

Taller 01 - Módulo de requerimientos

Presentado por:

Nicolas Quezada Mora - nquezada@unal.edu.co

Sharick Yelixa Torres Monroy - shtorres@unal.edu.co

Laura Sofia Vargas Rodriguez - lavargasro@unal.edu.co

Jeronimo Bermudez Hernandez- jebermudez@unal.edu.co

Profesor:

Oscar Eduardo Alvarez Rodriguez - oalvarezr@unal.edu.co

Viernes 20 de Diciembre



Universidad Nacional de Colombia

Facultad de Ingeniería

Departamento de Industrial y Sistemas

2024

1. Elegimos la conversación del GP2

2. Levantamiento de requerimientos:

1. Gestión de Pedidos

Organizar los pedidos de manera clara y accesible tanto para el cliente como para sus compradores.

- Datos a registrar:
 - Número de pedido único (ID).
 - Fecha del pedido.
 - Productos solicitados con cantidades.
 - Estado del pedido (Pendiente, Pagado, Enviado, Completado, Cancelado).
 - Fecha límite de entrega.
 - Monto total del pedido.
 - Medio de pago (transferencia, efectivo, etc.).
- Procesos asociados:
 - Crear pedidos a través de un formulario llenado por el cliente.
 - Permitir que los pedidos sean modificables (cambio de estado, cantidades, productos).
 - Notificar automáticamente al cliente cuando se complete una tarea asociada al pedido.
 - Posibilidad de generar un comprobante o resumen del pedido para el cliente final.

2. Control de Inventario

Supervisar la disponibilidad de productos en tiempo real.

- Datos a registrar:
 - Nombre del producto.
 - Precio.
 - Cantidad disponible en stock.
 - Descripción breve.
 - Fotos del producto (opcional).
- Procesos asociados:
 - Actualizar automáticamente el inventario cuando un pedido es completado.
 - Generar alertas cuando un producto esté cerca de agotarse (notificaciones por WhatsApp).

- Visualizar el estado del inventario en un formato sencillo y filtrable (por categorías, cantidad, etc.).
- Permitir al cliente agregar, modificar o eliminar productos desde un formulario.

3. Notificaciones

Informar al cliente y a los compradores sobre eventos importantes relacionados con los pedidos y el inventario.

- Formato y medios:
 - Cliente: Notificaciones por WhatsApp y correos electrónicos (resúmenes semanales).
 - Compradores: Mensajes automáticos de confirmación de pedidos vía WhatsApp, incluyendo un enlace al estado del pedido.
- Contenido de notificaciones:
 - Productos agotados o con baja disponibilidad.
 - Confirmación de pedidos nuevos para los clientes finales.
 - Cambios en el estado de los pedidos.

4. Catálogo en Línea

Facilitar la visualización de productos disponibles por parte de los compradores sin requerir envío manual de imágenes.

- Características principales:
 - Página web accesible desde cualquier dispositivo (móvil o computadora).
 - Mostrar productos con información relevante: nombre, precio, descripción y foto.
 - Vinculación directa con el inventario para ocultar automáticamente productos agotados.
 - Opción para generar un enlace compartible del catálogo.

5. Gestión de Clientes

Centralizar y administrar la información de los clientes.

- Datos a registrar:
 - Nombre completo.
 - Número de contacto.
 - Dirección de envío.
 - Historial de pedidos.
- Procesos asociados:

- Crear y editar registros de clientes mediante un formulario.
- Consultar el historial de compras asociado a cada cliente.

6. Finanzas (funcionalidad futura)

Llevar un registro básico de ingresos y gastos asociados al negocio.

- Datos a registrar:
 - Ingresos por ventas (registrados automáticamente por los pedidos).
 - Gastos relacionados (por ejemplo, compra de materiales o productos).
 - Balance financiero mensual.
- Procesos asociados:
 - Generar reportes simples de ganancias y pérdidas.
 - Categorizar ingresos y gastos por tipo.

7. Accesibilidad y Usabilidad

Requerimientos específicos:

- Acceso desde dispositivos móviles y computadoras mediante un navegador web.
- Interfaz intuitiva y sencilla, con formularios claros y botones bien definidos.
- Opcional: Funcionalidad offline básica para casos de desconexión temporal.

3. Análisis de requerimientos:

1. Must have (imprescindibles):

- Registro, edición y eliminación de productos.
- Gestión de usuarios con roles y permisos.
- Generación de reportes básicos de inventario.

2. Should have (importantes):

- Integración con un sistema de notificaciones para alertar sobre bajo inventario.
- Filtros avanzados para búsquedas de productos.

