



MANUAL TÉCNICO

PROTOTIPO DEL GEOPORTAL

UNIVERSIDAD CATÓLICA DE CUENCA
CUENCA - ECUADOR
2024

Tabla de Contenido

INTRODUCCIÓN	3
OBJETIVOS.....	4
1. REQUERIMIENTOS TÉCNICOS.....	5
2. HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO DEL GEOPORTAL.....	6
Visual Studio Code.....	6
Geoserver	6
GitHub	6
3. ARQUITECTURA DEL SISTEMA.....	7
4. COMPONENTES DEL SOFTWARE	9
Catálogo de Proyectos.....	9
Geoservicios	11
Manual	16
Geovisor	20
Contactos	23

INTRODUCCIÓN

En este manual describe los pasos necesarios para cualquier persona que tengan ciertas bases de sistemas para que pueda realizar los cambios, la edición de todo el prototipo.

En este manual se mencionan las especificaciones mínimas de hardware y software para el correcto uso del prototipo. Y describe las herramientas clave utilizadas en el desarrollo, como Visual Studio Code, GeoServer, y GitHub. Explica brevemente la arquitectura del sistema y detalla las tecnologías y frameworks implementados, incluyendo Node.js, React, y Tailwind CSS, entre otros. Además, proporciona una visión general de los componentes principales del software, como el catálogo de proyectos, los geoservicios, y el geovisor, enfocándose en su propósito y contribución al proyecto. Este manual sirve como una referencia esencial para realizar cambios y ediciones en el prototipo, asegurando su correcta utilización.

OBJETIVOS

- Brindar la información necesaria para poder realizar la instalación y configuración del prototipo.
- Describir las herramientas utilizadas para el diseño y desarrollo del prototipo.
- Brindar instrucciones paso a paso y orientación práctica que permitan a los usuarios realizar cambios, ediciones y personalizaciones en el prototipo, promoviendo la adaptabilidad y mejora continua del sistema.

1. REQUERIMIENTOS TÉCNICOS

REQUERIMIENTOS MÍNIMOS DE HARDWARE

- **Procesador:** 1.6 GHz or faster processor
- **Memoria RAM:** 1 GB of RAM
- **Memoria Interna:** 240Gb

REQUERIMIENTOS MÍNIMOS DE SOFTWARE

- **Sistema Operativo:**
Windows 10 and 11 (64-bit)
macOS versions con Apple security update support.
Linux (Debian): Ubuntu Desktop 20.04, Debian 10
Linux (Red Hat): Red Hat Enterprise Linux 8, Fedora 36

2. HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO DEL GEOPORTAL

Visual Studio Code

Microsoft lo desarrolló, un editor avanzado de código fuente distribuido de forma gratuita y compatible con sistemas operativos como Windows, Linux y macOS. Este editor se caracteriza por su capacidad para facilitar el proceso de desarrollo de software, gracias a características como la depuración integrada, el manejo eficiente de versiones con Git, el resaltado de sintaxis, y el autocompletado de código, conocido también como IntelliSense. Además, ofrece personalizar la interfaz de usuario, los atajos de teclado y las configuraciones de trabajo mediante un amplio catálogo de extensiones disponibles en su tienda en línea, adaptando el entorno de desarrollo a muchos lenguajes de programación y marcos de trabajo. Visual Studio Code es una herramienta indispensable para los desarrolladores que buscan optimizar su flujo de trabajo, desde la edición de simples fragmentos de código hasta la gestión de proyectos complejos, promoviendo la colaboración y la eficiencia en equipos de desarrollo.

Geoserver

GeoServer es un servidor de software libre enfocado en la compartición y edición de datos geoespaciales, promoviendo la interoperabilidad mediante la adhesión a los estándares abiertos del Open Geospatial Consortium (OGC), como Web Feature Service (WFS), Web Map Service (WMS), y Web Coverage Service (WCS). Esta plataforma facilita el acceso y análisis de información geográfica a través de internet, permitiendo a los usuarios visualizar mapas y conectar con diversas fuentes de datos geoespaciales, incluyendo bases de datos PostGIS, archivos shapefile y mosaicos de imágenes. GeoServer es una herramienta esencial en el ámbito de los Sistemas de Información Geográfica (SIG) para el desarrollo de aplicaciones web GIS, publicación de datos espaciales en la red y la integración de estos datos en diferentes plataformas y servicios, sirviendo como solución versátil para organizaciones que desean compartir información geográfica ampliamente o facilitar la colaboración en equipos de trabajo.

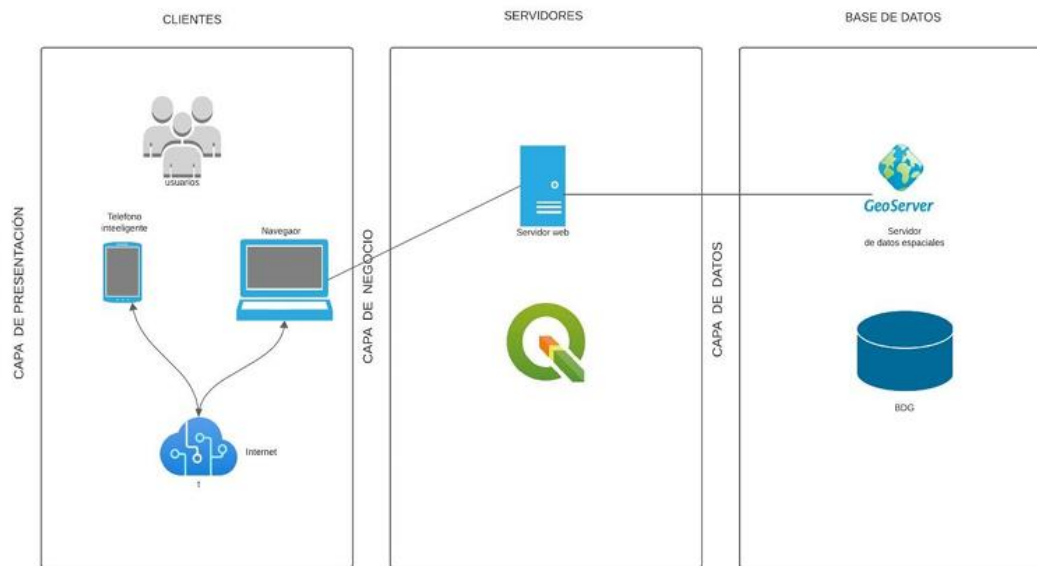
GitHub

Se trata de una plataforma de desarrollo colaborativo diseñada para alojar proyectos de software, aprovechando el sistema de control de versiones Git. Esta herramienta es fundamental para gestionar repositorios de código, ofreciendo diversas funcionalidades que facilitan el trabajo en equipo dentro de un proyecto. Uno de los usos más frecuentes es mantener el repositorio actualizado con las contribuciones de cada miembro del equipo, para lo cual se emplean comandos como git commit, git merge, git pull y git push, entre otros, que permiten integrar y compartir el trabajo realizado de manera eficiente.

Para la IDE-UCACUE, se gestiona un repositorio privado al que se le aportan diversos commits con ajustes o modificaciones efectuadas en el software, facilitando así la colaboración y el trabajo en equipo en el desarrollo de la aplicación.

3. ARQUITECTURA DEL SISTEMA

Diagramas de Arquitectura: Incluye diagramas que ilustran la estructura del sistema.



Tecnologías Utilizadas: Detalla las tecnologías, frameworks y herramientas implementadas.

