

Taller 01 - Requerimientos
ingeniería de software 1

Autores:

Adrian Ramirez Gonzalez

Adrian Alexander Benavides

Andrés Hernando Borda Muñoz

Brayan Alejandro Muñoz Pérez

Viernes 20 de diciembre de 2024

Universidad Nacional de Colombia

PUNTO 1

Conversación escogida: Conversación del grupo 2 (GP 2)

PUNTO 2

Levantamiento de requerimientos

La cliente es propietaria de un pequeño emprendimiento de venta de artesanías, accesorios y objetos para decoración del hogar. Los principales canales de venta de este emprendimiento son Instagram y WhatsApp, y además se hace uso de Excel para realizar la gestión de los pedidos. Esta forma de gestionar las ventas no ha sido efectiva para la cliente, haciéndole perder información valiosa sobre algunos pedidos realizados. Por esta razón, la propietaria del emprendimiento está interesada en tener una aplicación web que le ayude a gestionar todo este proceso de ventas, incluyendo la gestión de inventarios. Además, dentro de este software le gustaría poder incluir un nuevo servicio de asesorías para decoración de espacios.

Funcionalidades

1. Gestión de pedidos

- a. La dueña quiere que la aplicación web le permita registrar pedidos por medio de un formulario el cual pueda ser llenado tanto por los clientes como por la dueña.
- b. Los pedidos realizados deberán ser catalogados en 5 estados posibles: pendiente, pagado, enviado, completado y cancelado.
- c. Una vez se realiza un pedido, tanto la dueña como el cliente deberán recibir notificaciones sobre la realización del pedido. Estas notificaciones se realizarán por medio de WhatsApp.
- d. Los datos de los pedidos deben poder ser modificados. Esta posibilidad de modificación aplica tanto para los datos personales del cliente como para los productos solicitados.
- e. Debe existir una tabla donde la dueña pueda ver los pedidos más recientes.
- f. Se debe contar con la posibilidad de usar medios de pago como Nequi, Daviplata o alguna otra plataforma de este tipo.
- g. Dentro de la aplicación web debe ser posible personalizar los llaveros que la dueña vende y visualizarlos directamente dentro de la página web antes de realizar el pedido.
- h. Sobre cada pedido se debe guardar la siguiente información:
 - i. Número de pedido (ID único).
 - ii. Fecha del pedido.
 - iii. Productos solicitados.
 - iv. Estado del pedido (pendiente, pagado, enviado, completado o cancelado).
 - v. Fecha límite de entrega.
 - vi. Monto total del pedido.
 - vii. Medio de pago.
- i. Los usuarios deberán quedar registrados automáticamente cuando realicen un pedido, aunque su información podrá ser editada posteriormente. La información que se guardará de los clientes será la siguiente:
 - i. Nombre completo.
 - ii. Número de contacto.

- iii. Dirección de envío.
 - iv. Historial de pedidos.
- j. Se debe contar con una calculadora que estime los precios de envío de cada pedido.

2. Control de inventarios

- a. Debe haber un sistema de notificaciones el cual avise cuando quede poco stock de algún producto.
- b. La dueña debe poder editar el inventario en cualquier momento.
- c. Cada semana debe generarse un reporte sobre el estado del inventario el cual será enviado por correo electrónico a la dueña.
- d. Debe haber una conexión entre el inventario y los pedidos realizados de tal manera que el inventario actualice su stock cada vez que se realiza una compra.
- e. Sobre cada uno de los pedidos se debe guardar la siguiente información:
 - i. Nombre del producto.
 - ii. Precio.
 - iii. Cantidad disponible.
 - iv. Descripción breve.
 - v. Foto del producto (opcional).

3. Catálogo en línea

- a. El catálogo debe estar vinculado con el inventario con el fin de poder mostrar todos los productos que aún están en stock.
- b. Los productos deben estar clasificados en artesanías, accesorios o decoración.
- c. El catálogo debe tener un banner donde se muestren los productos más vendidos o los productos que se encuentren en promoción.
- d. El catálogo debe poder integrarse con redes sociales para aumentar el alcance de la tienda.

4. Agendamiento de asesorías

- a. La aplicación web debe contar con un calendario interactivo que permita agendar citas para las asesorías de decoración de espacios.

5. Gestión financiera

- a. La aplicación web debe almacenar y mostrar información sobre ingresos, gastos y ganancias netas de los productos.

6. Gestión de contenido

- a. Se deberá contar con una sección donde la dueña pueda publicar imágenes de los compradores junto con sus productos adquiridos.

- b. Debe haber una sección donde los clientes puedan compartir sus experiencias con los productos. Esta sección debe permitir la publicación tanto de comentarios como de fotos.

Especificaciones técnicas

- La aplicación web debe poder ser usada tanto en computador como en dispositivos móviles.
- La aplicación debe funcionar de manera online, aunque sería bueno que algunas funcionalidades básicas se puedan usar sin conexión a internet.
- Se debe implementar una interfaz sencilla que haga cómoda la navegación a lo largo de la aplicación web.
- Se debe facilitar la integración de nuevas funcionalidades que se vayan ideando a lo largo del tiempo.

Diseño

- Hacer poco uso de texto y centrarse más en el uso de imágenes.
- Colores principales:
 - o Beige claro o crema
 - o Terracota o salmón suave
- Colores secundarios:
 - o Gris suave o blanco para fondos.
 - o Verde oliva o musgo para botones y títulos.
- La tipografía usada deber ser Montserrat o Poppins

Por último, la cliente nos indica el siguiente orden de prioridad para el desarrollo de los distintos aspectos de su aplicación web:

1. Organización de los pedidos.
2. Gestión del inventario.
3. Gestión de ventas y pagos.
4. Catálogo en línea.
5. Promoción de sus productos.
6. Agendamiento de citas.

PUNTO 3

Análisis de requerimientos

FUNCIONALIDAD	MOSCOW	TIEMPO ESTIMADO (DÍAS)	RECURSOS NECESARIOS	TECNOLOGÍAS PROPUESTAS	COMENTARIOS
Registrar pedidos por formulario	Must have	8	1 Backend 1 Frontend	Backend: Node.js con Express Frontend: React.js	Incluye formulario para clientes y dueña; validación de datos.
Clasificación y gestión de pedidos	Must have	5	1 Backend	Node.js	Manejo de los estados: pendiente, pagado, enviado, completado, cancelado.
Notificaciones por WhatsApp	Must have	5	1 Backend	WhatsApp Business	Notificaciones para cliente y dueña al registrar el pedido.
Modificación de datos del pedido	Must have	5	1 Backend 1 Frontend	Node.js, React.js.	Permite editar productos o datos personales de los clientes.
Visualización de tabla de pedidos recientes	Must have	3	1 Frontend	React.js con diseño adaptativo (mobile-friendly).	Tabla en el dashboard para visualizar últimos pedidos.
Calculadora de costos de envío	Should have	3	1 Backend 1 Frontend	Lógica en Node.js, visualización en React.js.	Herramienta para estimar costos basados en ubicación.
Gestión de	Must	8	1 Backend	MongoDB	Incluye

inventarios	have		1 Frontend	(base de datos), Node.js, React.js.	notificaciones de bajo stock y actualización al realizar un pedido.
Reportes semanales del inventario	Should have	5	1 Backend	Node.js, integración con API de correo (Gmail o SendGrid gratuito).	Generación automática de reportes enviados por correo.
Vinculación del catálogo con el inventario	Should have	8	1 Backend 1 Frontend	MongoDB para sincronización, Node.js, React.js.	Sincronización para mostrar solo productos en stock.
Personalización de llaveros	Could have	8	1 Frontend	React.js con herramientas como Fabric.js o Konva para diseño interactivo.	Funcionalidad interactiva para diseñar llaveros.
Integración con pasarelas de pago	Should have	8	1 Backend	API PSE (sincronización con Node.js).	Compatible con Nequi y Daviplata; integra APIs externas.
Agendamiento de citas	Won't have	—	—		No priorizado en esta fase, debido a complejidad y bajo impacto inmediato.
Gestión financiera	Could have	8	1 Backend	Node.js y MongoDB para cálculos y almacenamiento de datos.	Almacena ingresos, gastos y ganancias netas de productos.
Sección de experiencias de	Could have	5	1 Backend 1 Frontend	React.js para diseño y	Los clientes pueden

clientes				MongoDB para almacenamiento.	publicar comentarios y fotos de productos.
----------	--	--	--	------------------------------------	---

