**Guía 2 Ingeniería de Software**

**Bryan Stiven Failach Durango Laura Valentina González Rojas Valentina Blanco Alvis**

**Julián Mauricio Montoya Rodríguez Bryan Andres Puentes Vargas**

**Ingeniería Web   
Diana Marcela Toquica**

**Universidad Manuela Beltrán Ingeniería de Software Bogotá D.C**

**4 de marzo de 2025**

**Tabla de contenido**

[Estructura y diseño 3](#_bookmark0)

[Acceso 3](#_bookmark1)

[Catálogo 3](#_bookmark2)

[Principal 3](#_bookmark3)

[Compras 3](#_bookmark4)

[Tecnologías 4](#_bookmark5)

[Consumo de APIS 4](#_bookmark6)

[API de Pago (UCommerce) 4](#_bookmark7)

[Conexión Frontend - Backend 4](#_bookmark8)

[Manejo de Estados y Errores 4](#_bookmark9)

[Interactividad y Validación 4](#_bookmark10)

[Interactividad en el Frontend (React.js) 4](#_bookmark11)

[Validaciones en Formularios 4](#_bookmark12)

[Inicio de Sesión / Registro 4](#_bookmark13)

[Carrito de Compra y Pagos: 5](#_bookmark14)

[Manejo de Estados de Carga y Errores 5](#_bookmark15)

[Experiencia de Usuario Mejorada 5](#_bookmark16)

[Adaptabilidad 5](#_bookmark17)

[Diseño Responsive con Media Queries 5](#_bookmark18)

[Compatibilidad en Distintos Dispositivos 5](#_bookmark19)

[Referencias 6](#_bookmark20)

# Estructura y diseño

El proyecto se organizará en las siguientes carpetas:

# Acceso

Contiene las páginas relacionadas con el inicio de sesión y el registro de usuarios.

» Iniciar Sesión: Formulario para ingresar con credenciales.

» Registro: Formulario de creación de cuenta.

» Conexión a base de datos: Gestión de autenticación y almacenamiento de usuarios.

# Catálogo

Maneja la visualización de libros y sus detalles.

» Catálogo principal: Listado de libros disponibles.

» Descripción del libro: Información detallada de cada libro.

» Reseñas: Opiniones de los usuarios sobre los libros.

» Recomendaciones: Sugerencias personalizadas basadas en intereses y compras previas.

# Principal

Incluye la funcionalidad de búsqueda y organización del catálogo.

» Búsqueda: Barra de búsqueda para encontrar libros específicos.

» Categorías: Organización de libros por géneros o temas.

» Filtros: Opciones para refinar la búsqueda según precio, autor, editorial, etc.

# Compras

Gestiona todo el proceso de compra.

» Carrito de compra: Vista y gestión de los libros agregados al carrito.

» Confirmación de pago: Página de revisión y confirmación antes de pagar.

» Detalle de compra: Información de la transacción y estado del pedido

» API de pago: Integración con pasarelas de pago seguras.

# Tecnologías

» Fronted: React.js para la construcción de la interfaz de usuario.

» Estilos: CSS para el diseño y la presentación visual.

# Consumo de APIs en nuestra aplicación

La integración de APIs en nuestra aplicación cumple un rol fundamental para ofrecer una experiencia de usuario segura, fluida y eficiente. A continuación, se detallan los aspectos clave en la implementación de la API de pago, la comunicación entre el frontend y backend, el manejo de estados y errores, la interactividad, la validación de formularios y la adaptabilidad del diseño.

**1. Integración de la API de Pago (UCommerce)**

Para procesar pagos en línea de manera segura, se integrará la API de **UCommerce**, esta es una plataforma robusta y flexible. Su implementación permitirá gestionar transacciones en tiempo real de manera eficiente, optimizando así la experiencia de compra de los usuarios.

**Flujo de implementación:**

**Configuración inicial de uCommerce:** Se configurar uCommerce permitiendo configurar diversos métodos de pago, como tarjetas de crédito, débito, transferencias bancarias y soluciones como PayPal.

**Gestión de pagos y transacciones:** UCommerce, maneja la creación y el procesamiento de pagos en tiempo real, garantizando que estas transacciones se gestionen de forma segura. Al realizar una compra, uCommerce se encargará de verificar la validez del pago y procesando el pago a través del método de pago seleccionado por el cliente.

**Autenticación y validación:** El api, de uCommerce, verificara los datos proporcionados por el cliente, como el numero de la tarjeta, para asegurarse que sean correctos y se cumplan con las políticas de seguridad del comercio y las normativas bancarias.

**Manejo de errores y reintentos:** En caso de que una transacción falle, uCommerce, ofrecerá un mecanismo para que el cliente sea notificado y así permita corregir los errores cometidos, y poder reintentar el pago o utilizar otro método.

**Beneficios de esta implementación:**

**Seguridad avanzada:** UCommerce integra protocolos de seguridad avanzados, para garantizar que los datos de pago estén protegidos, de esta forma se cumplirán los estándares internacionales, lo que asegurara la protección de la información financiera.

