

Curso: Desarrollo Web - 12025-9989-031-A

LABORATORIO #5



Walter Rene Rosales Hurtarte
Carné: 9989-08-7438

¿Por qué usamos Vite + React en el desarrollo?

1. React: Biblioteca para construir interfaces de usuario

- React es una de las tecnologías más populares para crear aplicaciones web modernas.
- Permite trabajar con **componentes reutilizables**, lo que facilita mantener y escalar proyectos grandes.
- Su enfoque declarativo hace que sea más sencillo describir “qué queremos ver en pantalla” y dejar que React gestione las actualizaciones de la interfaz.
- Se integra fácilmente con librerías externas como **Chakra UI** para estilos, **React Router** para rutas, y herramientas de estado global (Redux, Context API).

2. Vite: Entorno de desarrollo moderno

- Vite es un **bundler y servidor de desarrollo rápido** que reemplaza a herramientas más pesadas como Webpack.
- Beneficios principales:
 - **Inicio rápido:** al correr npm run dev levanta el servidor casi instantáneamente.
 - **Hot Module Replacement (HMR):** los cambios en el código se reflejan en el navegador sin recargar toda la aplicación.
 - **Optimización de build:** usa **esbuild** y **Rollup**, que generan aplicaciones rápidas y optimizadas para producción.

3. Razones pedagógicas (nivel universitario)

- **Facilidad de aprendizaje:** React tiene una curva de aprendizaje amigable y mucha documentación.
- **Adopción en la industria:** empresas reales lo usan masivamente, lo cual prepara a los estudiantes para el campo laboral.
- **Proyectos modulares:** React fomenta la separación del código en pequeños componentes fáciles de probar.
- **Eficiencia de desarrollo:** Vite permite que los estudiantes se concentren en aprender React, sin perder tiempo configurando entornos complejos.

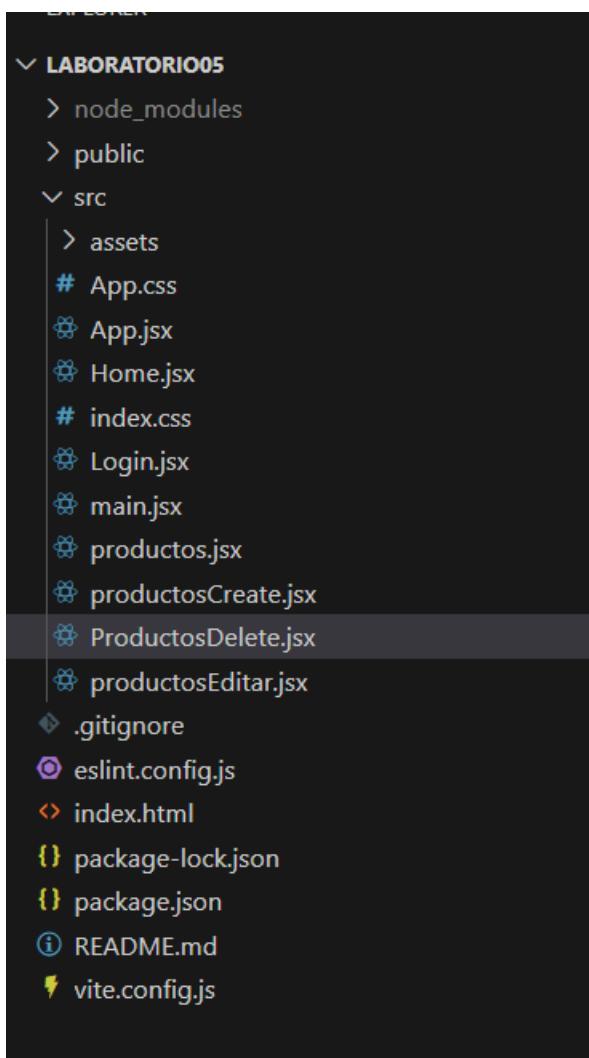
4. Ejemplo práctico en el laboratorio

- Usamos React para construir **pantallas dinámicas** (Login, Home, Productos).
- Vite hace que el entorno se configure con un simple npm create vite@latest.

Repositorio Git

[wallas2022/laboratorio06: Proyecto en React acceso a token y login](#)

Estructura general del proyecto:



1. Capturas de pantallas de consumo de apis

Captura de prueba de tocken

```

POST https://dummyjson.com/auth/login
200 OK
494 ms
930 B
Just Now

Params Body Auth Headers 4 Scripts Docs
Preview Headers 26 Cookies 2 Tests 0 / 0 → Mock Console

JSON
1
2 +
{
  "username": "emillys",
  "password": "emillyspass"
}
3
4
5
6
7

1 +
{
  "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJpZC16MswidXNlcm5hbWUiOjJlbWlseXMlCJlbWFpbG16ImVtaW5LmpvaG5zb25AeC5kdWtceMpbz24uY29tIiw1Zmlyc3ROYW11fjojRwphK1lC3syXN0TmfzSI6Ikpvag5zb241LCJnZWSKZXiiOjMzw1hbgU1lCpbWFnZSI6Imh0dHbzO18ZHvtbxLqc29uLmNb59pY29uL2vtaw5cy8xHjg1CJpyXQiojE3NTU2NjUxMjcsImV4cCI6MTc1NTY200cyN38.AJYbVLUpq9JssXXGhpNfHeGT-7g6whrxDRVcomJdk",
  "refreshToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJpZC16MswidXNlcm5hbWUiOjJlbWlseXMlCJlbWFpbG16ImVtaW5LmpvaG5zb25AeC5kdWtceMpbz24uY29tIiw1Zmlyc3ROYW11fjojRwphK1lC3syXN0TmfzSI6Ikpvad5zb241LCJnZWSKZXiiOjMzw1hbgU1lCpbWFnZSI6Imh0dHbzO18ZHvtbxLqc29uLmNb59pY29uL2vtaw5cy8xHjg1CJpyXQiojE3NTU2NjUxMjcsImV4cCI6MTc1ODI1NzEyN38.6VH2aQACaa82PDDgkdgHu2rG8LtoWvoacPdKZ_VOHU",
  "id": 1,
  "username": "emillys",
  "email": "emily.johnson@x.dummyjson.com",
  "firstName": "Emily",
  "lastName": "Johnson",
  "gender": "female",
  "image": "https://dummyjson.com/icon/emillys/128"
}