Estimación de recursos:

Equipo Propuesto:

1. Backend Dev (1 persona): Encargado de implementar la lógica del negocio, integración con APIs y base de datos.
2. Frontend Dev (1 persona): Diseña e implementa interfaces interactivas y visualizaciones.
3. Product Manager/Analista (1 persona): Monitorea el progreso, asegura cumplimiento del tiempo y gestiona requisitos del cliente. Es un rol compartido con alguno de los anteriores debido a la limitación del presupuesto.

Duración Total

- El desarrollador Backend tardaría 68 días en cumplir con la creación de las funcionalidades mencionadas, mientras que el desarrollador Frontend tardaría 48 días, por lo cual se esperaría que al finalizar los 68 días la aplicación se encontrara totalmente construida.

Costo Estimado

1. **Salario por día laboral (freelancer promedio en Colombia):**
 - Backend Dev: \$100,000/día.
 - Frontend Dev: \$100,000/día.
2. **Cálculo total:**
 - Costo desarrollador Frontend: $\$100,000 \times 48 = \$4,800,000$
 - Costo desarrollador Backend: $\$100,000 \times 68 = \$6,800,000$

El cliente tiene un presupuesto máximo de \$1,000,000, por lo cual será necesario renegociar el presupuesto.

Alternativas de reducción de costos

1. Reutilización de código: Usar plantillas gratuitas para React.js o Node.js.
2. Módulos escalonados: Priorizar las funcionalidades Must have para la primera fase.
 - a. Costo desarrollador Frontend (solo Must have): $\$100,000 \times 24 = \$2,400,000$
 - b. Costo desarrollador Backend (solo Must have): $\$100,000 \times 31 = \$3,100,000$

Tecnologías Propuestas

1. Backend: Node.js con Express.
2. Frontend: React.js.

3. Base de datos: MongoDB (gratis para pequeños proyectos).
4. Notificaciones: API WhatsApp Business.
5. Correo y reportes: SendGrid gratuito o integración con Gmail.

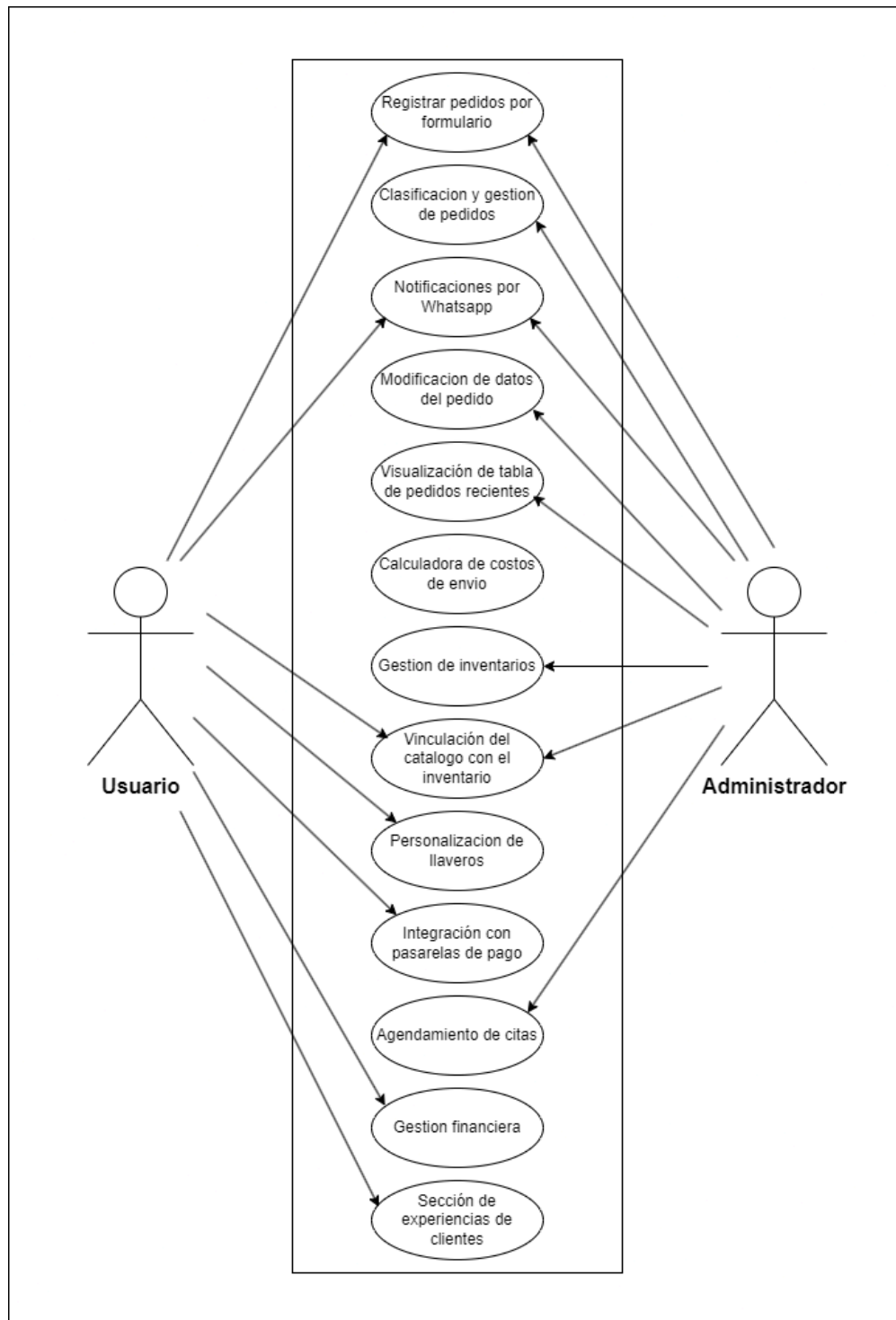
Alcance y Consideraciones

1. Fase inicial: Desarrollo solo de las funcionalidades Must have en 31 días, ajustando al presupuesto limitado.
2. Fase futura: Incorporar Should have y Could have como actualizaciones escalonadas.
3. Escalabilidad: Asegurar que el diseño permita agregar funcionalidades sin reestructurar el proyecto.