3. Could have (deseables):

- Exportar reportes en diferentes formatos (PDF, Excel).
- Modo oscuro en la interfaz de usuario.

4. Won't have (no ahora):

- Aplicación móvil complementaria.
- Integración con sistemas de facturación externos.

Estimado de tiempo y complejidad

Realizamos un análisis comparativo:

A nivel de desarrollo COMPLETO para un sistema con todos los módulos especificados:

Equipo de desarrollo:

- 2 desarrolladores junior (incluyéndome).
- 1 desarrollador senior (líder técnico).
- 1 diseñador UI/UX.
- 1 tester para pruebas.

Tiempo estimado:

- Fase de análisis y diseño: 3 semanas.
- Desarrollo (funcionalidades Must): 6 semanas.
- Desarrollo (funcionalidades Should y Could): 3 semanas adicionales.
- Pruebas y ajustes finales: 2 semanas.

Total: 14 semanas (3.5 meses).

Costo aproximado: Supongamos que cada desarrollador gana 20,000 COP/hora y trabaja 40 horas a la semana: Equipo total: 20,000 COP x 5 personas x 14 semanas x 40 horas. Aproximadamente 56,000,000 COP en desarrollo, diseño y pruebas.

Ajustado al contexto del taller, puesto que el cliente dice tener un presupuesto cercano a 1'000.000(COP) y el tiempo de desarrollo es más corto:

Equipo de desarrollo:

- 1 desarrollador junior (líder de proyecto).
- 1 desarrollador senior (horas específicas para revisión y guía técnica).

Tiempo estimado:

- Fase de análisis y diseño: 10 horas estimadas.
- Desarrollo principal (funcionalidades Must): 70 horas estimadas.
- Desarrollo (funcionalidades Should y Could): 15 horas estimadas.
- Pruebas y ajustes finales: 5 horas estimadas.

Total: 5 semanas.

Costo aproximado:

Supongamos que un desarrollador junior gana 20,000 COP/hora y trabaja 20 horas a la semana: $20,000 \text{ COP} \times 20 \text{ horas/semana} \times 5 \text{ semanas} = 2'000,000 \text{ COP}$.

Mientras que un desarrollador senior gana 35,000 COP/hora y trabaja 10 horas a lo largo del proyecto: $35,000 \text{ COP} \times 10 \text{ horas/proyecto} = 350,000$

Esto resulta en un monto total de 2'350,000 COP en desarrollo, diseño y pruebas. El cual es un presupuesto más acorde a la naturaleza del proyecto.

Tecnologías propuestas:

- Backend: Java con Spring Boot (por robustez y escalabilidad).
- Frontend: React.js (para una interfaz dinámica).
- Base de datos: MySQL (relacional, confiable).

Impacto en la triada de desarrollo

1. **Tiempo:**

Mantener el desarrollo dentro del cronograma propuesto dependerá de cumplir las etapas de análisis y diseño sin atrasos.

