

## Product backlog

### **(Prioridad alta)**

- **PBI-001:** El enfoque del proyecto debe ser sobre la venta de productos online con tecnologías web.
- **PBI-002:** Como cliente, deseo poder registrarme a la página web con información básica como correo, contraseña y nombre para posteriormente poder iniciar sesión.
- **PBI-003:** Como cliente, deseo poder entrar a la página web con mi correo y contraseña previamente registrados en la web para navegar libremente por la web.
- **PBI-004:** Como cliente, deseo poder añadir diferentes productos a un carrito de compras que después podré revisar en un resumen.
- **PBI-005:** Como comprador, quiero poder pagar mi carrito de compras usando una tarjeta de crédito/débito.

### **(Prioridad media)**

- **PBI-006:** Como cliente deseo obtener una confirmación vía correo electrónico de la compra una vez que haga la termine y revalide para saber que mi compra fue procesada exitosamente.
- **PBI-007:** Como cliente deseo tener acceso al historial de pedidos hechos por mí para consultar detalles y tener acceso a otras opciones.
- **PBI-008:** Como cliente deseo tener recomendaciones basadas en mis compras hechas para poder interesarme en nuevos productos.
- **PBI-009:** Como cliente deseo poder cancelar pedidos realizados en cualquier momento.

### **(Prioridad baja)**

- **PBI-010:** Podrá visualizarse de forma remota y acceder desde cualquier dispositivo y momento sin restricción.
- **PBI-011:** La interfaz debe responder y mostrarse correctamente al tamaño.

## Sprint backlog 1

Página E-commerce	
<b>No. Sprint: 1</b>	<b>Programador: Armando</b>
Como cliente deseo tener acceso a una página web de productos en línea para poder revisados y posteriormente comprados.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"><li>✓ Que tenga una variedad de productos online</li><li>✓ Se pueda obtener una descripción y precio de manera previa</li></ul>	

Registro de usuario	
<b>No. Sprint: 1</b>	<b>Programador: Armando</b>
Como cliente deseo poder registrarme a la página web con información básica como correo, contraseña y nombre para posteriormente poder iniciar sesión.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"><li>✓ Que pida al menos correo, contraseña y nombre</li><li>✓ Los campos de texto sean validados antes de enviarse</li><li>✓ Se envíe un correo de confirmación</li></ul>	

Inicio de sesión	
<b>No. Sprint: 1</b>	<b>Programador: Armando</b>
Como cliente deseo poder entrar a la página web con mi correo y contraseña previamente registrados en la web para navegar libremente por la web.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"><li>✓ Que se inicie sesión con datos ya registrados</li><li>✓ Los campos de texto sean validados antes de enviarse</li></ul>	

Carrito de compras	
<b>No. Sprint: 1</b>	<b>Programador: Armando</b>
Como cliente deseo poder añadir diferentes productos a un carrito de compras que después podré revisar en un resumen para pasar a la etapa de compra.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"><li>✓ Que cualquier producto enlistado en la web pueda añadirse</li><li>✓ Se presente un resumen previo a la compra</li></ul>	

Compras y métodos de pago	
<b>No. Sprint:</b> 1	<b>Programador:</b> Armando
Como cliente deseo que después de haber un hecho carrito proceder al pago donde me pedirá que escoja un método de pago, datos de entrega y pueda confirmar la compra para asegurarme que mi compra es válida.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"> <li>✓ El método de pago debe ser uno previamente registrado en el perfil o registrado en ese momento</li> <li>✓ La dirección debe ser real y válida</li> <li>✓ El método de pago debe ser de alguna financiera existente</li> </ul>	

**Sprint goal:** "Permitir que los usuarios se registren, inicien sesión y añadan productos a un carrito de compras desde una página principal para proceder a hacer la compra con un método de pago y datos de envío válidos"

#### Elementos seleccionados del Product Backlog:

- **PBI-002:** Registro de usuario.
- **PBI-003:** Inicio de sesión de usuario.
- **PBI-004:** Añadir artículos al artículo de compras.
- **PBI-005:** Compras y métodos de pago.