PUNTO 4

Diagrama de casos de uso y especificación de un caso de uso del administrador y el usuario

Diagrama de casos de uso



Caso de uso del administrador

Gestión de inventarios
Actor Administrador
Breve descripción El administrador podrá gestionar todo el inventario de productos que ofrece la tienda
Descripción paso por paso <ol style="list-style-type: none">1. El administrador va a la sección de inventario en la aplicación.2. El sistema despliega una lista con todos los productos junto con su información más importante. Cada producto tiene 3 botones llamados “consultar”, “eliminar” y “editar”. Además, al inicio de la lista hay un botón llamado “agregar”.<ol style="list-style-type: none">a. Botón “consultar”<ol style="list-style-type: none">i. El administrador escoge el botón consultar.ii. El sistema muestra toda la información del producto.iii. El administrador puede darle al botón “Volver” para regresar al listado de productos (Volver al paso 2).b. Botón “eliminar”<ol style="list-style-type: none">i. El administrador escoge el botón “eliminar”.ii. El sistema elimina el producto del listado y de la base de datos.iii. Se actualiza el stock del catálogo.iv. El administrador puede darle al botón “Volver” para regresar al listado de productos (Volver al paso 2).c. Botón “editar”<ol style="list-style-type: none">i. El administrador escoge el botón “editar”.ii. El sistema muestra toda la información del producto en cuadros de texto los cuales pueden ser modificados.iii. El administrador realiza las modificaciones.iv. El administrador confirma las modificaciones<ol style="list-style-type: none">1. Si las modificaciones no son válidas, el sistema le pide volver a realizar las modificaciones (Volver al paso 2.c.iii).v. Se actualiza el listado y la base de datos.vi. El sistema redirige al usuario al listado inicial (Volver al paso 2).d. Botón “agregar”<ol style="list-style-type: none">i. El administrador escoge el botón “agregar”.ii. El sistema despliega un formulario con todos los campos que deben ser rellenados para crear un nuevo producto.iii. El administrador llena el formulario.iv. El administrador confirma la creación del producto<ol style="list-style-type: none">1. Si alguno de los campos no fue llenado, el sistema le pide llenar los campos vacíos (Volver al paso 2.d.iii).v. Se actualiza el listado y la base de datos.vi. El sistema redirige al usuario al listado inicial (Volver al paso 2).

Caso de uso del usuario

Transacciones por medio de la pasarela de pago
Actor Usuario
Breve descripción El usuario podrá hacer de una pasarela de pago para poder realizar las compras de los productos ofrecidos en la tienda
Precondiciones El usuario debe haber seleccionado uno o más productos de la tienda
Descripción paso por paso <ol style="list-style-type: none">1. El usuario va a la sección de pagos.2. El sistema le muestra los diferentes métodos de pagos disponibles.3. El usuario escoge su método de pago de preferencia<ol style="list-style-type: none">a. Si el método de pago no se encuentra disponible, el usuario es regresado a la sección inicial para que escoja otro método de pago (Volver al paso 2).4. El usuario ingresa la información necesaria para hacer uso del método de pago de interés.<ol style="list-style-type: none">a. Si la información no es válida, se le pide al usuario ingresar su información nuevamente (Volver al paso 4).5. El sistema valida la información y se completa la transacción.6. El sistema genera el pedido
Se notifica tanto al usuario como al administrador sobre el pedido. Se actualiza el stock

PUNTO 5 HU 1 (Adrian Ramirez Gonzalez)

Gestión de inventarios

Anexo de documentos relacionados

1. Caso de uso: Gestión de inventarios

Descripción conceptual

Módulo	Módulo de gestión de inventario
Descripción de la(s) funcionalidad(es) requerida(s):	<p>El administrador podrá gestionar los productos que posee. Esto implica la realización de las siguientes operaciones:</p> <ol style="list-style-type: none">1. Consulta de productos2. Edición de la información de los productos3. Eliminación de productos4. Agregación de nuevos productos

Descripción técnica

Backend

1. Consulta de producto

URL	Método	Código html
/productos/{id}	GET	200
Al realizar una consulta se debe retornar un código 200 con toda la información del producto consultado		
Datos de entrada	Datos de salida	
	<pre>200: { "status": "success", "data": { "id": 1, "nombre": "Producto X", "precio": 100.0, "cantidad": 20 "descripcion": "descripcion",</pre>	

	<pre> "foto": "imagen.jpg" } } </pre>
--	---

2. Eliminar producto

URL	Método	Código html
/productos/{id}	DELETE	200 404
Al realizar la eliminación se debe retornar un código 200 indicando que la eliminación fue exitosa o un código 404 indicando un error en la eliminación del producto		
Datos de entrada	Datos de salida 200: <pre> { "status": "success", "message": "El producto ha sido eliminado." } </pre>	
Datos de entrada	404: <pre> { "status": "error", "message": "Producto no encontrado." } </pre>	

3. Editar producto

URL	Método	Código html
/productos/{id}	PATCH	200 422
Al realizar la modificación, se debe retornar un código 200 con la información del producto modificada o un código 422 indicando que los datos de modificación no son válidos		
Datos de entrada 200: { "nombre": "Producto Actualizado", "precio": 120.0, "cantidad": 15, "descripcion": "descripcion", "foto": "imagen.jpg" }	Datos de salida 200: { "status": "success", "data": { "id": 1, "nombre": "Producto Actualizado", "precio": 120.0, "cantidad": 15, "descripcion": "descripcion", "foto": "imagen.jpg" } }	
422: { "nombre": "", "precio": -50, "cantidad": "abc", "descripcion": "descripcion", "foto": "imagen.jpg" }	422: { "status": "error", "message": "Datos de entrada no válidos" }	

4. Agregar producto

URL	Método	Código html
/productos	POST	201 422
Al realizar la creación de un nuevo producto se debe retornar un código 201 con la información del producto agregado o un código 422 indicando que hay campos que no han sido llenados		
Datos de entrada		Datos de salida
201: { "nombre": "Nuevo Producto", "precio": 150.0, "cantidad": 10, "descripcion": "descripcion", "foto": "imagen.jpg" }		201: { "status": "success", "data": { "id": 2, "nombre": "Nuevo Producto", "precio": 150.0, "cantidad": 10, "descripcion": "descripcion", "foto": "imagen.jpg" } }
422: { "nombre": "", "precio": "", "cantidad": "-5", "descripcion": "descripcion", "foto": "imagen.jpg" }		422: { "status": "error", "message": "Faltan campos requeridos" }

Frontend

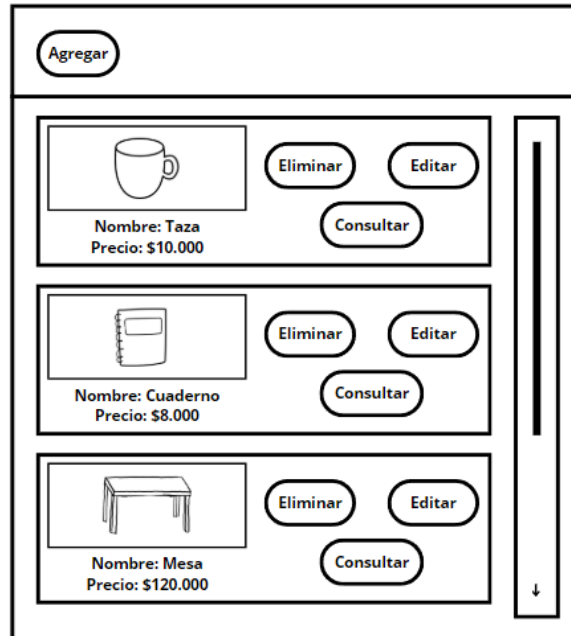
Interacción esperada:

1. **Consultar:** Al hacer clic en el botón "Consultar", se abre una ventana con la información del producto seleccionado.
2. **Eliminar:** Al hacer clic en el botón "Eliminar", el producto se elimina del listado.