```

Captrua de api me

```

My first collection | Run
POST My first request | GET New Request
GET https://dummyjson.com/auth/me
200 OK
515 ms
1403 B
Preview Headers 23 Cookies Tests 0 / 0 → Mock Console

Params Body Auth Headers 3 Scripts Docs
Preview

Bearer Token
ENABLED
TOKEN: ...
PREFIX

1 +
{
  "id": 1,
  "firstName": "Emily",
  "lastName": "Johnson",
  "maidenName": "Smith",
  "age": 28,
  "gender": "female",
  "email": "emily.johnson@x.dummyjson.com",
  "phone": "+81 965-431-3024",
  "username": "emillys",
  "password": "emillyspass",
  "birthDate": "1996-5-30",
  "image": "https://dummyjson.com/icon/emillys/128",
  "bloodGroup": "O-",
  "height": 193.24,
  "weight": 63.16,
  "eyeColor": "Green",
  "hair": {
    "color": "Brown",
    "type": "curly"
  },
  "ip": "42.48.100.32",
  "address": {
    "address": "626 Main Street",
    "city": "Phoenix",
    "state": "Mississippi",
    "statusCode": "MS",
    "postalCode": "29112",
    "coordinates": {
      "lat": -77.16213
    }
  }
}

```

Lista de productos

The screenshot shows the Postman interface with a collection named "My first collection". A GET request to <https://dummyjson.com/products> is selected. The response status is 200 OK, with a duration of 123 ms and a size of 43.1 KB. The preview pane shows a JSON response object representing a product.

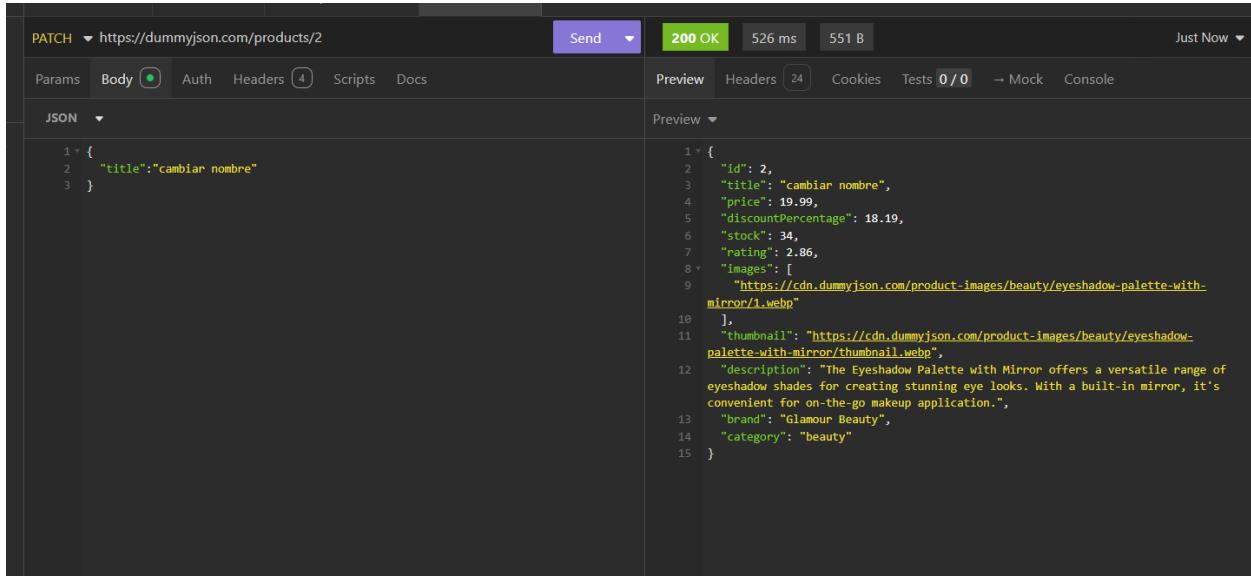
```
1: {  
2:   "products": [  
3:     {  
4:       "id": 1,  
5:       "title": "Essence Mascara Lash Princess",  
6:       "description": "The Essence Mascara Lash Princess is a popular mascara known for its volumizing and lengthening effects. Achieve dramatic lashes with this long-lasting and cruelty-free formula.",  
7:       "category": "beauty",  
8:       "price": 9.99,  
9:       "discountPercentage": 10.48,  
10:      "rating": 2.56,  
11:      "stock": 99,  
12:      "tags": [  
13:        "beauty",  
14:        "mascara"  
15:      ],  
16:      "brand": "Essence",  
17:      "sku": "BEA-ESS-ESS-001",  
18:      "weight": 4,  
19:      "dimensions": (  
20:        "width": 15.14,  
21:        "height": 13.08,  
22:        "depth": 22.99  
23:      ),  
24:      "warrantyInformation": "1 week warranty",  
25:      "shippingInformation": "Ships in 3-5 business days",  
26:      "availabilityStatus": "In Stock",  
27:      "reviews": [  
28:        {  
29:          "rating": 3,  
30:          "comment": "Would not recommend!"  
31:        }  
32:      ]  
33:    }  
34:  ]  
35: }  
36: }
```

Crear

The screenshot shows the Postman interface with a collection named "My first collection". A POST request to <https://dummyjson.com/products/add> is selected. The response status is 201 Created, with a duration of 501 ms and a size of 27 B. The preview pane shows a JSON response object representing a newly created product.

```
1: {  
2:   "id": 195,  
3:   "title": "prueba"  
4: }
```