#### Plan de tareas técnicas:

##### Tareas para (PBI-000) - Página principal

- **T-01 [Frontend/HTML]:** Estructurar el index.html básico (header, footer, y una sección de cuadrícula para productos).
- **T-02 [Frontend/CSS]:** Crear la hoja de estilos (style.css) con un enfoque *Mobile First* para la cuadrícula de productos, asegurando que sea responsiva.
- **T-03 [Backend/API]:** Crear un endpoint de API GET /api/products que devuelva un listado de productos.
- **T-04 [Database/Cloud]:** Definir y crear la tabla "productos" en la base de datos de la nube.
- **T-05 [Frontend/JS]:** Escribir la lógica en app.js (fetch) para consumir la API GET /api/products y mostrar dinámicamente los productos en index.html.

### **Tareas para PBI-002: Registro de usuario**

- **T-06 [Frontend/HTML]:** Crear el formulario de registro.
- **T-07 [Frontend/CSS]:** Aplicar estilos responsivos al formulario de registro, asegurando que sea fácil de usar en móviles.
- **T-08 [Frontend/JS]:** Implementar validación de campos en el cliente (correo válido, contraseñas coinciden, contraseña segura).
- **T-09 [Backend/API]:** Crear el endpoint POST /api/register que reciba los datos del usuario.
- **T-10 [Backend/Seguridad]:** Implementar el hashéo de contraseñas (ej. bcrypt) antes de guardarlas en la BD.
- **T-11 [Database/Cloud]:** Definir y crear la colección/tabla "usuariosDatos" en la base de datos.
- **T-12 [Frontend/JS]:** Programar el fetch para enviar los datos de registro al API y manejar la respuesta (éxito o error).

### **Tareas para PBI-003: Inicio de sesión de usuario**

- **T-13 [Frontend/HTML]:** Crear el formulario de inicio de sesión.
- **T-14 [Backend/API]:** Crear el endpoint POST /api/login.
- **T-15 [Backend/Seguridad]:** Implementar la lógica de autenticación: buscar usuario, comparar contraseña hasheada (bcrypt.compare).
- **T-16 [Backend/Seguridad]:** Generar y devolver un JSON Web Token (JWT) si la autenticación es exitosa.
- **T-17 [Frontend/JS]:** Programar el fetch para enviar los datos de login y, si es exitoso, guardar el JWT en localStorage.
- **T-18 [Frontend/JS]:** Actualizar la UI dinámicamente para mostrar el estado "autenticado".

### **Tareas para PBI-004: Añadir artículos al carrito**

- **T-19 [Frontend/HTML]:** Añadir un ícono de "carrito" en el header principal.
- **T-20 [Frontend/HTML/CSS]:** Crear la vista del carrito.
- **T-21 [Frontend/JS]:** Programar la lógica del botón "Añadir al carrito" en cada producto.

- **T-22 [Backend/API]:** Crear endpoints para el carrito: POST /api/cart (para añadir) y GET /api/cart (para ver).
- **T-23 [Database/Cloud]:** Definir la estructura para guardar carritos.
- **T-24 [Frontend/JS]:** Escribir la lógica para que la vista del carrito (cart.html) consuma el GET /api/cart y se actualice dinámicamente.

