DuocUC® INFORMÁTICA Y TELECOMUNICACIONES

Diseño Responsivo

- Desarrollo Fullstack II
- DSY1104

FORMAMOS PERONAS PARA UNA SOCIEDAD MEJOR

01

Repaso de la sesión anterior

Repaso sesión anterior

REFLEXIONEMOS

1. **¿Cuál es el propósito principal de React?**

Reflexiona sobre cómo React facilita el desarrollo de interfaces de usuario y su papel en las aplicaciones de una sola página (SPA).

2. **¿Por qué es importante que React sea una biblioteca de código abierto?**

Considera los beneficios y desafíos del uso de software de código abierto en el desarrollo de aplicaciones web.

3. **¿Cómo contribuye la capacidad de React para crear componentes reutilizables al desarrollo de aplicaciones web modernas?**

Piensa en ejemplos específicos donde los componentes reutilizables pueden mejorar la eficiencia y la mantenibilidad del código.

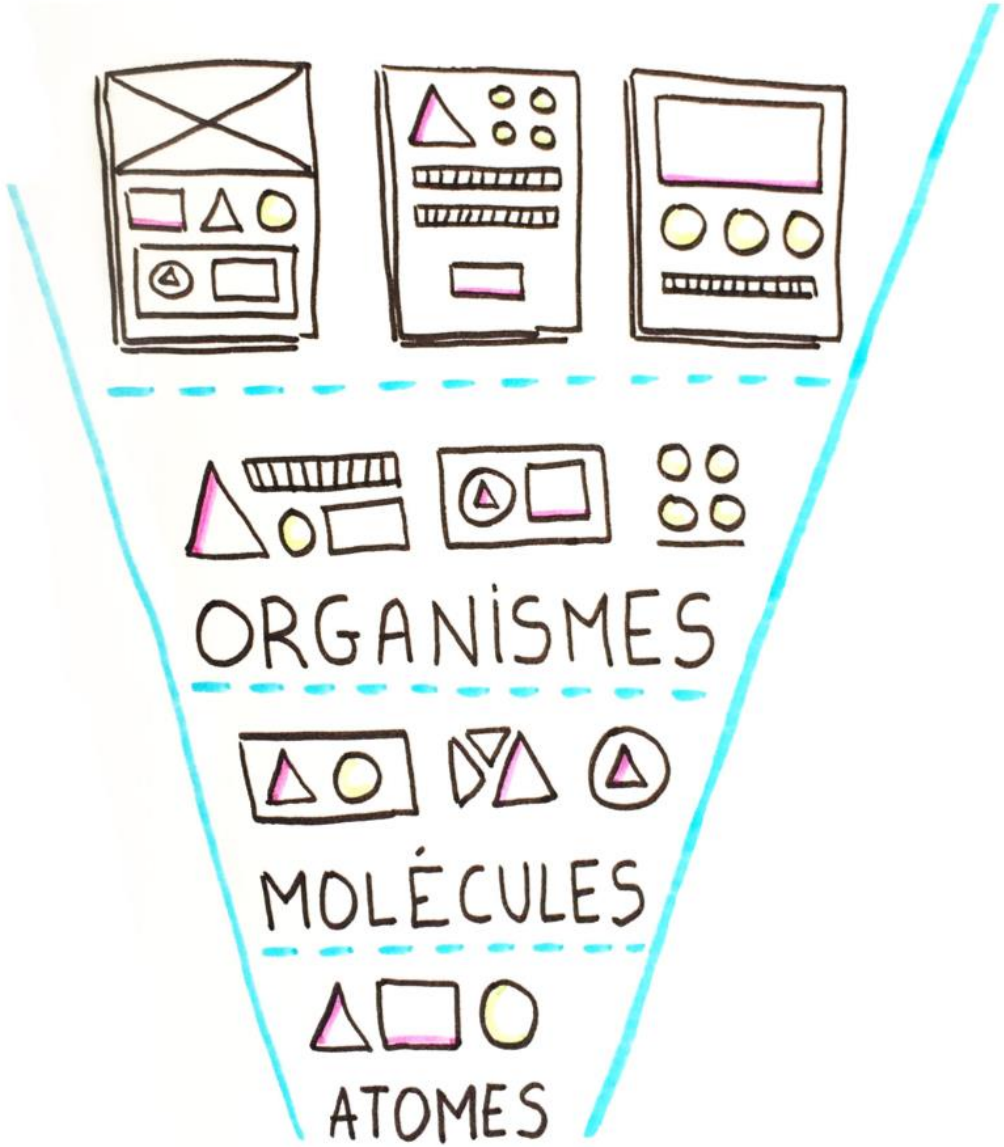
02

Explorando React – Parte II

• Atomic Design

Es una metodología propuesta por el diseñador Brad Frost para crear sistemas de diseño eficientes y escalables.

Esta metodología se basa en la idea de descomponer una interfaz en sus elementos más básicos, llamados "átomos", y luego combinarlos en componentes más complejos.



