Universidad de la Cuenca del Plata

Autorización Definitiva. Decreto Poder Ejecutivo Nacional N°091/2006

Título del Trabajo Práctico

CARRERA: Ingeniería en Sistemas de Información

MATERIA: Paradigmas y Lenguajes de programación 3

COMISIÓN: “U” (única) “A” PROFESOR:

ESTUDIANTE: Escallier Alejandro

FECHA: 18-11-2025

Informe del Proyecto Web

Amictus – Tienda de Ropa Online

Introducción a las mejoras implementadas

Luego del planteamiento inicial del proyecto Amictus – Tienda de Ropa Online, se avanzó hacia el desarrollo técnico de las primeras funcionalidades reales del sistema. Esta etapa se enfocó en transformar el sitio desde un prototipo estático a una aplicación web funcional, conectada a una base de datos y con módulos dinámicos para la administración de productos, gestión del carrito, persistencia de datos y estructura backend.

El objetivo de esta actualización fue sentar las bases técnicas del e-commerce, asegurando escalabilidad para las funcionalidades futuras propuestas: gestión completa de usuarios, pasarelas de pago, integración con APIs de envíos y despliegue en hosting.

Arquitectura general del sistema

El sistema se diseñó bajo una arquitectura cliente-servidor, donde:

- Cliente: HTML, CSS, JavaScript, localStorage, renderizado dinámico.
- Servidor: PHP, manejo de sesiones, validación de datos, CRUD.
- Base de datos: MySQL/MariaDB (XAMPP).

La comunicación sigue este flujo:

1. El cliente solicita recursos a PHP.
2. PHP consulta o modifica datos en MySQL.
3. PHP devuelve información estructurada al frontend.
4. El frontend la presenta dinámicamente.

Esta arquitectura permite aislar la lógica de negocios en el backend, mientras que el frontend gestiona la experiencia visual y la interacción del usuario.

Base de datos: estructura y justificación técnica

Se implementó una base de datos MySQL con las siguientes tablas:

Tabla: categorías

- id (PK)
- nombre

Permite agrupar productos por tipo (abrigos, pantalones, remeras, etc.).
Justificación: normalización, integridad referencial, escalabilidad.

Tabla: productos

- id (PK)
- nombre
- precio
- emoji
- categoria_id (FK → categorías.id)

Se eligió almacenar emojis en vez de imágenes para reducir peso y acelerar el desarrollo inicial. Gracias al uso de UTF-8, la base admite caracteres Unicode sin inconvenientes .

Esta base de datos también está preparada para futuras tablas:

- usuarios
- órdenes
- items_orden
- direcciones
- métodos de pago

Backend PHP: conexión y arquitectura interna

Se desarrolló un archivo conexion.php, encargado de establecer la conexión con MySQL utilizando credenciales locales (XAMPP).

Este archivo se reutiliza en:

- index.php: obtiene los productos para mostrarlos.
- obtener_productos.php: sirve como pseudo-API.
- admin_productos.php: operaciones CRUD.
- validar_login.php: acceso al panel administrativo.

Justificación técnica:

- Evita duplicar código.
- Permite migrar la base fácilmente (solo se modifica un archivo).
- Mantiene la modularidad del backend.

Frontend dinámico: renderizado en tiempo real

El sitio reemplazó completamente los archivos HTML por archivos PHP.
El motivo es simple: permitir que el frontend reciba datos dinámicos desde la base de datos.

El catálogo de productos ya no es estático:

```
<?php while ($p = mysqli_fetch_assoc($productos)): ?>
   <!-- Tarjeta del producto dinámica -->
<?php endwhile; ?>
```

Esto convierte a Amictus en una tienda actualizable «desde el panel admin», sin tocar el código.

Carrito de compras avanzado

El carrito fue uno de los mayores avances del sistema:

✓ LocalStorage para persistencia

Permite que el carrito se mantenga incluso al cerrar la pestaña.

✓ Estructura del producto dentro del carrito

Se guardan:

- id
- nombre
- precio
- categoría
- emoji
- cantidad

Esto permitió que el carrito sea completamente funcional.