### **Tareas para PBI-005: Compras y métodos de pago**

- **T-25 [Frontend/HTML]:** Crear la página de checkout (checkout.html) con dos secciones: "datos de envío" y "método de pago".
- **T-26 [Frontend/CSS]:** Aplicar estilos responsivos al formulario de checkout para que sea fácil de completar en móviles.
- **T-27 [Frontend/JS]:** Implementar la validación de todos los campos de envío (dirección, ciudad, código postal, etc.).
- **T-28 [Frontend/JS-Backend]:** Integrar una pasarela de pago. *Ya que es una simulación: crear un formulario de tarjeta de crédito falso.*
- **T-29 [Backend/API]:** Crear el endpoint POST /api/checkout que reciba los datos de envío y el token de pago.
- **T-30 [Backend/API]:** Lógica del lado del servidor para procesar el pago y, si es exitoso, guardar la orden.
- **T-31 [Database/Cloud]:** Definir y crear las tablas "pedidos" y "detallespedido".
- **T-32 [Backend/API]:** Al confirmar la orden, la lógica debe vaciar el carrito del usuario.
- **T-33 [Frontend/HTML]:** Crear una página simple de "compra exitosa" (success.html) a la cual redirigir al usuario.

### **Tareas globales (no funcionales)**

- **T-34 [Infra/Cloud]:** Configurar el servicio de hosting para los archivos estáticos (HTML/CSS/JS).
- **T-35 [Infra/Cloud]:** Configurar el entorno del API y la base de datos en la nube.
- **T-36 [Pruebas/Responsive]:** Realizar pruebas de responsividad en todas las vistas (index, cart, checkout, modales) en múltiples tamaños de pantalla (móvil, tablet, desktop) usando DevTools.

- **T-37 [CSS/Global]:** Asegurar que la tipografía, los botones y los campos de formulario sean legibles y fáciles de usar en todos los tamaños de pantalla.

## Sprint backlog 2

Confirmación de compras	
<b>No. Sprint:</b> 2	<b>Programador:</b> Armando
Como cliente deseo obtener una confirmación vía correo electrónico de la compra una vez que haga la termine y revalide para saber que mi compra fue procesada exitosamente.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"> <li>✓ Se envíe un correo electrónico a la dirección del usuario con detalles y confirmación de compra</li> </ul>	

Historial de pedidos	
<b>No. Sprint:</b> 2	<b>Programador:</b> Armando
Como cliente deseo tener acceso al historial de pedidos hechos por mí para consultar detalles y tener acceso a otras opciones.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"> <li>✓ La pestaña “historial” sea integrada en el perfil de usuario</li> <li>✓ Se enlisten todos los pedidos hechos con sus detalles correspondientes</li> </ul>	

Recomendaciones basadas en compras	
<b>No. Sprint:</b> 2	<b>Programador:</b> Armando
Como cliente deseo tener recomendaciones basadas en mis compras hechas para poder interesarme en nuevos productos.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"> <li>✓ Que las recomendaciones sean basadas en compras recientes</li> </ul>	

Cancelación de pedidos	
<b>No. Sprint:</b> 2	<b>Programador:</b> Armando
Como cliente deseo poder cancelar pedidos realizados y confirmados para asegurar mi satisfacción.	
<b>Criterios de aceptación:</b> <ul style="list-style-type: none"> <li>✓ Que se puedan cancelar pedidos hechos y se devuelva el pago</li> <li>✓ Una vez cancelados se muestren en el historial como cancelado</li> </ul>	

**Sprint goal:** "Permitir que los usuarios reciban un correo de confirmación de compra, tengan acceso a su historial de pedidos donde puedan cancelar pedidos pendientes y se les recomienden productos nuevos relacionados a sus compras"

#### Elementos seleccionados del Product Backlog:

- **PBI-006:** Confirmación de compras.
- **PBI-007:** Historial de pedidos.
- **PBI-008:** Recomendaciones basadas en compras.
- **PBI-009:** Cancelación de pedidos.

#### Plan de tareas técnicas:

##### Tareas para PBI-006: Confirmación de compras

- **T-01 [Infra/Cloud]:** Configurar un servicio de envío de correos en la nube y obtener las claves de API.
- **T-02 [Backend/API]:** Modificar el endpoint POST /api/checkout (creado en el sprint anterior).
- **T-03 [Backend/API]:** Dentro de la lógica de POST /api/checkout, después de que el pago sea exitoso y se guarde la orden, agregar la llamada al servicio de email (del T-01) para enviar la confirmación.
- **T-04 [Frontend/HTML]:** Crear la plantilla de correo (confirmation\_template.html). Debe ser HTML simple y responsivo (para clientes de correo móvil).
- **T-05 [Backend/API]:** Implementar la lógica para poblar la plantilla de correo con los datos reales de la orden (nombre del cliente, productos comprados, total, dirección de envío).