Herramienta	¿Qué es?	Uso	Importancia	Función Principal	Recurso
Node.js	Entorno de ejecución para JS	Desarrollo de aplicaciones web y servidores	Fundamental en el desarrollo del lado del servidor	Permite ejecutar código JavaScript fuera del navegador	https://nodejs.org/en
React	Biblioteca de JavaScript	Construcción de interfaces de usuario interactivas	Dominante en el desarrollo de interfaces web	Facilita la creación de componentes reutilizables en aplicaciones web	https://es.react.dev/
Tailwind CSS	Framework de estilos en CSS	Estilización rápida y consistente de interfaces	Simplifica la gestión de personalización del estilo	Proporciona clases y utilitarias para estilos predefinidos	https://tailwindcss.com/
Leaflet	- Es una biblioteca de JavaScript. - Se utiliza para crear mapas interactivos en la web.	Permite a los usuarios interactuar con los mapas, como hacer zoom, arrastrar, y agregar marcadores	Proporciona una solución sencilla y eficiente para la integración de mapas.	Permite personalizar mapas con varias capas, como imágenes satelitales, rutas de transporte, y otros puntos de interés.	https://leafletjs.com/

Nodemailer	<p>Librería de Node.js para enviar emails.</p>	<p>Enviar emails desde aplicaciones Node.js.</p>	<p>Notificar a usuarios vía email (registros, contraseñas, notificaciones).</p>	<p>Facilitar envío de emails vía SMTP o servicios de correo.</p>	<p>https://www.nodemailer.com/</p>
------------	------------------------------------------------	--------------------------------------------------	---------------------------------------------------------------------------------	------------------------------------------------------------------	------------------------------------------------------------------------------

4. COMPONENTES DEL SOFTWARE

Descripción de Componentes: Enumera y describe cada componente del software y su propósito.

Catálogo de Proyectos

En este componente se muestra lo que son la categoría de proyectos, publicaciones destacadas, los filtros y un ejemplo de proyecto que pertenece a esa categoría.

Categoría de Proyectos

Este fragmento de código está dentro de la clase “**categories.jsx**” la cual esta parte define una constante “**categories**” la cual es un arreglo que contiene objetos. Cada objeto representa una categoría de información o funcionalidad dentro de una aplicación, que está relacionada con información geográfica o servicios de mapas.

Documentación Específica

- **Propósito del Arreglo “categories”:** Explicar que este arreglo tiene como objetivo agrupar y organizar distintas categorías de información o funcionalidades que ofrece la aplicación. Es especialmente útil para la construcción dinámica de menús, navegación, o presentaciones de categorías en la interfaz de usuario.
- **Usos de la Interfaz de Usuario:** Sugerir formas en las que este arreglo podría ser utilizado en la interfaz de usuario, como la generación de una lista o cuadrícula de categorías, incluyendo la presentación de la imagen, el nombre, y la descripción de cada categoría.

```
const categories = [  
  {  
    id: 1,  
    name: "Riesgos",  
    description: "Consulta los datos espaciales relacionados con los riesgos",  
    image: imgriesgos,  
    to: "/projects/riesgos",  
  },  
]
```

Publicaciones Destacadas

Este fragmento de código está dentro de la clase “**CatalogoPage.jsx**” es donde se muestra de que se trata las publicaciones destacadas relacionadas con datos geoespaciales, y luego de eso está la línea de código que es el componente “**Post**” donde es instanciado, pasando por un arreglo que contiene el primer elemento de “**proyectos**” y arreglo completo “**categories**”. Esto permite que el componente “**Post**” tenga acceso a esos datos para su funcionamiento interno, como la renderización de información sobre el primer proyecto y categorizarlo o filtrarlo basado en las “**categories**” proporcionadas.

```

<div className="mx-auto mt-20 w-full max-w-[1100px] px-3 pb-20">
  <h3 className="mb-6 text-center text-4xl font-semibold text-black dark:text-white">
    Publicaciones destacadas
  </h3>
  <p className="lead pb-10 pt-5 text-justify font-sans dark:text-white">
    En esta sección, le ofrecemos un vistazo a las publicaciones
    destacadas de datos geoespaciales derivados de investigaciones
    recientes que están transformando nuestra comprensión del mundo que
    habitamos. Desde estudios de cambio climático hasta investigaciones en
    biodiversidad y urbanismo sostenible. Estas publicaciones representan
    la vanguardia del conocimiento geoespacial.
  </p>
  <Post projects={projectos.slice(0, 1)} categories={categories} />
</div>

```

Filtros

Este fragmento de código está dentro de la clase “**ProjectPage.jsx**” tiene la función para manejar la selección de una opción por parte del usuario, filtrar un conjunto de datos basándose en esta selección y actualizar el estado de la aplicación para reflejar los resultados filtrados.

- **Filtrado de Proyectos:** La función primero verifica si el valor seleccionado es diferente de "Seleccione la fuente". Si es así, procede a filtrar el arreglo “**projects**”, buscando proyectos cuya propiedad “**fuentes**” contenga al menos una coincidencia con el valor seleccionado. La comparación se realiza de manera insensible a mayúsculas y minúsculas.

```

const handleOptionChange = (selectedValue) => {
  setSelectedOption(selectedValue);

  // Filtra los proyectos según la fuente seleccionada
  if (selectedValue !== "Seleccione la fuente") {
    const filteredProjects = projects.filter((project) =>
      project.fuentes.some(
        (fuente) => fuente.name.toLowerCase() === selectedValue.toLowerCase()
      )
    );
    setItems(filteredProjects);
  } else {
    // Si se selecciona "Seleccione la fuente", muestra todos los proyectos sin filtrar
    setItems(projects);
  }

  <span className="mr-2 text-[12px]">
    <strong>Fuente:</strong> {selectedValue}
  </span>
};

```

Búsqueda de Proyectos

Este fragmento de código que utiliza hooks para gestionar el estado y la lógica de visualización de una lista paginada de proyectos, junto con capacidades de filtrado y búsqueda.

- **Documentación Específica:** Describir cómo los hooks de estado se utilizan para manejar la interactividad de la lista de proyectos, incluyendo la paginación, el filtrado y la búsqueda.

```

const [items, setItems] = useState(projects); // Lista original de proyectos
const [currentPage, setCurrentPage] = useState(0);
const itemsPerPage = 9; // item por página
const pageCount = Math.ceil(items.length / itemsPerPage);
const [selectedOption, setSelectedOption] = useState(""); // Filtro de fuente
const [filtros, setFiltros] = useState([]); // Lista de filtros seleccionados

const handlePageClick = ({ selected }) => {
  setCurrentPage(selected);
};

const startIndex = currentPage * itemsPerPage;
const endIndex = startIndex + itemsPerPage;
const itemsToDisplay = items.slice(startIndex, endIndex);

//Busqueda de proyectos
const search2 = (searchQuery) => {
  // Filtra los elementos que coinciden con la consulta de búsqueda en el título
  const filteredItems = projects.filter(
    (item) =>
      item.title.toLowerCase().includes(searchQuery.toLowerCase()) ||
      item.description.toLowerCase().includes(searchQuery.toLowerCase())
  );
  // Actualiza la lista de elementos y restablece la página actual
  setItems(filteredItems);
  setCurrentPage(0);
};

```

Ejemplo de Proyecto

Este fragmento de código está dentro de la clase “**projects.jsx**” que contiene objetos, cada uno representando un proyecto específico. Este arreglo está orientado a ser utilizado en un contexto donde se manejan datos geoespaciales o proyectos relacionados con la geografía, como se indica por el contenido del primer objeto del arreglo.