3. **Editar:** Al hacer clic en el botón "Editar", se abre un formulario prellenado con la información del producto. El usuario realiza los cambios y guarda.
4. **Agregar:** Al hacer clic en el botón "Agregar", se abre un formulario en blanco. El usuario ingresa los datos y confirma para agregar el nuevo producto.

Mockups/Prototipos

1. Página principal con la lista de productos



The mockup shows a mobile application interface for a product list. At the top, there is a button labeled "Agregar". Below it, there is a list of three products, each in a separate card. Each card contains an image of the product, its name, its price, and three action buttons: "Eliminar", "Editar", and "Consultar". The products listed are: a cup (Taza) for \$10.000, a notebook (Cuaderno) for \$8.000, and a table (Mesa) for \$120.000. A vertical scrollbar is visible on the right side of the list.

Agregar

Eliminar Editar Consultar

Nombre: Taza
Precio: \$10.000

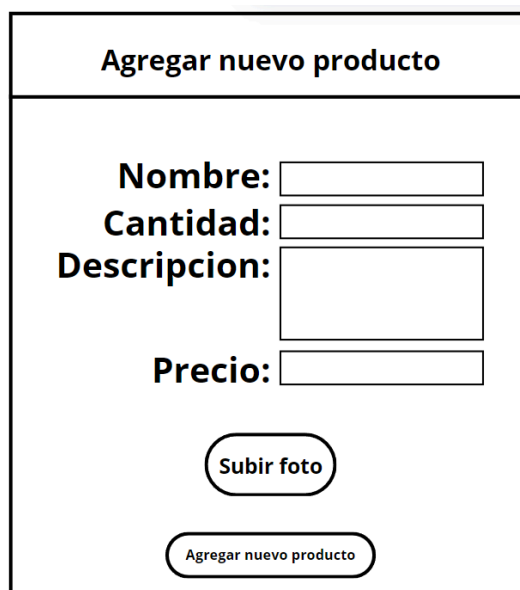
Eliminar Editar Consultar

Nombre: Cuaderno
Precio: \$8.000

Eliminar Editar Consultar

Nombre: Mesa
Precio: \$120.000

2. Página para la adición de nuevos productos



The mockup shows a mobile application interface for adding a new product. The title is "Agregar nuevo producto". Below the title, there are four input fields: "Nombre:", "Cantidad:", "Descripcion:", and "Precio:". Below the input fields, there are two buttons: "Subir foto" and "Agregar nuevo producto".

Agregar nuevo producto

Nombre:

Cantidad:

Descripcion:

Precio:

Subir foto

Agregar nuevo producto

3. Ventana emergente que indica la eliminación de un producto

El producto ha sido eliminado

Aceptar

4. Página para realizar la modificación de productos

Editar producto Taza

Nombre:

Cantidad:

Descripción:

Precio:

Subir foto

Editar producto

5. Página para realizar la consulta de productos

Taza



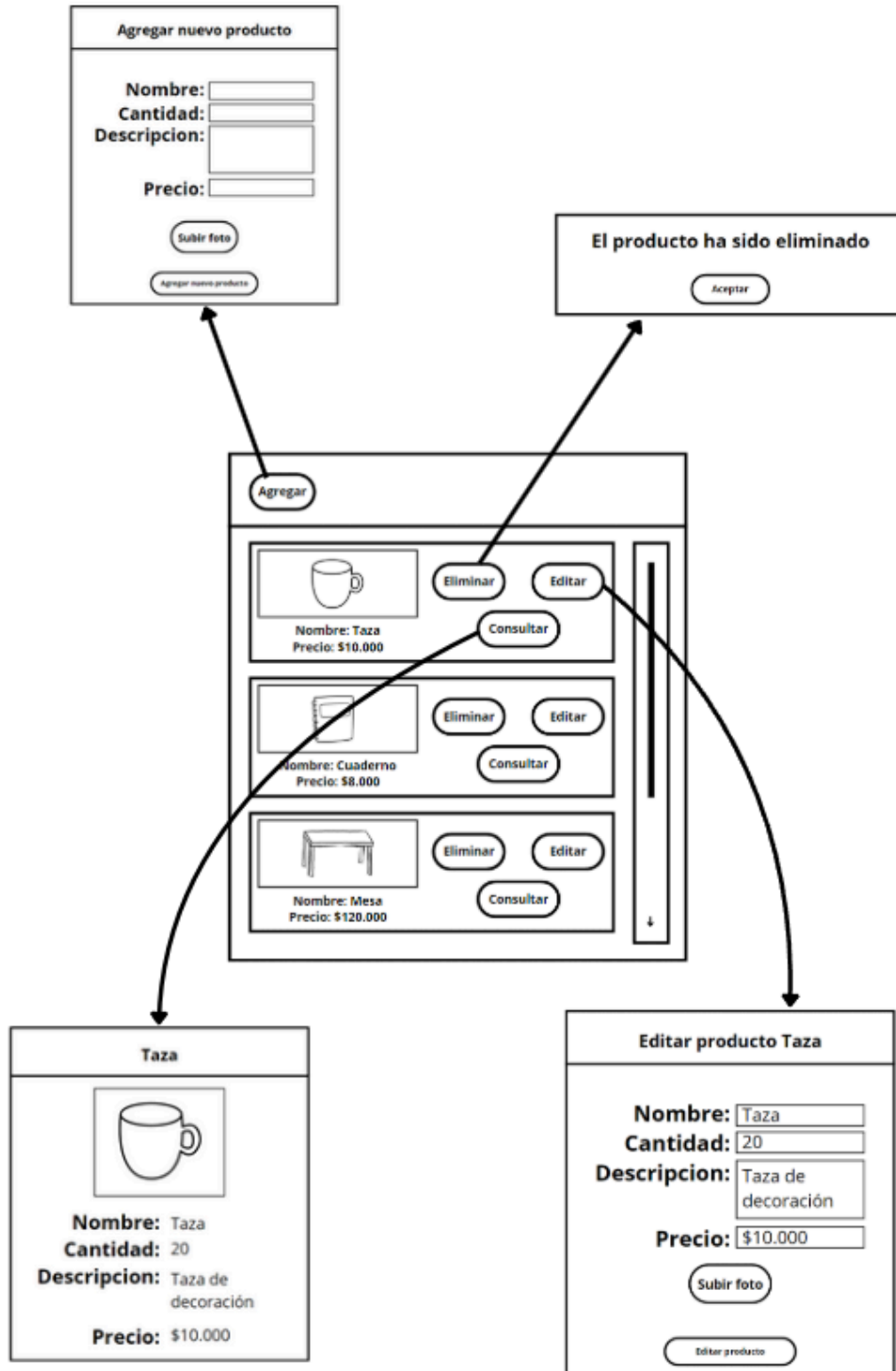
Nombre: Taza

Cantidad: 20

Descripción: Taza de decoración

Precio: \$10.000

Flujo visual y eventos



Notas Adicionales:

- Validar campos requeridos tanto en frontend como en backend.

