Sistema de Gestión de Maquinaria Agrícola

Una guía completa paso a paso para desarrollar tu proyecto individual del SENA - Centro de Biotecnología Agropecuaria (CBA)

Instructor: Iván Malavé Fierro
Módulo: Desarrollo Backend con PHP
Tecnología en Análisis y Desarrollo de Software

Introducción al Proyecto

Bienvenido, futuro tecnólogo en desarrollo de software. Este proyecto te permitirá demostrar todo lo aprendido en el módulo de Backend con PHP, creando una aplicación real que resuelve problemas del sector agrícola.

¿Por qué es útil aprender esto? Porque combinarás programación, bases de datos, y tecnología móvil para crear un sistema que empresas reales necesitan: gestionar su maquinaria agrícola sin depender de internet.

Al finalizar esta guía, habrás construido un sistema completo que funciona tanto online como offline, con reportes en PDF, sincronización de datos, y una interfaz amigable para operadores en el campo.

Antes de Empezar - Requisitos Previos

MySQL

01

Conocimientos Básicos

Saber programar en PHP (variables, funciones, arrays)

Conocimientos básicos de HTML y CSS

Experiencia básica con bases de datos

02

Software Necesario

XAMPP (servidor local con PHP y MySQL)

Editor de código (Visual Studio Code recomendado)

Navegador web moderno (Chrome o Firefox)

Preparación del Entorno

1. Instala XAMPP desde apachefriends.org

2. Inicia Apache y MySQL desde el panel de control

3. Crea una carpeta "proyecto_maquinaria" en htdocs

03

Mapa del Tema - ¿Qué Aprenderás?

Base de Datos Híbrida

MySQL para el servidor central y SQLite para modo offline

Mantenimientos

Programación automática y registro de revisiones técnicas

App Móvil Híbrida

Funciona sin internet y sincroniza cuando hay conexión

Gestión de Maquinaria

Registro, control de estado y asignación de equipos agrícolas

Control de Combustible

Registro de consumos y alertas de rendimiento

Reportes Avanzados

Generación automática de PDF con estadísticas

Lección 1: ¿Qué es un Sistema de Gestión de Maquinaria?

Qué es

Es una aplicación web que permite a empresas agrícolas controlar el estado, mantenimiento y uso de sus tractores, cosechadoras y otros equipos, incluso cuando no hay internet en el campo.

Para qué sirve

- Evitar que la maquinaria se dañe por falta de mantenimiento
- Controlar costos de combustible y operación
- Saber dónde está cada máquina en tiempo real

Lección 1: Cómo Funciona (Continuación) Cómo se hace

Piensa en esto como el "cuaderno digital" de una empresa agrícola. En lugar de anotar en papel cuándo se le hizo mantenimiento a un

tractor, todo queda guardado en la aplicación.

Paso a Paso - Arquitectura del Sistema

Creas una base de datos para guardar información de las máquinas1.

Programas formularios web para registrar datos2.

Implementas sincronización offline con SQLite3.

Agregas reportes automáticos en PDF4.

Pruebas todo en dispositivos móviles5.

Lección 1: Ejemplo Mínimo Reproducible

Salida Esperada: Una página web que muestra "Sistema de Maquinaria Agrícola" y conecta con la base de datos.

```
<?php
// index.php - Archivo principal del sistema
session_start(); // Inicia sesiones para login

// Configuración de base de datos
$host = 'localhost';
$dbname = 'maquinaria_agricola';
$username = 'root';
$password = '';
```

```
try {
 // Conexión PDO (más segura que mysqli)
 $pdo = new PDO("mysql:host=$host;dbname=$dbname",
 $username, $password);
 $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
 echo "
```

Errores Comunes y Soluciones

Error: "could not find driver"

Causa: PDO no está habilitado en PHP

Solución: En php.ini descomenta la línea: extension=pdo_mysql

Error: "Access denied"

Causa: Usuario o contraseña incorrecta

Solución: Verifica que XAMPP esté ejecutándose y usa 'root' sin contraseña

Error: "Unknown database"

Causa: La base de datos no existe

Solución: Créala primero en phpMyAdmin

Comprueba tu comprensión: ¿Puedes ver el mensaje "✅Conexión exitosa" en tu navegador? Si no, revisa que XAMPP esté corriendo.

Lección 2: Diseño de Base de Datos

Qué es

La estructura donde guardaremos toda la información: máquinas, mantenimientos, combustible y usuarios. Es como los archivadores de una oficina, pero digitales.

Para qué sirve

- Organizar información de forma eficiente
- Permitir búsquedas rápidas
- Mantener la integridad de los datos

Lección 2: Tablas Principales del Sistema Cómo se hace

Piensa en las tablas como hojas de cálculo Excel, donde cada fila es un registro y cada columna es una característica.

Tabla: maquinaria

Guarda información básica de cada máquina: ID, tipo, marca, modelo, año, número de serie, horas de uso, estado

Tabla: combustible

Controla el consumo: fecha, cantidad, precio, rendimiento por hora, operador

Tabla: mantenimientos

Registra cada servicio realizado: fecha, tipo de mantenimiento, costo, técnico responsable, próxima revisión

Maneja accesos: nombre, email, contraseña encriptada, rol (admin/operador)

Tabla: usuarios

Paso a Paso: Creación de la Base de Datos

01

Accede a phpMyAdmin

Abre tu navegador y ve a: http://localhost/phpmyadmin

Usuario: root (sin contraseña)

02

Crea la Base de Datos

Haz clic en "Nueva" en el menú izquierdo Nombre: maquinaria_agricola

Codificación: utf8mb4_spanish_ci

03

Ejecuta el Script SQL

Ve a la pestaña "SQL"

Pega el código de creación de tablas Haz clic en "Continuar"

Ejemplo: Script de Creación de Tablas

```
-- Crear base de datos
CREATE DATABASE IF NOT EXISTS maquinaria_agricola;
USE maquinaria_agricola;

-- Tabla de maquinaria
```

```
CREATE TABLE maquinaria (
 id INT PRIMARY KEY AUTO_INCREMENT,
 tipo ENUM('tractor', 'cosechadora', 'sembradora', 'fumigadora') NOT NULL,
 marca VARCHAR(50) NOT NULL,
 modelo VARCHAR(50) NOT NULL,
 año YEAR NOT NULL,
 numero_serie VARCHAR(100) UNIQUE NOT NULL,
 horas_uso INT DEFAULT 0,
 estado ENUM('disponible', 'en_mantenimiento', 'alquilado', 'baja_tecnica') DEFAULT 'disponible',  fecha_registro TIMESTAMP DEFAULT
CURRENT_TIMESTAMP,
 latitud DECIMAL(10,8) NULL,
 longitud DECIMAL(11,8) NULL
);

-- Tabla de mantenimientos
CREATE TABLE mantenimientos (
 id INT PRIMARY KEY AUTO_INCREMENT,
 maquinaria_id INT NOT NULL,
 tipo_mantenimiento ENUM('preventivo', 'correctivo', 'revision_tecnica') NOT NULL,  fecha_realizado DATE NOT NULL,
 proxima_fecha DATE NULL,
 costo DECIMAL(10,2) DEFAULT 0,
 tecnico VARCHAR(100) NOT NULL,
 descripcion TEXT,
 repuestos_usados TEXT,
```

FOREIGN KEY (maquinaria_id) REFERENCES maquinaria(id) ON DELETE CASCADE);

Salida Esperada: "Se han creado las tablas exitosamente" en phpMyAdmin.

Lección 3: Sistema de Autenticación

Qué es

Un sistema que verifica quién puede acceder a la aplicación. Como el portero de un edificio que pide credencial antes de dejar pasar. **Para qué sirve**

- Proteger información sensible de la empresa
- Controlar qué funciones puede usar cada usuario
- Mantener un registro de quién hizo qué cambios

Cómo Implementar Login Seguro **Paso a Paso**

1

Formulario de Login

Crea un formulario HTML con campos de email y

contraseña 3

2

Validación PHP

Verifica que los datos existan en la base de

datos 4

Protección de Rutas

Verifica sesión en cada página protegida

Sesiones Seguras

Guarda datos del usuario en $_SESSION

TIP: Nunca guardes contraseñas en texto plano. Usa password_hash() de PHP.