- **Estructura del Objeto de Proyecto:** Cada objeto dentro del arreglo “**projects**” incluye varias propiedades que describen detalles importantes de un proyecto.

```

const projects = [
  {
    id: 1,
    title:
      "Implementación de un geovisor para la visualización de las propiedades geodinámicas y geomorfológicas del subsuelo: caso de estudio Cuenca, Azuay, Ecuador",
    imageUrl: imgjoseproject,
    publication: "2023-08-03",
    description:
      "El crecimiento urbano y la falta de estudios de suelos adecuados antes de construir edificaciones generan vulnerabilidad estructural. Para abordar este problema, se propone la creación de una base
    fuente: [
      { id: 1, name: "SIGDATA" },
      { id: 2, name: "C2MAD" },
    ],
    layeritem: "Leer más",
    category: 1,
    autores: [{ id: 2, name: "José Guaman" }],
    to: "/mapas/jose",
    visitas: 0,
  },
];

```

Geoservicios

En esta parte encontramos los servicios web que utiliza este geoportal, nuestro proyecto de los geoservicios tiene el nombre de “**GeoservicesPage.jsx**”, para iniciar la estructura de la clase vamos a ver cada una de las partes.

Descripción

Este fragmento de código describe una sección de una interfaz de usuario diseñada para la plataforma que trabaja con información geográfica (IG). La sección se titula "GEOSERVICIOS" y proporciona una breve descripción de lo que implica un sistema de geoservicios. A continuación, se ofrece una guía general para documentar esta sección de código:

Descripción General

- **Propósito y Contenido:** La sección está destinada a informar al usuario acerca de los geoservicios.

Elementos del Código

- **Encabezado h3:** El encabezado `<h3>` destaca el título de la sección, "GEOSERVICIOS", utilizando clases de Tailwind CSS para estilización (`p-8`, `text-base`, `font-bold`, `dark:text-white`). Estas clases aplican un padding, establecen el tamaño del texto, el peso de la fuente, y ajustan el color del texto para modos claros y oscuros.
- **Descripción span:** El elemento `` contiene la descripción de los geoservicios. Al igual que el encabezado, este elemento utiliza clases de Tailwind CSS para asegurar que el texto sea legible en temas oscuros (`dark:text-white`).

```
return (
  <div className="mx-auto w-full max-w-[1400px] px-3">
    <h3 className="p-8 text-base font-bold dark:text-white">
      GEOSERVICIOS
    </h3>
    <span className="dark:text-white">
      Es un sistema compuesto por hardware, software y procedimientos
      diseñados para facilitar la captura, gestión, análisis y visualización
      de IG para resolver problemas de la planificación y gestión.
    </span>
  </div>
)
```

Pestañas

Este fragmento se utiliza para permitir a los usuarios cambiar entre diferentes vistas o secciones de contenido dentro de la misma página web. En este caso, está configurado para alternar entre diferentes tipos de servicios relacionados con sistemas de información geográfica: WMS (Web Map Service), WFS (Web Feature Service), y WCS (Web Coverage Service). A continuación, se detalla una guía general para documentar este código:

Estructura y Funcionamiento

- **Contenedor Principal:** Se usa un `"div"` con clases de Tailwind CSS para estilos de borde y margen, actuando como contenedor del componente de pestañas.

- **Lista de Pestañas:** Las pestañas están implementadas como elementos “li” dentro de un elemento “ul”, facilitando una estructura semántica accesible y clara. Cada pestaña es un botón que, al hacer clic, activa un cambio de la vista mostrada al usuario.
- **Estilos Condicionales:** Se utilizan plantillas literales y expresiones ternarias para aplicar estilos condicionales a las pestañas. Los estilos cambian dependiendo de cuál pestaña está activa (activeTab), lo que indica visualmente al usuario cuál sección está viendo actualmente.
- **Manejador de Eventos:** Cada botón de pestaña tiene un evento onClick que invoca la función handleTabChange, pasando el identificador del servicio correspondiente (WMS, WFS, WCS). Esta función es responsable de cambiar el estado del componente y actualizar la vista activa.

Consideraciones de Implementación

- **Flexibilidad:** Sugerir la posibilidad de extender el componente para incluir más pestañas o servicios, modificando la estructura de la lista y las funciones correspondientes.

```

{ /* Código de las pestañas */
<div className="mb-4 border-b ■ border-gray-200 □ dark:border-gray-700">
  <ul>
    <li className="me-2" role="presentation">
      <button
        className={`inline-block rounded-t-lg border-b-2 p-4 ${
          activeTab === "profile"
            ? "■ border-gray-700"
            : "border-transparent"
        }`}
        onClick={() => handleTabChange("WMS")}
      >
        Servicios WMS
      </button>
    </li>
    <li className="me-2" role="presentation">
      <button
        className={`inline-block rounded-t-lg border-b-2 p-4 ${
          activeTab === "dashboard"
            ? "■ border-gray-700"
            : "border-transparent"
        }`}
        onClick={() => handleTabChange("WFS")}
      >
        Servicios WFS
      </button>
    </li>
    <li className="me-2" role="presentation">
      <button
        className={`inline-block rounded-t-lg border-b-2 p-4 ${
          activeTab === "dashboard"
            ? "■ border-gray-700"
            : "border-transparent"
        }`}
        onClick={() => handleTabChange("WCS")}
      >
        Servicios WCS
      </button>
    </li>
  </ul>
</div>

```

Contenido de las Pestañas

Esta sección específica pertenece a la pestaña “**Profile**” de una página web y se centra en proporcionar información sobre el “**Servicio de Mapas Web**”, junto con un ejemplo específico de proyecto que utiliza WMS. A continuación, se detalla la estructura y el propósito de este código:

Estructura General

- **Descripción del WMS:** Se inicia con un párrafo breve que introduce qué es un WMS.
- **Tabla de Ejemplo:** A continuación, se presenta una tabla con estilo personalizado que contiene tres columnas: AUTOR, PROYECTO y URL. Esta tabla muestra un ejemplo de proyecto que utiliza WMS, incluyendo el nombre del autor (José Guaman), el nombre del proyecto, y una URL que parece apunta un servidor GeoServer configurado para ofrecer servicios WMS.

Elementos de Código

- **Uso de clases CSS para estilización:** El código hace un uso intensivo de clases de Tailwind CSS para aplicar estilos, como colores de texto (text-gray-500, dark:text-gray-400), fondos (bg-slate-400, dark:bg-gray-700), bordes (border-solid, dark:border-gray-700), y otros para espaciado, redondeado de bordes, etc.
- **Elementos React:** El fragmento está construido utilizando JSX, una extensión de sintaxis para React que permite escribir estructuras similares al HTML en el código JavaScript.

```

/* Contenido de las pestañas */
{activeTab === "WMS" && (
  <div className="rounded-lg bg-gray-50 p-4 dark:bg-gray-800">
    /* Contenido para la pestaña Profile */
    <p className="text-sm text-gray-500 dark:text-gray-400">
      <strong className="font-bold text-gray-800 dark:text-white">
        ¿Qué es WMS?
      </strong>
    </p>
    <p>
      <strong className="font-normal text-gray-800 dark:text-white">
        Un servicio de mapas web (WMS, por sus siglas en inglés Web
        Map Service) es un estándar que proporciona una interfaz de
        protocolo HTTP para solicitar imágenes de mapas
        georreferenciados de uno o varios servidores. Este estándar
        fue desarrollado por el Open Geospatial Consortium (OGC) y es
        ampliamente utilizado en la comunidad de información
        geográfica.{" "}
      </strong>
    </p>
    <p></p>
    <div className="rounded-lg border-solid p-4">
      <div className="overflow-x-auto">
        <table className="w-full rounded-lg border border-solid text-left text-sm text-gray-500 dark:text-gray-400">
          <thead className="bg-slate-400 text-xs uppercase text-white dark:bg-gray-700 dark:text-gray-200">
            <tr>
              <th scope="col" className="px-6 py-3">
                AUTOR
              </th>
              <th scope="col" className="px-6 py-3">
                PROYECTO
              </th>
              <th scope="col" className="px-6 py-3">
                URL
              </th>
            </tr>
          </thead>
          <tbody className="bg-gray-200">
            <tr className="border-b dark:border-gray-700 dark:bg-gray-800">
              <th
                scope="row"
                className="whitespace-nowrap px-6 py-4 font-medium text-gray-900 dark:text-white"
              >
                José Guaman
              </th>
              <td className="px-6 py-4">
                {" "}
                Implementación de un geovisor para la visualización de
                las propiedades geodinámicas y geomorfológicas del
                subsuelo: caso de estudio Cuenca, Azuay, Ecuador
              </td>
              <th>
                <a href="http://192.168.10.4:8085/geoserver/jose/ows?service=WMS&request=GetCapabilities">http://192.168.10.4:8085/geoserver/jose/ows?service=WMS&request=GetCapabilities</a>
              </th>
            </tr>
          </tbody>
        </table>
      </div>
    </div>
  </div>
)
)

