RESUMEN SESIÓN EN LÍNEA

Fecha de la Sesión:	03/12/2024
Competencia:	TÉCNICA
Tema Central:	GRABACIÓN Y RESUMEN CONFERENCIA CODIFICAR EL FRONTEND
Instructor Responsable:	ILMER CUELLO

Desarrollo de la Sesión

- ✓ **COMPETENCIA: 220501096 - DESARROLLAR LA SOLUCIÓN DE SOFTWARE DE ACUERDO CON EL DISEÑO Y METODOLOGÍAS DE DESARROLLO.**

- ✓ **RESULRADO DE APRENDIZAJE:** 220501096- 04 CODIFICAR EL SOFTWARE DE ACUERDO CON EL DISEÑO ESTABLECIDO.

    - ✓ Se inicia la sesión en línea a las 8:00 pm con los aprendices de la ficha: 2869399, del programa ANÁLISIS Y DESARROLLO DE SOFTWARE. Posterior a la solicitud de aprobación para grabar la sesión en línea, se inicia explicando las evidencias:

        - RAP 17_ GA7-220501096-AA4-EV01_IC_ Taller sobre componentes frontend.
        - RAP 17_ GA7-220501096-AA4-EV02_RM_ Verificación de procedimientos para la definición de componentes frontend de la aplicación (Listas de chequeo).
        - RAP 17_ GA7-220501096-AA4-EV03_IC_ Componente frontend del proyecto formativo y proyectos de clase (Listas de chequeo).

ENLACE GRABACIÓN:

https://drive.google.com/file/d/1hkRNBNFO6-UycKr92ISmsT9CLvZhMAFE/view?usp=sharing

RESUMEN Y GRABACIÓN CONFERENCIA WEB RAP 17 COMPONENTE FRONT-END 03/12/2024

- ✓ **RAP 17_ GA7-220501096-AA4-EV01_IC_ Taller sobre componentes frontend.**

Con base en lo visto en el componente formativo “Desarrollo de frontend con React JS”, realizar un documento que responda los aspectos descritos a continuación:

1. Diferencia entre React y JSX
2. ¿Qué son clases en React?
3. Principales eventos de React
4. Mapa conceptual de React

EVIDENCIA.
Documento Word o pdf.

- ✓ Portada
- ✓ Introducción
- ✓ Tabla de contenido
- ✓ Objetivos
- ✓ Desarrollo “Taller sobre componente frontend.”
- ✓ Conclusiones
- ✓ Referencias bibliográficas.

Nota: enviar archivo .zip con la evidencia y los archivos ejecutables de la actividad

1. Diferencia entre React y JSX
React:

- Es una biblioteca de JavaScript desarrollada por Facebook para construir interfaces de usuario (UI).
- Proporciona las herramientas necesarias para crear componentes reutilizables, administrar el estado de las aplicaciones y actualizar la UI de manera eficiente.
- React es el "motor" que permite estructurar y manejar tu aplicación.

JSX (JavaScript XML):

- Es una extensión de sintaxis para JavaScript que permite escribir código similar a HTML dentro de los archivos JavaScript.
- JSX hace que sea más fácil construir componentes visuales al combinar el diseño HTML con la lógica de React.
- Aunque React puede funcionar sin JSX, este hace que el código sea más legible y cómodo.

2. ¿Qué son clases en React?
En React, clases son una forma de definir componentes. Aunque React utiliza cada vez más funciones para crear componentes (funcionales), las clases fueron muy comunes en versiones anteriores.

Componentes de clase:

- Son componentes definidos como una clase de JavaScript que extiende de React.Component.

- Incluyen un método obligatorio llamado render() que define qué debe mostrarse en la UI.
- Pueden tener su propio estado y utilizar métodos del ciclo de vida de React.

Ejemplo:

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

class MiComponente extends Component {
  constructor(props) {
    super(props);
    this.state = { mensaje: 'Hola, estudiantes!' };
  }

  render() {
    return <h1>{this.state.mensaje}</h1>;
  }
}

export default MiComponente;
```

3. Principales eventos en React

React permite manejar eventos de la UI de forma similar a los eventos de JavaScript nativo, pero con una sintaxis especial. Algunos eventos comunes son:

Evento	Uso	Ejemplo
onClick	Se ejecuta al hacer clic.	<button onClick={handleClick}>Clic</button>
onChange	Detecta cambios en inputs.	<input onChange={handleChange} />
onSubmit	Captura el envío de formularios.	<form onSubmit={handleSubmit}>...</form>
onMouseOver	Detecta cuando el mouse pasa sobre un elemento.	Texto
onKeyPress	Detecta teclas presionadas.	<input onKeyPress={handleKeyPress} />

Ejemplo:

```
function MiBoton() {
  function manejarClick() {
    alert('¡Hiciste clic!');
  }

  return <button onClick={manejarClick}>Clic aquí</button>;
```

}

4. Elementos para crear un mapa conceptual de React

Un mapa conceptual te ayuda a visualizar los elementos clave de React y cómo se relacionan. Aquí hay una descripción de un posible mapa:

1. React
    - Componentes
        - Funcionales (usando funciones y hooks como useState y useEffect).
        - De clase (usando class y this.state).
    - JSX
        - Sintaxis para combinar HTML y JavaScript.
    - Estado (State)
        - Controla los datos internos del componente.
    - Props (Propiedades)
        - Permiten pasar datos entre componentes.
    - Eventos
        - Interacciones de usuario (clics, teclado, etc.).
    - Hooks
        - Ejemplo: useState, useEffect, useContext.
    - Virtual DOM
        - Mecanismo para actualizar eficientemente la UI.

****

- ✓ **RAP 17_ GA7-220501096-AA4-EV02_RM_ Verificación de procedimientos para la definición de componentes frontend de la aplicación**

Teniendo en cuenta las características del software a desarrollar, realice un documento en donde describa los componentes frontend a utilizar en la codificación del proyecto, cada uno de los componentes seleccionados debe tener su respectiva justificación

EVIDENCIA.
Documento Word o pdf.

- ✓ Portada
- ✓ Introducción
- ✓ Tabla de contenido
- ✓ Objetivos
- ✓ Desarrollo “Verificación de procedimientos para la definición de componentes frontend de la aplicación”
- ✓ Conclusiones
- ✓ Referencias bibliográficas.

Nota: enviar archivo .zip con la evidencia y los archivos ejecutables de la actividad

La actividad "Verificación de procedimientos para la definición de componentes frontend de la aplicación" se refiere al proceso de revisión y aseguramiento de calidad en el diseño y estructura de los componentes de la interfaz de usuario (frontend) de una aplicación. Esta actividad asegura que los componentes estén correctamente definidos, funcionales y alineados con los objetivos del proyecto y las mejores prácticas.

Puntos clave que abarca:

1. **Identificación de Componentes**
   Verificar que los componentes necesarios han sido identificados (como Navbar, Footer, Cards, etc.) y que cumplen con los requisitos funcionales y de diseño establecidos.
2. **Estandarización de Procedimientos**
   Asegurarse de que los procedimientos para la creación y configuración de componentes sigan estándares específicos, como principios de diseño modular, reutilización de código y alineación con guías de estilo.
3. **Validación del Código**
   Revisar que el código de cada componente esté limpio, bien documentado y optimizado. Esto incluye validar la sintaxis (HTML, CSS, JavaScript, o frameworks como React), estructuras, y cumplimiento de estándares como W3C.
4. **Compatibilidad**
   Comprobar que los componentes funcionan correctamente en diferentes navegadores y dispositivos (diseño responsivo).
5. **Integración y Funcionamiento**

Verificar que los componentes interactúan de manera correcta con otros elementos del sistema, como la lógica del backend o las API utilizadas.

6. **Alineación con el Diseño**
   Confirmar que los componentes se ajusten al diseño previamente definido (mockups o wireframes) en términos de estilo, funcionalidad, y usabilidad.
7. **Pruebas de Usabilidad**
   Realizar pruebas básicas para garantizar que los componentes son intuitivos y cumplen con la experiencia de usuario (UX) deseada.

Objetivo:
Asegurar que la definición de los componentes frontend sea sólida, funcional y eficiente, para reducir errores y garantizar un desarrollo más fluido de la aplicación. Si esta actividad está dentro de un contexto específico (educativo, profesional, de un proyecto), puedo personalizar aún más esta explicación

Los componentes frontend más utilizados, incluyendo básicos como barra de navegación, header y footer, junto con su descripción y aplicación:

Componente Frontend	Descripción	Aplicación
Header	Sección superior de una página que generalmente incluye el título, logotipo, y enlaces principales.	Mostrar la identidad del sitio, navegación primaria o accesos rápidos.
Navbar	Barra de navegación utilizada para proporcionar enlaces a diferentes secciones o páginas.	Navegación entre páginas de sitios web y aplicaciones. Ejemplo: barra de menú superior.