PUNTO 5 HU 2 (Adrian Alexander Benavides)

Sección de experiencia de clientes

Descripción conceptual

Módulo	Sección de experiencia de clientes
Descripción de la(s) funcionalidad(es) requerida(s):	<p>Los usuarios deben tener un espacio en la aplicación donde puedan compartir sus experiencias con los productos. Dentro de esta sección los usuarios pueden realizar las siguientes acciones:</p> <ol style="list-style-type: none">1. Publicar comentarios sobre los productos2. Subir fotos con sus productos adquiridos3. Ver los comentarios y fotos de otros usuarios <p>Esta sección fomenta la confianza de los clientes al permitirles compartir reseñas reales y experiencias sobre los productos. Además, impulsa la imagen de la marca al facilitar interacciones entre usuarios.</p>

Descripción técnica

Backend

1. Publicar experiencias (comentarios y fotos)

URL	Método	Código html
/experiencias	POST	201 422
Al realizar una publicación se puede retornar un código 201 donde se confirma que se realizó la publicación junto con información relacionada al comentario o se puede retornar un código 422 indicando que no se llenaron los campos mínimos		
Datos de entrada		Datos de salida
201: { "cliente_id": 123,		201: { "status": "success",

<pre> "producto_id": 1, "comentario": "Excelente producto, lo recomiendo mucho.", "foto": "url_foto.jpg" } </pre>	<pre> "data": { "id": 456, "cliente_id": 123, "producto_id": 1, "comentario": "Excelente producto, lo recomiendo mucho.", "foto": "url_foto.jpg", "fecha": "2024-12-20" } } </pre>
<pre> 422: { "cliente_id": "", "producto_id": 1, "comentario": "", "foto": "" } </pre>	<pre> 422: { "status": "error", "message": "Faltan campos requeridos." } </pre>

2. Visualizar otras experiencias

URL	Método	Código html
/experiencias? producto_id=1	GET	200
Al realizar la visualización de experiencias se debe retornar un código 200 junto con un array que contiene todos las experiencias asociadas a un producto en particular		
Datos de entrada	Datos de salida	
	<pre> 200: { "status": "success", "data": [{ "id": 456, "cliente_id": 123, "producto_id": 1, "comentario": "Excelente producto, lo recomiendo mucho.", "foto": "url_foto.jpg", "fecha": "2024-12-20" }, </pre>	

	<pre> { "id": 789, "cliente_id": 124, "producto_id": 1, "comentario": "Cumple con las expectativas.", "foto": "url_foto2.jpg", "fecha": "2024-12-19" } 1 } </pre>
--	---

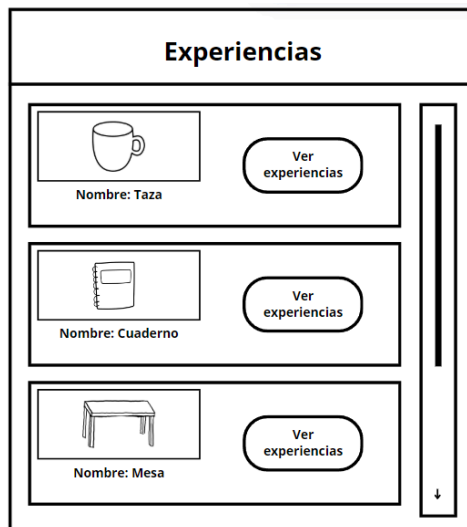
Frontend

Interacción esperada:

1. **Publicar experiencia:** El usuario llena un formulario con su comentario y sube una foto. Al enviarlo, se muestra un mensaje de confirmación.
2. **Ver experiencias:** En la página del producto, se despliega un listado de experiencias con comentarios y fotos.

Mockups/Prototipos

1. Lista con los productos



2. Experiencias de un producto

Experiencias con la taza

The diagram illustrates a feedback loop for a cup experience. It is structured as follows:

- Top Section:** A large box containing a simple line drawing of a cup.
- Bottom Section:** A large box containing two smaller boxes, each with an icon and a text label:
 - Left Box:** An icon of a cup on a table. The text next to it reads: "Excelente producto, lo recomiendo mucho."
 - Right Box:** An icon of a person holding a cup. The text next to it reads: "Cumple con lo prometido."
- Right Side:** A vertical bar with a downward-pointing arrow, indicating a flow or continuation of the process.

3. Formulario para la publicación de una experiencia

Publicar experiencia

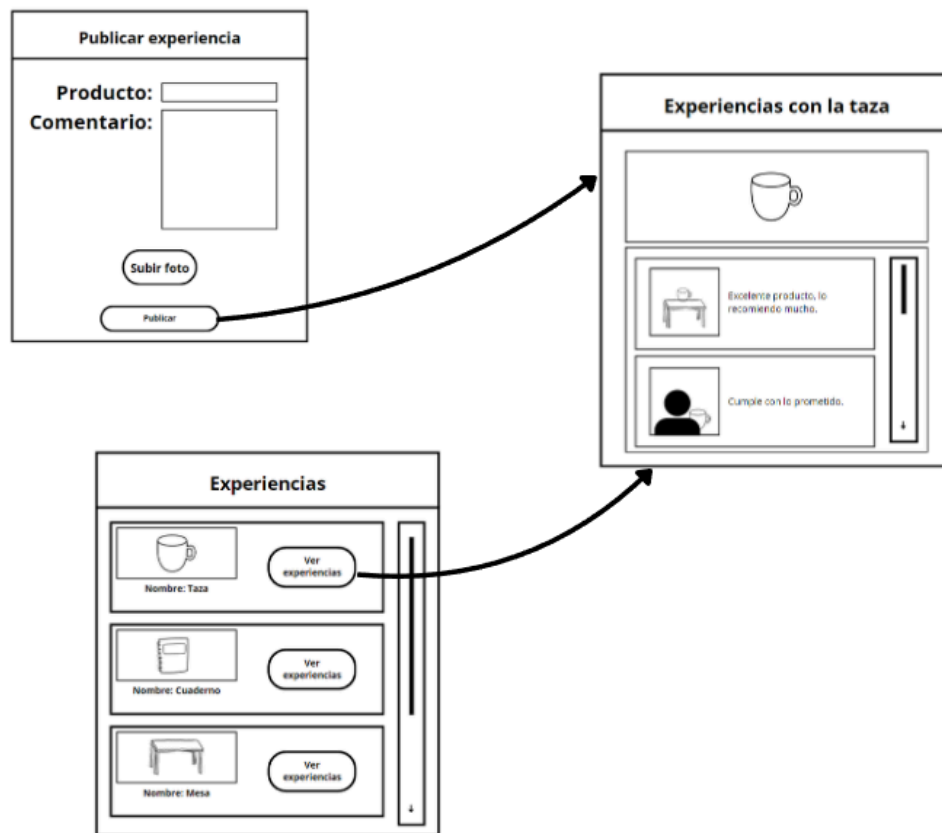
Producto:

Comentario:

Subir foto

Publicar

Flujo visual y eventos



Notas Adicionales:

- Validar los comentarios y fotos para evitar contenido ofensivo.
- Implementar filtros automáticos de palabras ofensivas en comentarios.
- Mostrar advertencias si el contenido publicado no cumple con las políticas.