```

Advertencia o Nota

Este fragmento es una advertencia o nota destinada a los usuarios, aclarando la función específica de un enlace proporcionado en el contexto de los geoservicios. A continuación, se detalla una guía para documentar esta sección de código:

Descripción General

- Propósito de la Nota:** La nota informa a los usuarios que el enlace proporcionado no conduce a una página web tradicional. En cambio, su propósito es ser utilizado dentro de un software específico para acceder a geoservicios como WMS y WFS. Esto es importante para evitar confusiones entre los usuarios que podrían esperar que el enlace abra una página web en su navegador.

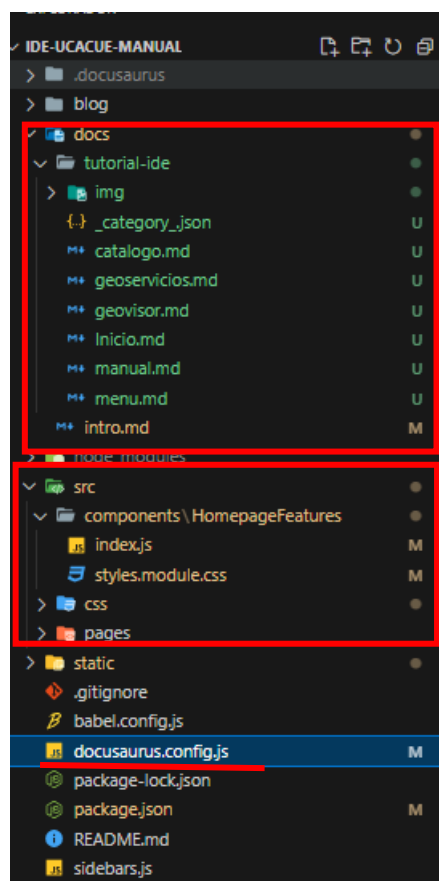

```
<span className="text-xxl items-center text-center">
  <strong>
    Nota: Este link no es una página web. Sirve únicamente para que se
    puedan consumir los diferentes geoservicios: WMS, WFS. Colocarlos en
    un software de presefencia para consumir los geoservicios
  </strong>
</span>
```

Manual

El manual se realizó utilizando Docusaurus que es una herramienta dedicada a la construcción, despliegue y mantenimiento de sitios web de documentación de código abierto. Está construida sobre React, lo que simplifica el proceso de escritura de la documentación al permitir a los usuarios escribir sus documentos en Markdown, que luego se convierten automáticamente en un sitio web estructurado.

Nuestro proyecto del manual tiene el nombre de “IDE-UCACUE-MANUAL”, para empezar, veremos la estructura de directorio que nos brinda:

- **docs:** donde pondremos nuestros archivos Markdown para la documentación.
- **src:** contiene los archivos fuente para el sitio, incluyendo páginas personalizadas.
- **docusaurus.config.js:** el archivo de configuración principal para el sitio, donde podemos definir el título, el tagline, el URL del proyecto, y más.



Como por ahora estamos desplegando de manera local, debemos irnos a la carpeta principal donde este nuestro “IDE-UCACUE-MANUAL”, abrimos el terminal y ejecutamos el comando *npm start* donde ya tenemos preconfigurado que corra en el puerto 3008.

```
PS D:\react\IDE-UCACUE-MANUAL> npm start

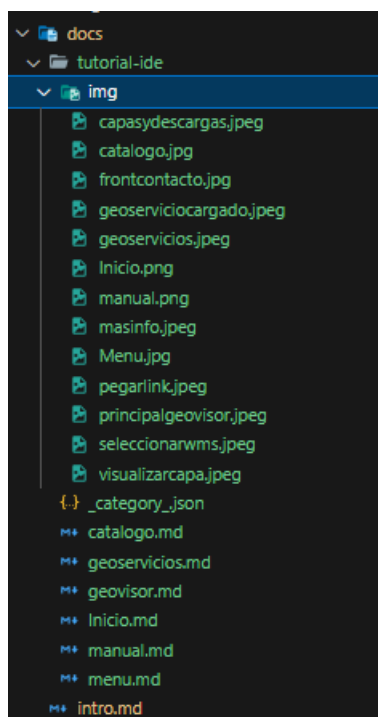
> my-website@0.0.0 start
> set PORT=3008 && docusaurus start

(node:5364) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
[INFO] Starting the development server...
[SUCCESS] Docusaurus website is running at: http://localhost:3008/

✓ Client
  Compiled successfully in 11.68s

client (webpack 5.90.3) compiled successfully
```

En esta carpeta de docs se encuentran todos los archivos de nuestra documentación como lo mencionamos antes:



Por cada componente que tenemos en la IDE, existe un archivo Markdown en el que explica el funcionamiento de este. Explicaremos en breve cada uno de ellos:

1) Inicio

Para editar la dirección IP en la que se publica el contenido, simplemente reemplace la dirección actual con la nueva, el inicio está en la primera posición del sidebar. Si desea modificar las imágenes, estas deben ser agregadas en la siguiente ruta: `../docs/tutorial-ide/img/nombreimagen.png`. Asegúrese de incluir el nombre correcto del archivo de la imagen que desea agregar.

```
docs > ** intro.md > # Introducción IDE-UCACUE > ## Empezamos > ### Lo que necesitarás
  Click here to ask Blackbox to help you code faster
1  ---
2  sidebar_position: 1
3  ---
4
5  # Introducción IDE-UCACUE
6
7  Es una pequeña introducción de como usar la IDE y el Geovisor.
8
9  ## Empezamos
10
11  ### Inicio y estructura
12
13  Para iniciar debemos ingresar a la dirección http://localhost:3000 (desplegado localmente).
14
15  O a la dirección en donde este desplegado el proyecto ejemplo \*\*\["ideucacue.edu.ec"\]\(http://www.ideucacue.edu.ec\)\*\*.
16
17  ![Logo UCACUE](../docs/tutorial-ide/img/Inicio.png)
18
19  Se estructura de la siguiente manera:
20
21  - **Navbar**: Contiene acceso directos a páginas de la IDE
22  - **Servicios**: Contiene los servicios que dispone la IDE
23  - **Footer**: Información disponible del laboratorio
24
25  ### Lo que necesitarás
26
27  - Conexión a internet y de ser necesario ArcGis o Qgis para que descargues y puedas visualizar los mapas.
28
29  **NOTA: En este caso esta desplegado de manera local**
```