Código del Sistema de Login

```
<?php
// auth.php - Manejo de autenticación
session_start();

function login($email, $password, $pdo) {
```

```
 // Buscar usuario en base de datos
 $stmt = $pdo->prepare("SELECT id, nombre, email, password, rol   FROM usuarios WHERE email = ?");
 $stmt->execute([$email]);
 $usuario = $stmt->fetch();

 // Verificar contraseña
 if ($usuario && password_verify($password, $usuario['password'])) {  // Crear sesión
 $_SESSION['usuario_id'] = $usuario['id'];
 $_SESSION['usuario_nombre'] = $usuario['nombre'];
 $_SESSION['usuario_rol'] = $usuario['rol'];
 return true;
 }
 return false;
}

function verificar_sesion() {
 if (!isset($_SESSION['usuario_id'])) {
 header('Location: login.php');
 exit();
 }
}

function logout() {
 session_unset();
```

```
 session_destroy();
 header('Location: login.php');
 exit();
}
?>
```

Formulario de Login Completo

```
<!-- login.php -->
<?php
require_once 'config/database.php';
require_once 'includes/auth.php';

if ($_POST) {
 $email = $_POST['email'];
 $password = $_POST['password'];

 if (login($email, $password, $pdo)) {
 header('Location: dashboard.php');
 exit();
 } else {
 $error = "Email o contraseña incorrectos";
 }
}
```

```
?>

<!DOCTYPE html>
<html lang="es">
<head>
 <title>Login - Sistema de Maquinaria</title>
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"> </head>
<body class="bg-light">
 <div class="container mt-5">
 <div class="row justify-content-center">
 <div class="col-md-6">
 <div class="card">
 <div class="card-body">
 <h3 class="card-title text-center">Iniciar Sesión</h3>
 <?php if(isset($error)): ?>
 <div class="alert alert-danger">
```

Comprobación del Sistema de Login

Salida Esperada:

- Formulario de login aparece correctamente
- Con datos correctos redirige al dashboard
- Con datos incorrectos muestra error

- Sesión se mantiene entre páginas

CUIDADO: Errores comunes al implementar login:

- No usar password_hash() para encriptar contraseñas
- Olvidar session_start() al inicio de cada página
- No validar datos antes de enviar a la base de datos

Comprueba tu comprensión: Crea un usuario de prueba con password_hash() y verifica que puedas hacer login correctamente.

Lección 4: Gestión de Maquinaria - CRUD

Básico Qué es CRUD

Son las cuatro operaciones básicas con datos: **C**rear, **R**ead (leer), **U**pdate (actualizar), **D**elete (eliminar). Como un fichero de oficina donde puedes agregar, consultar, modificar y eliminar documentos.

Para qué sirve

- Registrar nuevas máquinas que llegan a la empresa
- Consultar información rápidamente

- Actualizar datos como horas de uso o ubicación
- Dar de baja equipos que ya no sirven

Estructura del Módulo de Maquinaria

Crear Nueva

maquinaria/crear.php

Formulario para registrar nueva

Eliminar

Listar Máquinas

maquinaria/index.php

Muestra tabla con todas las máquinas registradas

Ver Detalles

maquinaria/ver.php

Información completa de una máquina específica

maquinaria

Editar Datos

maquinaria/editar.php

Modificar información existente

maquinaria/eliminar.php

Dar de baja una máquina del sistema

Código: Listar Maquinaria

```
<?php
```

```
// maquinaria/index.php
require_once '../includes/auth.php';
require_once '../config/database.php';

verificar_sesion(); // Verificar que el usuario esté logueado

// Obtener todas las máquinas
$stmt = $pdo->prepare("
 SELECT
 id, tipo, marca, modelo, año,
 numero_serie, horas_uso, estado,
 DATE_FORMAT(fecha_registro, '%d/%m/%Y') as fecha_registro
 FROM maquinaria
 ORDER BY fecha_registro DESC
");
$stmt->execute();
$maquinas = $stmt->fetchAll();
?>

<!DOCTYPE html>
<html lang="es">
<head>
 <title>Gestión de Maquinaria</title>
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"> </head>
```

```
<body>
 <div class="container mt-4">
 <div class="d-flex justify-content-between align-items-center mb-4">
 <h2>Gestión de Maquinaria</h2>
 <a href="crear.php" class="btn btn-primary">
```

Formulario para Crear Maquinaria

```
<?php
// maquinaria/crear.php
require_once '../includes/auth.php';
require_once '../config/database.php';

verificar_sesion();

if ($_POST) {
 try {
 $stmt = $pdo->prepare("
 INSERT INTO maquinaria
 (tipo, marca, modelo, año, numero_serie, horas_uso, estado)
 VALUES (?, ?, ?, ?, ?, ?, ?)
 ");

 $resultado = $stmt->execute([
```

```
$_POST['tipo'],
$_POST['marca'],
$_POST['modelo'],
$_POST['año'],
$_POST['numero_serie'],
$_POST['horas_uso'],
$_POST['estado']
]);

if ($resultado) {
$mensaje = "
```

Lección 5: Control de Mantenimientos

Qué es

Un sistema que programa y registra todos los servicios técnicos de las máquinas: cambios de aceite, revisiones, reparaciones. Como el recordatorio del celular para citas médicas, pero para tractores.

Para qué sirve

- Evitar averías costosas por falta de mantenimiento
- Cumplir con revisiones técnicas obligatorias

▪ Controlar costos de repuestos y servicios

▪ Programar mantenimientos por horas de uso

Sistema de Alertas de Mantenimiento Cómo funciona

El sistema calcula automáticamente cuándo una máquina necesita servicio, basándose en:

Horas de Uso

Cada 250 horas → Servicio menor Cada 500 horas → Servicio mayor

1

Tiempo Transcurrido

Cada 3 meses → Revisión preventiva Cada 6 meses → Inspección completa

2

Revisión Técnica

Anual → Certificación obligatoria Semestral → Equipos pesados

3

TIP: Las alertas aparecen 15 días antes de la fecha límite, dando tiempo para programar el servicio.

Código: Sistema de Alertas

```
<?php
// includes/alertas_mantenimiento.php

function obtener_alertas_mantenimiento($pdo) {
 // Consulta para obtener máquinas que necesitan mantenimiento
 $stmt = $pdo->prepare("
 SELECT
 m.id, m.tipo, m.marca, m.modelo, m.horas_uso,
 COALESCE(MAX(man.fecha_realizado), m.fecha_registro) as ultimo_mantenimiento,

 -- Calcular días desde último mantenimiento
 DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) as dias_sin_mantenimiento,
```

```
-- Determinar tipo de alerta
CASE
WHEN m.horas_uso % 500 = 0 AND m.horas_uso > 0 THEN 'SERVICIO_MAYOR'
WHEN m.horas_uso % 250 = 0 AND m.horas_uso > 0 THEN 'SERVICIO_MENOR'
WHEN DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 90   THEN 'REVISION_PREVENTIVA'
WHEN DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 365   THEN 'REVISION_TECNICA'
ELSE NULL
END as tipo_alerta,

-- Nivel de urgencia
CASE
WHEN DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 365   THEN 'CRITICO'
WHEN DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 180   THEN 'ALTO'
WHEN DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 90   THEN 'MEDIO'
ELSE 'BAJO'
END as urgencia

FROM maquinaria m
LEFT JOIN mantenimientos man ON m.id = man.maquinaria_id
WHERE m.estado != 'baja_tecnica'
GROUP BY m.id
HAVING tipo_alerta IS NOT NULL
ORDER BY urgencia DESC, dias_sin_mantenimiento DESC
```

```
 ");

 $stmt->execute();
 return $stmt->fetchAll();
}

function mostrar_alertas($alertas) {
 if (empty($alertas)) {
 echo '<div class="alert alert-success">
```

Dashboard con Alertas Integradas

```
<?php
// dashboard.php - Página principal del sistema
require_once 'includes/auth.php';
require_once 'config/database.php';
require_once 'includes/alertas_mantenimiento.php';

verificar_sesion();

// Obtener estadísticas generales
$stmt = $pdo->query("
 SELECT
 COUNT(*) as total_maquinas,
```

```
 SUM(CASE WHEN estado = 'disponible' THEN 1 ELSE 0 END) as disponibles,
 SUM(CASE WHEN estado = 'alquilado' THEN 1 ELSE 0 END) as alquiladas,
 SUM(CASE WHEN estado = 'en_mantenimiento' THEN 1 ELSE 0 END) as en_mantenimiento  FROM maquinaria
");
$stats = $stmt->fetch();

// Obtener alertas de mantenimiento
$alertas = obtener_alertas_mantenimiento($pdo);
?>

<!DOCTYPE html>
<html lang="es">
<head>
 <title>Dashboard - Sistema de Maquinaria</title>
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"> </head>
<body>
 <nav class="navbar navbar-dark bg-dark">
 <div class="container">
 <a class="navbar-brand" href="#">
```

Lección 6: Control de Combustible

Qué es

Un módulo que registra cada carga de combustible y calcula el rendimiento de las máquinas. Como llevar el control de gastos de gasolina de tu carro, pero para toda una flota de tractores.