Actualizar



PATCH <https://dummyjson.com/products/2> Send 200 OK 526 ms 551 B Just Now

Params	Body	Auth	Headers	Scripts	Docs
			24		

Preview Headers Cookies Tests 0 / 0 → Mock Console

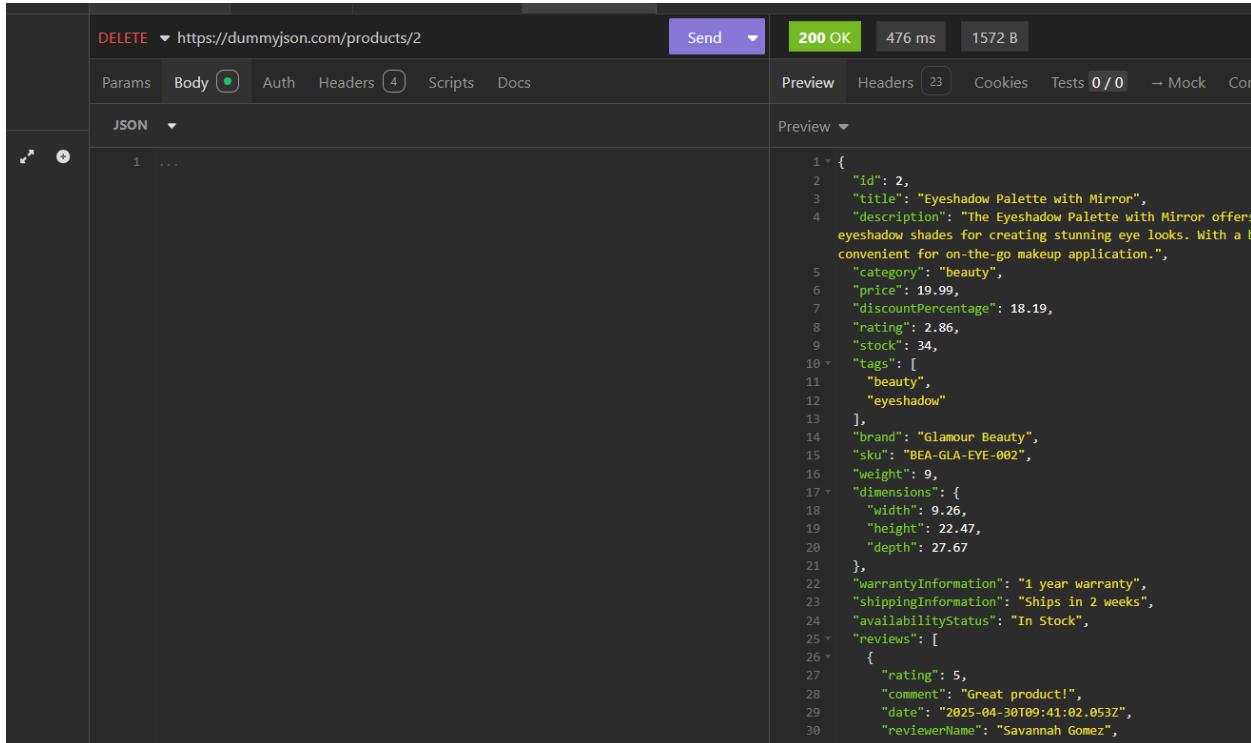
JSON

```
1 + {  
2   "title": "cambiar nombre"  
3 }
```

Preview

```
1 + {  
2   "id": 2,  
3   "title": "cambiar nombre",  
4   "price": 19.99,  
5   "discountPercentage": 18.19,  
6   "stock": 34,  
7   "rating": 2.86,  
8   "images": [  
9     "https://cdn.dummyjson.com/product-images/beauty/eyeshadow-palette-with-mirror/1.webp"  
10    ],  
11    "thumbnail": "https://cdn.dummyjson.com/product-images/beauty/eyeshadow-palette-with-mirror/thumbnail.webp",  
12    "description": "The Eyeshadow Palette with Mirror offers a versatile range of eyeshadow shades for creating stunning eye looks. With a built-in mirror, it's convenient for on-the-go makeup application.",  
13    "brand": "Glamour Beauty",  
14    "category": "beauty"  
15 }
```

Eliminar



DELETE <https://dummyjson.com/products/2> Send 200 OK 476 ms 1572 B

Params	Body	Auth	Headers	Scripts	Docs
			23		

Preview Headers Cookies Tests 0 / 0 → Mock Console

JSON

```
1 + ...
```

Preview

```
1 + {  
2   "id": 2,  
3   "title": "Eyeshadow Palette with Mirror",  
4   "description": "The Eyeshadow Palette with Mirror offers eyeshadow shades for creating stunning eye looks. With a built-in mirror, it's convenient for on-the-go makeup application.",  
5   "category": "beauty",  
6   "price": 19.99,  
7   "discountPercentage": 18.19,  
8   "rating": 2.86,  
9   "stock": 34,  
10  "tags": [  
11    "beauty",  
12    "eyeshadow"  
13  ],  
14  "brand": "Glamour Beauty",  
15  "sku": "BEA-GLA-EYE-002",  
16  "weight": 9,  
17  "dimensions": {  
18    "width": 9.26,  
19    "height": 22.47,  
20    "depth": 27.67  
21  },  
22  "warrantyInformation": "1 year warranty",  
23  "shippingInformation": "Ships in 2 weeks",  
24  "availabilityStatus": "In Stock",  
25  "reviews": [  
26    {  
27      "rating": 5,  
28      "comment": "Great product!",  
29      "date": "2025-04-30T09:41:02.053Z",  
30      "reviewerName": "Savannah Gomez",  
31    }  
32  ]  
33}
```

Buscar con paginado y filtros

```

1: [
2:   {
3:     "products": [
4:       {
5:         "id": 11,
6:         "title": "Annibale Colombo Bed",
7:         "price": 1899.99
8:       },
9:       {
10:        "id": 12,
11:        "title": "Annibale Colombo Sofa",
12:        "price": 2499.99
13:      },
14:      {
15:        "id": 13,
16:        "title": "Bedside Table African Cherry",
17:        "price": 299.99
18:      },
19:      {
20:        "id": 14,
21:        "title": "Knoll Saarinen Executive Conference Chair",
22:        "price": 499.99
23:      },
24:      {
25:        "id": 15,
26:        "title": "Wooden Bathroom Sink With Mirror",
27:        "price": 799.99
28:      },
29:      {
30:        "id": 16,
31:        "title": "Apple",
32:        "price": 1.99
33:     }
34:   ]
35: }

```

2.- Explicacion del código y capturas

```

// src/Login.jsx
import React, { useState } from 'react';
import {
  Box, Button, Flex, FormControl, FormLabel, Input,
  Heading, Text, Image, VStack
} from '@chakra-ui/react';
import { useNavigate } from 'react-router-dom';

const Login = () => {
  const [username, setUsername] = useState('emilys'); // demo
  const [password, setPassword] = useState('emilyspass'); // demo
  const [error, setError] = useState('');
  const [loading, setLoading] = useState(false);
  const navigate = useNavigate();

  const handleSubmit = async (e) => {
    e.preventDefault();
    setError('');
    setLoading(true);

    try {
      const r = await fetch('https://dummyjson.com/auth/login', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ username, password, expiresInMins: 30 })
      })
    }
  }
}

