de nuestro proyecto (por favor no experimentar con esto sin tener conocimientos sobre webpack ya que esto afecta directamente el proceso de compilado y ejecución).
- Para ejecutar nuestra aplicación simplemente ejecutaremos *npm start*.



React - Ejecución de Aplicaciones

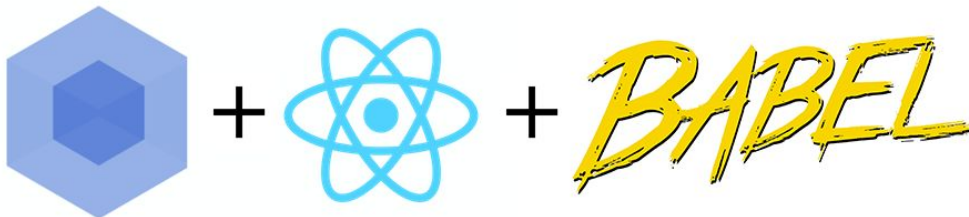
- Una vez nuestro entorno de desarrollo se esté ejecutando, nuestra app debería abrir el navegador web que tengamos configurado por defecto en la url ***localhost:3000***.
- Deberíamos ver lo siguiente:





React - JSX

- Es una extensión de la sintaxis de JavaScript (.jsx o .js)
- Prácticamente es JavaScript pero con más funcionalidades.
 - Nos permite embeber HTML dentro de nuestros archivos con JSX.
 - Nos permite acceder a funcionalidades propias de React.
 - Produce “elementos” de React.
- En últimas JSX es compilado a bloques de JS a través de Babel y de Webpack:





React - JSX

- Para crear un componente es necesario que este tenga su inicial en mayúscula y retorne una expresión JSX.
- Las expresiones JSX normalmente se encuentran encerradas en paréntesis ().
- En JSX se puede utilizar la misma sintaxis que en HTML puesto que el equipo de react se ha encargado de implementar los elementos básicos.
- Para añadir una clase de CSS a un elemento se usa `className` y no `class` ya que `class` es una palabra reservada de javascript y JSX en últimas es javascript.



React - JSX

- Los componentes pueden incluir otros componentes usando la prop **children**.
- A un componente se le pueden agregar estilos en línea usando el formato **json**.
- Para los nombres de las propiedades con un guión “-”, como **background-color**, se puede usar camelCase y entonces se usaría **backgroundColor** o se podría usar una string “**background-color**”.
- Un componente sólo puede retornar un elemento o nodo HTML padre.
- Antes del return se puede definir cualquier lógica que JS permite puesto que en últimas estamos es ejecutando JS.



React - Renderizar Elementos

- Crearemos un archivo **Demo.js** en **src/components** con la siguiente estructura:

```
import React, {useState} from 'react';

const Demo = () => {
  const [name, setName] = useState('Developer');
  return <p>Hola {name}</p>;
};

export default Demo;
```



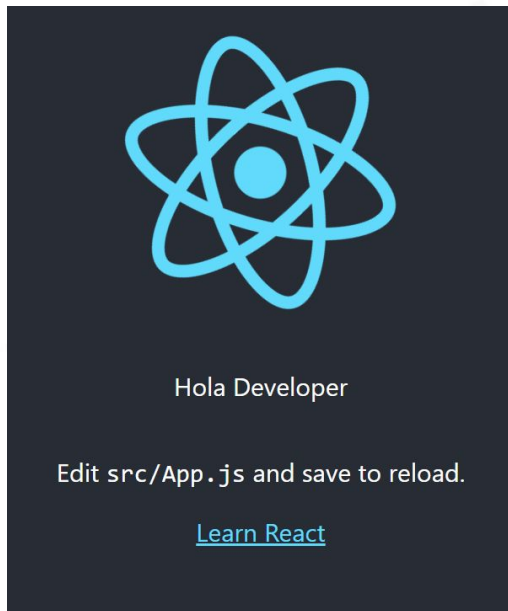

React - Renderizar Elementos

- Luego en `App.js`:
 - Importamos Demo en la primera línea:
 - `import Demo from './components/Demo';`
 - Escribimos lo siguiente en la décima línea:
 - `<Demo />`



React - Renderizar Elementos

- Si observamos nuevamente nuestra aplicación notaremos un cambio:



- Como podemos observar se añadió la línea “Hola Developer”.
- Esto es posible debido a que lo que acabamos de hacer fue crear un componente de React.
- Así mismo, creamos una variable name y la utilizamos dentro de nuestro JSX (código con un formato similar a HTML), usando llaves.



React - Componentes

- React introduce dos tipos de componentes, componentes basados en clases y componentes funcionales.
- La diferencia entre ambos componentes afecta directamente la forma en como trabajamos en react.
 - Diferencias en formato.
 - Diferencias en funcionalidades de React.
 - Diferencias en ciclo de vida.
 - Entre otros.
- Cabe resaltar que en **Demo.js** definimos el componente **<Demo>** de manera funcional.



React - Componentes basados en Clases

- Para ver directamente la diferencia entre componentes basados en clase y funcionales, reescribimos <Demo> como un componente basado en clases de la siguiente forma:

```
import React, { Component } from 'react';

class Demo extends Component {
  state = { name: 'Developer' };

  render() {
    return <p>Hola {this.state.name}</p>;
  }
}

export default Demo;
```



React - Componentes basados en Clases

- Contienen más **boilerplate** debido a que es más tipado con respecto a los componentes funcionales.
- Cómo definimos clases y no funciones directamente, entonces requerimos de utilizar el keyword **this** para referirnos al estado de nuestro componente.



React - Componentes Funcionales

- Los componentes funcionales son considerados como la forma moderna de desarrollar React.
- Esto se debe a que:
 - Es más sencillo de leer y escribir.
 - Nos facilita controlar el estado de nuestros componentes y evitamos conflictos con el keyword `this` debido al alcance o `scope` del mismo.
 - React ha estado introduciendo nuevas funcionalidades para este tipo de componentes en sus versiones más recientes.
- Por estas razones, utilizaremos componentes funcionales en estas sesiones.



El futuro digital
es de todos

MinTIC

UN UNIVERSIDAD
DEL NORTE

Vigilada Mineducación

Ejercicios de práctica

Mision
TIC2022



Referencias

- <https://es.reactjs.org/docs/getting-started.html>
- <https://create-react-app.dev/docs/getting-started/#quick-start>
- <https://create-react-app.dev/docs/folder-structure>
- <https://create-react-app.dev/docs/available-scripts>
- <https://reactjs.org/docs/introducing-jsx.html>
- <https://reactjs.org/docs/rendering-elements.html>
- <https://reactjs.org/docs/components-and-props.html#function-and-class-components>



Seguimiento Habilidades Digitales en Programación

* De modo general, ¿Cuál es grado de satisfacción con los siguientes aspectos?

	Nada Satisfecho	Un poco satisfecho	Neutra	Muy satisfecho	Totalmente satisfecho
Sesiones técnicas sincrónicas	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Sesiones técnicas asincrónicas	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Sesiones de inglés	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Apoyo recibido	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Material de apoyo: diapositivas	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Material de apoyo: ejercicios prácticos	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Completa la siguiente encuesta para darnos retroalimentación sobre esta semana ▼▼▼

<https://www.questionpro.com/t/ALw8TZIxOJ>



El futuro digital
es de todos

MinTIC

UN UNIVERSIDAD
DEL NORTE

Vigilada Mineducación

¡GRACIAS
POR SER PARTE DE
ESTA EXPERIENCIA
DE APRENDIZAJE!



Misión
TIC 2022