Módulo 5
Desarrollo de aplicaciones Front-End con React

Consumo de API

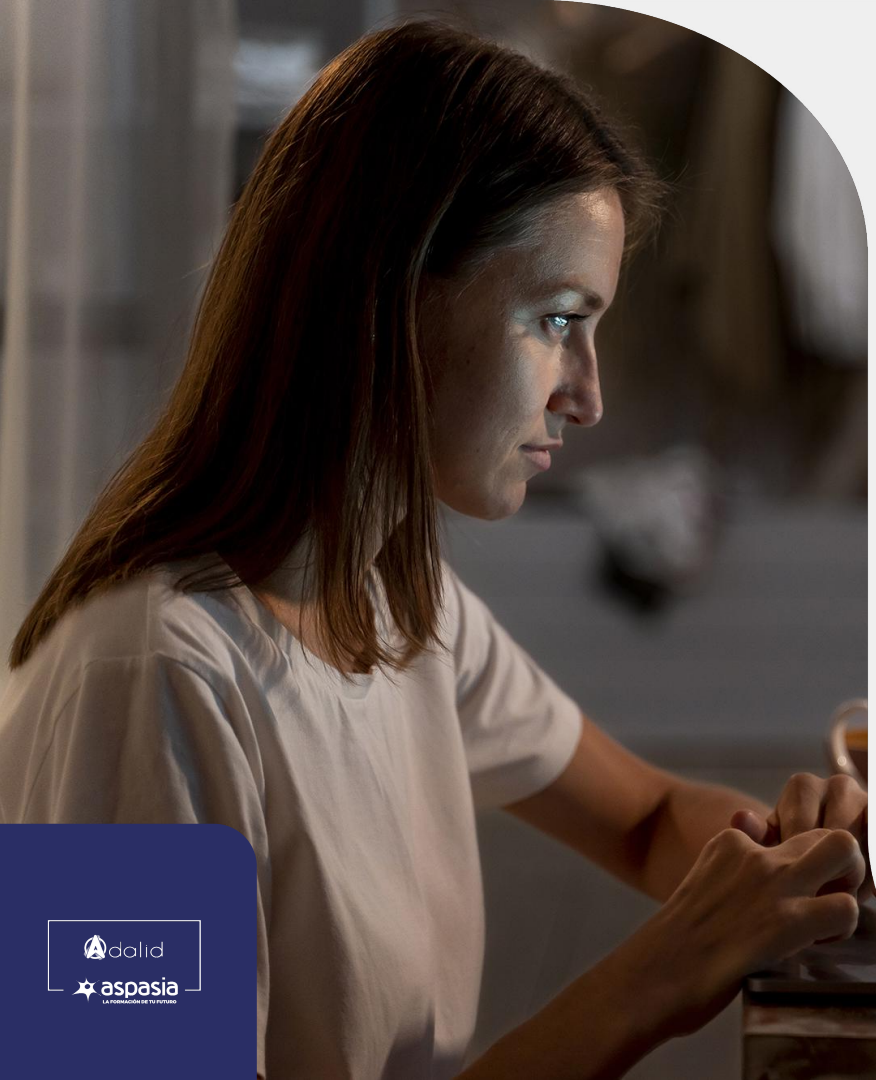
Módulo 5

AE 1.2

OBJETIVOS

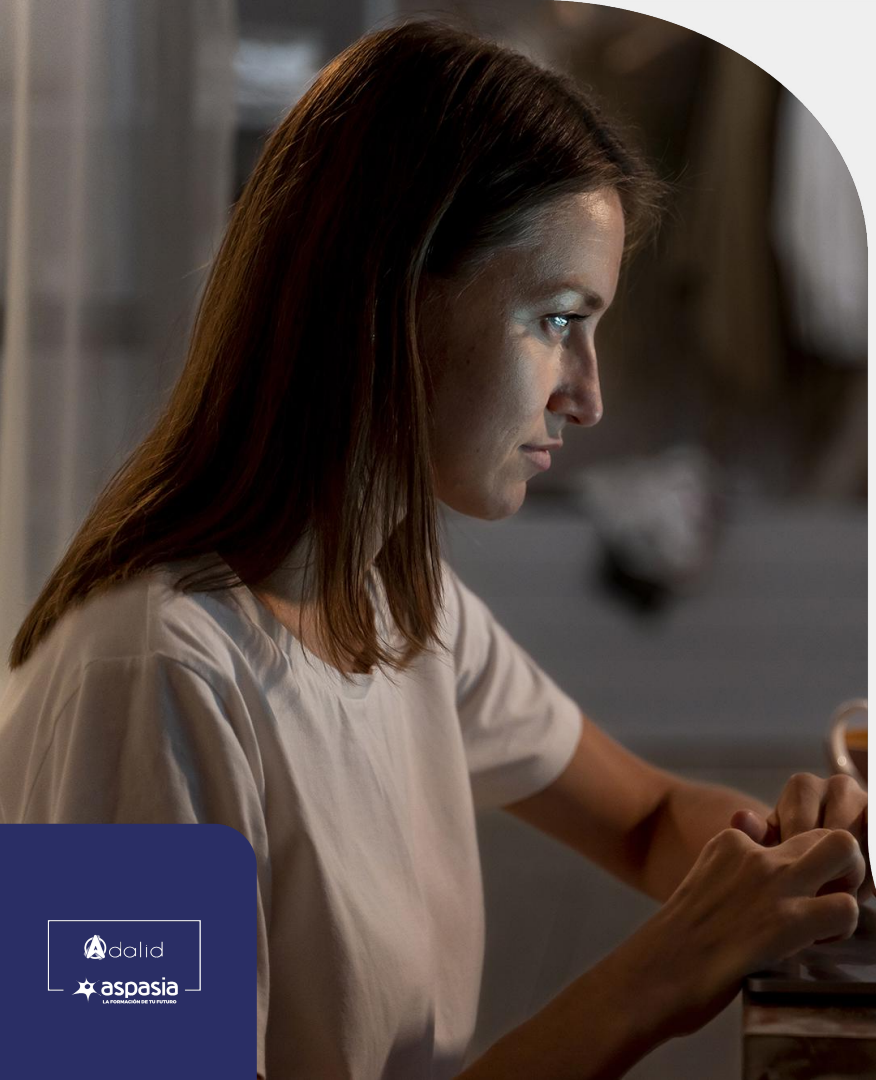
Aprender a consumir APIs en React usando Fetch