```

```
});

if (!r.ok) {
  let msg = 'Error de autenticación';
  try {
    const err = await r.json();
    msg = err?.message || msg;
  } catch (_) {}
  throw new Error(msg);
}

const data = await r.json();
const { accessToken } = data || {};
if (!accessToken) throw new Error('No se recibió accessToken');

localStorage.setItem('token', accessToken);

// Validar token
const me = await fetch('https://dummyjson.com/auth/me', {
  headers: { Authorization: `Bearer ${accessToken}` },
});
if (!me.ok) throw new Error('No se pudo cargar el perfil');

navigate('/home');
} catch (err) {
  setError(err.message || 'Hubo un problema al iniciar sesión');
} finally {
  setLoading(false);
}
};

return (
<Flex minH="100vh" align="center" justify="center" bg="gray.50">
<Box
  as="form"
  onSubmit={handleSubmit}
  bg="white"
  p={8}
  rounded="xl"
  shadow="md"
  minW={{ base: '90%', sm: '400px' }}
  textAlign="center"
>
<VStack spacing={4} mb={6}>
```

```

        {/* ◇ Logo UMG (puedes reemplazar por tu archivo local en /assets)
*/}
        <Image
            src="https://play-lh.googleusercontent.com/PAgEDMao5gLi5N-9x-EdPIihJHe0CRqscma-BQPunQoV887HW58Wi8ccdAtU2UwBnwo=w480-h960-rw"
            alt="Logo UMG"
            boxSize="80px"
            objectFit="contain"
        />
        <Heading size="md">Laboratorio 05 - APIs DummyJSON</Heading>
    </VStack>

    <FormControl mb={4}>
        <FormLabel>Usuario</FormLabel>
        <Input
            value={username}
            onChange={(e) => setUsername(e.target.value)}
            placeholder="usuario"
        />
    </FormControl>

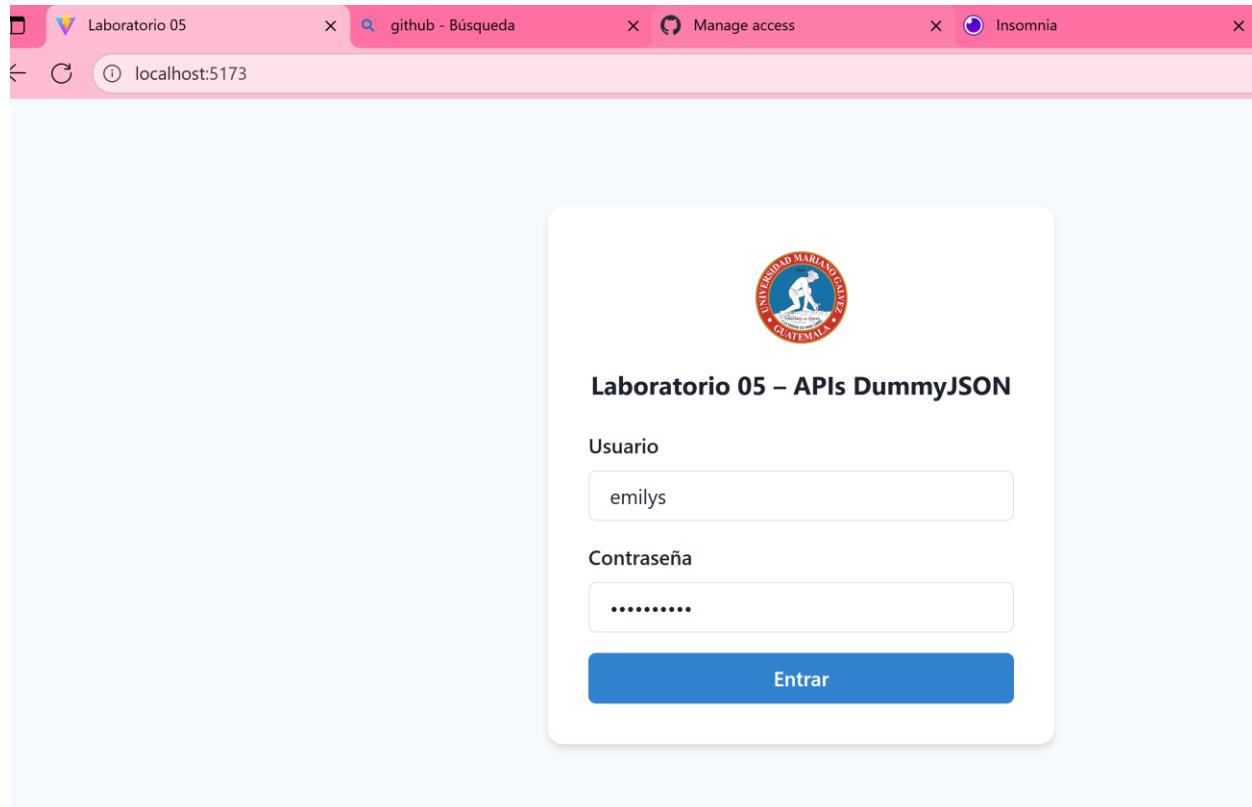
    <FormControl mb={4}>
        <FormLabel>Contraseña</FormLabel>
        <Input
            type="password"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
            placeholder="contraseña"
        />
    </FormControl>

    {error && <Text color="red.500" mb={4}>{error}</Text>}

    <Button type="submit" w="full" colorScheme="blue" isLoading={loading}>
        Entrar
    </Button>
</Box>
</Flex>
);
};

export default Login;