### Tareas para PBI-007 (historial) y PBI-009 (cancelación)

- **T-06 [Frontend/HTML]:** Crear una nueva página `account.html` que tendrá el historial de pedidos (y las recomendaciones).
- **T-07 [Frontend/CSS]:** Aplicar estilos responsivos a `account.html` para que la lista de pedidos sea clara y legible en dispositivos móviles.
- **T-08 [Backend/API]:** Crear un nuevo endpoint protegido GET `/api/orders/my-history` que verifique el JWT del usuario y devuelva solo las órdenes que le pertenecen.
- **T-09 [Frontend/JS]:** Escribir la lógica en `account.js` para llamar a GET `/api/orders/my-history` al cargar la página.
- **T-10 [Frontend/JS]:** Implementar la lógica para renderizar dinámicamente la lista de pedidos en `account.html` usando los datos de la API.
- **T-11 [Frontend/JS]:** En la lógica de renderizado (T-10), agregar un botón de "Cancelar pedido" en cada ítem de pedido.
- **T-12 [Frontend/JS]:** Implementar una lógica condicional para que el botón "Cancelar Pedido" solo aparezca si el estado del pedido es "pendiente".
- **T-13 [Backend/API]:** Crear un nuevo endpoint protegido POST `/api/orders/:orderId/cancel`.
- **T-14 [Backend/API]:** Implementar la lógica en el T-13 para:
  1. Verificar que el usuario (del JWT) sea el dueño del: `orderId`.
  2. Verificar que el estado actual del pedido sea "pendiente".
  3. Actualizar el estado del pedido en la BD de "pendiente" a "cancelado".
- **T-15 [Frontend/JS]:** Programar el fetch del botón "cancelar pedido" para que llame al T-13 y, si tiene éxito, actualice la vista del pedido en la UI (cambiando el estado a "cancelado") sin recargar la página.

### Tareas para PBI-008: Recomendaciones basadas en compras

- **T-16 [Backend/API]:** Crear un nuevo endpoint GET `/api/recommendations`.
- **T-17 [Backend/Lógica]:** Implementar la lógica de recomendación (MVP) para T-16. Por ejemplo:
  1. Obtener el historial de pedidos del usuario (usando su JWT).
  2. Identificar las categorías de los últimos productos comprados.



3. Buscar en la BD de "productos" otros productos de esas mismas categorías (excluyendo los que ya compró).
  4. Devolver una lista de 3-5 productos recomendados.
- **T-18 [Frontend/HTML]:** En la página account.html, agregar una sección `<section id="recommendations">`.
  - **T-19 [Frontend/CSS]:** Aplicar estilos responsivos a la sección de recomendaciones (probablemente reutilizando los estilos de la cuadrícula de productos del index.html).
  - **T-20 [Frontend/JS]:** En account.js, después de cargar el historial, llamar a `GET /api/recommendations` y renderizar los productos en la sección correspondiente.

#### **Tareas globales (no funcionales y pruebas)**

- **T-21 [Pruebas/Responsive]:** Probar la nueva página account.html en Chrome DevTools (o similar) para asegurar que la vista de historial y las recomendaciones se reordenen correctamente en tamaños de móvil, tablet y computadoras.
- **T-22 [Pruebas/Email]:** Probar el correo de confirmación (T-04) en clientes de correo reales (Gmail) tanto en web como en móvil para asegurar que se vea bien.
- **T-23 [Seguridad/Backend]:** Revisar todos los nuevos endpoints protegidos (T-08, T-13, T-16) para asegurar que devuelvan un error 401/403 si el usuario no está autenticado o intenta acceder a datos que no son suyos.