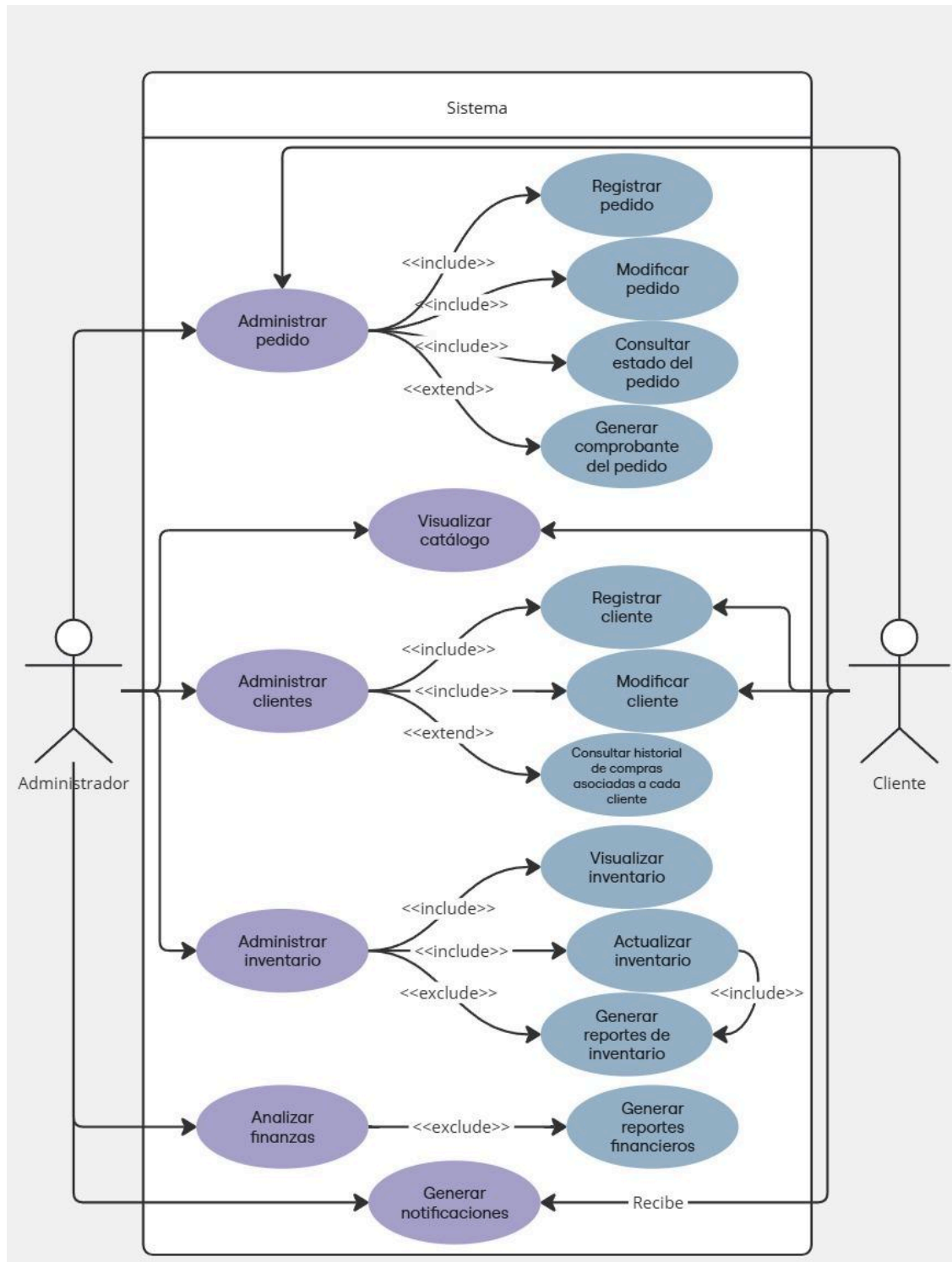
2. **Costo:**

Si se encuentran problemas inesperados, el costo podría aumentar si necesitamos más tiempo o personal adicional.

3. **Alcance:**

Priorizar las funcionalidades con MoSCoW asegura que el sistema funcione incluso si el tiempo o los recursos se limitan, ya que lo esencial se desarrolla primero.

4. Diagrama de Casos de Uso:



Especificación de un Caso de Uso:

Registrar pedido
Actor Cliente Administrador
Breve descripción Se realiza la obtención de datos del pedido para posteriormente verificar la información y guardar el registro en la base de datos.
Descripción paso por paso <ol style="list-style-type: none"> 1. El usuario proporciona los datos del pedido <ol style="list-style-type: none"> a. Fecha del pedido. b. Productos solicitados con sus respectivas cantidades. c. Estado del pedido (Pendiente, Pagado, Enviado, Completado, Cancelado). d. Fecha límite de entrega. e. Medio de pago (transferencia, efectivo, etc.). 2. El sistema genera el ID del pedido <ol style="list-style-type: none"> a. Número de pedido único (ID). 3. El sistema calcula el monto total del pedido <ol style="list-style-type: none"> a. Valor numérico válido según los productos y las cantidades. 4. Se realiza la verificación de la fecha del pedido <ol style="list-style-type: none"> a. La fecha debe estar en formato (AAAA/MM/DD). 5. Se realiza la verificación de disponibilidad de productos <ol style="list-style-type: none"> a. Si algún producto no está disponible en el inventario, el sistema notificará al usuario y se solicitará seleccionar otro producto o sacar a este del registro (Volver al paso 1). 6. Se realiza la verificación del estado del pedido <ol style="list-style-type: none"> a. La opción elegida debe encontrarse entre las opciones aceptadas (Pendiente, Pagado, Enviado, Completado, Cancelado). <ol style="list-style-type: none"> i. Si la opción no es válida, el sistema indicará que el estado del pedido debe ser una de las categorías determinadas previamente (Volver al paso 1). 7. Se realiza la verificación de la fecha límite de entrega del pedido <ol style="list-style-type: none"> a. La fecha debe estar en formato (AAAA/MM/DD). b. El sistema verifica que la fecha límite de entrega es posterior a la fecha del pedido.