```



Código:

```
// src/Login.jsx

import React, { useState } from 'react';
import {
  Box, Button, Flex, FormControl, FormLabel, Input,
  Heading, Text, Image, VStack
} from '@chakra-ui/react';
import { useNavigate } from 'react-router-dom';

const Login = () => {

  const [username, setUsername] = useState('emilys'); // demo
  const [password, setPassword] = useState('emilyspass'); // demo
  const [error, setError] = useState("");
  const [loading, setLoading] = useState(false);

  const navigate = useNavigate();

  
```

```
const handleSubmit = async (e) => {
  e.preventDefault();
  setError("");
  setLoading(true);

  try {
    const r = await fetch('https://dummyjson.com/auth/login', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ username, password, expiresInMins: 30 })
    });

    if (!r.ok) {
      let msg = 'Error de autenticación';
      try {
        const err = await r.json();
        msg = err?.message || msg;
      } catch (_) {}

      throw new Error(msg);
    }

    const data = await r.json();
    const { accessToken } = data || {};
    if (!accessToken) throw new Error('No se recibió accessToken');

    localStorage.setItem('token', accessToken);

    // Validar token
  }
}
```

```
const me = await fetch('https://dummyjson.com/auth/me', {  
  headers: { Authorization: ` Bearer ${accessToken}` },  
});  
  
if (!me.ok) throw new Error('No se pudo cargar el perfil');  
  
  
  navigate('/home');  
}  
} catch (err) {  
  setError(err.message || 'Hubo un problema al iniciar sesión');  
}  
} finally {  
  setLoading(false);  
}  
};  
  
  
return (  
  <Flex minH="100vh" align="center" justify="center" bg="gray.50">  
    <Box  
      as="form"  
      onSubmit={handleSubmit}  
      bg="white"  
      p={8}  
      rounded="xl"  
      shadow="md"  
      minW={{ base: '90%', sm: '400px' }}  
      textAlign="center"  
    >  
    <VStack spacing={4} mb={6}>  
      {/* ♦ Logo UMG (puedes reemplazar por tu archivo local en /assets) */}  
      <Image
```

```
src="https://play-lh.googleusercontent.com/PAgEDMAo5gLi5N-9x-EdPljhJHe0CRqscma-BQPunQoV887HW58Wi8cccdAtU2UwBnwo=w480-h960-rw"
alt="Logo UMG"
boxSize="80px"
objectFit="contain"
/>
<Heading size="md">Laboratorio 05 – APIs DummyJSON</Heading>
</VStack>

<FormControl mb={4}>
  <FormLabel>Usuario</FormLabel>
  <Input
    value={username}
    onChange={(e) => setUsername(e.target.value)}
    placeholder="usuario"
  />
</FormControl>

<FormControl mb={4}>
  <FormLabel>Contraseña</FormLabel>
  <Input
    type="password"
    value={password}
    onChange={(e) => setPassword(e.target.value)}
    placeholder="contraseña"
  />
</FormControl>

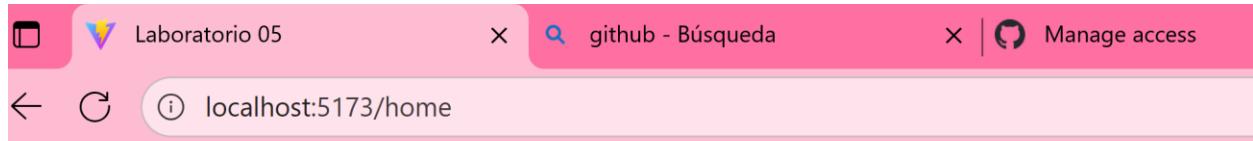
{error && <Text color="red.500" mb={4}>{error}</Text>}
```

```

<Button type="submit" w="full" colorScheme="blue" isLoading={loading}>
    Entrar
</Button>
</Box>
</Flex>
);
};

export default Login;

```



Home

Token presente: Sí

[Cerrar sesión](#)

[Ir a Productos](#)

CODIGO:

Pantalla Home – Explicación de componentes

1. Contenedor principal

- **Box:** se usa como un “div” con padding (p={8}), que da espacio interno y organiza los elementos en bloque.

2. Encabezado

- **Heading:** título grande que muestra “Home”. Sirve como cabecera de la página principal después de hacer login.

3. Estado de sesión

- **Text:** muestra un mensaje indicando si existe o no un token en localStorage.
 - Ejemplo: “Token presente: Sí” si hay sesión activa.
 - “Token presente: No” si no hay sesión (token borrado o no autenticado).

4. Acciones principales (Botones)

- **Flex con gap={4}:** organiza los botones en fila con espacio entre ellos.
- **Button (colorScheme="red") → Cerrar sesión**
 - Al hacer clic, ejecuta:
 - `localStorage.removeItem('token');`
 - `navigate('/');`
 - Esto **borra la sesión** (token) y redirige automáticamente al **Login**.
- **Button (colorScheme="blue") → Ir a Productos**
 - Usa Link de react-router-dom para navegar a /productos.
 - Permite entrar al listado con búsqueda, paginación y CRUD de productos.

1. Estados principales en el consumo de APIs

Loading (Cargando):

Se activa cuando se inicia la petición al endpoint de la API. Durante este estado normalmente se muestra un spinner o un mensaje indicando que los datos están cargando.

Success (Éxito):

Ocurre cuando la API responde correctamente (`status 200 OK`). En este punto se procesan los datos recibidos y se muestran en la interfaz (por ejemplo, la lista de productos).

Error (Fallo):

Sucede cuando la API devuelve un error (ej. `401 Unauthorized`, `404 Not Found`) o hay un problema de red. En la interfaz se despliega un mensaje de error al usuario.

Página de Productos

The screenshot shows a web browser window with the URL `localhost:3173/productos`. The page title is "Productos". The interface includes:

- Heading:** "Productos" (labeled "Heading")
- InputGroup + Input:** A search bar with placeholder "Buscar por título (DummyJSON)" and a dropdown menu set to "Por página: 10" (labeled "Input").
- Badge:** A badge indicating "TOTAL: 194" and "PAGINA 1 DE 20" with navigation links "Anterior" and "Siguiente" (labeled "Badge").
- Table:** A table listing products with columns "NOMBRE", "PRECIO", and "ACCIONES".
- Select:** A dropdown menu for selecting the number of items per page (labeled "Select").
- Buttons:** "Volver a Home", "Crear Producto", and several "Editar" and "Eliminar" buttons.
- Table Headers:** "NOMBRE", "PRECIO", and "ACCIONES".
- Table Data:**

NOMBRE	PRECIO	ACCIONES
Essence Mascara Lash Princess	\$9.99	<button>Editar</button> <button>Eliminar</button>
Eyeshadow Palette with Mirror	\$19.99	<button>Editar</button> <button>Eliminar</button>
Powder Canister	\$14.99	<button>Editar</button> <button>Eliminar</button>
Red Lipstick	\$12.99	<button>Editar</button> <button>Eliminar</button>
Red Nail Polish	\$8.99	<button>Editar</button> <button>Eliminar</button>
Calvin Klein CK One	\$49.99	<button>Editar</button> <button>Eliminar</button>
- Table Buttons:** "ton.", "Button", "Table", "Button", "Button", "Button".