Átomos

Los elementos más básicos, como botones, etiquetas, o entradas de texto.

Moléculas

Combinaciones simples de átomos que forman unidades funcionales, como un formulario de búsqueda con una etiqueta y un campo de entrada.

Organismos

Componentes más complejos compuestos por múltiples moléculas y átomos, como una barra de navegación.

Plantillas

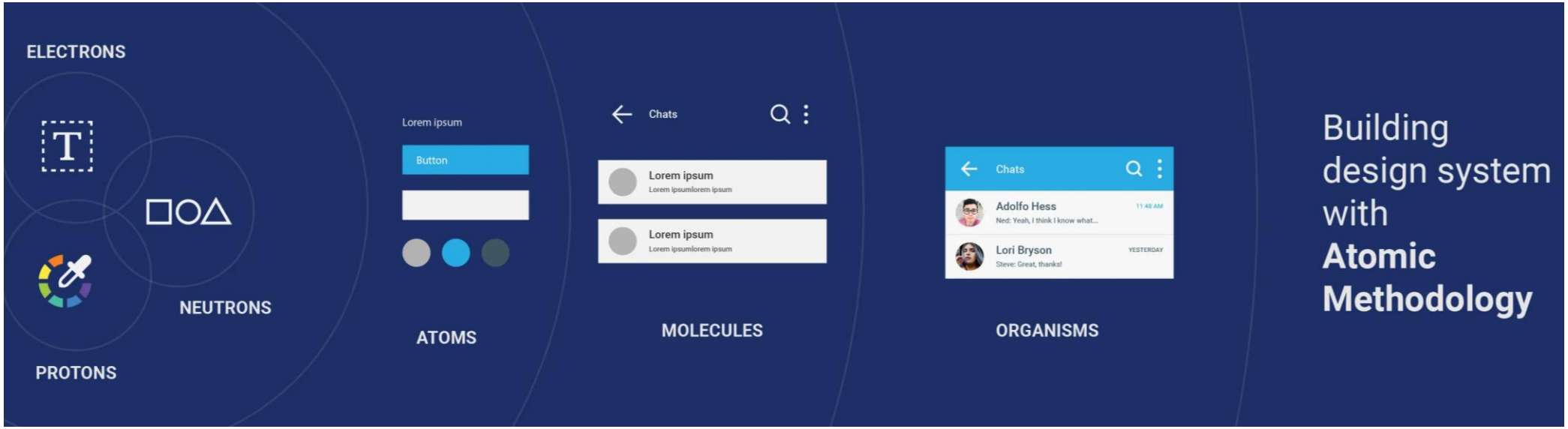
Estructuras de página que contienen grupos de organismos, mostrando el diseño general.

Páginas

Versiones finales de las plantillas con contenido real, representando el producto terminado.

• Ventajas de Atomic Design

- Consistencia en el diseño
- Reutilización de componentes
- Facilita la colaboración entre diseñadores y desarrolladores



• ¿Qué son los componentes?

Son bloques de construcción de la interfaz de usuario, encapsulan la lógica y la presentación.

Tipos de componentes

- Componentes funcionales
- Componentes de clase

Ejemplo de un componente funcional ***.jsx***

```jsx
function Greeting(props) {
    return <h1>Hola, {props.name}!</h1>;
}
```

• Props y State

Props

- Datos pasados de un componente padre a un hijo
- Inmutables dentro del componente

State

- Datos gestionados internamente por el componente
- Pueden cambiar con el tiempo

```
function Counter() {
  const [count, setCount] = useState(0);
  return (
    <div>
      <p>Has hecho clic {count} veces</p>
      <button onClick={() => setCount(count + 1)}>
        Haz clic
      </button>
    </div>
  );
}
```

Ejemplo de uso de ***props*** y ***state***

• Persistencia en JavaScript

Métodos de almacenamiento local

- localStorage
- sessionStorage

Ejemplo de uso de localStorage ***.js***

```
// Guardar datos
localStorage.setItem('usuario', JSON.stringify({nombre: 'Juan', edad: 30}));
// Recuperar datos
const usuario = JSON.parse(localStorage.getItem('usuario'));
```

• Navegación con React Router

React Router es una biblioteca de enrutamiento para aplicaciones React que facilita la creación de aplicaciones de una sola página (SPA) con múltiples rutas. Permite la navegación dinámica sin recargar la página completa, proporcionando una experiencia de usuario fluida y rápida.

Ejemplo de configuración de rutas

```jsx
import { BrowserRouter, Route, Link } from 'react-router-dom';

function App() {
  return (
    <BrowserRouter>
      <nav>
        <Link to="/">Inicio</Link>
        <Link to="/about">Acerca de</Link>
      </nav>
      <Route exact path="/" component={Home} />
      <Route path="/about" component={About} />
    </BrowserRouter>
  );
}
```

React Router permite definir y gestionar rutas en una aplicación React. Algunas de sus características principales incluyen:

- Declaración de rutas
- Navegación dinámica
- Parámetros de ruta
- Anidación de rutas
- Protección de rutas

Componentes principales de React Router

- BrowserRouter
- Route
- Link

• Tomar parámetros GET del navegador

Para acceder a los parámetros de la URL, podemos utilizar los hooks ***useParams*** y ***useLocation*** de React Router.