Para qué sirve

- Detectar máquinas que consumen más combustible del normal
- Calcular costos operativos reales por hora de trabajo
- Programar cargas de combustible según rutas de trabajo
- Generar reportes de eficiencia para la gerencia

Base de Datos para Combustible

```
-- Tabla para registro de combustible
CREATE TABLE combustible (
 id INT PRIMARY KEY AUTO_INCREMENT,
 maquinaria_id INT NOT NULL,
 fecha DATE NOT NULL,
 cantidad_litros DECIMAL(8,2) NOT NULL,
 precio_por_litro DECIMAL(6,2) NOT NULL,
 costo_total DECIMAL(10,2) GENERATED ALWAYS AS (cantidad_litros * precio_por_litro) STORED,
 horas_uso_inicio INT NOT NULL,
```

```
 horas_uso_fin INT NULL,
 rendimiento_calculado DECIMAL(6,2) NULL COMMENT 'Litros por hora',
 operador VARCHAR(100) NOT NULL,
 ubicacion VARCHAR(200) NULL,
 observaciones TEXT NULL,
 fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (maquinaria_id) REFERENCES maquinaria(id) ON DELETE CASCADE
);

-- Índices para consultas rápidas
CREATE INDEX idx_combustible_fecha ON combustible(fecha);
CREATE INDEX idx_combustible_maquinaria ON combustible(maquinaria_id);

-- Vista para cálculos automáticos de rendimiento
CREATE VIEW vista_rendimiento_combustible AS
SELECT
 c.id,
 c.maquinaria_id,
 m.marca,
 m.modelo,
 c.fecha,
 c.cantidad_litros,
 c.costo_total,
 c.horas_uso_inicio,
```

```
 c.horas_uso_fin,

 -- Calcular rendimiento si hay horas de fin
 CASE
 WHEN c.horas_uso_fin IS NOT NULL AND c.horas_uso_fin > c.horas_uso_inicio
 THEN c.cantidad_litros / (c.horas_uso_fin - c.horas_uso_inicio)
 ELSE c.rendimiento_calculado
 END AS rendimiento_litros_hora,

 -- Calcular costo por hora
 CASE
 WHEN c.horas_uso_fin IS NOT NULL AND c.horas_uso_fin > c.horas_uso_inicio
 THEN c.costo_total / (c.horas_uso_fin - c.horas_uso_inicio)
 ELSE NULL
 END AS costo_por_hora

FROM combustible c
JOIN maquinaria m ON c.maquinaria_id = m.id;
```

TIP: La columna costo_total se calcula automáticamente multiplicando cantidad por precio. Esto evita errores de cálculo manual.

Formulario de Registro de Combustible

```php
<?php
// combustible/registrar.php
require_once '../includes/auth.php';
require_once '../config/database.php';

verificar_sesion();

// Obtener máquinas disponibles
$stmt = $pdo->query("
 SELECT id, CONCAT(marca, ' ', modelo, ' (', tipo, ')') as nombre_completo,
 horas_uso
 FROM maquinaria
 WHERE estado IN ('disponible', 'alquilado')
 ORDER BY marca, modelo
");
$maquinas = $stmt->fetchAll();

if ($_POST) {
 try {
 $stmt = $pdo->prepare("
 INSERT INTO combustible
 (maquinaria_id, fecha, cantidad_litros, precio_por_litro,
 horas_uso_inicio, operador, ubicacion, observaciones)
```

```
VALUES (?, ?, ?, ?, ?, ?, ?, ?)
");

$resultado = $stmt->execute([
$_POST['maquinaria_id'],
$_POST['fecha'],
$_POST['cantidad_litros'],
$_POST['precio_por_litro'],
$_POST['horas_uso_inicio'],
$_POST['operador'],
$_POST['ubicacion'],
$_POST['observaciones']
]);

if ($resultado) {
$mensaje = "
```

Análisis de Rendimiento de Combustible

```
<?php
// combustible/analisis.php - Página de análisis de rendimiento
require_once '../includes/auth.php';
require_once '../config/database.php';
```

```
verificar_sesion();

// Obtener estadísticas de rendimiento por máquina
$stmt = $pdo->query("
 SELECT
 m.id,
 CONCAT(m.marca, ' ', m.modelo) as maquina,
 m.tipo,
 COUNT(c.id) as total_cargas,
 SUM(c.cantidad_litros) as total_litros,
 SUM(c.costo_total) as total_costo,

 -- Rendimiento promedio (solo cargas con horas de fin registradas)
 AVG(CASE
 WHEN c.horas_uso_fin IS NOT NULL AND c.horas_uso_fin > c.horas_uso_inicio
 THEN c.cantidad_litros / (c.horas_uso_fin - c.horas_uso_inicio)
 ELSE NULL
 END) as rendimiento_promedio,

 -- Costo promedio por hora
 AVG(CASE
 WHEN c.horas_uso_fin IS NOT NULL AND c.horas_uso_fin > c.horas_uso_inicio
 THEN c.costo_total / (c.horas_uso_fin - c.horas_uso_inicio)
```

```
 ELSE NULL
 END) as costo_promedio_hora,

 MAX(c.fecha) as ultima_carga

 FROM maquinaria m
 LEFT JOIN combustible c ON m.id = c.maquinaria_id
 WHERE m.estado != 'baja_tecnica'
 GROUP BY m.id
 HAVING total_cargas > 0
 ORDER BY rendimiento_promedio DESC
");

$analisis = $stmt->fetchAll();

// Obtener alertas de consumo excesivo
$stmt_alertas = $pdo->query("
 SELECT
 m.id,
 CONCAT(m.marca, ' ', m.modelo) as maquina,
 AVG(c.cantidad_litros / (c.horas_uso_fin - c.horas_uso_inicio)) as rendimiento_actual,

 -- Rendimiento histórico promedio (últimos 6 meses)
 (SELECT AVG(c2.cantidad_litros / (c2.horas_uso_fin - c2.horas_uso_inicio))
```

```
 FROM combustible c2
 WHERE c2.maquinaria_id = m.id
 AND c2.fecha >= DATE_SUB(CURDATE(), INTERVAL 6 MONTH)
 AND c2.horas_uso_fin IS NOT NULL
 AND c2.horas_uso_fin > c2.horas_uso_inicio
 ) as rendimiento_historico

 FROM maquinaria m
 JOIN combustible c ON m.id = c.maquinaria_id
 WHERE c.fecha >= DATE_SUB(CURDATE(), INTERVAL 1 MONTH)
 AND c.horas_uso_fin IS NOT NULL
 AND c.horas_uso_fin > c.horas_uso_inicio
 GROUP BY m.id
 HAVING rendimiento_actual > rendimiento_historico * 1.2 -- 20% más consumo
");

$alertas_consumo = $stmt_alertas->fetchAll();
?>

<!DOCTYPE html>
<html lang="es">
<head>
 <title>Análisis de Rendimiento - Combustible</title>
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">  <script
```

```
src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
 <div class="container mt-4">
 <h2>
```

Lección 7: Sistema Offline con SQLite

Qué es

Una funcionalidad que permite usar el sistema sin internet, guardando datos localmente en SQLite. Como tener una copia del sistema en cada tablet que funciona aunque no haya señal en el campo.

Para qué sirve

▪ Trabajar en zonas rurales sin cobertura de internet

▪ Registrar datos de campo en tiempo real

▪ Sincronizar automáticamente cuando vuelva la conexión

▪ Mantener operatividad 24/7 sin depender de la red

Arquitectura del Sistema Híbrido

Proceso automático que envía y recibe cambios

SQLite Local

Base de datos ligera que guarda datos sin conexión

Aplicación Móvil

Interfaz web que funciona como PWA (Progressive Web App)

Servidor Central

MySQL que consolida información de todas las tablets

Sincronización

CUIDADO: La sincronización debe manejar conflictos cuando el mismo registro se edita en múltiples dispositivos.