y Axios, gestionando estado con hooks como useEffect y useState, para construir interfaces dinámicas y eficientes.

Chile
Adalid
aspasia
LA FORMACIÓN DE TU FUTURO



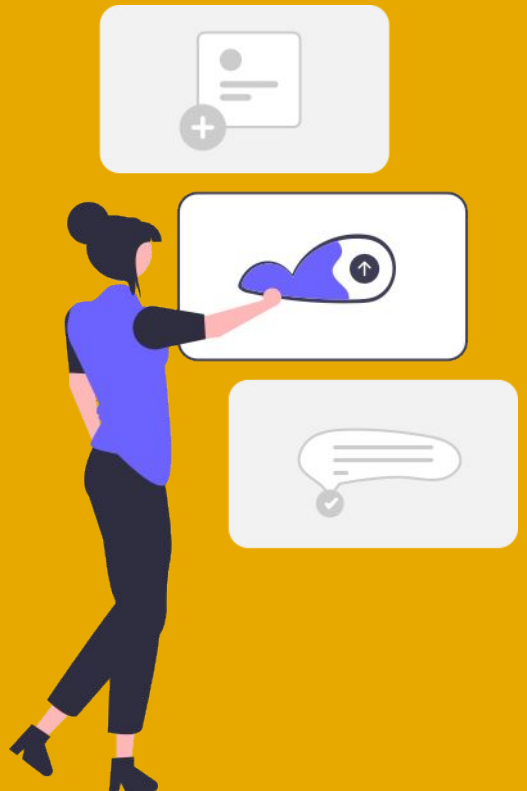
¿QUÉ VAMOS A VER?