PUNTO 5 HU 3 (Andrés Hernando Borda Muñoz)
Gestión de pedidos

Anexo de documentos relacionados

2. Caso de uso: Gestión de pedidos.

Descripción conceptual

Módulo	Módulo de gestión de pedidos
Descripción de la(s) funcionalidad(es) requerida(s):	<p>El administrador podrá gestionar los pedidos realizados por los clientes para productos como tazas, llaveros personalizados, entre otros. Esto implica la realización de las siguientes operaciones:</p> <ul style="list-style-type: none">● Consulta de pedidos existentes.● Edición de la información de los pedidos.● Cancelación de pedidos.● Registro de nuevos pedidos.

Descripción técnica

Backend

- **Consulta de pedidos existentes**

URL	Método	Código html
/pedidos/{id}	GET	200 404
200 (Consulta exitosa). 404 (Pedido no encontrado)		
Datos de entrada id del pedido.	Datos de salida 200 (Consulta exitosa): { "status": "success", "data": { "id": 1, "cliente": "Ana Pérez", "productos": [

	<pre>{ "nombre": "Taza personalizada", "cantidad": 2, "precio": 15000 }, { "fecha_pedido": "2024-12-20", "estado": "Pendiente", "monto_total": 30000, "medio_pago": "Nequi" } }</pre> <p>404 (Pedido no encontrado):</p> <pre>{ "status": "error", "message": "Pedido no encontrado." }</pre>
--	--

● Edición de la información de los pedidos

URL	Método	Código html
/pedidos/{id}	PATCH	200 422
200 (Actualización exitosa). 422 (Datos inválidos).		
Datos de entrada <pre>{ "productos": [{ "nombre": "Taza personalizada", "cantidad": 3 }], "estado": "Pagado" }</pre>	Datos de salida 200 (Actualización exitosa): <pre>{ "status": "success", "data": { "id": 1, "cliente": "Ana Pérez", "productos": [{ "nombre": "Taza personalizada", "cantidad": 3, "precio": 15000 }], "estado": "Pagado", "monto_total": 45000 } }</pre>	

	<pre>} }</pre> <p>422 (Datos inválidos):</p> <pre>{ "status": "error", "message": "Los datos proporcionados no son válidos." }</pre>
--	---

- **Cancelación de pedidos.**

URL	Método	Código html
/pedidos/{id}/cancelar	PATCH	200 404
200 (Actualización exitosa). 404 (Pedido no encontrado).		
Datos de entrada	Datos de salida	
id del pedido.	<p>200 (Cancelación exitosa):</p> <pre>{ "status": "success", "message": "El pedido ha sido cancelado." }</pre> <p>404 (Pedido no encontrado):</p> <pre>{ "status": "error", "message": "Pedido no encontrado." }</pre>	

- **Registro de nuevos pedidos.**

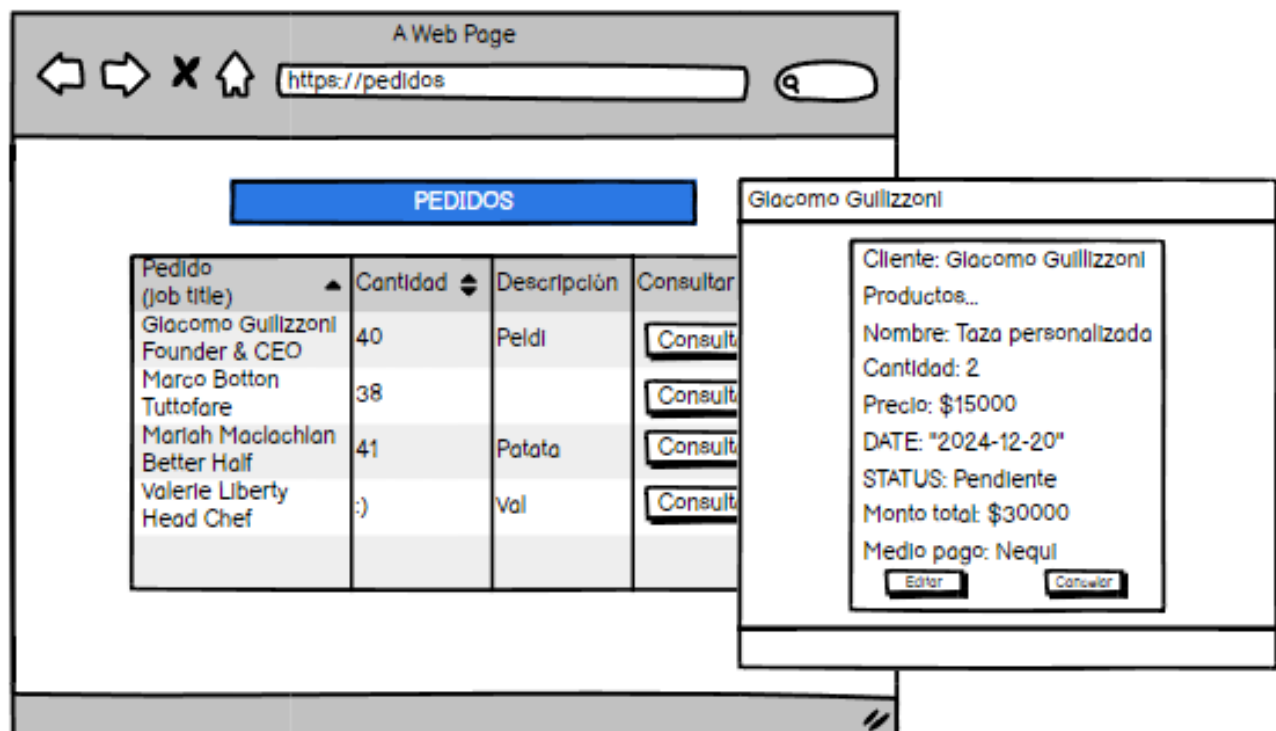
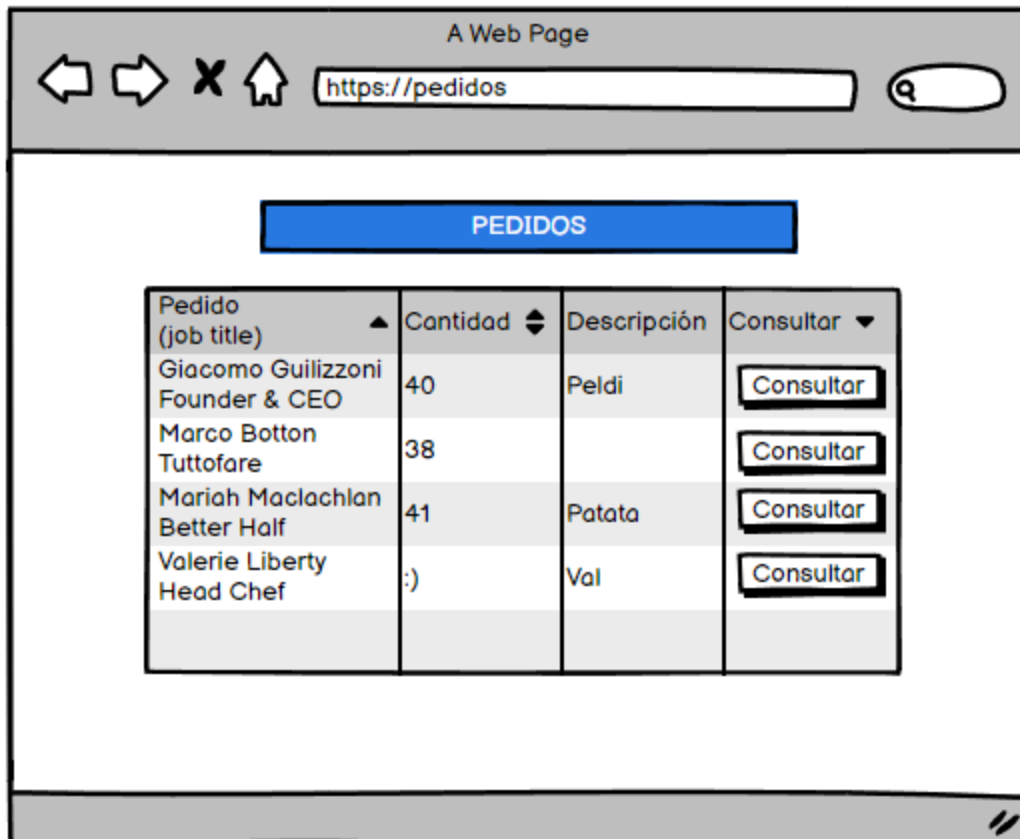
URL	Método	Código html
/pedidos	POST	201 422
201 (Creación exitosa). 422 (Datos faltantes o inválidos).		