✓ Funcionalidades implementadas

- Agregar productos
- Incrementar cantidad
- Decrementar cantidad
- Eliminar un producto
- Calcular subtotal
- Sumar costo de envío
- Renderizar items en pantalla
- Actualización automática de totales
- Modal de checkout (inicio de simulación)

✓ Justificación técnica

Se eligió LocalStorage porque:

- No requiere login de usuario todavía.

- Es rápido y liviano.
- Evita tráfico innecesario a la base.
- Reduce carga en el servidor.

En el futuro, cuando se implemente login, el carrito podrá sincronizarse con la base de datos.

Panel de administración (CRUD completo)

Se desarrolló un panel administrativo moderno en admin_productos.php, con:

✓ Crear productos

✓ Editar productos

✓ Eliminar productos

✓ Ver listado con datos de BD

✓ Interfaz profesional con modales

✓ Validación de datos y sanitización

✓ Integración directa con MySQL

Todo esto permite administrar la tienda sin necesidad de entrar a phpMyAdmin.

Justificación técnica:

- Un CRUD es indispensable para escalabilidad.
- Evita manipulación manual de datos sensibles.
- Permite delegar roles a usuarios del sistema.
- Es parte fundamental de todo e-commerce real.

Sistema de login para administradores

Para proteger el panel administrativo, se implementó:

- login.php: formulario
- validar_login.php: autenticación
- logout.php: cierre de sesión
- Sistema de sesiones basado en $_SESSION
- Validación en el panel:

```
session_start();
if (!isset($_SESSION['admin'])) {
    header('Location: login.php');
```

}

Este mecanismo bloquea el acceso al CRUD para usuarios no autorizados.

En esta etapa inicial, el administrador es:

- Usuario: admin
- Contraseña: admin

En futuras versiones se añadirá:

- Hash seguro con password_hash()
- Tabla de usuarios administradores
- Roles y permisos

Checkout y modal de compra

Aunque aún no se integraron pasarelas de pago reales, se implementó:

- Formulario de datos personales
- Campos dinámicos según método de pago
- Validación en tiempo real (DNI, tarjeta, vencimiento)
- Confirmación de compra con número de orden
- Reseteo del carrito
- Flujo de compra completo (simulado)

Este módulo será clave cuando ingresen APIs externas.

Tecnologías utilizadas

Tecnología	Motivo de elección
PHP 8	Fácil integración con MySQL, soporte amplio, ideal para backend rápido
MySQL	Relacional, estable y compatible con XAMPP
JavaScript	Manipulación dinámica del DOM y carrito
LocalStorage	Persistencia sin login
HTML + CSS	Interfaz responsive
XAMPP	Servidor local rápido para desarrollo
UTF-8	Soporte para emojis como imágenes

Escalabilidad futura

A nivel técnico, Amictus ya está preparado para integración de:

✓ APIs de pago

- MercadoPago
- PayPal
- Stripe

Requerirá: backend seguro, webhooks, tokens, validación.

✓ APIs de envíos

- Andreani
- OCA
- Correo Argentino

Requerirá: cálculo dinámico de costos y tiempos.

✓ Registro de usuarios

Autenticación completa, carrito asociado a usuarios, historial de compras.

✓ Hosting y despliegue

El sistema podrá hostearse en:

- Hostinger
- InfinityFree
- 000webhost
- Servidores VPS
- Apache / Nginx

Migración simple debido a la arquitectura modular.

Conclusión

El proyecto Amictus evolucionó de una idea inicial a un sistema Web funcional, con backend, base de datos, CRUD, login administrativo, carrito dinámico y procesos de compra simulados.

Las bases técnicas están sólidamente implementadas, permitiendo que en fases futuras se integren servicios externos, sistemas de pago, envíos y perfiles de usuario, transformando este proyecto en un e-commerce completamente operativo.

La arquitectura modular y la separación lógica entre frontend, backend y base de datos garantizan que el sistema pueda escalarse sin reescrituras innecesarias.