- Consumo de API.
- El rol del front en una aplicación Cliente/Servidor.
- Interacción a través de APIs.
- Usando el Hook useEffect.
  - Qué hace useEffect.
  - Omite efectos para optimizar el rendimiento.
  - Cómo realizar peticiones en React con useEffect.
  - Cómo realizar peticiones a partir de eventos del usuario.
  - Manejando errores.



¿QUÉ VAMOS A VER?

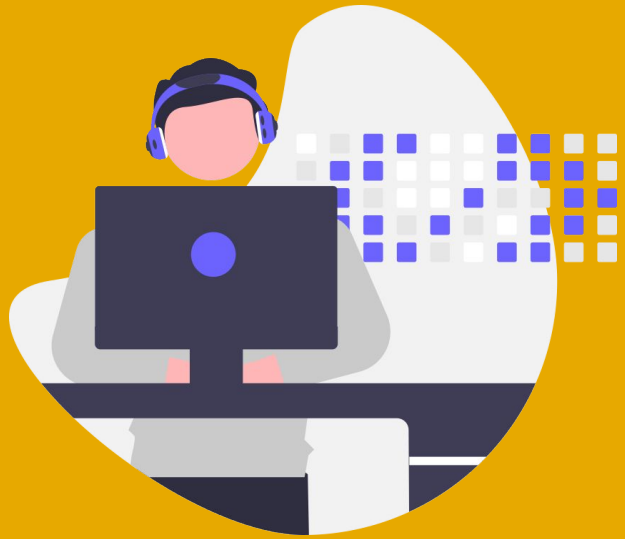
- Usando el Hook use Sate.
- Usando Fetch API.
  - Qué es Fetch API.
  - Cómo utilizar Fetch API.
  - Cómo manejar errores con Fetch API.
- Usando Axios.
  - Sintaxis.
  - Invocación.
  - Manejo de errores.
  - Usando Async/Await.
- Principales diferencias entre Fetch.
- API y Axios



Consumo de API

Adalid
aspasia
LA FORMACIÓN DE TU FUTURO

Pongamos a prueba lo aprendido 😊!!!



Adalid aspasia

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Desarrollaremos paso a paso una aplicación React usando Vite para **consumir APIs** y aplicar los conceptos aprendidos. Seguiremos el orden de las temáticas y avanzaremos integrando características en el proyecto.

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 1: Configuración inicial con Vite

Crea el proyecto con Vite:

```
npm create vite@latest api-consumption --template react
```

Instala las dependencias necesarias:

```
cd api-consumption
npm install
```

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 3: Consumo básico de API (1. Consumo de API)

Crea un componente **PostList.jsx** para obtener y mostrar datos.

```jsx
import { useState, useEffect } from 'react';

function PostList() {
  const [posts, setPosts] = useState([]);

  useEffect(() => {
    fetch('https://jsonplaceholder.typicode.com/posts')
      .then((response) => response.json())
      .then((data) => setPosts(data))
      .catch((error) => console.error('Error:', error));
  }, []);
```

```jsx
  return (
    <div>
      <h1>Lista de Posts</h1>
      <ul>
        {posts.slice(0, 10).map((post) => (
          <li key={post.id}>
            <h3>{post.title}</h3>
            <p>{post.body}</p>
          </li>
        ))}
      </ul>
    </div>
  );
}

export default PostList;
```

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 3: Consumo básico de API (1. Consumo de API)

Crea un componente **PostList.jsx** para obtener y mostrar datos.

Explicación del código:

- **useState([]):** Declara un estado inicial vacío para los posts.
- **useEffect:** Ejecuta una solicitud GET a la API cuando el componente se monta.
- **fetch:** Realiza la solicitud y convierte la respuesta a JSON.
- **Renderizado:** Muestra los primeros 10 posts en una lista.