Datos de entrada	Datos de salida
<pre>{ "cliente": { "nombre": "Ana Pérez", "contacto": "+573001234567", "direccion": "Calle 123 #45-67, Bogotá" }, "productos": [{ "id": 1, "nombre": "Taza personalizada", "cantidad": 2, "precio": 15000 }], "fecha_pedido": "2024-12-20", "monto_total": 30000, "estado": "Pendiente", "medio_pago": "Nequi" }</pre>	<p>201 (Creación exitosa):</p> <pre>{ "status": "success", "data": { "id": 2, "cliente": "Ana Pérez", "productos": [{ "nombre": "Taza personalizada", "cantidad": 2, "precio": 15000 }], "fecha_pedido": "2024-12-20", "estado": "Pendiente", "monto_total": 30000, "medio_pago": "Nequi" } }</pre> <p>422 (Datos faltantes o inválidos):</p> <pre>{ "status": "error", "message": "Faltan campos obligatorios o datos no válidos." }</pre>

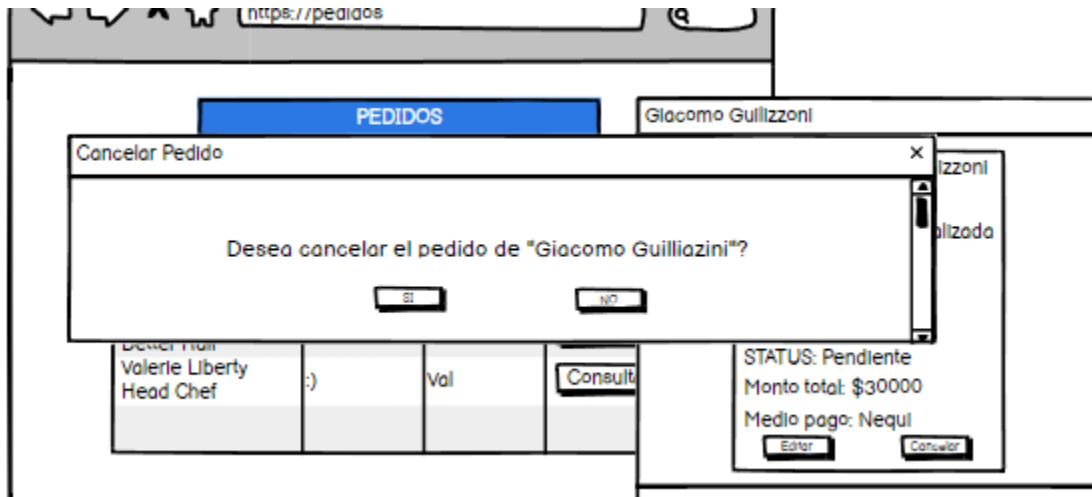
Frontend

Interacción esperada:

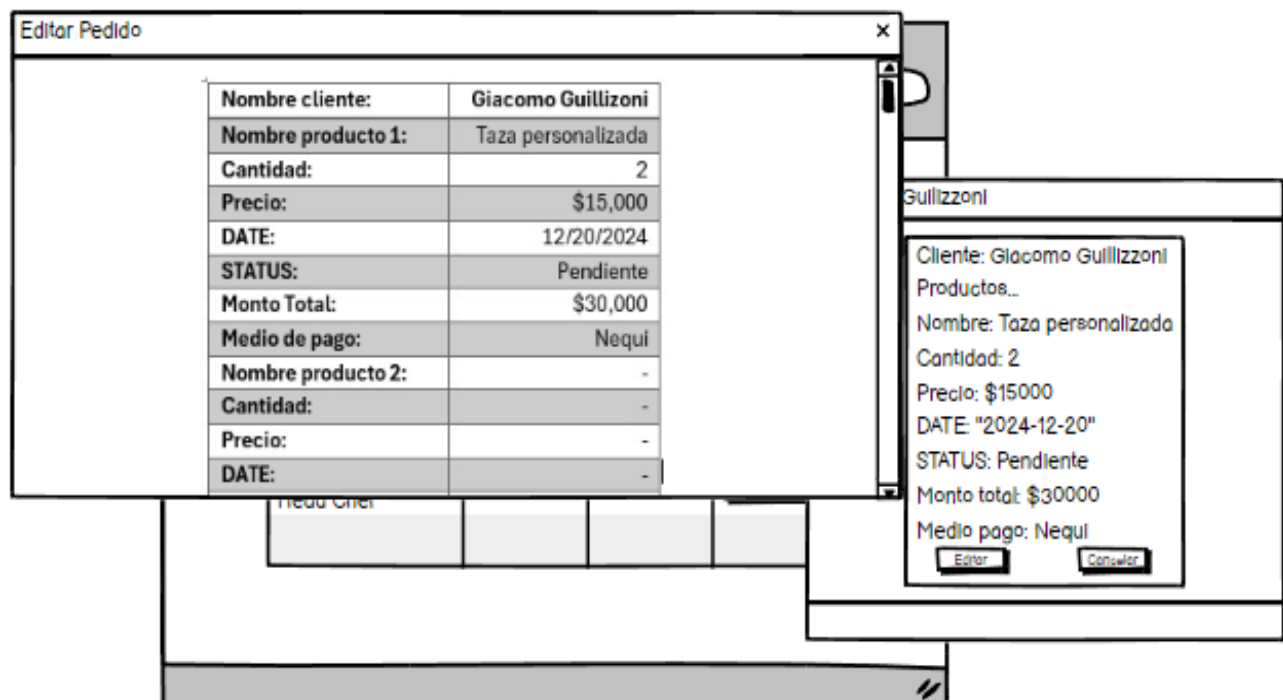
- **Consultar:** Al hacer clic en "Consultar", se despliega una ventana emergente con los detalles del pedido seleccionado.



- **Cancelar:** Al hacer clic en "Cancelar", se solicita confirmación antes de proceder con la acción.



- **Editar:** Al hacer clic en "Editar", se muestra un formulario prediligenciado con los datos del pedido.



- **Crear:** Al hacer clic en "Registrar Pedido", se despliega un formulario vacío para ingresar los datos del nuevo pedido.

The diagram illustrates a user interface for registering a new order. It features a main form titled "Registrar pedido nuevo" and a summary panel titled "Gullizzoni".