Configuración de SQLite para Modo Offline

```
<?php
// config/sqlite_config.php
class SQLiteManager {
 private $sqlite_db;
 private $mysql_pdo;

 public function __construct($mysql_pdo = null) {
```

```
// Crear base de datos SQLite local
$db_path = __DIR__ . '/../data/maquinaria_local.sqlite';

// Crear directorio si no existe
if (!is_dir(dirname($db_path))) {
mkdir(dirname($db_path), 0777, true);
}

try {
$this->sqlite_db = new PDO("sqlite:$db_path");
$this->sqlite_db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$this->mysql_pdo = $mysql_pdo;

// Crear tablas si no existen
$this->crearTablasSQLite();

} catch(PDOException $e) {
throw new Exception("Error al conectar SQLite: " . $e->getMessage());
}
}

private function crearTablasSQLite() {
$tablas = [
// Tabla de maquinaria con campos adicionales para sincronización
```

```
"CREATE TABLE IF NOT EXISTS maquinaria_local (
id INTEGER PRIMARY KEY AUTOINCREMENT,
id_servidor INTEGER NULL,
tipo TEXT NOT NULL,
marca TEXT NOT NULL,
modelo TEXT NOT NULL,
año INTEGER NOT NULL,
numero_serie TEXT UNIQUE NOT NULL,
horas_uso INTEGER DEFAULT 0,
estado TEXT DEFAULT 'disponible',
latitud REAL NULL,
longitud REAL NULL,
fecha_registro DATETIME DEFAULT CURRENT_TIMESTAMP,
fecha_modificacion DATETIME DEFAULT CURRENT_TIMESTAMP,
sincronizado INTEGER DEFAULT 0,
accion TEXT DEFAULT 'INSERT' -- INSERT, UPDATE, DELETE
)",

// Tabla de combustible local
"CREATE TABLE IF NOT EXISTS combustible_local (
id INTEGER PRIMARY KEY AUTOINCREMENT,
id_servidor INTEGER NULL,
maquinaria_id_local INTEGER NOT NULL,
maquinaria_id_servidor INTEGER NULL,
```

```
fecha DATE NOT NULL,
cantidad_litros REAL NOT NULL,
precio_por_litro REAL NOT NULL,
costo_total REAL NOT NULL,
horas_uso_inicio INTEGER NOT NULL,
horas_uso_fin INTEGER NULL,
operador TEXT NOT NULL,
ubicacion TEXT NULL,
observaciones TEXT NULL,
fecha_registro DATETIME DEFAULT CURRENT_TIMESTAMP,
fecha_modificacion DATETIME DEFAULT CURRENT_TIMESTAMP,
sincronizado INTEGER DEFAULT 0,
accion TEXT DEFAULT 'INSERT',
FOREIGN KEY (maquinaria_id_local) REFERENCES maquinaria_local(id)
)",

// Tabla de mantenimientos local
"CREATE TABLE IF NOT EXISTS mantenimientos_local (
id INTEGER PRIMARY KEY AUTOINCREMENT,
id_servidor INTEGER NULL,
maquinaria_id_local INTEGER NOT NULL,
maquinaria_id_servidor INTEGER NULL,
tipo_mantenimiento TEXT NOT NULL,
fecha_realizado DATE NOT NULL,
```

```
proxima_fecha DATE NULL,
costo REAL DEFAULT 0,
tecnico TEXT NOT NULL,
descripcion TEXT,
repuestos_usados TEXT,
fecha_registro DATETIME DEFAULT CURRENT_TIMESTAMP,
fecha_modificacion DATETIME DEFAULT CURRENT_TIMESTAMP,
sincronizado INTEGER DEFAULT 0,
accion TEXT DEFAULT 'INSERT',
FOREIGN KEY (maquinaria_id_local) REFERENCES maquinaria_local(id)
)",

// Tabla de log de sincronización
"CREATE TABLE IF NOT EXISTS sync_log (
id INTEGER PRIMARY KEY AUTOINCREMENT,
tabla TEXT NOT NULL,
registro_id INTEGER NOT NULL,
accion TEXT NOT NULL,
fecha_sync DATETIME DEFAULT CURRENT_TIMESTAMP,
resultado TEXT NOT NULL,
error_message TEXT NULL
)"
];
```

```
foreach ($tablas as $sql) {
$this->sqlite_db->exec($sql);
}
}

public function obtenerConexionSQLite() {
return $this->sqlite_db;
}

public function verificarConexionInternet() {
// Intentar hacer ping al servidor
$conexion = @fsockopen('www.google.com', 80, $errno, $errstr, 5);
if ($conexion) {
fclose($conexion);
return true;
}
return false;
}

public function modoOperacion() {
if ($this->mysql_pdo && $this->verificarConexionInternet()) {
return 'ONLINE';
} else {
```

```
    return 'OFFLINE';
    }
    }
   }
   ?>
```

Funciones de Sincronización

```
<?php
// includes/sincronizacion.php
require_once '../config/sqlite_config.php';

class SincronizadorDatos {
 private $sqlite;
 private $mysql;
 private $log = [];

 public function __construct($mysql_pdo) {
 $this->sqlite = new SQLiteManager($mysql_pdo);
 $this->mysql = $mysql_pdo;
 }

 public function sincronizarTodo() {
 try {
```

```
// Verificar conexión
if (!$this->sqlite->verificarConexionInternet()) {
throw new Exception("No hay conexión a internet");
}

$this->log[] = "
```

PWA - Progressive Web App

Qué es una PWA

Una aplicación web que se comporta como app nativa del celular. Funciona sin internet, se puede instalar en el escritorio, y envía notificaciones push. Como WhatsApp Web, pero más avanzado.

```
// manifest.json - Configuración de PWA
{
 "name": "Sistema de Maquinaria Agrícola",
 "short_name": "MaquinariaApp",
 "description": "Gestión de maquinaria agrícola offline",
 "start_url": "./",
 "display": "standalone",
 "background_color": "#ffffff",
 "theme_color": "#1A2D7A",
 "orientation": "portrait",
```

```
"icons": [
{
"src": "icons/icon-72.png",
"sizes": "72x72",
"type": "image/png"
},
{
"src": "icons/icon-144.png",
"sizes": "144x144",
"type": "image/png"
},
{
"src": "icons/icon-192.png",
"sizes": "192x192",
"type": "image/png"
},
{
"src": "icons/icon-512.png",
"sizes": "512x512",
"type": "image/png"
}
],
"categories": ["business", "productivity"],
"lang": "es-ES"
```

```
}
```

TIP: El manifest.json debe estar en la carpeta raíz y enlazado en el HTML con `<link rel="manifest" href="manifest.json">`

Service Worker para Funcionamiento Offline

```
// service-worker.js - Controla el funcionamiento offline
const CACHE_NAME = 'maquinaria-app-v1';
const STATIC_CACHE = 'static-v1';
const DYNAMIC_CACHE = 'dynamic-v1';

// Archivos a cachear al instalar la PWA
const STATIC_FILES = [
 '/',
 '/index.php',
 '/dashboard.php',
 '/maquinaria/index.php',
 '/assets/css/bootstrap.min.css',
 '/assets/js/app.js',
 '/assets/icons/icon-192.png',
 '/offline.html'
];
```

```
// Instalación del Service Worker
self.addEventListener('install', event => {
 console.log('
```

Lección 8: Generación de Reportes PDF

Qué es

Un sistema que crea automáticamente documentos PDF con información de las máquinas, mantenimientos y costos. Como generar facturas automáticas, pero para reportes gerenciales.

Para qué sirve

- Entregar reportes mensuales a la gerencia
- Generar certificados de mantenimiento para clientes
- Crear informes técnicos para seguros
- Exportar datos para auditorías

Instalación y Configuración de TCPDF Paso a Paso

01

Descargar TCPDF

Ve a tcpdf.org y descarga la librería

O usa Composer: composer require tecnickcom/tcpdf

```
// config/pdf_config.php
```

02

Instalar en el Proyecto

Crea carpeta /vendor/tcpdf/ en tu proyecto Extrae todos los archivos de TCPDF ahí

03

Configurar Autoload

Incluye require_once 'vendor/tcpdf/tcpdf.php' Configura fonts y cache directories

```
require_once __DIR__ . '/../vendor/tcpdf/tcpdf.php';

class PDFGenerator extends TCPDF {

 // Configuración personalizada de header
```

```
 public function Header() {
 $this->SetFont('helvetica', 'B', 20);
 $this->SetTextColor(26, 45, 122); // Color #1A2D7A
 $this->Cell(0, 15, 'Sistema de Maquinaria Agrícola', 0, 1, 'C');
 $this->Ln(5);

 // Línea separadora
 $this->SetLineWidth(0.5);
 $this->SetDrawColor(26, 45, 122);
 $this->Line(15, 35, 195, 35);
 $this->Ln(10);
 }

 // Configuración personalizada de footer
 public function Footer() {
 $this->SetY(-15);
 $this->SetFont('helvetica', 'I', 8);
 $this->SetTextColor(128, 128, 128);
 $this->Cell(0, 10, 'Página ' . $this->getAliasNumPage() . ' de ' . $this->getAliasNbPages(), 0, 0, 'C');

 $this->SetY(-20);
 $this->Cell(0, 10, 'Generado el: ' . date('d/m/Y H:i:s'), 0, 0, 'R');
 }
}
```