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

function PostList() {
  const [posts, setPosts] = useState([]);

  useEffect(() => {
    fetch('https://jsonplaceholder.typicode.com/posts')
      .then((response) => response.json())
      .then((data) => setPosts(data))
      .catch((error) => console.error('Error:', error));
  }, []);

  return (
    <div>
      <h1>Lista de Posts</h1>
      <ul>
        {posts.slice(0, 10).map((post) => (
          <li key={post.id}>
            <h3>{post.title}</h3>
            <p>{post.body}</p>
          </li>
        ))}
      </ul>
    </div>
  );
}

export default PostList;
```

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 4: Agregar un formulario para publicar datos

Crea un componente **PostForm.jsx** para enviar nuevos posts.

Explicación:

- **fetch con POST:** Envía los datos al servidor para crear un nuevo recurso.
- **Renderizado:** Muestra un formulario simple para capturar información.

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

function PostForm() {
  const [title, setTitle] = useState('');
  const [body, setBody] = useState('');

  const handleSubmit = (e) => {
    e.preventDefault();
    fetch('https://jsonplaceholder.typicode.com/posts', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ title, body }),
    })
      .then((response) => response.json())
      .then((data) => console.log('Post creado:', data))
      .catch((error) => console.error('Error:', error));
  };

  return (
    <form onSubmit={handleSubmit}>
      <h2>Crear un nuevo post</h2>
      <input
        type="text"
        placeholder="Título"
        value={title}
        onChange={(e) => setTitle(e.target.value)}
        required
      />
      <textarea
        placeholder="Contenido"
        value={body}
        onChange={(e) => setBody(e.target.value)}
        required
      ></textarea>
      <button type="submit">Publicar</button>
    </form>
  );
}

export default PostForm;
```

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 5: Integrar los componentes en la aplicación principal

Edita **App.jsx** para integrar **PostList** y **PostForm**.

```
import PostList from './components/PostList';
import PostForm from './components/PostForm';

function App() {
  return (
    <div>
      <h1>Gestión de Posts</h1>
      <PostForm />
      <PostList />
    </div>
  );
}

export default App;
```

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 5: Integrar los componentes en la aplicación principal

Edita **App.jsx** para integrar **PostList** y **PostForm**.

```
import PostList from './components/PostList';
import PostForm from './components/PostForm';

function App() {
  return (
    <div>
      <h1>Gestión de Posts</h1>
      <PostForm />
      <PostList />
    </div>
  );
}

export default App;
```

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 6: Agregar optimizaciones (4. Usando el Hook useEffect)

Optimización de peticiones:

Modifica **PostList** para evitar realizar múltiples solicitudes innecesarias añadiendo dependencias en **useEffect**.

```
useEffect(() => {
  console.log('Cargando datos...');
  fetch('https://jsonplaceholder.typicode.com/posts')
    .then((response) => response.json())
    .then((data) => setPosts(data));
}, []); // La dependencia vacía asegura que se ejecute solo una vez.
```

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 7: Diferencias entre Fetch y Axios

Puedes reemplazar **fetch** con **Axios** en cualquier parte del proyecto para manejar errores y configuraciones avanzadas con mayor facilidad.

```
import axios from 'axios';

useEffect(() => {
  axios
    .get('https://jsonplaceholder.typicode.com/posts')
    .then((response) => setPosts(response.data))
    .catch((error) => console.error('Error:', error));
}, []);
```

Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 7: Diferencias entre Fetch y Axios

Puedes reemplazar **fetch** con **Axios** en cualquier parte del proyecto para manejar errores y configuraciones avanzadas con mayor facilidad.