- i. Si la fecha límite es superior a la fecha del pedido el sistema indicará la incoherencia y solicitará cambiar el dato (Volver al paso 1).
- 8. Se realiza la verificación del medio de pago
 - a. La opción elegida debe encontrarse entre las opciones aceptadas (transferencia, efectivo).
 - i. Si la opción elegida no es válida, el sistema solicitará corregir el dato (Volver al paso 1).
- 9. Se registra la información del pedido en la base de datos.

5. Historias de usuario

Historia de Usuario #1: Gestor de Pedidos- Nicolas Quezada Mora

Descripción conceptual

Módulo: Gestor de Pedidos

Descripción de la(s) funcionalidad(es) requerida(s): El sistema debe permitir registrar, gestionar y actualizar pedidos de manera clara y organizada. Los pedidos deben incluir detalles como el nombre del cliente, productos solicitados, cantidades, estado del pedido, fecha de entrega, y monto total. Además, debe enviar notificaciones al cliente para confirmar la recepción y actualizaciones del pedido.

El objetivo es facilitar tanto la administración interna como la experiencia del cliente, asegurando que la información sea precisa y accesible.

Descripción técnica

Backend

- **URL:** localhost:8080/pedidos
- **Método:** POST
- **Código HTTP esperado:** 201 (Creado)

Caso de uso técnico: El endpoint debe permitir registrar un nuevo pedido en la base de datos, generando un ID único de manera automática. Además, se deben validar los datos de entrada antes de guardar el pedido.

Datos de entrada:

```
{  
  
  "cliente": "Juan Pérez",  
  
  "productos": [  
  
    {"nombre": "Pulsera", "cantidad": 2, "precio_unitario": 15000},  
  
    {"nombre": "Maceta pintada", "cantidad": 1, "precio_unitario": 25000}  
  
  ],  
  
  "estado": "Pendiente",
```

```
"fecha_pedido": "2024-12-20",  
"fecha_entrega": "2024-12-25",  
"monto_total": 55000,  
"medio_pago": "Transferencia"  
}
```

Datos de salida (201):

```
{  
  "status": "success",  
  "data": {  
    "id_pedido": 12345,  
    "cliente": "Juan Pérez",  
    "productos": [  
      {"nombre": "Pulsera", "cantidad": 2, "precio_unitario": 15000},  
      {"nombre": "Maceta pintada", "cantidad": 1, "precio_unitario": 25000}  
    ],  
    "estado": "Pendiente",  
    "fecha_pedido": "2024-12-20",  
    "fecha_entrega": "2024-12-25",  
    "monto_total": 55000,  
    "medio_pago": "Transferencia"  
  }  
}
```

Frontend

Interacción esperada: El usuario (administrador) deberá interactuar con un formulario accesible desde una página web. Este formulario contendrá los campos necesarios para registrar un pedido: nombre del cliente, productos, cantidades, estado del pedido, fecha de entrega y medio de pago. Al guardar el pedido, el sistema debe mostrar una confirmación visual.

Mockups/Prototipos:

1. Pantalla principal del gestor de pedidos:

- Tabla con los pedidos existentes, mostrando ID, cliente, estado y fecha de entrega.
- Botón flotante “Nuevo Pedido”.

2. Formulario para registrar un pedido:

- Campos:
 - Nombre del cliente (texto).
 - Productos (lista desplegable + cantidad).
 - Estado del pedido (desplegable con opciones: Pendiente, Pagado, Enviado, Completado, Cancelado).
 - Fecha de entrega (selector de fecha).
 - Medio de pago (desplegable).
- Botones:
 - “Guardar” (color principal: verde oliva).
 - “Cancelar” (color secundario: gris claro).

En el estante

Gestión de pedidos

Pedidos Existentes

Identificación del pedido	Nombre del cliente	Fecha de pedido	Estado	Total
#001	Ana López	10/10/2023	colgante	\$150.00
#002	Carlos García	11/10/2023	Pagado	\$200.00
#003	Maria Torres	12/10/2023	Enviado	\$175.00

Agregar pedido

Registrar Nuevo Pedido

Nombre

Escribe el nombre del cliente

Productos

Manzanas

138

Estado del pedido

colgante

Fecha deEntrega

YYYY-MM-DD

Medio de pago

Tarjeta de crédito

Guardar

Cancelar

Flujo visual y eventos:

1. El administrador hace clic en “Nuevo Pedido”.
2. Se despliega el formulario para registrar un nuevo pedido.

3. Al completar los campos y dar clic en “Guardar”:
 - Se muestra un spinner mientras el pedido se registra.
 - Al completarse el proceso, aparece un mensaje: “Pedido registrado exitosamente”.
 - El pedido se agrega automáticamente a la tabla principal.
4. Si ocurre un error en la validación (por ejemplo, campos vacíos), se muestra un mensaje de alerta indicando los campos que deben corregirse.