2) Menú

Como el inicio las imágenes se ubican en la ruta: `../docs/tutorial-ide/img/nombreimagen.png`. Aquí se detalla submenús para una navegación directa hacia áreas específicas: el submenú "IDE" facilita el acceso a la página de inicio del IDE, "Servicios" conduce a una sección que ofrece geoservicios y documentación técnica, y "Contacto" abre una página dedicada a la comunicación con los usuarios. La sección de contacto, además de mostrar un formulario de consulta cuyo propósito es canalizar preguntas y comentarios hacia el correo electrónico del laboratorio a cargo, proporciona información de localización física del CIITT, permitiendo consultas en persona. Este enfoque estructural y de navegación busca optimizar la experiencia del usuario al proporcionar un acceso claro y directo a los recursos y servicios ofrecidos por el IDE.

```
docs > tutorial-ide > ** menu.md > # Menú > ## Página de contacto
  Click here to ask Blackbox to help you code faster
1  ---
2  sidebar_position: 2
3  ---
4
5  # Menú
6
7  ![Imagen del menú](../tutorial-ide/img/Menu.jpg)
8
9  Aquí vamos a tener los submenús que van a tener acceso a componentes directos:
10
11  - 'IDE' → 'Redirige a la página principal de la IDE'
12  - 'Servicios' → 'Geoservicios y manual'
13  - 'Contacto'
14
15  ## Página de contacto
16
17  ![Imagen del front de Contacto](../tutorial-ide/img/frontcontacto.jpg)
18
19  Cuando se tenga una duda o consulta referente a la IDE vamos a llenar el formulario, esta información es enviada al correo del laboratorio encargado.
20  Por otro lado tenemos la ubicación del CIITT.
21
```

3) Catalogo

Se encuentra configurado para ocupar la tercera posición en el panel lateral, se presenta el "Catálogo de proyectos", acompañado de una imagen representativa. Este catálogo incluye un buscador para localizar proyectos por palabras clave, una sección de filtros que permite refinar la búsqueda según la fuente o laboratorio, y un listado completo de proyectos. Cada entrada en el listado detalla el título, una breve descripción, la fecha de publicación, la categoría, la fuente, y los autores del proyecto. Al seleccionar un

proyecto, el usuario es redirigido a un geovisor que ofrece información detallada sobre el mismo, facilitando la navegación y el acceso a los datos de interés de manera eficiente y estructurada.

```
catálogo.md X
docs > tutorial-ide > ** catálogo.md > # Catálogo de proyectos
1
2 ---
3 sidebar_position: 3
4 ---
5 # Catálogo de proyectos
6
7 ![Imagen del catálogo](../tutorial-ide/img/catalogo.jpg)
8
9 Dentro del **catálogo de proyectos** tenemos:
10
11 - El **buscador**
12 - **Sección de filtros**
13 - **Listado de los proyectos**
14
15 # Buscador
16
17 Nos facilita la búsqueda por palabras o frases claves que tenga en el título o descripción del proyecto
18
19 # Filtros
20
21 Nos facilita filtrar los proyectos por la fuente o laboratorio, dándonos una respuesta rápida y fácil de comprender!
22
23 # Listado de proyectos
24
25 Si bien su nombre lo indica, aquí nos muestra todo el listado de los proyectos, dentro de cada uno se encuentra el:
26
27 - **Título**
28 - **Breve descripción del proyecto**
29 - **Fecha de publicación**
30 - **Categoría a la que pertenece**
31 - **La fuente**
32 - **Autor/es**
33
34 **Nota: Cuando decidamos el proyecto que vamos a ver, solo debemos dar click en uno de ellos y nos redirige a un geovisor con más información sobre ese proyecto.**
```

4) Geoservicios

Este componente está configurado para ser el quinto elemento en el panel lateral, detalla la sección "Geoservicios" con una guía técnica sobre cómo acceder y usar servicios geoespaciales como WMS, WFS y WCS a través de QGIS. Se inicia con una introducción a los geoservicios ofrecidos, seguida de un ejemplo práctico que incluye una imagen con un cuadro de autor, proyecto y URL para facilitar la referencia. Luego, se ofrece un tutorial paso a paso ilustrado para consumir un geoservicio WMS en QGIS, desde la apertura de la aplicación hasta la visualización de la capa geoespacial. Este enfoque instruccional proporciona a los usuarios un método claro y directo para integrar datos geoespaciales en sus proyectos de QGIS, demostrando el proceso a través de imágenes explicativas que acompañan cada paso.

```
geoservicios.md
docs > tutorial-ide > geoservicios.md > # Geoservicios > # Consumir WMS desde Qgis
1
2 ---
3 sidebar_position: 5
4 ---
5 # Geoservicios
6
7 En la sección de geoservicios, mostraremos diferentes proyectos, proporcionando un enlace específico para cada uno, permitiendo a los usuarios consumir diversos tipos de servicios como **WMS, WFS, y WCS**. En la parte inferior de la imagen, se presenta un cuadro con tres columnas tituladas **autor, proyecto y url**, actuando como un directorio para acceder a proyectos específicos. Por ejemplo, el **autor** José Guzmán lidera el **proyecto** "Implementación de un geovisor para la visualización de las propiedades geotécnicas y geomorfológicas del subsuelo: caso de estudio Cuenca, Azuay, Ecuador", del cual se puede obtener el **link** directo al geoservicio.
8
9
10 ![Geoservicios](../tutorial-ide/img/geoservicios.jpg)
11
12 # Consumir WMS desde Qgis
13
14 Para consumir el geoservicio disponible en una URL específica, se debe utilizar la herramienta QGIS. El proceso es el siguiente:
15
16 1. Para consumir el geoservicio disponible en una URL específica, se debe utilizar la herramienta QGIS. El proceso es el siguiente:
17
18 2. Abra QGIS y diríjase a la opción Capas en el menú principal.
19
20 Seleccione Añadir capa y luego elija Añadir capa WMS/WFS....
21
22 ![seleccionarwms](../tutorial-ide/img/seleccionarwms.jpg)
23
24 Al seleccionar la opción para añadir una capa, se abrirá una ventana que permite cargar la URL del geoservicio seleccionado. Aquí, se debe introducir un nombre para la conexión y copiar la URL del geoservicio.
25
26 ![añadircapa](../tutorial-ide/img/añadircapa.jpg)
27
28 Una vez añadida la capa, ésta se cargará automáticamente en QGIS, permitiéndole visualizar y trabajar con la capa deseada. En la interfaz de QGIS, podrá ver las capas cargadas y seleccionar cualquiera de ellas para visualizar y analizar los datos geoespaciales que contiene.
29
30 ![vercapa](../tutorial-ide/img/geoserviciocargado.jpg)
31
32 ![añadircapa](../tutorial-ide/img/visualizarcapa.jpg)
33
34 Para cargar otro tipo de geoservicio, como un WFS, simplemente repita los mismos pasos, seleccionando esta vez la opción adecuada para Añadir capa WFS. Esto le permitirá visualizar y trabajar con datos geoespaciales vectoriales proporcionados a través del geoservicio WFS.
35
36 Este proceso facilita el acceso y la visualización de múltiples capas de información geoespacial directamente desde QGIS, aprovechando los geoservicios disponibles en distintas URLs para explorar y analizar datos geográficos específicos.
```