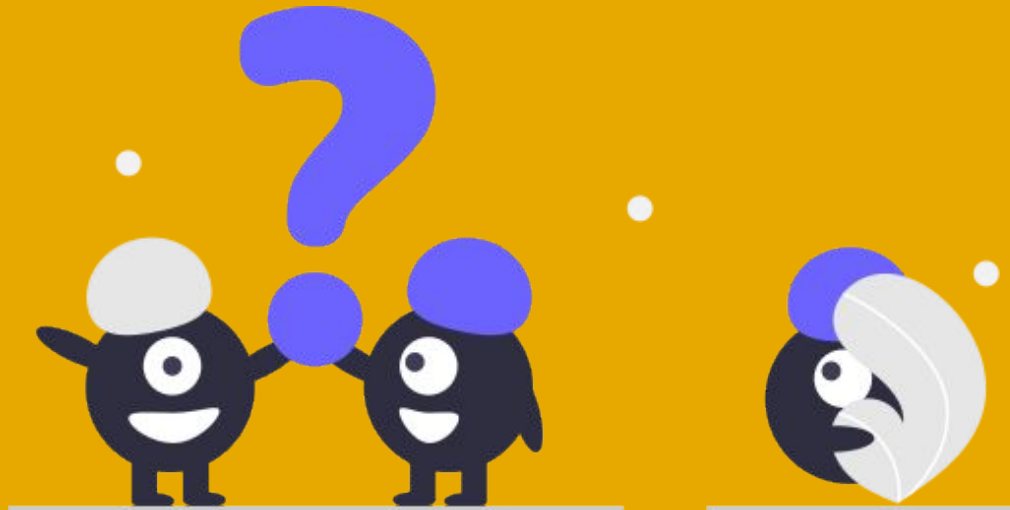
```
import axios from 'axios';

useEffect(() => {
  axios
    .get('https://jsonplaceholder.typicode.com/posts')
    .then((response) => setPosts(response.data))
    .catch((error) => console.error('Error:', error));
}, []);
```

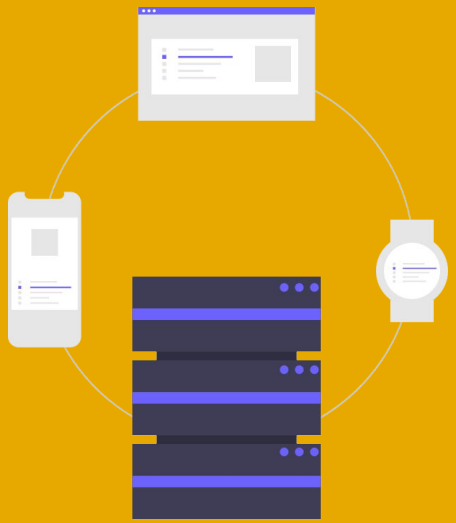
Ejercicio Guiado: Construcción de una Aplicación React para Consumo de APIs

Paso 7: Ejecuta el proyecto

Revisa la **consola del navegador** cuando generes un nuevo **Post.**

¿Como te fue con el Ejercicio guiado?

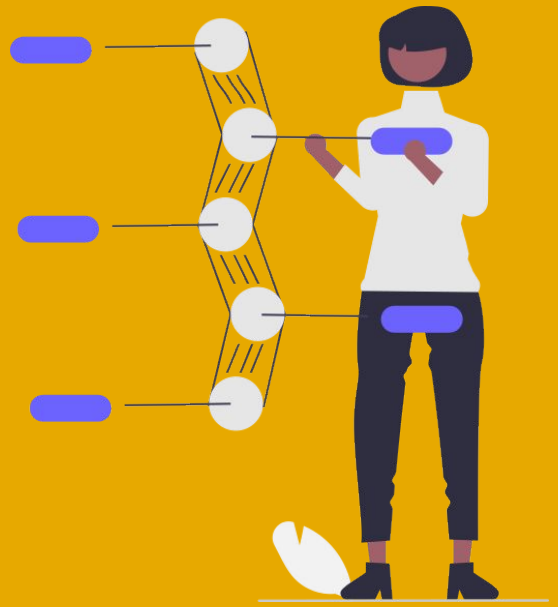




Visita el Repositorio de GitHub para ver ejemplos

Adalid
aspasia
LA FORMACIÓN DE TU FUTURO

Resumen de lo aprendido 🧑‍🎓



Adalid aspasia

Resumen de lo aprendido

- Aprendiste cómo conectar tu aplicación React con APIs externas para obtener y enviar datos de manera eficiente.
- Entendiste el rol clave del frontend en la comunicación cliente/servidor y cómo gestionar esa interacción.
- Descubriste cómo usar herramientas como useEffect, Fetch y Axios para manejar peticiones HTTP y sus respuestas.
- Aprendiste a gestionar errores y optimizar el rendimiento al consumir datos en React.

GRACIAS POR TU ATENCIÓN

Nos vemos en la próxima clase

Chile
Adalid
aspasia
LA FORMACIÓN DE TU FUTURO