**Facilidad de integración:** El api de uCommerce facilita la integración con una amplia variedad de pasarelas de pago proporcionando de esta forma flexibilidad para seleccionar la opción mas adecuada para cada tipo de negocio.

**Optimización de la experiencia de pago:** UCommerce proporciona una experiencia de pago fluida y rápida para los usuarios, permitiendo transacciones seguras, sin interrupciones ni complicaciones, esto contribuye a una buena experiencia de pago para el cliente

**Escalabilidad:** La plataforma UCommerce está diseñada para crecer junto entre líneas permitiendo gestionar un mayor volumen de transacciones sin sacrificar el rendimiento o la seguridad

**2. Comunicación entre Frontend y Backend**

Para la interacción eficiente entre la interfaz de usuario y el servidor, se utilizará **Fetch API** junto con métodos HTTP como **GET, POST, PUT y DELETE**. Esta comunicación permitirá la gestión de usuarios, compras y pedidos en tiempo real.

**Estrategia de implementación:**

* **Estructura de peticiones asíncronas:** Se realizarán solicitudes **fetch** con promesas o async/await para evitar bloqueos en la UI.
* **Manejo de tokens de autenticación:** Se usará un **JSON Web Token (JWT)** para asegurar que solo usuarios autenticados puedan acceder a ciertas funcionalidades.
* **Optimización de tráfico:** Se aplicará **caché de datos** para evitar sobrecargar el servidor con peticiones innecesarias.

**Ejemplo de flujo de comunicación:**

1. **Un usuario realiza una compra:** La UI genera una solicitud **POST** con los detalles del pedido.
2. **El backend procesa la solicitud:** Verifica la disponibilidad de los productos y confirma el pedido.
3. **Se genera un pago:** Se redirige al usuario a la pasarela de UCommerce.
4. **Recepción de confirmación:** UCommerce envía una respuesta al backend con el estado del pago, el cual se almacena en la base de datos.
5. **Notificación al usuario:** El frontend recibe la respuesta y muestra una confirmación de compra o un mensaje de error.

**Ventajas:**

* **Mejor rendimiento:** Reduciendo peticiones innecesarias y optimizando la comunicación con el servidor.
* **Mayor seguridad:** Implementando HTTPS y tokens de autenticación.
* **Experiencia fluida:** Los usuarios recibirán respuestas rápidas sin recargas de página.

**3. Manejo de Estados de Carga y Errores**

Para mejorar la experiencia del usuario, se implementará un sistema de manejo de estados y errores en **React.js**, permitiendo una navegación intuitiva y sin interrupciones.

**Estados de Carga:**

* Indicadores visuales para mostrar cuando la aplicación está procesando información.
* Skeleton loaders para mejorar la percepción de velocidad en la UI.

**Manejo de errores:**

* Captura de errores de red y fallos en la API.
* Reintentos automáticos en caso de problemas con la conexión.
* Mensajes de error específicos según el tipo de problema detectado.

**Ejemplo:** Si la API de UCommerce no responde, en lugar de mostrar un error genérico, la aplicación mostrará:  
"No pudimos procesar tu pago en este momento. Intenta nuevamente más tarde o utiliza otro método de pago.

**4. Interactividad y Validación en Formularios**

**Interactividad en el Frontend:**

Se implementarán eventos dinámicos en React.js para mejorar la usabilidad de la aplicación:

* **Efectos visuales y transiciones suaves** para una experiencia más atractiva.
* **Interacción en tiempo real con los formularios**, como autocompletado y validaciones instantáneas.
* **Feedback inmediato** cuando el usuario comete errores al ingresar datos.

**Validaciones de formularios:**

1. **Inicio de sesión y registro:**
   * Verificación de **campos obligatorios** antes de enviar el formulario.
   * Validación de **correos electrónicos** con expresiones regulares.
   * Restricción de contraseñas débiles para mejorar la seguridad.
2. **Carrito de compras y pagos:**
   * Validación en tiempo real de la **cantidad de productos disponibles**.
   * Confirmación de datos antes de proceder al pago.
   * **Manejo de errores en la API de pago** para evitar cobros duplicados.

**5. Experiencia de Usuarios Mejorada:**

Para garantizar una experiencia fluida, se implementarán varias mejoras en la interfaz y funcionalidad:

* **Autocompletado en búsqueda** para encontrar productos más rápido.
* **Sugerencias y recomendaciones personalizadas** basadas en compras previas.
* **Notificaciones en tiempo real** sobre el estado del pedido y cambios en la cuenta del usuario.

1. **Adaptabilidad y Diseño Responsive**

La aplicación será completamente **responsiva**, garantizando su correcto funcionamiento en distintos dispositivos y tamaños de pantalla.

**Estrategias de adaptación:**

* **Uso de CSS Media Queries** para ajustar el diseño en móviles, tablets y desktops.
* **Grid y Flexbox** para distribuir los elementos de manera flexible.
* **Optimización de imágenes y recursos** para reducir tiempos de carga en dispositivos móviles.

Además, se realizarán **pruebas en distintos navegadores y dispositivos** para asegurar compatibilidad total.

# Referencias

Diana Marcela Toquica Rodríguez. 2025. Material de clase. Universidad Manuela Beltrán, ingeniera