5) Geovisor

En la cuarta posición del panel lateral, el documento de Markdown presenta la sección "Geovisor", describiendo sus principales características y funcionalidades. Inicia con una visión general y una imagen del geovisor, seguido de detalles sobre la interfaz, que incluye una barra lateral con información del proyecto y controles de mapa para la navegación. La sección "Capas y descarga" ilustra cómo activar capas específicas y acceder a opciones de descarga de datos en formatos como GeoTIFF, JPEG, GeoJSON

y Shapefile. Este resumen ofrece una guía concisa para el usuario sobre cómo navegar y utilizar el geovisor, resaltando sus capacidades de visualización y descarga de datos geoespaciales.

```
geovisor.md X
docs > tutorial-ide > M geovisor.md > # Geovisor > ## Información
1 ---
2 sidebar_position: 4
3 ---
4
5 # Geovisor
6 Esta es una vista principal del geovisor
7
8 ![Imagen del geovisor](../tutorial-ide/img/principalgeovisor.jpeg)
9
10 A continuación, una vez que se ingresa a un proyecto se obtendrá el **Geovisor**, a la izquierda se encontrará:
11
12 ## Información
13 Es una barra lateral que ofrece información detallada del proyecto, incluyendo el título y el año de publicación. Debajo
14 de toda esta información está un botón que dice **ver más**... lo que redirige a otra página que muestra toda la información del proyecto, como se muestra a continuación:
15
16 ![Imagen de información](../tutorial-ide/img/principalgeovisor.jpeg)
17
18 Además, en la derecha, hay controles de mapa para la navegación, incluyendo herramientas para acercar o alejar, y otros
19 controles para ajustar la visualización de capas o acceder a herramientas analíticas adicionales.
20
21
22 ## Capas y descarga
23
24 ![Imagen de capas y descarga](../tutorial-ide/img/capasydescargas.jpeg)
25
26 El siguiente apartado de capas nos permite activar o desactivar cada una de ellas dependiendo de cual se escoja
27 se cargará la información de esa capa. Además, se habilitará un botón con el nombre de la capa y al darle click se habilitará los enlaces para descargar datos relacionados en varios
    formatos como GeoTIFF, JPEG, GeoJSON y Shapefile.
```

6) Manual

En la sexta y última posición del panel lateral, el documento de Markdown introduce la sección "Manual", complementada con una imagen que ilustra el acceso al mismo.

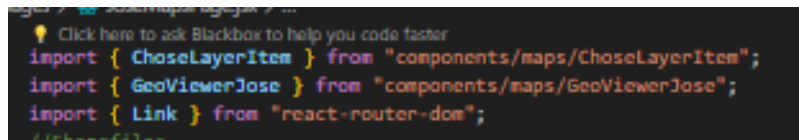
```
manual.md X
docs > tutorial-ide > M manual.md > ...
1 ---
2 sidebar_position: 6
3 ---
4
5 # Manual
6
7 ![Imagen del manual](../tutorial-ide/img/manual.png)
8
9 Aquí en el boton ver manual de usuario redirige a esta página que esta hecha
10 para explicar los conceptos básicos y mostrar cómo utilizar la herramienta.
11
```

Geovisor

Empezamos a describir la estructura y funcionalidad del componente ‘JoseMapsPage’ que también lo conoceremos como geovisor, las funcionalidades principales son visualización de un panel lateral con opciones para activar/desactivar capas de datos geográficos, descarga de datos geoespaciales en diferentes formatos (GeoTIFF, JPEG, GeoJson, Shapefile) y la interacción con un mapa geográfico para visualizar las capas de datos seleccionados.

Componentes importados:

- **ChoseLayerItem:** Componente para cada item de la lista de capas de datos. Permite al usuario activar/desactivar la visualización de la capa en el mapa.
- **GeoViewerJose:** Componente encargado de mostrar el mapa y las capas de datos activas.
- **Link:** Componente de react-router-dom utilizado para la navegación entre componentes sin recargar la página.



```

Click here to ask Blackbox to help you code faster
import { ChoseLayerItem } from "components/maps/ChoseLayerItem";
import { GeoViewerJose } from "components/maps/GeoViewerJose";
import { Link } from "react-router-dom";

```

Hooks

- **useState:** Se utiliza para manejar el estado de la visibilidad de las capas de datos y del submenú.
- **useExpCalicatas, useGeoAzuay, useGeoLocal, useInfoGeotecnica, useIsoPeriodos, useParrAzuay, usePerfilEstra:** Hooks personalizados que probablemente se encargan de cargar y administrar los datos para cada tipo de capa geográfica, incluyendo el manejo de estados de carga.



```

import {
  useExpCalicatas,
  useGeoAzuay,
  useGeoLocal,
  useInfoGeotecnica,
  useIsoPeriodos,
  useParrAzuay,
  usePerfilEstra,
} from "hooks/useMap";
import { useState } from "react";
import projects from "utils/projects";

export const JoseMapsPage = () => {
  const [key0, setKey0] = useState(false);
  const [key1, setKey1] = useState(false);
  const [key2, setKey2] = useState(false);
  const [key3, setKey3] = useState(false);
  const [key4, setKey4] = useState(false);
  const [key5, setKey5] = useState(false);
  const [key6, setKey6] = useState(false);

  // Estado para controlar la visibilidad del submenú
  const [isSubmenuVisible, setIsSubmenuVisible] = useState(false);

  // Función para alternar la visibilidad del submenú
  const toggleSubmenu = () => {
    setIsSubmenuVisible(!isSubmenuVisible);
  };

  const { data: infoGeotecnica, isLoading: isLoadingKey0 } = useInfoGeotecnica({
    enabled: key0,
  });

  const { data: expCalicatas, isLoading: isLoadingKey1 } = useExpCalicatas({
    enabled: key1,
  });

  const { data: perfilEstra, isLoading: isLoadingKey2 } = usePerfilEstra({
    enabled: key2,
  });

  const { data: isoPeriodos, isLoading: isLoadingKey3 } = useIsoPeriodos({
    enabled: key3,
  });

  const { data: geoLocal, isLoading: isLoadingKey4 } = useGeoLocal({
    enabled: key4,
  });

  const { data: geoAzuay, isLoading: isLoadingKey5 } = useGeoAzuay({
    enabled: key5,
  });

  const { data: parrAzuay, isLoading: isLoadingKey6 } = useParrAzuay({
    enabled: key6,
  });
}

```

Importaciones

- Diversos archivos ZIP y TIFF: Representan los datos geoespaciales que se pueden visualizar en el mapa y descargar por el usuario.

```

pages > JoseMapsPage.jsx > ...
  ⚡ Click here to ask Blackbox to help you code faster
import { ChoseLayerItem } from "components/maps/ChoseLayerItem";
import { GeoViewerJose } from "components/maps/GeoViewerJose";
import { Link } from "react-router-dom";
//Shapefiles
import shgeotecnica from "../utils/Shapefile/info_geotecnica.zip";
import shgeoazuay from "../utils/Shapefile/geo_azuay.zip";
import shgeolocal from "../utils/Shapefile/geo_local.zip";
import shcalicatas from "../utils/Shapefile/exploracion_calicatas.zip";
import shiso from "../utils/Shapefile/iso_periodos.zip";
import shazuay from "../utils/Shapefile/parroquias_azuay.zip";
import shperfil from "../utils/Shapefile/perfil_estratigrafico.zip";
//Tifs
import tiffpgs from "../utils/jose-info_geotecnica.tif";
import tiffgeoazuay from "../utils/jose-geo_azuay.tif";
import tiffgeolocal from "../utils/jose-geo_local.tif";
import tiffcalicatas from "../utils/jose-exploracion_calicatas.tif";
import tiffiso from "../utils/jose-iso_periodos.tif";
import tiffazuay from "../utils/jose-parroquias_azuay.tif";
import tiffperfil from "../utils/jose-perfil_estratigrafico.tif";