Footer	Sección al final de la página que contiene información adicional como derechos de autor o enlaces.	Mostrar políticas, contacto, enlaces útiles o redes sociales.
Sidebar	Panel lateral que contiene menús, herramientas o contenido adicional.	Navegación secundaria, herramientas rápidas o filtros en dashboards y aplicaciones complejas.
Buttons	Elementos interactivos que permiten disparar acciones al ser clickeados.	Iniciar procesos, enviar formularios o activar funciones específicas.
Forms	Conjuntos de campos que recopilan datos del usuario.	Registro de usuarios, inicio de sesión o recopilación de datos en formularios de contacto.
Cards	Contenedores visuales compactos que agrupan contenido relacionado.	Presentar productos, servicios o publicaciones en galerías o listados.
Tables	Elementos estructurados para organizar datos en filas y columnas.	Mostrar reportes, listas de usuarios, inventarios u otra información tabular.
Modals	Ventanas emergentes superpuestas a la interfaz principal para mostrar contenido o acciones.	Confirmaciones, formularios rápidos o mensajes importantes sin redirigir la página.
Dropdowns	Menús desplegables que muestran opciones adicionales al interactuar.	Selección de opciones en configuraciones, filtros o listas de acciones.
Alerts	Mensajes de notificación para informar al usuario sobre cambios o eventos importantes.	Mostrar errores, advertencias, confirmaciones o éxitos en procesos interactivos.

Tabs	Componentes que organizan contenido en secciones navegables.	Alternar entre vistas o categorías, como configuraciones por secciones o perfiles de usuario.
Carousels	Contenedores deslizables que presentan contenido como imágenes o textos en secuencia.	Mostrar productos destacados, promociones o galerías de imágenes.
Icons	Pequeños gráficos o símbolos que representan acciones o conceptos.	Indicar funciones de forma visual, como buscar, configurar, guardar o eliminar.
Progress Bars	Indicadores visuales que muestran el avance de una tarea o proceso.	Mostrar el progreso en procesos como carga de archivos o envío de formularios largos.
Tooltips	Mensajes flotantes que proporcionan información adicional al pasar el mouse sobre un elemento.	Explicar la funcionalidad de botones o enlaces sin sobrecargar la interfaz.
Breadcrumbs	Indicadores de navegación que muestran la ruta actual dentro de la jerarquía del sitio.	Ayudar a los usuarios a orientarse y regresar a secciones anteriores.
Accordions	Componentes que expanden o contraen contenido para mostrar detalles adicionales.	Presentar preguntas frecuentes, listas extensas o contenido de manera organizada y compacta.
Search Bars	Campo de entrada que permite a los usuarios buscar contenido específico dentro de un sitio.	Búsquedas en catálogos, bases de datos o cualquier contenido disponible en el sitio web.
Sliders	Controles interactivos que permiten seleccionar valores en un rango continuo.	Ajustar configuraciones como precios, volumen o tiempo en filtros y configuraciones.
Pagination	Navegación estructurada que divide contenido en páginas para una mejor usabilidad.	Gestionar listas largas como catálogos de productos, publicaciones o resultados de búsqueda.

****

- ✓ **RAP 17_ GA7-220501096-AA4-EV03_IC_ Componente frontend del proyecto formativo y proyectos.**

Teniendo en cuenta el tipo de proyecto a desarrollar móvil o web realice la codificación del módulo del proyecto aplicando lo visto en el componente "Desarrollo de frontend con React JS."

EVIDENCIA.
Documento Word o pdf.

- ✓ Portada
- ✓ Introducción
- ✓ Tabla de contenido
- ✓ Objetivos
- ✓ Desarrollo "Componente frontend del proyecto formativo y proyectos"
- ✓ Conclusiones
- ✓ Referencias bibliográficas.

Nota: enviar archivo .zip con la evidencia y los archivos ejecutables de la actividad

Para la codificación del módulo debe tener en cuenta los artefactos del ciclo del software realizados con anterioridad: diagrama de clases, diagramas de casos de uso, historias de usuario, diseños, prototipos, Informe técnico de plan de trabajo para construcción de software con tecnologías seleccionadas etc.

- El código debe contener comentarios
- El código debe cumplir con estándares de codificación
- Se debe crear el proyecto utilizando herramientas de versionamiento.