Descripción conceptual

Módulo: Gestión de usuarios.- Jeronimo Bermudez Hernandez

Descripción de las funcionalidades requeridas: Una interfaz que permita al usuario ver la lista de clientes y hacer cambios sobre ella, así como registrar un nuevo cliente, editarlo, ver sus datos y su pedidos o eliminarlo.

Descripción técnica

Consulta de usuarios

- **URL:** localhost:8080/users
- **Método:** GET
- **Código HTTP esperado:** 200 (Creado)

Caso de uso técnico: Al consultar, debe retornar un código 200 y en data un array con los datos de los usuarios registrados.

Datos de entrada:

No requiere datos de entrada.

Datos de salida (200):

```
{  
  "status": "success",  
  "data": [  
    {  
      "id": 1,  
      "nombre": "Juan Pérez",  
      "url": "imagen.html"  
    }  
  ]  
}
```

Edición de usuario

- **URL:** localhost:8080/users/{id}
- **Método:** PATCH
- **Código HTTP:** 200: El usuario ha sido actualizado correctamente.
- 422: La solicitud no incluye parámetros requeridos.

Caso de uso técnico: Al consultar, el sistema debe verificar que el id dado exista en la base de datos y validar que la solicitud contenga los parámetros requeridos, además, debe actualizar la información y retornar un código según el estado de la operación.

Datos de entrada (200):

```
{  
  
  "data": {  
  
    "nombre": "Juan Pérez 2"  
  
  }  
}
```

Datos de salida (200):

```
{  
  
  "status": "success",  
  
  "data": {  
  
    "id": 1,  
  
    "nombre": "Juan Pérez 2",  
  
    "url": "imagen.html"  
  
  }  
}
```

Datos de entrada (422):

```
{  
  "data": {}  
}
```

Datos de salida (422):

```
{  
  "status": "unprocessed",  
  "message": "No existe el parámetro <nombre>"  
}
```

Eliminación de usuario

- **URL:** localhost:8080/users/{id}
- **Método:** DELETE
- **Código HTTP:** 200: El usuario ha sido eliminado correctamente.
- 404: El usuario no existe.

Caso de uso técnico: Al consultar, el sistema debe verificar que el id dado exista en la base de datos, si existe, debe eliminar el registro y retornar un código según el estado de la operación.

Datos de entrada :

No requiere datos adicionales más allá del parámetro {id} en la URL.

Datos de salida (200):

```
{  
  "status": "success",  
  "message": "Usuario eliminado correctamente"  
}
```

Datos de salida (404):

```
{  
  "status": "error",
```

```
"message": "El usuario no fue encontrado"
}
```

Eliminación de usuario

- **URL:** localhost:8080/users/{id}/orders
- **Método:** DELETE
- **Código HTTP:** 200: El usuario ha sido eliminado correctamente.
- 404: El usuario no existe.

Caso de uso técnico: Al consultar, el sistema debe verificar que el id dado exista en la base de datos, si existe, debe buscar el registro de pedidos asociados al usuario, retornando un código 200 junto a un array con los datos solicitados.

Datos de entrada :

No requiere datos adicionales más allá del parámetro {id} en la URL.

Datos de salida (200):