Registrar pedido nuevo Form:

Nombre cliente:	
Nombre producto 1:	-
Cantidad:	-
Precio:	-
DATE:	-
STATUS:	-
Monto Total:	-
Medio de pago:	-
Nombre producto 2:	-
Cantidad:	-
Precio:	-

Below the form is a navigation bar with a button labeled "Registrar pedido nuevo". An arrow points from this button to the form.

Gullizzoni Summary Panel:

Cliente: Giacomo Gullizzoni
Productos...
Nombre: Taza personalizada
Cantidad: 2
Precio: \$15000
DATE: "2024-12-20"
STATUS: Pendiente
Monto total: \$30000
Medio pago: Nequi

At the bottom of the summary panel are two buttons: "Editar" and "Cancelar".

PUNTO 5 HU 4 (Brayan Alejandro Muñoz Pérez)

Notificaciones automáticas

Descripción conceptual

Módulo	Notificaciones automáticas
Descripción de la(s) funcionalidad(es) requerida(s):	El sistema debe enviar notificaciones automáticas a la dueña y a los clientes cada vez que se registre un nuevo pedido. Estas notificaciones deben ser enviadas por WhatsApp y contener información relevante del pedido, como el estado inicial y un mensaje de confirmación.

Descripción técnica

Backend

1. Notificación a la dueña

URL /notifications/owner	Método POST	Código html 201
Descripción Envía un mensaje de confirmación a la dueña con los detalles del pedido.		
Datos de entrada { "orderId": "12345", "customerName": "Ana Pérez", "products": ["Llavero personalizado", "Pulsera"], "totalAmount": 45000, "status": "Pendiente" }	Datos de salida { "status": "success", "message": "Notificación enviada a la dueña." }	

2. Notificación al cliente

URL /notifications/customer	Método POST	Código html 201
Descripción Envía un mensaje de confirmación al cliente con los detalles de su pedido.		
Datos de entrada { "customerName": "Ana Pérez", }	Datos de salida { "status": "success", }	

<pre> "phoneNumber": "+573001234567", "orderSummary": { "orderId": "12345", "products": ["Llavero personalizado", "Pulsera"], "totalAmount": 45000 }, "status": "Pendiente" } </pre>	<pre> "message": "Notificación enviada al cliente." } </pre>
--	--

Frontend

Interacción esperada:

- La dueña no necesita realizar acciones directas para enviar las notificaciones, estas se generan automáticamente al registrar un pedido.
- El cliente recibe un mensaje por WhatsApp con un resumen de su pedido.

Flujo visual y eventos:

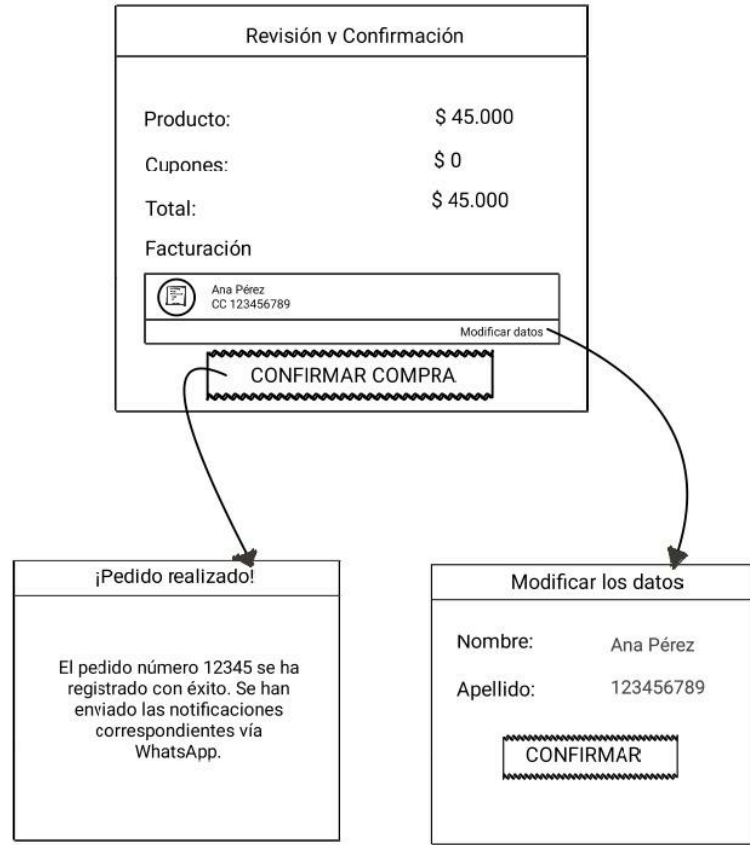
1. Registrar el pedido desde el formulario.
2. El backend procesa el registro y envía automáticamente las notificaciones por WhatsApp.
3. La dueña recibe confirmación visual en la interfaz y los mensajes llegan tanto a su número como al del cliente.

Notas adicionales:

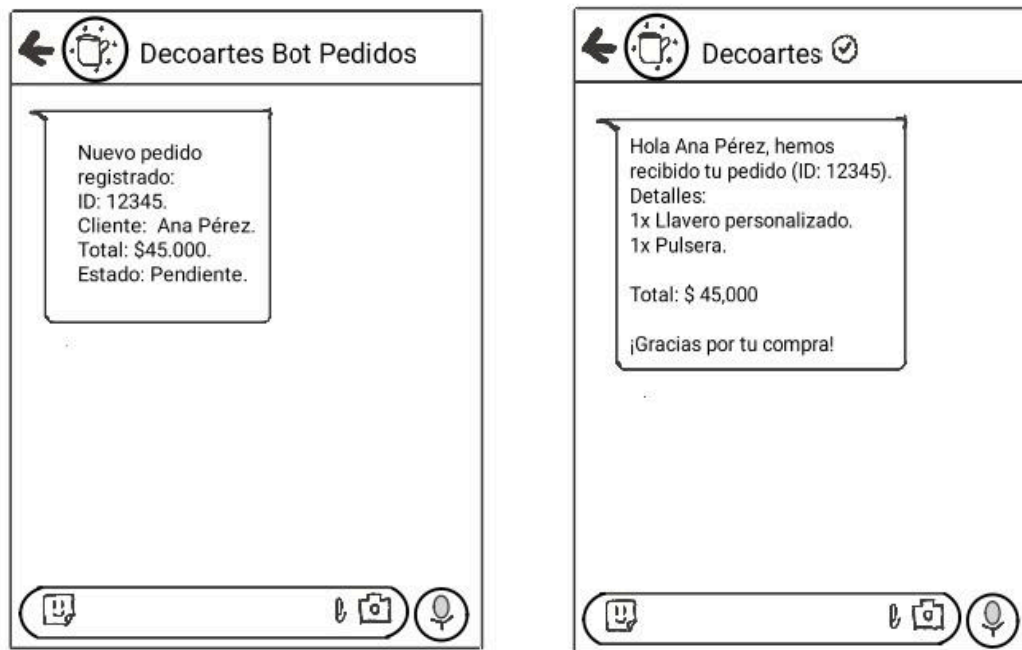
- Validar que los números de teléfono del cliente y la dueña tengan una cuenta asociada existente en WhatsApp
- Tener en cuenta el límite de número de mensajes permitidos por la API gratuita seleccionada.

Mockups/Prototipos:

1. Confirmación de la compra.



2. Mensajes de notificación vía WhatsApp a cliente y dueña.



Referencias

OpenAI. (2024). *ChatGPT* (v2). OpenAI. <https://openai.com>