Reporte de Mantenimientos Pendientes

```
<?php
// reportes/mantenimientos_pendientes.php
require_once '../config/pdf_config.php';
require_once '../config/database.php';
require_once '../includes/auth.php';

verificar_sesion();

// Obtener datos para el reporte
$stmt = $pdo->query("
 SELECT
 m.id,
 CONCAT(m.marca, ' ', m.modelo) as maquina,
 m.tipo,
 m.horas_uso,
 m.estado,
 COALESCE(MAX(man.fecha_realizado), m.fecha_registro) as ultimo_mantenimiento,
 DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) as dias_sin_mantenimiento,

 -- Próximo mantenimiento programado
 CASE
```

```
WHEN m.horas_uso >= 500 THEN 'SERVICIO MAYOR URGENTE'
WHEN m.horas_uso >= 250 THEN 'SERVICIO MENOR REQUERIDO'
WHEN DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 180   THEN 'REVISIÓN PREVENTIVA'
ELSE 'AL DÍA'
END as estado_mantenimiento,

-- Prioridad
CASE
WHEN DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 365   THEN 'CRÍTICO'
WHEN DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 180   THEN 'ALTO'
WHEN m.horas_uso >= 500 THEN 'ALTO'
WHEN m.horas_uso >= 250 THEN 'MEDIO'
ELSE 'BAJO'
END as prioridad

FROM maquinaria m
LEFT JOIN mantenimientos man ON m.id = man.maquinaria_id
WHERE m.estado != 'baja_tecnica'
GROUP BY m.id
HAVING estado_mantenimiento != 'AL DÍA'
ORDER BY
CASE prioridad
WHEN 'CRÍTICO' THEN 1
WHEN 'ALTO' THEN 2
```

```
 WHEN 'MEDIO' THEN 3
 ELSE 4
 END,
 dias_sin_mantenimiento DESC
");

$mantenimientos_pendientes = $stmt->fetchAll();

// Crear PDF
$pdf = new PDFGenerator('P', 'mm', 'A4');
$pdf->SetCreator('Sistema de Maquinaria Agrícola');
$pdf->SetAuthor($_SESSION['usuario_nombre']);
$pdf->SetTitle('Reporte de Mantenimientos Pendientes');
$pdf->SetSubject('Mantenimientos Requeridos');

$pdf->AddPage();

// Título del reporte
$pdf->SetFont('helvetica', 'B', 16);
$pdf->SetTextColor(26, 45, 122);
$pdf->Cell(0, 10, 'REPORTE DE MANTENIMIENTOS PENDIENTES', 0, 1, 'C');
$pdf->Ln(5);

// Información del reporte
```

```
$pdf->SetFont('helvetica', '', 10);
$pdf->SetTextColor(0, 0, 0);
$pdf->Cell(40, 6, 'Fecha del reporte:', 0, 0, 'L');
$pdf->Cell(0, 6, date('d/m/Y H:i:s'), 0, 1, 'L');
$pdf->Cell(40, 6, 'Generado por:', 0, 0, 'L');
$pdf->Cell(0, 6, $_SESSION['usuario_nombre'], 0, 1, 'L');
$pdf->Cell(40, 6, 'Total de alertas:', 0, 0, 'L');
$pdf->Cell(0, 6, count($mantenimientos_pendientes), 0, 1, 'L');
$pdf->Ln(5);

// Estadísticas rápidas
$criticos = array_filter($mantenimientos_pendientes, function($item) { return $item['prioridad'] == 'CRÍTICO'; }); $altos =
array_filter($mantenimientos_pendientes, function($item) { return $item['prioridad'] == 'ALTO'; });

$pdf->SetFillColor(255, 236, 236); // Fondo rojo claro
$pdf->SetFont('helvetica', 'B', 11);
$pdf->Cell(60, 8, 'Mantenimientos CRÍTICOS:', 1, 0, 'L', true);
$pdf->SetFont('helvetica', '', 11);
$pdf->Cell(30, 8, count($criticos), 1, 0, 'C', true);

$pdf->SetFillColor(255, 248, 220); // Fondo amarillo claro
$pdf->SetFont('helvetica', 'B', 11);
$pdf->Cell(60, 8, 'Prioridad ALTA:', 1, 0, 'L', true);
$pdf->SetFont('helvetica', '', 11);
```

```
$pdf->Cell(30, 8, count($altos), 1, 1, 'C', true);

$pdf->Ln(10);

// Tabla de mantenimientos
$pdf->SetFont('helvetica', 'B', 9);
$pdf->SetFillColor(26, 45, 122);
$pdf->SetTextColor(255, 255, 255);

// Cabeceras de tabla
$pdf->Cell(15, 8, 'ID', 1, 0, 'C', true);
$pdf->Cell(45, 8, 'MÁQUINA', 1, 0, 'C', true);
$pdf->Cell(25, 8, 'TIPO', 1, 0, 'C', true);
$pdf->Cell(20, 8, 'HORAS', 1, 0, 'C', true);
$pdf->Cell(25, 8, 'DÍAS SIN', 1, 0, 'C', true);
$pdf->Cell(40, 8, 'TIPO MANTENIMIENTO', 1, 0, 'C', true);
$pdf->Cell(20, 8, 'PRIORIDAD', 1, 1, 'C', true);

// Contenido de tabla
$pdf->SetFont('helvetica', '', 8);
$pdf->SetTextColor(0, 0, 0);

foreach ($mantenimientos_pendientes as $item) {
 // Color de fondo según prioridad
```

```
 switch ($item['prioridad']) {
 case 'CRÍTICO':
 $pdf->SetFillColor(255, 205, 205);
 break;
 case 'ALTO':
 $pdf->SetFillColor(255, 235, 153);
 break;
 case 'MEDIO':
 $pdf->SetFillColor(255, 255, 204);
 break;
 default:
 $pdf->SetFillColor(240, 240, 240);
 }

 $pdf->Cell(15, 6, $item['id'], 1, 0, 'C', true);
 $pdf->Cell(45, 6, substr($item['maquina'], 0, 20), 1, 0, 'L', true);
 $pdf->Cell(25, 6, ucfirst($item['tipo']), 1, 0, 'C', true);
 $pdf->Cell(20, 6, number_format($item['horas_uso']), 1, 0, 'C', true);
 $pdf->Cell(25, 6, $item['dias_sin_mantenimiento'] . ' días', 1, 0, 'C', true);
 $pdf->Cell(40, 6, substr($item['estado_mantenimiento'], 0, 18), 1, 0, 'L', true);
 $pdf->Cell(20, 6, $item['prioridad'], 1, 1, 'C', true);
}

// Recomendaciones
```

```
$pdf->Ln(10);
$pdf->SetFont('helvetica', 'B', 12);
$pdf->SetTextColor(26, 45, 122);
$pdf->Cell(0, 8, 'RECOMENDACIONES:', 0, 1, 'L');

$pdf->SetFont('helvetica', '', 10);
$pdf->SetTextColor(0, 0, 0);

$recomendaciones = [
 '• Priorizar mantenimientos CRÍTICOS en las próximas 48 horas',
 '• Programar servicios de prioridad ALTA dentro de una semana',
 '• Verificar disponibilidad de repuestos para mantenimientos mayores',
 '• Contactar técnicos especializados para equipos con más de 500 horas de uso',
 '• Considerar alquiler temporal de equipos durante mantenimientos críticos'
];

foreach ($recomendaciones as $recomendacion) {
 $pdf->Cell(0, 6, $recomendacion, 0, 1, 'L');
 $pdf->Ln(1);
}

// Generar y descargar PDF
$filename = 'mantenimientos_pendientes_' . date('Y-m-d_H-i-s') . '.pdf';
$pdf->Output($filename, 'D'); // 'D' = Forzar descarga
```