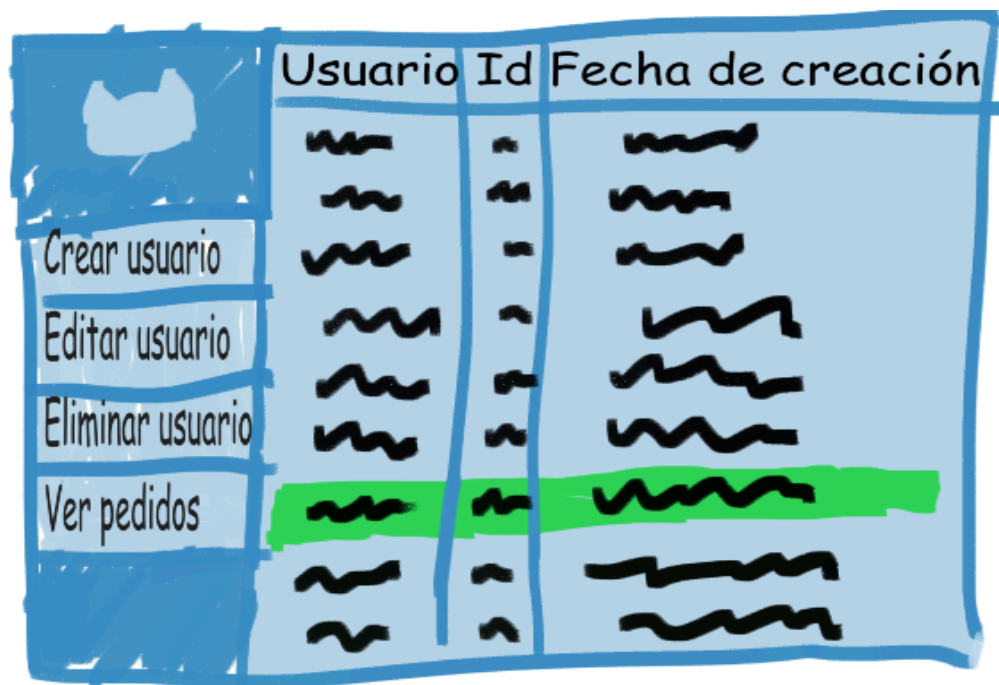
```
{
  "status": "success",
  "data": [
    {
      "orderId": 101,
      "producto": "Laptop",
      "cantidad": 1,
      "precio": 1200.50,
      "estado": "Enviado"
    },
    {
      "orderId": 102,
```

```
"producto": "Teclado",  
  
"cantidad": 2,  
  
"precio": 80.00,  
  
"estado": "Pendiente"  
  
}  
  
]  
  
}
```

Frontend

Interacción esperada: El usuario (administrador) debe ver un menú desplegable en donde se encuentran las opciones para la gestión de los usuarios, junto con un formulario para proporcionar la información de entrada para cada caso.

Mockups:



	Usuario	Id	Fecha de creación
	~~~~~	~	~~~~~
	~~~~~	~	~~~~~
Crear usuario	~~~~~	~	~~~~~
Editar usuario	~~~~~	~	~~~~~
Eliminar usuario	~~~~~	~	~~~~~
Ver pedidos	~~~~~	~	~~~~~
	~~~~~	~	~~~~~
	~~~~~	~	~~~~~

Flujo visual:

1. El usuario selecciona la opción de “Crear usuario”
2. Se abre un formulario donde el usuario pone los datos

3. Los datos quedan registrados en la base de datos
4. El usuario desea actualizar la información del cliente, entonces lo selecciona y selecciona la opción “Editar usuario”
5. Se abre una ventana donde se permite editar los datos asociados.
6. Los datos se actualizan.
7. El usuario desea saber más acerca de los pedidos asociados a un id, por lo que lo selecciona y selecciona la opción de “Ver pedidos”
8. Aparece una ventana con una lista de los pedidos asociados al usuario seleccionado.
9. El administrador desea eliminar un usuario, por lo que lo selecciona y selecciona la opción de “Eliminar usuario”.
10. Se abre una ventana de confirmación
11. Si se confirma, da una alerta indicando que la eliminación fue correcta.

Historia de Usuario #3: módulo de Gestión de productos- Laura Sofia Vargas Rodriguez

El sistema debe permitir a los administradores gestionar los productos en la base de datos mediante la creación, edición, consulta y eliminación de estos, además, se debe garantizar que la información se valide antes de almacenarse o actualizarse en el sistema, junto con herramientas específicas para la administración del inventario:

- Actualización automática del inventario cuando se registren ventas o pedidos.
- Generador de alertas automáticas para stock bajo, enviadas al usuario por WhatsApp.
- Visualización clara de productos con indicadores de estado: stock adecuado, stock bajo y agotado.
- Registro mediante escaneo de código de barras o QR para agilizar la entrada de productos.
- Generador de reportes automáticos de inventario y ventas enviados por email
- Predicción básica de demanda para anticipar agotamientos según tendencias
- Panel de control interactivo con estadísticas clave.

1. Consulta de productos

URL:localhost:8080/products

Método:GET

Código HTML:200

Datos de entrada:

(Sin parámetros)

Datos de salida:

```
{  
  
  "status": "success",  
  
  "data": [  
  
    {  
  
      "id": 101,  
  
      "nombre": "Producto A",  
  
      "precio": 19.99,
```

```
    "stock": 97,  
    "categoria": "Electrónica",  
    "estado": "Stock Adecuado"  
  }  
]  
}
```

2. Creación de producto

URL:localhost:8080/products

Método:POST

Código HTML:201: Producto creado exitosamente.