```

Funcionalidad de la UI

- El componente organiza su interfaz en dos partes principales: un panel lateral y un visor de mapas. El panel lateral contiene una lista desplegable de proyectos y capas de datos, cada una con opciones para activar/desactivar la visualización en el mapa y descargar los datos. El visor de mapas muestra las capas activas seleccionadas por el usuario.

```

pages > JoseMapsPage.jsx > ...
export const JoseMapsPage = () => {
  <div className="flex space-x-0 p-3 md:space-x-3 lg:space-x-3">
    <div className="hidden h-[calc(100vh_-_80px)] w-[400px] overflow-hidden overflow-y:auto rounded-lg bg-inherit p-3 text-white shadow-md md:lg:block lg:block">
      <div
        id="accordion-flush"
        data-accordion="collapse"
        data-active-classes="bg-white dark:bg-gray-900 text-gray-900 dark:text-white"
        data-inactive-classes="text-gray-500 dark:text-gray-400"
      >
        <h2 id="accordion-flush-heading-1">
          <button
            type="button"
            className="flex w-full items-center justify-between border-b border-gray-200 py-5 text-left font-medium text-gray-500 dark:border-gray-700 dark:text-gray-400"
            data-accordion-target="#accordion-flush-body-1"
            aria-expanded="true"
            aria-controls="accordion-flush-body-1"
          >
            <span>Información</span>
            <svg
              data-accordion-icon
              className="h-3 w-3 shrink-0 rotate-180"
              aria-hidden="true"
              xmlns="http://www.w3.org/2000/svg"
              fill="none"
              viewBox="0 0 10 6"
            >
              <path stroke="currentColor" d="M5 5 1 1 5 5"/>
            </svg>
          </button>
        </h2>
        {projects.map((project, index) => {
          <div
            key={index}
            id="accordion-flush-body-1"
            className="hidden"
            aria-labelledby="accordion-flush-heading-1"
          >
            <div className="border-b border-gray-200 py-5 dark:border-gray-700">
              <p className="text-lg font-semibold text-gray-500">Título</p>
              <p className="text-base font-normal text-gray-500">{project.title}</p>
              <p>
                <div className="text-lg font-semibold text-gray-500">Publicación</div>
                <p>
                  <div className="text-base font-normal text-gray-500">{project.publication}</div>
                </p>
              </p>
              <div className="text-5 inline-flex items-center">
                <Link to="/moreinfo">
                  <button
                    type="button"
                    className="inline-flex items-center rounded-lg bg-red px-5 py-2.5 text-center text-sm font-medium text-white hover:bg-red focus:outline-none focus:ring-4 focus:ring-red dark:bg-red dark:hover:bg-red dark:focus:ring-red"
                  >
                    Ver más
                  </button>
                </Link>
              </div>
            </div>
          </div>
        })}</div>

```

Manejo de Estados y Datos

- El componente utiliza estados para controlar la activación de las capas y la visibilidad del submenú. Los hooks personalizados probablemente hacen peticiones a una API o a un servidor de mapas para obtener los datos geoespaciales correspondientes a cada capa activada.

```

177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Contactos

Estructura del Componente

El componente se divide en dos secciones principales:

- Formulario de Contacto:** Permite al usuario introducir sus datos personales, consulta, y detalles de afiliación.


```

</div>
<div className="mb-6">
  <label>
    htmlFor="institucion"
    className="block text-sm font-medium text-white"
  </label>
  <select>
    name="institucion"
    id="institucion"
    value={institucion}
    onChange={(e) => setInstitucion(e.target.value)}
    className="w-full rounded-lg border border-gray-300 bg-white px-4 py-2 shadow-sm transition duration-150 ease-in-out focus:border-indigo-500 focus:outline-none focus:ring-2 focus:ring-indigo-500 focus:ring-opacity-50"
    required
  >
    <option value="">Seleccione una opción</option>
    <option value="Universidad de Cuenca">
      Universidad de Cuenca
    </option>
    <option value="Universidad Politécnica Salesiana">
      Universidad Politécnica Salesiana
    </option>
    <option value="Universidad del Azuay">
      Universidad del Azuay
    </option>
    <option value="Gad Municipal de Cuenca">
      Gad Municipal de Cuenca
    </option>
    <option value="Otra">Otra...</option>
  </select>
  <div institution === "Otra" && {
    <input
      type="text"
      name="nuevaInstitucion"
      id="nuevaInstitucion"
      value={nuevaInstitucion}
      onChange={(e) => setNuevaInstitucion(e.target.value)}
      className="w-full rounded-lg border border-gray-300 bg-white px-4 py-2 shadow-sm focus:border-indigo-500 focus:ring focus:ring-indigo-500 focus:ring-opacity-50"
      placeholder="Escriba el nombre de la institución"
    >
  </div>
}
</div>
<button
  type="submit"
  className="hover:bg-red-700 focus:ring-red-500 inline-flex w-full items-center justify-center rounded-lg bg-red px-5 py-3 text-sm font-medium text-white transition focus:outline-none focus:ring-4 focus:ring-opacity-50"
>
  Enviar correo
</button>
</form>

```

- **Ubicación:** Presenta un mapa de Google Maps embebido que muestra la ubicación relevante para el usuario.

```

<div className="w-full px-4 lg:w-1/2">
  <div className="rounded-lg bg-inherit p-6 shadow-lg">
    <h2 className="mb-6 text-2xl font-semibold text-white">Ubicanos</h2>
    <div className="overflow-hidden rounded-lg shadow-lg">
      <iframe
        src="https://www.google.com/maps/embed?pb=!1m14!1m8!1m3!1d7969.724696605074!2d-78.9656163!3d-2.8560806!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x91cd1733f70eca17k3A8xb544f95d8070fc6!2sCI111K20-K20UniversidadK20CatK3K3allicaK28deK20Cuenca!5e0!3m2!1ses!2sec!4v1693337053326!5m2!1ses!2sec"
        width="100%"
        height="400"
        loading="lazy"
        title="mapa"
        className="shadow-sm" // Puedes aplicar sombras directamente al iframe si lo deseas
      ></iframe>
    </div>
  </div>
</div>

```

Estados

El componente maneja varios estados locales con `useState` para almacenar la información introducida por el usuario:

- **nombre:** Almacena el nombre del usuario.
- **apellido:** Almacena el apellido del usuario.
- **consultadescarga:** Contiene la consulta o mensaje que el usuario desea enviar.
- **email:** Guarda el correo electrónico del usuario.
- **telefono:** Almacena el número de teléfono del usuario.
- **institucion:** Guarda la afiliación institucional del usuario.
- **sectorpertenece:** Este estado no se utiliza en el código proporcionado y podría ser removido o implementado en futuras extensiones del componente.
- **nuevaInstitucion:** En caso de que el usuario seleccione "Otra" institución, este estado almacena el nombre de esa nueva institución.

```
export const ContactPage = () => {
  const [nombre, setNombre] = useState("");
  const [apellido, setApellido] = useState("");
  const [consultadescarga, setConsultadescarga] = useState("");
  const [email, setEmail] = useState("");
  const [telefono, setTelefono] = useState("");
  const [institucion, setInstitucion] = useState("");
  const [sectorpertenece, setSectorpertenece] = useState("");
  const [nuevaInstitucion, setNuevaInstitucion] = useState("");

```

Funcionalidades Principales

- Envío de Datos: Al enviar el formulario, se recopilan los datos del estado y se envían a un servidor o API especificado, en este caso, a `http://localhost:3005/send-email`, usando `fetch` con método `POST`.
- Validación y Limpieza de Formulario: El formulario realiza validaciones básicas usando el atributo `required` de HTML5 en los campos y limpia los campos después de un envío exitoso.