Explicación de los componentes en la pantalla Productos

Encabezado

Heading (Chakra UI): “Productos” en la parte superior izquierda.

Se usa para indicar el título de la sección.

Barra superior de acciones

InputGroup + Input: Campo de búsqueda con placeholder “Buscar por título (DummyJSON)”.

Incluye SearchIcon (cuando está vacío) o CloseIcon (cuando hay texto, para limpiar).

Select: Dropdown “Por página: 10” que permite seleccionar cuántos registros ver por página (5, 10, 15, 20).

Button (outline): “Volver a Home”, que navega a la pantalla Home.

Button (colorScheme="blue"): “Crear Producto”, abre el formulario de creación.

Indicadores de paginación

Badge (colorScheme="purple"): muestra el total de registros (TOTAL: 194).

Badge (colorScheme="teal"): muestra la página actual (Página 1 de 20).

Button: “Anterior” (deshabilitado en la primera página).

Button: “Siguiente” (activo si hay más páginas).

Tabla de productos (Table)

Encabezado (Thead y Th):

ID

Nombre

Precio

Acciones

Cuerpo (Tbody y Tr con Td):

Cada fila representa un producto traído del API DummyJSON.

Ejemplo:

ID: 1

Nombre: Essence Mascara Lash Princess

Precio: \$9.99

Acciones en cada fila

Button (colorScheme="yellow"): “Editar” → lleva a /productos/:id/edit

Button (colorScheme="red"): “Eliminar” → lleva a /productos/:id/delete.

Código:

```
// src/Productos.jsx

import React, { useEffect, useMemo, useState } from "react";
import {
  Box, Button, Flex, Heading, Spinner, Table, Tbody, Td, Th, Thead, Tr, Text,
  Input, InputGroup, InputRightElement, IconButton, Select, HStack, Spacer, Badge
} from "@chakra-ui/react";
import { Link } from "react-router-dom";
import { CloseIcon, SearchIcon } from "@chakra-ui/icons";

const Productos = () => {
  const [productos, setProductos] = useState([]);
  const [total, setTotal] = useState(0);

  const [loading, setLoading] = useState(false);
  const [error, setError] = useState("");

  // UI state
  const [query, setQuery] = useState("");
  const [debouncedQ, setDebouncedQ] = useState("") // debounce para la búsqueda
  const [page, setPage] = useState(1);
  const [limit, setLimit] = useState(10);

  const pages = useMemo(() => Math.max(1, Math.ceil(total / limit || 1)), [total, limit]);
  const skip = useMemo(() => (page - 1) * limit, [page, limit]);

  // Debounce de la búsqueda
  useEffect(() => {
    const t = setTimeout(() => {
      setPage(1); // al cambiar búsqueda, volver a página 1
      setDebouncedQ(query.trim());
    }, 400);
    return () => clearTimeout(t);
  }, [query]);

  const fetchProductos = async () => {
    setLoading(true);
    setError("");
    try {

```

```
const base = "https://dummyjson.com";
const url = debouncedQ
  ?
`${base}/products/search?q=${encodeURIComponent(debouncedQ)}&limit=${limit}&skip=${skip}`
  : `${base}/products?limit=${limit}&skip=${skip}`;

const res = await fetch(url);
if (!res.ok) throw new Error("No se pudieron cargar los productos");
const data = await res.json();

// DummyJSON retorna { products, total, skip, limit }
setProductos(data.products || []);
setTotal(Number(data.total ?? 0));
} catch (err) {
  setError(err.message || "Error al cargar productos");
  setProductos([]);
  setTotal(0);
} finally {
  setLoading(false);
}
};

useEffect(() => {
  fetchProductos();
  // eslint-disable-next-line react-hooks/exhaustive-deps
}, [debouncedQ, page, limit]);

const canPrev = page > 1;
const canNext = page < pages;

const resetSearch = () => {
 setQuery("");
  setDebouncedQ("");
  setPage(1);
};

return (
<Box p={6}>
  <Flex direction={{ base: "column", md: "row" }} gap={3} align={{ md: "center" }} mb={4}>
    <Heading size="lg">Productos</Heading>
    <Spacer />
    <HStack spacing={3}>
      <Button as={Link} to="/home" variant="outline">Volver a Home</Button>
```

```
        <Button as={Link} to="/productos/create" colorScheme="blue">Crear
Producto</Button>
    </HStack>
</Flex>

        <Flex direction={{ base: "column", md: "row" }} gap={3} mb={4} align={{ md:
"center" }}>
    <InputGroup maxWidth="420px">
        <Input
            placeholder="Buscar por título (DummyJSON)"
            value={query}
            onChange={(e) => setQuery(e.target.value)}
        />
        <InputRightElement>
            {query ? (
                <IconButton
                    aria-label="Limpiar búsqueda"
                    size="sm"
                    variant="ghost"
                    icon={<CloseIcon boxSize={2.5} />}
                    onClick={resetSearch}
                />
            ) : (
                <SearchIcon opacity={0.6} />
            )}
        </InputRightElement>
    </InputGroup>

    <HStack>
        <Text>Por página:</Text>
        <Select
            value={limit}
            onChange={(e) => { setLimit(Number(e.target.value)); setPage(1); }}
            width="90px"
        >
            {[5,10,15,20].map(n => <option key={n} value={n}>{n}</option>)}
        </Select>
    </HStack>

    <Spacer />

    <HStack>
        <Badge colorScheme="purple">Total: {total}</Badge>
        <Badge colorScheme="teal">Página {page} de {pages}</Badge>
    </HStack>