422: Error en los parámetros enviados.

Datos de entrada (201):

```
{  
  "nombre": "Producto B",  
  "precio": 29.99,  
  "stock": 50,  
  "categoria": "Hogar",  
  "imagen": "producto_b.jpg"  
}
```

Datos de salida (201):

```
{  
  "status": "success",  
  "data": {
```

```
"id": 102,  
  
"nombre": "Producto B",  
  
"precio": 29.99,  
  
"stock": 50,  
  
"categoria": "Hogar",  
  
"estado": "Stock Adecuado"  
  
}  
  
}
```

3. Actualización de producto

URL:localhost:8080/products/{id}

Método:PATCH

Código HTML:200: Actualización exitosa.

422: Error en los parámetros enviados.

Datos de entrada (200):

```
{  
  
  "stock": 30  
  
}
```

Datos de salida (200):

```
{  
  
  "status": "success",  
  
  "data": {  
  
    "id": 101,  
  
    "nombre": "Producto A",
```

```
    "precio": 19.99,  
    "stock": 30,  
    "categoria": "Electrónica",  
    "estado": "Stock Bajo"  
  }  
}
```

4. Eliminación de producto

URL:localhost:8080/products/{id}

Método:DELETE

Código HTML:200

Datos de salida:

```
{  
  "status": "success",  
  "message": "Producto eliminado correctamente"  
}
```

Frontend

Interacción esperada: Pantalla principal del módulo de productos:

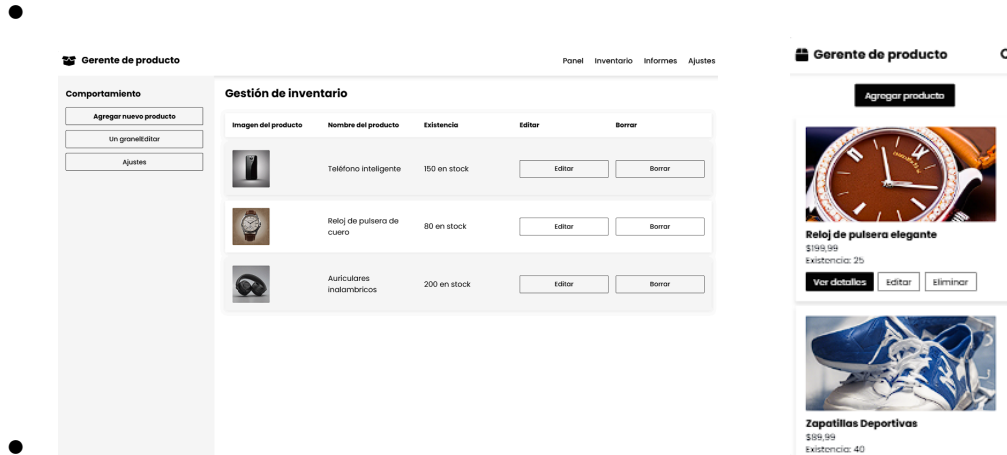
- Una tabla muestra los productos con columnas para el nombre, precio, stock, categoría, miniatura de la imagen, y estado del inventario.

Acciones disponibles:

- Crear producto: Un botón que abre un formulario en un modal para registrar un nuevo producto.
- Editar producto: Un ícono de lápiz que abre un modal para actualizar datos específicos del producto.

- Eliminar producto: Un ícono de papelera que solicita confirmación antes de eliminar el producto seleccionado.

Mockups/Prototipos:



Flujo visual y eventos:

- Al cargar el módulo, se muestra un spinner mientras se consultan los productos
- Al hacer clic en el botón "Crear Producto", se abre un modal con campos para nombre, precio, stock, categoría e imagen.
- Al guardar cambios (crear o editar), se muestra un mensaje de éxito en la parte superior y se actualiza la tabla dinámicamente.
- Al eliminar un producto, se abre un modal de confirmación; si el administrador confirma, el producto se elimina sin recargar la página.
- El sistema envía alertas automáticas por WhatsApp a la usuaria cuando un producto alcanza un nivel de stock bajo o se agota.
- Indicadores visuales de estado de inventario en la tabla: verde (adecuado), amarillo (bajo), rojo (agotado).
- Panel de control interactivo con estadísticas de productos más vendidos, niveles de stock y alertas pendientes.