?>

Reporte de Costos Operativos

```
<?php
// reportes/costos_operativos.php
require_once '../config/pdf_config.php';
require_once '../config/database.php';
require_once '../includes/auth.php';

verificar_sesion();

// Parámetros del reporte
$fecha_inicio = $_GET['fecha_inicio'] ?? date('Y-m-01'); // Primer día del mes actual
$fecha_fin = $_GET['fecha_fin'] ?? date('Y-m-t'); // Último día del mes actual

// Consulta principal de costos
$stmt = $pdo->prepare("
 SELECT
 m.id,
 CONCAT(m.marca, ' ', m.modelo) as maquina,
 m.tipo,

 -- Costos de combustible
```

```
COALESCE(SUM(c.costo_total), 0) as costo_combustible,
COALESCE(SUM(c.cantidad_litros), 0) as litros_consumidos,

-- Costos de mantenimiento
COALESCE(SUM(man.costo), 0) as costo_mantenimiento,
COUNT(man.id) as mantenimientos_realizados,

-- Cálculos de rendimiento
CASE
WHEN SUM(c.horas_uso_fin - c.horas_uso_inicio) > 0
THEN SUM(c.cantidad_litros) / SUM(c.horas_uso_fin - c.horas_uso_inicio)
ELSE NULL
END as rendimiento_promedio,

-- Horas trabajadas
COALESCE(SUM(c.horas_uso_fin - c.horas_uso_inicio), 0) as horas_trabajadas,

-- Costo total
(COALESCE(SUM(c.costo_total), 0) + COALESCE(SUM(man.costo), 0)) as costo_total

FROM maquinaria m
LEFT JOIN combustible c ON m.id = c.maquinaria_id
AND c.fecha BETWEEN ? AND ?
AND c.horas_uso_fin IS NOT NULL
```

```
 LEFT JOIN mantenimientos man ON m.id = man.maquinaria_id
 AND man.fecha_realizado BETWEEN ? AND ?
 WHERE m.estado != 'baja_tecnica'
 GROUP BY m.id
 HAVING costo_total > 0
 ORDER BY costo_total DESC
");

$stmt->execute([$fecha_inicio, $fecha_fin, $fecha_inicio, $fecha_fin]);
$costos_maquinas = $stmt->fetchAll();

// Estadísticas generales
$total_combustible = array_sum(array_column($costos_maquinas, 'costo_combustible'));
$total_mantenimiento = array_sum(array_column($costos_maquinas, 'costo_mantenimiento')); $total_general = $total_combustible +
$total_mantenimiento;
$total_horas = array_sum(array_column($costos_maquinas, 'horas_trabajadas'));

// Crear PDF en formato horizontal (landscape) para más espacio
$pdf = new PDFGenerator('L', 'mm', 'A4');
$pdf->SetCreator('Sistema de Maquinaria Agrícola');
$pdf->SetAuthor($_SESSION['usuario_nombre']);
$pdf->SetTitle('Reporte de Costos Operativos');
$pdf->SetSubject('Análisis de Costos por Período');
```

```
$pdf->AddPage();

// Título del reporte
$pdf->SetFont('helvetica', 'B', 16);
$pdf->SetTextColor(26, 45, 122);
$pdf->Cell(0, 10, 'REPORTE DE COSTOS OPERATIVOS', 0, 1, 'C');
$pdf->Ln(3);

// Período del reporte
$pdf->SetFont('helvetica', 'B', 12);
$pdf->Cell(0, 8, 'Período: ' . date('d/m/Y', strtotime($fecha_inicio)) . ' - ' . date('d/m/Y', strtotime($fecha_fin)), 0, 1, 'C'); $pdf->Ln(8);

// Resumen ejecutivo en cajas
$pdf->SetFont('helvetica', 'B', 11);

// Primera fila de estadísticas
$box_width = 70;
$box_height = 20;

// Caja 1: Costo Total
$pdf->SetFillColor(26, 45, 122);
$pdf->SetTextColor(255, 255, 255);
$pdf->Rect($pdf->GetX(), $pdf->GetY(), $box_width, $box_height, 'F');
```

```
$pdf->Cell($box_width, 8, 'COSTO TOTAL', 0, 0, 'C');
$pdf->Ln(6);
$pdf->SetFont('helvetica', 'B', 16);
$pdf->Cell($box_width, 8, '$' . number_format($total_general, 2), 0, 0, 'C');

// Caja 2: Combustible
$pdf->SetXY($pdf->GetX() + 5, $pdf->GetY() - 14);
$pdf->SetFillColor(220, 53, 69);
$pdf->SetTextColor(255, 255, 255);
$pdf->SetFont('helvetica', 'B', 11);
$pdf->Rect($pdf->GetX(), $pdf->GetY(), $box_width, $box_height, 'F');
$pdf->Cell($box_width, 8, 'COMBUSTIBLE', 0, 0, 'C');
$pdf->Ln(6);
$pdf->SetFont('helvetica', 'B', 14);
$pdf->Cell($box_width, 8, '$' . number_format($total_combustible, 2), 0, 0, 'C');

// Caja 3: Mantenimiento
$pdf->SetXY($pdf->GetX() + 5, $pdf->GetY() - 14);
$pdf->SetFillColor(255, 193, 7);
$pdf->SetTextColor(0, 0, 0);
$pdf->SetFont('helvetica', 'B', 11);
$pdf->Rect($pdf->GetX(), $pdf->GetY(), $box_width, $box_height, 'F');
$pdf->Cell($box_width, 8, 'MANTENIMIENTO', 0, 0, 'C');
$pdf->Ln(6);
```

```
$pdf->SetFont('helvetica', 'B', 14);
$pdf->Cell($box_width, 8, '$' . number_format($total_mantenimiento, 2), 0, 0, 'C');

// Caja 4: Horas Trabajadas
$pdf->SetXY($pdf->GetX() + 5, $pdf->GetY() - 14);
$pdf->SetFillColor(40, 167, 69);
$pdf->SetTextColor(255, 255, 255);
$pdf->SetFont('helvetica', 'B', 11);
$pdf->Rect($pdf->GetX(), $pdf->GetY(), $box_width, $box_height, 'F');
$pdf->Cell($box_width, 8, 'HORAS TRABAJADAS', 0, 0, 'C');
$pdf->Ln(6);
$pdf->SetFont('helvetica', 'B', 14);
$pdf->Cell($box_width, 8, number_format($total_horas) . ' hrs', 0, 1, 'C');

$pdf->Ln(15);

// Tabla detallada
$pdf->SetFont('helvetica', 'B', 9);
$pdf->SetFillColor(26, 45, 122);
$pdf->SetTextColor(255, 255, 255);

// Cabeceras de tabla (ajustadas para formato horizontal)
$pdf->Cell(20, 8, 'ID', 1, 0, 'C', true);
$pdf->Cell(50, 8, 'MÁQUINA', 1, 0, 'C', true);
```

```
$pdf->Cell(25, 8, 'TIPO', 1, 0, 'C', true);
$pdf->Cell(25, 8, 'HORAS TRAB.', 1, 0, 'C', true);
$pdf->Cell(30, 8, 'COMBUSTIBLE', 1, 0, 'C', true);
$pdf->Cell(20, 8, 'LITROS', 1, 0, 'C', true);
$pdf->Cell(30, 8, 'MANTENIMIENTO', 1, 0, 'C', true);
$pdf->Cell(20, 8, 'SERVICIOS', 1, 0, 'C', true);
$pdf->Cell(30, 8, 'COSTO TOTAL', 1, 0, 'C', true);
$pdf->Cell(25, 8, 'COSTO/HORA', 1, 1, 'C', true);

// Contenido de tabla
$pdf->SetFont('helvetica', '', 8);
$pdf->SetTextColor(0, 0, 0);
$pdf->SetFillColor(248, 249, 250);

foreach ($costos_maquinas as $index => $maquina) {
 $fill = ($index % 2 == 0);
 $costo_por_hora = $maquina['horas_trabajadas'] > 0 ? $maquina['costo_total'] / $maquina['horas_trabajadas'] : 0;

 $pdf->Cell(20, 6, $maquina['id'], 1, 0, 'C', $fill);
 $pdf->Cell(50, 6, substr($maquina['maquina'], 0, 22), 1, 0, 'L', $fill);
 $pdf->Cell(25, 6, ucfirst($maquina['tipo']), 1, 0, 'C', $fill);
 $pdf->Cell(25, 6, number_format($maquina['horas_trabajadas']), 1, 0, 'C', $fill);
 $pdf->Cell(30, 6, '$' . number_format($maquina['costo_combustible'], 2), 1, 0, 'R', $fill);
 $pdf->Cell(20, 6, number_format($maquina['litros_consumidos']), 1, 0, 'C', $fill);
```

```
 $pdf->Cell(30, 6, '$' . number_format($maquina['costo_mantenimiento'], 2), 1, 0, 'R', $fill);  $pdf->Cell(20, 6,
$maquina['mantenimientos_realizados'], 1, 0, 'C', $fill);
 $pdf->Cell(30, 6, '$' . number_format($maquina['costo_total'], 2), 1, 0, 'R', $fill);
 $pdf->Cell(25, 6, '$' . number_format($costo_por_hora, 2), 1, 1, 'R', $fill);
}

// Fila de totales
$pdf->SetFont('helvetica', 'B', 9);
$pdf->SetFillColor(26, 45, 122);
$pdf->SetTextColor(255, 255, 255);
$pdf->Cell(120, 8, 'TOTALES:', 1, 0, 'R', true);
$pdf->Cell(30, 8, '$' . number_format($total_combustible, 2), 1, 0, 'R', true);
$pdf->Cell(20, 8, '', 1, 0, 'C', true);
$pdf->Cell(30, 8, '$' . number_format($total_mantenimiento, 2), 1, 0, 'R', true);
$pdf->Cell(20, 8, '', 1, 0, 'C', true);
$pdf->Cell(30, 8, '$' . number_format($total_general, 2), 1, 0, 'R', true);
$pdf->Cell(25, 8, '$' . number_format($total_horas > 0 ? $total_general / $total_horas : 0, 2), 1, 1, 'R', true);

// Análisis y observaciones
$pdf->Ln(10);
$pdf->SetFont('helvetica', 'B', 12);
$pdf->SetTextColor(26, 45, 122);
$pdf->Cell(0, 8, 'ANÁLISIS Y OBSERVACIONES:', 0, 1, 'L');
```

```
$pdf->SetFont('helvetica', '', 10);
$pdf->SetTextColor(0, 0, 0);

// Calcular máquina más costosa
$maquina_costosa = $costos_maquinas[0] ?? null;
$porcentaje_combustible = $total_general > 0 ? ($total_combustible / $total_general) * 100 : 0;

$observaciones = [
 "• La máquina con mayor costo operativo es: " . ($maquina_costosa['maquina'] ?? 'N/A') .   " con $" .
number_format($maquina_costosa['costo_total'] ?? 0, 2),
 "• El combustible representa el " . number_format($porcentaje_combustible, 1) . "% del costo total",  "• Costo promedio por hora de operación:
$" . number_format($total_horas > 0 ? $total_general / $total_horas : 0, 2),  "• Total de máquinas operativas en el período: " .
count($costos_maquinas),
 "• Se recomienda optimizar el consumo de combustible para reducir costos operativos"
];

foreach ($observaciones as $observacion) {
 $pdf->Cell(0, 6, $observacion, 0, 1, 'L');
 $pdf->Ln(1);
}

// Generar y descargar PDF
$filename = 'costos_operativos_' . $fecha_inicio . '_' . $fecha_fin . '.pdf';
```

```
$pdf->Output($filename, 'D');
?>
```