```

```
<Button onClick={() => setPage(p => p - 1)}  
isDisabled={!canPrev}>Anterior</Button>  
    <Button onClick={() => setPage(p => p + 1)}  
isDisabled={!canNext}>Siguiente</Button>  
    </HStack>  
</Flex>  
  
{loading && <Spinner />}  
{error && <Text color="red.500" mb={3}>{error}</Text>}  
{!loading && !error && productos.length === 0 && (  
    <Text>No hay productos para mostrar.</Text>  
)}  
  
{productos.length > 0 && (  
    <Table variant="striped" colorScheme="gray">  
        <Thead>  
            <Tr>  
                <Th>ID</Th>  
                <Th>Nombre</Th>  
                <Th isNumeric>Precio</Th>  
                <Th>Acciones</Th>  
            </Tr>  
        </Thead>  
        <Tbody>  
            {productos.map((p) => (  
                <Tr key={p.id}>  
                    <Td>{p.id}</Td>  
                    <Td>{p.title}</Td>  
                    <Td isNumeric>{p.price != null ? `$$\{p.price\}` : "-"}</Td>  
                    <Td>  
                        <Button  
                            as={Link}  
                            to={`/productos/${p.id}/edit`}  
                            size="sm"  
                            mr={2}  
                            colorScheme="yellow"  
                        >  
                            Editar  
                        </Button>  
                        <Button  
                            as={Link}  
                            to={`/productos/${p.id}/delete`}  
                            size="sm"  
                            colorScheme="red"  
                        >  
                    </Td>  
                </Tr>  
            ))}  
        </Tbody>  
    </Table>  
)}
```

```
        Eliminar
      </Button>
    </Td>
  </Tr>
)
)
</Table>
)
<Box>
);
};

export default Productos;
```

CREAR NUEVO PRODUCTO

The screenshot shows a browser window with a pink header bar. The title bar includes the project name "Laboratorio 05", a GitHub search bar, and a "Manage access" button. The main content area has a light gray background and displays a "Crear Producto" form. The form consists of several input fields: "Nombre (title) *", "Precio", "Marca", "Categoría", "Descripción", and a "Guardar" button.

Nombre (title) *
Ej: TESLA X

Precio
Ej: 999

Marca
Ej: Tesla

Categoría
Ej: autos

Descripción
Detalles del producto

Guardar

CODIGO:

```
// src/ProductosCreate.jsx

import React, { useState } from "react";
import {
  Box, Button, FormControl, FormLabel, Input, Heading, Textarea,
  Text, Flex, SimpleGrid, NumberInput, NumberInputField, useToast
} from "@chakra-ui/react";
import { useNavigate, Link } from "react-router-dom";

const ProductosCreate = () => {

  const [title, setTitle] = useState("");
  const [price, setPrice] = useState("");
  const [brand, setBrand] = useState("");
  const [category, setCategory] = useState("");
  const [description, setDescription] = useState("");
  const [saving, setSaving] = useState(false);
  const [error, setError] = useState("");
  const toast = useToast();
  const navigate = useNavigate();

  const handleSubmit = async (e) => {
    e.preventDefault();
    setError("");
    if (!title.trim()) {
      setError("El nombre (title) es obligatorio");
      return;
    }

    setSaving(true);
    try {
```

```
const payload = {
    title: title.trim(),
    // DummyJSON acepta campos extra; price opcional
    ...(price !== "" ? { price: Number(price) } : {}),
    ...({brand.trim() ? { brand: brand.trim() } : {}},
    ...({category.trim() ? { category: category.trim() } : {}},
    ...({description.trim() ? { description: description.trim() } : {}}),
};

const res = await fetch("https://dummyjson.com/products/add", {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(payload),
});

if (!res.ok) throw new Error("No se pudo crear el producto");
await res.json(); // respuesta simulada

toast({
    title: "Producto creado (simulado).",
    status: "success",
    duration: 2000,
    isClosable: true,
});

navigate("/productos");

} catch (e){
    setError(e.message || "Error al crear el producto");
} finally{
```

```

    setSaving(false);

}

};

return (
<Box p={6}>
<Flex justify="space-between" align="center" mb={4}>
<Heading size="lg">Crear Producto</Heading>
<Button as={Link} to="/productos" variant="outline">
    Volver
</Button>
</Flex>

{error && <Text color="red.500" mb={4}>{error}</Text>}

<Box as="form" onSubmit={handleSubmit} maxW="720px">
<SimpleGrid columns={{ base: 1, md: 2 }} spacing={4}>
<FormControl isRequired>
<FormLabel>Nombre (title)</FormLabel>
<Input
    value={title}
    onChange={(e) => setTitle(e.target.value)}
    placeholder="Ej: TESLA X"
/>
</FormControl>

<FormControl>
<FormLabel>Precio</FormLabel>
<NumberInput min={0} value={price} onChange={(v) => setPrice(v)}>

```

```
<NumberInputField placeholder="Ej: 999" />
</NumberInput>
</FormControl>

<FormControl>
<FormLabel>Marca</FormLabel>
<Input
  value={brand}
  onChange={(e) => setBrand(e.target.value)}
  placeholder="Ej: Tesla"
/>
</FormControl>

<FormControl>
<FormLabel>Categoría</FormLabel>
<Input
  value={category}
  onChange={(e) => setCategory(e.target.value)}
  placeholder="Ej: autos"
/>
</FormControl>
</SimpleGrid>

<FormControl mt={4}>
<FormLabel>Descripción</FormLabel>
<Textarea
  value={description}
  onChange={(e) => setDescription(e.target.value)}
  placeholder="Detalles del producto"
/>
</FormControl>
```

```
    />

</FormControl>

<Button
  type="submit"
  colorScheme="blue"
  mt={6}
  isLoading={saving}
  isDisabled={!title.trim()}>
  Guardar
</Button>
</Box>
</Box>
);

};

export default ProductosCreate;
```