```
const handleSubmit = async (e) => {
  e.preventDefault();
  const formData = {
    nombre,
    apellido,
    consultadescarga,
    email,
    telefono,
    institucion,
    nuevaInstitucion,
  };

  try {
    // Enviar los datos al servidor
    const response = await fetch("http://localhost:3005/send-email", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify(formData),
    });

    if (response.ok) {
      // El correo se envió con éxito
      alert("Correo electrónico enviado con éxito");

      // Limpiar el formulario restableciendo los estados a una cadena vacía
      setNombre("");
      setApellido("");
      setConsultadescarga("");
      setEmail("");
      setTelefono("");
      setInstitucion("");
      setSectorpertenece("");
    } else {
      // Hubo un error al enviar el correo
      alert(
        "Error al enviar el correo electrónico. Por favor, inténtalo de nuevo."
      );
    }
  } catch (error) {
    console.error("Error al enviar el correo electrónico", error);
    alert("Error de red al intentar enviar el correo electrónico.");
  }
};

```

Estilos y Layout

- El componente utiliza estilos definidos en `../assets/css/home.css`, lo que implica que su apariencia está dictada por las clases CSS definidas en ese archivo.
- Se emplean clases de Tailwind CSS para el layout y diseño del formulario, como `rounded-lg`, `shadow-lg`, `text-white`, entre otras, para darle una apariencia moderna y responsive.

```
<div className="background-contact justify-center p-4">
  <div className="container mx-auto flex flex-wrap items-center justify-between">
    <div className="mb-4 w-full px-4 lg:w-1/2">
      <div className="rounded-lg bg-inherit p-6 shadow-lg">

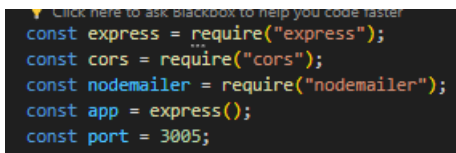
```

- **Server.js**

Este servidor se encarga de recibir solicitudes HTTP POST con información de contacto y consultas de usuarios para luego enviar esta información a través de un correo electrónico.

Configuración del Entorno

- **Express:** Marco de trabajo para aplicaciones web en Node.js que facilita la creación de servidores HTTP.
- **CORS (Cross-Origin Resource Sharing):** Módulo para habilitar CORS, permitiendo que el servidor acepte solicitudes de dominios cruzados.
- **Nodemailer:** Módulo para el envío de correos electrónicos desde Node.js.

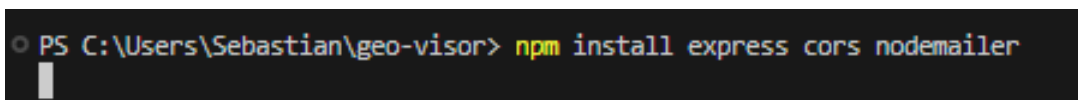


```
▼ Click here to ask Blackbox to help you code faster
const express = require("express");
const cors = require("cors");
const nodemailer = require("nodemailer");
const app = express();
const port = 3005;
```

Instalación de Dependencias

Para instalar las dependencias necesarias (express, cors, nodemailer), se debe ejecutar el siguiente comando en el terminal:

- `npm install express cors nodemailer`



```
PS C:\Users\Sebastian\geo-visor> npm install express cors nodemailer
```

Configuración del Servidor

- **Puerto:** El servidor se configura para escuchar en el puerto 3005.
- **Middleware:**
 - **express.json():** Permite al servidor manejar solicitudes que tengan un cuerpo en formato JSON.
 - **express.urlencoded({ extended: true }):** Habilita el procesamiento de cuerpos de solicitudes que usen codificación URL.
 - **cors():** Aplica CORS a todas las rutas del servidor, permitiendo solicitudes de cualquier origen.

```
const app = express();
const port = 3005;

// Configuración para Express >= 4.16
app.use(express.json()); // Para parsing de application/json
app.use(express.urlencoded({ extended: true })); // Para parsing de application/x-www-form-urlencoded
app.use(cors()); // Habilita CORS en todas las peticiones a la API
```

Configuración de Nodemailer

Se configura un transporte de Nodemailer utilizando el servicio de correo electrónico de Gmail. Credenciales del gmail:

Correo: pruebaside23@gmail.com

Contraseña: ciitt-server23

Es necesario proporcionar un usuario y una contraseña (en este caso, se usa una contraseña de aplicación generada para mayor seguridad).

```
app.use(cors()); // Habilita CORS en todas las peticiones
// Configuración de Nodemailer
const transporter = nodemailer.createTransport({
  service: "gmail",
  auth: {
    user: "pruebaside23@gmail.com",
    pass: "ccsjqdkrfotonwe",
  },
});
```

Ruta del Servidor

- **POST /send-email:** Esta ruta maneja las solicitudes POST para enviar correos electrónicos. Recibe datos de un formulario que incluye nombre, apellido, consulta, correo electrónico, teléfono, institución y, si corresponde, el nombre de una nueva institución.
- **Validación:** Se verifica que todos los campos requeridos estén presentes. Si falta alguno, se retorna un error HTTP 400.
- **Envío de Correo:** Si todos los campos están correctos, se procede a enviar un correo electrónico usando Nodemailer con los detalles proporcionados. En caso de error en el envío, se retorna un error HTTP 500.
- **Respuesta:** Se envía una respuesta indicando si el correo se envió correctamente o si ocurrió un error.

```

app.post("/send-email", (req, res) => {
  const {
    nombre,
    apellido,
    consultadescarga,
    email,
    telefono,
    institucion,
    nuevaInstitucion,
  } = req.body;

  // Verificar que todos los campos están llenos y que se ha seleccionado una institución
  if (
    !nombre ||
    !apellido ||
    !email ||
    !telefono ||
    !consultadescarga ||
    !institucion ||
    (institucion === "otra" && !nuevaInstitucion)
  ) {
    return res
      .status(400)
      .send(
        "Todos los campos son obligatorios y se debe seleccionar una institución."
      );
  }

  let institucionNombre =
    institucion === "otra" ? nuevaInstitucion : institucion;

  const mailOptions = {
    from: "pruebaside23@gmail.com",
    to: "pruebaside23@gmail.com", // 0 cualquier otro destinatario
    subject: "Confirmación de consulta",
    html: `
    <div style="background-color: #f0f0f0; padding: 20px;">
      <div style="max-width: 600px; margin: auto; background: white; padding: 20px; font-family: Arial, sans-serif; color: #333; border-radius: 8px;">
        <h2 style="color: #4F8A10;">Hola ${nombre} ${apellido}</h2>
        <p>Hemos recibido tu consulta sobre "<strong>${consultadescarga}</strong>". Te contactaremos en breve al número <strong>${telefono}</strong> o al correo <strong>${email}</strong>.</p>
        <p><strong>Institución:</strong> ${institucionNombre}</p>
        <hr>
        <p>Gracias por contactarnos.</p>
      </div>
    </div>
  `;
  // Aquí puedes usar también HTML para estructurar mejor el mensaje
  };

  transporter.sendMail(mailOptions, (error, info) => {
    if (error) {
      console.log(error);
      res.status(500).send("Error al enviar el email");
    } else {
      console.log("Email enviado: " + info.response);
      res.status(200).send("Email enviado correctamente");
    }
  });
});

app.listen(port, () => {
  console.log(`Servidor escuchando en http://localhost:${port}`);
});

```

Ejecución del Servidor

Para iniciar el servidor, se ejecuta el siguiente comando:

- `node server.js`

```

PS C:\Users\Sebastian\geo-visor> cd .\src\
PS C:\Users\Sebastian\geo-visor\src> node server.js
Servidor escuchando en http://localhost:3005

```