El hook **useParams** se utiliza para acceder a los parámetros de la ruta. Este hook devuelve un objeto de parámetros de la URL.

Ejemplo de uso de useParams

```
import { useParams } from 'react-router-dom';

function UserProfile() {
  let { id } = useParams();
  return <div>Perfil del usuario {id}</div>;
}
```

Supongamos que tenemos una ruta definida como ***/user/:id***, donde ***:id*** es un parámetro de la URL. Podemos acceder a este parámetro en un componente de la siguiente manera:

jsx

```jsx
import React from 'react';
import { useParams } from 'react-router-dom';

const UserDetail = () => {
  const { id } = useParams();

  return (
    <div>
      <h1>User ID: {id}</h1>
    </div>
  );
};

export default UserDetail;
```

```jsx
import React from 'react';
import { useLocation } from 'react-router-dom';

const useQuery = () => {
  return new URLSearchParams(useLocation().search);
};

const SearchPage = () => {
  const query = useQuery();
  const name = query.get('name');
  const age = query.get('age');

  return (
    <div>
      <h1>Search Parameters:</h1>
      <p>Name: {name}</p>
      <p>Age: {age}</p>
    </div>
  );
};

export default SearchPage;
```

El hook ***useLocation*** se utiliza para acceder a la ubicación actual, que incluye la URL completa, el objeto de búsqueda, etc. Si deseas acceder a los parámetros de búsqueda de la URL (query params), puedes usar useLocation, por ejemplo:

← Ejemplo de useLocation

Combinando ambos hooks useParams y useLocation.

Si necesitas acceder tanto a los parámetros de la ruta como a los parámetros de búsqueda:

```jsx
import React from 'react';
import { useParams, useLocation } from 'react-router-dom';

const useQuery = () => {
  return new URLSearchParams(useLocation().search);
};

const UserDetail = () => {
  const { id } = useParams();
  const query = useQuery();
  const referrer = query.get('referrer');

  return (
    <div>
      <h1>User ID: {id}</h1>
      <p>Referrer: {referrer}</p>
    </div>
  );
};

export default UserDetail;
```

• Configuración del Router

Asegúrate de tener configurado tu router de manera adecuada para que tus rutas funcionen correctamente. Un ejemplo básico:

```jsx
import React from 'react';
import { BrowserRouter as Router, Route, Switch } from 'react-router-dom';
import UserDetail from './UserDetail';
import SearchPage from './SearchPage';

const App = () => {
  return (
    <Router>
      <Switch>
        <Route path="/user/:id" component={UserDetail} />
        <Route path="/search" component={SearchPage} />
      </Switch>
    </Router>
  );
};

export default App;
```

03

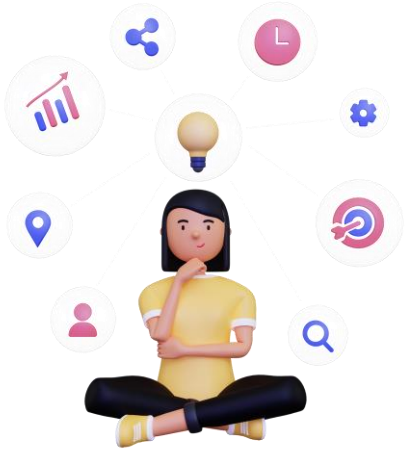
Actividad 2.2.2

ESCUELA DE INFORMÁTICA Y TELECOMUNICACIONES

Actividad 2.2.2

Ingresa al AVA de esta Actividad y desarrolla la actividad descrita en la guía 2.2.2 Actividad Individual creación de sitios web responsivos

Consulta con tu docente las dudas que tengas al desarrollar la actividad.



DuocUC®

• Bibliografía

Libros Digitales Biblioteca Duoc. (Con tu cuenta de Duoc puedes consultar)

- Larsson, M. (2023). Microservices with Spring Boot 3 and Spring Cloud: Build resilient and scalable microservices using Spring Cloud, Istio, and Kubernetes (3a ed.). Packt Publishing.

Recursos de información.

- “REACT - W3Schools." https://www.w3schools.com/react/ Se consultó el 28 julio 2024.
- Curso de React Básico Gratis: https://codigofacilito.com/programas/react-g3

Duoc UC®

CERCANÍA. LIDERAZGO. FUTURO.

duoc.cl

7 AÑOS ACREDITADO

Comisión Nacional de Acreditación CNA-Chile

DESDE AGOSTO 2017 HASTA AGOSTO 2024.
DOCENCIA DE PREGRADO. GESTIÓN INSTITUCIONAL. VINCULACIÓN CON EL MEDIO.