EDITAR



localhost:5173/productos/1/edit

Editar Producto #1

Nombre *

Essence Mascara Lash Princess

Precio

9.99

Guardar cambios

CODIGO:

```
// src/ProductosEdit.jsx
import React, { useEffect, useState } from "react";
import {
  Box, Button, FormControl, FormLabel, Input, Heading, Text, Spinner, Flex
} from "@chakra-ui/react";
import { useNavigate, useParams, Link } from "react-router-dom";

const ProductosEditar = () => {
  const { id } = useParams();
  const navigate = useNavigate();
  const [title, setTitle] = useState("");
  const [price, setPrice] = useState("");
  const [error, setError] = useState("");
  const [loading, setLoading] = useState(true);
  const [saving, setSaving] = useState(false);

  // Cargar datos actuales
  useEffect(() => {
```

```

const load = async () => {
  setError("");
  try {
    const res = await fetch(`https://dummyjson.com/products/${id}`);
    if (!res.ok) throw new Error("No se pudo cargar el producto");
    const data = await res.json();
    setTitle(data.title ?? "");
    setPrice(data.price ?? "");
  } catch (e) {
    setError(e.message || "Error cargando producto");
  } finally {
    setLoading(false);
  }
};

load();
}, [id]);

const handleSubmit = async (e) => {
  e.preventDefault();
  setSaving(true);
  setError("");
  try {
    const res = await fetch(`https://dummyjson.com/products/${id}`, {
      method: "PUT",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({
        title,
        price: price === "" ? undefined : Number(price),
      }),
    });
    if (!res.ok) throw new Error("No se pudo actualizar el producto");
    await res.json();
    navigate("/productos");
  } catch (e) {
    setError(e.message || "Error al guardar");
  } finally {
    setSaving(false);
  }
};

if (loading) return <Spinner ml={6} mt={6} />

return (
  <Box p={6}>
    <Flex justify="space-between" align="center" mb={4}>

```

```

<Heading size="lg">Editar Producto #{id}</Heading>
<Button as={Link} to="/productos" variant="outline">Volver</Button>
</Flex>

{error && <Text color="red.500" mb={4}>{error}</Text>}

<Box as="form" onSubmit={handleSubmit} maxW="520px">
  <FormControl mb={4} isRequired>
    <FormLabel>Nombre</FormLabel>
    <Input value={title} onChange={(e) => setTitle(e.target.value)} />
  </FormControl>

  <FormControl mb={6}>
    <FormLabel>Precio</FormLabel>
    <Input
      type="number"
      value={price}
      onChange={(e) => setPrice(e.target.value)}
      placeholder="Ej: 999"
    />
  </FormControl>

  <Button type="submit" colorScheme="blue" isLoading={saving}>
    Guardar cambios
  </Button>
</Box>
</Box>
);

};

export default ProductosEditar;

```

ELIMINAR

The screenshot shows a web browser window with a pink header bar. The title bar includes the text "Laboratorio 05", "github - Búsqueda", "Manage access", and a close button. The main content area has a light gray background and displays the following information:

Eliminar Producto #1

Nombre: Essence Mascara Lash Princess

Precio: \$9.99

Marca: Essence

Categoría: beauty

¿Confirmas que deseas eliminar este producto? (Operación simulada)

Sí, eliminar

```
// src/ProductosDelete.jsx
import React, { useEffect, useState } from "react";
import {
  Box, Button, Heading, Text, Spinner, Flex, Stack
} from "@chakra-ui/react";
import { useNavigate, useParams, Link } from "react-router-dom";

const ProductosDelete = () => {
  const { id } = useParams();
  const navigate = useNavigate();
  const [prod, setProd] = useState(null);
  const [loading, setLoading] = useState(true);
  const [deleting, setDeleting] = useState(false);
  const [error, setError] = useState("");

  useEffect(() => {
    const load = async () => {
      setError("");
      try {
        const res = await fetch(`https://dummyjson.com/products/${id}`);
        if (!res.ok) throw new Error("No se pudo cargar el producto");
        const data = await res.json();
        setProd(data);
      } catch (e) {
        setError(e.message);
      }
    };
    load();
  }, [id]);
  if (deleting) return (
    <Text>Eliminando producto...</Text>
  );
  if (loading) return (
    <Spinner>
      <Text>Cargando producto...</Text>
    </Spinner>
  );
  return (
    <Stack direction="column" spacing={4}>
      <Text>Nombre:</Text>
      <Text>Essence Mascara Lash Princess</Text>
      <Text>Precio:</Text>
      <Text>$9.99</Text>
      <Text>Marca:</Text>
      <Text>Essence</Text>
      <Text>Categoría:</Text>
      <Text>beauty</Text>
      <Text>¿Confirmas que deseas eliminar este producto? (Operación simulada)</Text>
      <Button color="red" onClick={() => setDeleting(true)}>Sí, eliminar</Button>
    </Stack>
  );
}

export default ProductosDelete;
```

```

        setError(e.message || "Error cargando producto");
    } finally {
      setLoading(false);
    }
  };
  load(),
}, [id]);

const handleDelete = async () => {
  setDeleting(true);
  setError("");
  try {
    const res = await fetch(`https://dummyjson.com/products/${id}`, {
      method: "DELETE",
    });
    if (!res.ok) throw new Error("No se pudo eliminar (simulado)");
    await res.json();
    navigate("/productos");
  } catch (e) {
    setError(e.message || "Error al eliminar");
  } finally {
    setDeleting(false);
  }
};

if (loading) return <Spinner ml={6} mt={6} />

return (
  <Box p={6}>
    <Flex justify="space-between" align="center" mb={4}>
      <Heading size="lg">Eliminar Producto #${id}</Heading>
      <Button as={Link} to="/productos" variant="outline">Cancelar</Button>
    </Flex>
    {error && <Text color="red.500" mb={4}>{error}</Text>}
    {prod ? (
      <Box borderWidth="1px" rounded="md" p={4} maxW="640px">
        <Stack spacing={2} mb={4}>
          <Text><b>Nombre:</b> {prod.title}</Text>
          {prod.price != null && <Text><b>Precio:</b> ${prod.price}</Text>}
          {prod.brand && <Text><b>Marca:</b> {prod.brand}</Text>}
          {prod.category && <Text><b>Categoría:</b> {prod.category}</Text>}
        </Stack>
    ) : <Text>Este producto no existe o ya fue eliminado.</Text>}
  </Box>
);

```

```
<Text mb={4} color="red.600">
  ¿Confirmas que deseas eliminar este producto? (Operación simulada)
</Text>

<Button colorScheme="red" onClick={handleDelete} isLoading={deleting}>
  Sí, eliminar
</Button>
</Box>
) : (
  <Text>No se encontró el producto.</Text>
)
</Box>
);
};

export default ProductosDelete;
```

Conclusión para el informe:

Usamos **Vite + React** porque juntos brindan un **entorno de desarrollo rápido, modular y eficiente**, ideal tanto para fines educativos como para proyectos reales. React aporta la **estructura y modularidad** para la interfaz, mientras que Vite ofrece un **flujo de trabajo ágil y moderno** que optimiza la experiencia del desarrollador.