Historia de Usuario #4 – Gestor de finanzas

Anexo de Documentos Relacionados:

- Módulo de finanzas: Finanzas (funcionalidad futura)
 - Llevar un registro básico de ingresos y gastos asociados al negocio.
 - Datos a registrar:
 - Ingresos por ventas (registrados automáticamente por los pedidos).
 - Gastos relacionados (por ejemplo, compra de materiales o productos).
 - Balance financiero mensual.
 - Procesos asociados:
 - **Generar reportes simples de ganancias y pérdidas. ← funcionalidad elegida**
 - Categorizar ingresos y gastos por tipo.

Descripción conceptual

Módulo	<i>Ejemplo: Módulo de Gestión de Finanzas</i>
Descripción de la(s) funcionalidad(es) requerida(s):	<i>El sistema debe permitir generar reportes simples de ganancias y pérdidas en base a los ingresos y gastos registrados. Estos reportes deben incluir una categorización de ingresos y gastos por tipo y mostrar un balance financiero mensual.</i>

Descripción técnica

Backend

URL	Método	Código html
localhost:8080/reportes/finanzas	POST	- 200 (Solicitud procesada correctamente) - 422 (Datos de entrada incorrectos)

Descripción	
Al consultar, el sistema debe verificar que los parámetros estén completos y existan, retorna código 200 y un data array con los datos del reporte financiero.	
Datos de entrada 200: <pre>{ "fechaInicio": "2024-01-01", "fechaFin": "2024-01-31", "categorias": ["Ventas", "Materiales"] }</pre>	Datos de salida 200: <pre>{ "status": "success", "data": { "ingresos": 500000, "gastos": 200000, "balance": 300000, "detalle": [{ "categoria": "Ventas", "monto": 500000 }, { "categoria": "Materiales", "monto": 200000 }] } }</pre>
422: <pre>{ "fechaFin": "2024-01-31", "categorias": ["Ventas", "Materiales"] }</pre>	422: <pre>{ "status": "unprocessed", "message": "Faltan parámetros o no existen" }</pre>

Frontend

1. El usuario accede al módulo de Finanzas desde el menú principal.
2. Se muestra un formulario para ingresar el rango de fechas y seleccionar las categorías que desea incluir en el reporte.
3. Al hacer clic en el botón "Generar reporte", se muestra un spinner mientras se procesa la solicitud.
4. Una vez generado, el sistema despliega el reporte de ganancias y pérdidas en un formato sencillo, con tablas, gráficas y recursos visuales.

Mockups/Prototipos

Teniendo en cuenta las recomendaciones del cliente:

- Colores: beige, terracota, salmón, blanco, verde oliva
- Botones llamativos
- Iconos pequeños

1. Pantalla inicial con formulario de entrada

The mockup shows a web application interface for 'ReportGenius'. At the top is a dark red header with the logo on the left and a 'Home' link on the right. The main content area is white and contains a centered form titled 'Generar reporte'. The form has two date input fields labeled 'Fecha de inicio' and 'Fecha de finalización', both with placeholder text 'YYYY-MM-DD'. Below these is a 'Categories' section with two radio buttons: 'Ventas' (selected) and 'Materiales'. At the bottom of the form is a large red button labeled 'Generar reporte' with a small icon. The footer is a dark red bar containing copyright information, contact details, and links to 'Privacy Policy' and 'Terms of Service'.

ReportGenius Home

Generar reporte

Fecha de inicio
YYYY-MM-DD

Fecha de finalización
YYYY-MM-DD

Categories
☒ Ventas ☐ Materiales

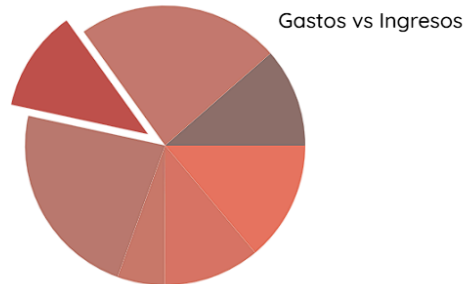
Generar reporte

© 2023 ReportGenius. All rights reserved.
Contact: support@reportgenius.com

Privacy Policy Terms of Service

2. Resultado del reporte

Reporte financiero



Detalles de ingresos y gastos

Categoría	Monto
Ventas	\$500,000
Materiales	\$200,000
-----	\$-----
-----	\$-----

Flujo visual y eventos

- Formulario de entrada:
 - Campos requeridos: Rango de fechas, Categorías.
 - Validaciones: Campos vacíos o fechas inconsistentes muestran mensajes de error.
- Botón "Generar reporte":
 - Mostrar spinner al hacer clic.
 - Ocultar spinner al recibir respuesta.
- Pantalla de resultados:
 - Gráfico interactivo (hover para ver detalles).
 - Opción de botón para exportar el reporte en PDF o Excel.