Ejercicios Guiados - Práctica Paso a Paso Ejercicio 1: Crear

Usuario Administrador

Objetivo: Registrar el primer usuario que tendrá acceso completo al sistema

01

Crear el Archivo

Crea crear_admin.php en la carpeta raíz del proyecto

<?php
02

Código Base

Copia el código de inserción con

password_hash()
03

Ejecutar Una Vez

Ejecuta el archivo y luego elimínalo por seguridad

```
// crear_admin.php - EJECUTAR SOLO UNA VEZ Y ELIMINAR
require_once 'config/database.php';

$nombre = 'Administrador';
$email = 'admin@empresa.com';
$password = 'admin123'; // CAMBIAR POR CONTRASEÑA SEGURA
$rol = 'administrador';
```

```
$password_hash = password_hash($password, PASSWORD_DEFAULT);

$stmt = $pdo->prepare("INSERT INTO usuarios (nombre, email, password, rol) VALUES (?, ?, ?,
?)"); $resultado = $stmt->execute([$nombre, $email, $password_hash, $rol]);

if ($resultado) {
 echo "
```

Ejercicio 2: Sistema de Navegación

Objetivo: Crear un menú de navegación consistente en todas las

páginas **Pistas:**

- Crea un archivo includes/navbar.php que incluyas en cada página
- Usa Bootstrap para el diseño responsive
- Incluye el nombre del usuario logueado
- Agrega enlaces a todas las secciones principales

Solución:

```
<!-- includes/navbar.php -->
<nav class="navbar navbar-expand-lg navbar-dark bg-primary">
```

<a class="navbar-brand" href="/dashboard.php">

Ejercicio 3: Validación de Formularios

Objetivo: Implementar validación tanto en cliente (JavaScript) como en servidor (PHP) **Enunciado:** Crea un sistema de validación que

verifique:

Campos obligatorios no estén vacíos

Números de serie únicos en la base de datos

Formatos de email válidos

Rangos de números (año entre 1990-2024)

Solución - Validación JavaScript:

```
// assets/js/validaciones.js
class ValidadorFormularios {

 static validarMaquinaria(formulario) {
 const errores = [];

 // Validar campos obligatorios
 const camposRequeridos = ['tipo', 'marca', 'modelo', 'año', 'numero_serie'];
```

```
camposRequeridos.forEach(campo => {
const elemento = formulario.querySelector(`[name="${campo}"]`);
if (!elemento.value.trim()) {
errores.push(`El campo ${campo} es obligatorio`);
elemento.classList.add('is-invalid');
} else {
elemento.classList.remove('is-invalid');
elemento.classList.add('is-valid');
}
});

// Validar año
const año = parseInt(formulario.querySelector('[name="año"]').value);
if (año < 1990 || año > new Date().getFullYear()) {
errores.push('El año debe estar entre 1990 y ' + new Date().getFullYear());
}

// Validar horas de uso
const horas = parseInt(formulario.querySelector('[name="horas_uso"]').value);
if (horas < 0 || horas > 50000) {
errores.push('Las horas de uso deben estar entre 0 y 50,000');
}

return errores;
```

```
}

static async validarNumeroSerie(numeroSerie, idMaquina = null) {
try {
const response = await fetch('/api/validar_serie.php', {
method: 'POST',
headers: {
'Content-Type': 'application/json',
},
body: JSON.stringify({
numero_serie: numeroSerie,
id_maquina: idMaquina
})
});

const resultado = await response.json();
return resultado.valido;

} catch (error) {
console.error('Error al validar número de serie:', error);
return false;
}
}
```

```
 static mostrarErrores(errores, contenedor) {
 if (errores.length > 0) {
 const html = `
 <div class="alert alert-danger">
 <strong>
```

Ejercicio 4: API de Validación del Servidor

Solución - Validación PHP:

```
<?php
// api/validar_serie.php
header('Content-Type: application/json');
require_once '../config/database.php';
require_once '../includes/auth.php';

// Verificar que sea petición POST
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
 http_response_code(405);
 echo json_encode(['error' => 'Método no permitido']);
 exit;
}
```

```
// Obtener datos JSON
$input = json_decode(file_get_contents('php://input'), true);
$numero_serie = $input['numero_serie'] ?? '';
$id_maquina = $input['id_maquina'] ?? null;

if (empty($numero_serie)) {
 echo json_encode(['valido' => false, 'mensaje' => 'Número de serie requerido']);
 exit;
}

try {
 // Verificar si el número de serie ya existe
 if ($id_maquina) {
 // Para edición: verificar que no exista en otra máquina
 $stmt = $pdo->prepare("
 SELECT COUNT(*) FROM maquinaria
 WHERE numero_serie = ? AND id != ?
 ");
 $stmt->execute([$numero_serie, $id_maquina]);
 } else {
 // Para creación: verificar que no exista en absoluto
 $stmt = $pdo->prepare("
 SELECT COUNT(*) FROM maquinaria
```

```
 WHERE numero_serie = ?
 ");
 $stmt->execute([$numero_serie]);
 }

 $existe = $stmt->fetchColumn() > 0;

 echo json_encode([
 'valido' => !$existe,
 'mensaje' => $existe ? 'Número de serie ya existe' : 'Número de serie disponible'
 ]);

} catch (PDOException $e) {
 http_response_code(500);
 echo json_encode([
 'valido' => false,
 'mensaje' => 'Error del servidor: ' . $e->getMessage()
 ]);
}
?>
```

Salida Esperada: JSON con {valido: true/false, mensaje: "descripción"}

Ejercicio 5: Sistema de Búsqueda Avanzada

Objetivo: Implementar búsqueda con filtros múltiples y resultados en tiempo real

Enunciado: Crear un sistema que permita buscar máquinas por:

- Texto libre (marca, modelo, número de serie)
- Tipo de máquina (dropdown)
- Estado (disponible, alquilado, etc.)
- Rango de años
- Rango de horas de uso

Solución - Interfaz de Búsqueda:

```
<!-- maquinaria/buscar.php -->
<div class="card mb-4">
 <div class="card-header">
 <h5>
```

Mini-Proyecto: Dashboard en Tiempo Real

Objetivo: Crear un panel de control que se actualice automáticamente cada 30 segundos con estadísticas del sistema **Criterios de**

Éxito:

- Se actualiza automáticamente sin recargar la página
- Muestra estadísticas en tarjetas coloridas
- Incluye gráficos interactivos con Chart.js
- Tiene indicador de estado de conexión
- Funciona en modo offline con datos cacheados

Paso 1: API de Estadísticas 1

Crea api/estadisticas.php que devuelva JSON con datos del sistema

2 Paso 2: Función JavaScript

Programa actualizaciones automáticas con setInterval()

Paso 3: Gráficos Dinámicos 3

Integra Chart.js para visualizaciones interactivas

4 Paso 4: Indicadores Visuales

Agrega iconos de estado y animaciones CSS

Solución del Mini-Proyecto

```php
<?php
// api/estadisticas.php
header('Content-Type: application/json');
require_once '../config/database.php';

try {
 // Estadísticas generales
 $stats = [];

 // Total de máquinas por estado
 $stmt = $pdo->query("
 SELECT estado, COUNT(*) as cantidad
 FROM maquinaria
 GROUP BY estado
 ");
 $stats['maquinas_por_estado'] = $stmt->fetchAll(PDO::FETCH_KEY_PAIR);

 // Alertas de mantenimiento
 $stmt = $pdo->query("
 SELECT COUNT(*) as alertas
 FROM maquinaria m
 LEFT JOIN mantenimientos man ON m.id = man.maquinaria_id
```

```
WHERE DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro)) >= 90  OR m.horas_uso % 250 = 0 AND m.horas_uso > 0
GROUP BY m.id
");
$stats['alertas_mantenimiento'] = $stmt->rowCount();

// Consumo combustible último mes
$stmt = $pdo->query("
SELECT
SUM(cantidad_litros) as litros_mes,
SUM(costo_total) as costo_mes,
COUNT(DISTINCT maquinaria_id) as maquinas_activas
FROM combustible
WHERE fecha >= DATE_SUB(CURDATE(), INTERVAL 1 MONTH)
");
$stats['combustible_mes'] = $stmt->fetch();

// Tendencia mensual (últimos 6 meses)
$stmt = $pdo->query("
SELECT
DATE_FORMAT(fecha, '%Y-%m') as mes,
SUM(costo_total) as costo_total
FROM combustible
WHERE fecha >= DATE_SUB(CURDATE(), INTERVAL 6 MONTH)
GROUP BY DATE_FORMAT(fecha, '%Y-%m')
```

```
ORDER BY mes
");
$stats['tendencia_costos'] = $stmt->fetchAll();

// Top 5 máquinas más costosas
$stmt = $pdo->query("
SELECT
CONCAT(m.marca, ' ', m.modelo) as maquina,
SUM(c.costo_total) as costo_total
FROM maquinaria m
JOIN combustible c ON m.id = c.maquinaria_id
WHERE c.fecha >= DATE_SUB(CURDATE(), INTERVAL 1 MONTH)
GROUP BY m.id
ORDER BY costo_total DESC
LIMIT 5
");
$stats['top_maquinas_costosas'] = $stmt->fetchAll();

// Timestamp de actualización
$stats['ultima_actualizacion'] = date('Y-m-d H:i:s');
$stats['exito'] = true;

echo json_encode($stats);
```

```
} catch (Exception $e) {
 http_response_code(500);
 echo json_encode([
 'exito' => false,
 'error' => $e->getMessage(),
 'ultima_actualizacion' => date('Y-m-d H:i:s')
 ]);
}
?>


// assets/js/dashboard-realtime.js
class DashboardTiempoReal {
 constructor() {
 this.intervalo = null;
 this.chartCostos = null;
 this.chartMaquinas = null;
 this.inicializar();
 }

 inicializar() {
 // Cargar datos iniciales
 this.actualizarEstadisticas();
```

```
// Configurar actualización automática cada 30 segundos
this.intervalo = setInterval(() => {
this.actualizarEstadisticas();
}, 30000);

// Mostrar indicador de estado
this.mostrarEstadoConexion('conectado');
}

async actualizarEstadisticas() {
try {
const response = await fetch('/api/estadisticas.php');
if (!response.ok) throw new Error('Error de red');

const data = await response.json();

if (data.exito) {
this.actualizarTarjetas(data);
this.actualizarGraficos(data);
this.mostrarUltimaActualizacion(data.ultima_actualizacion);
this.mostrarEstadoConexion('conectado');
}
```

```
} catch (error) {
console.error('Error al actualizar estadísticas:', error);
this.mostrarEstadoConexion('error');

// Intentar cargar datos del cache offline
this.cargarDatosOffline();
}
}

actualizarTarjetas(data) {
// Total máquinas
const totalMaquinas = Object.values(data.maquinas_por_estado || {})
.reduce((sum, val) => sum + parseInt(val), 0);
this.actualizarTarjeta('total-maquinas', totalMaquinas, '
```

Errores Frecuentes y Cómo Evitarlos

Error: Headers already sent

Causa: Espacios o texto antes de <?php

Solución: Asegúrate que

Error: undefined index

Causa: Acceder a $_POST['campo'] sin verificar si existe

Solución: Usa isset($_POST['campo']) o $_POST['campo'] ?? null

Error: Sincronización infinita

Causa: No marcar registros como sincronizados

Solución: Actualiza flag sincronizado = 1 después de enviar

Error: PDF se corrompe

Causa: Output HTML antes de generar PDF

Solución: No hagas echo antes de $pdf->Output()

Checklist Final del

Error: SQLite locked

Causa: Transacciones abiertas sin cerrar

Solución: Usa try-catch y siempre cierra conexiones

Error: PWA no se instala

Causa: manifest.json mal configurado o HTTPS requerido **Solución:** Valida manifest y usa HTTPS en producción

Error: Session no funciona

Causa: session_start() después de output

Solución: Llama session_start() antes de cualquier HTML

Proyecto Marca cada elemento cuando lo hayas

completado exitosamente:

01

Base de datos MySQL creada con todas las tablas Sistema de login funcional con sesiones

CRUD completo de maquinaria (crear, leer, actualizar, eliminar)

03

Sistema offline con SQLite y sincronización

PWA instalable que funciona sin internet

Generación de reportes PDF profesionales

02

Módulo de mantenimientos con alertas automáticas Control de combustible con cálculo de rendimiento Dashboard con estadísticas en tiempo real

04

Validaciones tanto en cliente como servidor Interfaz responsive compatible con tablets

Documentación técnica y manual de usuario completos

Glosario de Términos Técnicos

API (Interfaz de Programación)

Conjunto de reglas que permite que diferentes programas se comuniquen entre sí

JSON (Notación de Objetos JavaScript) Formato ligero para intercambiar datos entre aplicaciones

PWA (Aplicación Web Progresiva)

Aplicación web que se comporta como una app nativa del celular

SQLite

Base de datos ligera que se almacena en un solo archivo

Hash (Cifrado)

Proceso de convertir texto legible en código irreversible

Chart.js

Librería JavaScript para crear gráficos interactivos en páginas web

Responsive Design

Diseño web que se adapta automáticamente a diferentes tamaños de pantalla

Manifest.json

Archivo de configuración que define cómo se comporta una PWA

Prepared Statement

Consulta SQL preparada que previene inyecciones de código malicioso

Timestamp

Marca de tiempo que indica cuándo ocurrió un evento específico

CRUD (Crear, Leer, Actualizar, Borrar)

Las cuatro operaciones básicas que se pueden realizar con datos en una base

PDO (Objetos de Datos PHP)

Interfaz uniforme para acceder a bases de datos desde PHP

Service Worker

Script que funciona en segundo plano y permite funcionalidad offline

Session (Sesión)

Mecanismo para mantener información del usuario entre páginas

Bootstrap

Framework CSS que facilita crear interfaces responsive

TCPDF

Librería PHP para generar documentos PDF de forma programática

AJAX (JavaScript Asíncrono)

Técnica para actualizar partes de página sin recargar completamente

Foreign Key (Clave Foránea)

Campo que conecta dos tablas en una base de datos

Cache (Caché)

Almacenamiento temporal de datos para acelerar futuras solicitudes

Debug (Depuración)

Proceso de encontrar y corregir errores en el código de programación

Mini-Quiz de Conocimientos

Evalúa tu comprensión del proyecto con estas preguntas:

1. ¿Cuál es la principal ventaja de usar PDO en lugar de mysqli?

a) Es más rápido
b) Funciona con múltiples tipos de bases de datos
c) Consume menos memoria
d) Es más fácil de escribir

2. ¿Qué función PHP se debe usar para encriptar contraseñas de forma segura?

a) md5()
b) sha1()
c) password_hash()
d) encrypt()

3. ¿Cuándo se debe ejecutar session_start() en PHP?

a) Al final del archivo
b) Antes de cualquier salida HTML
c) Solo cuando necesites la sesión
d) Después de verificar el login

4. ¿Qué archivo permite que una aplicación web funcione como PWA?

a) config.json
b) app.json
c) manifest.json
d) pwa.json

5. ¿Qué tipo de base de datos es mejor para modo offline?

a) MySQL
b) PostgreSQL
c) SQLite

d) Oracle

Respuestas del Quiz

Pregunta 1

Respuesta: b) Funciona con múltiples tipos de bases de datos

PDO es una capa de abstracción que permite cambiar de MySQL a PostgreSQL o SQLite sin modificar el código.

Respuesta: c) manifest.json

Pregunta 2

Respuesta: c) password_hash()

md5() y sha1() son vulnerables. password_hash() usa algoritmos seguros como bcrypt automáticamente.

Respuesta: c) SQLite

Pregunta 3

Respuesta: b) Antes de cualquier salida HTML

session_start() debe ejecutarse antes de echo, print, o cualquier HTML para evitar el error "headers already sent".

Pregunta 4

El archivo manifest.json define nombre, iconos, colores y comportamiento de la PWA cuando se instala.

Pregunta 5

SQLite se almacena en un archivo local y no requiere servidor, perfecto para funcionalidad offline.

Puntuación:

5 correctas: ¡Excelente! Dominas los conceptos 4 correctas: Muy bien, solo repasa algunos detalles 3 correctas: Buen trabajo, revisa los temas fallados 2 o menos: Necesitas repasar más el material

Preparación para la Sustentación

Estructura Recomendada de la Presentación

1. Introducción (3 min) 1

- Presentación personal y del proyecto
- Problemática que resuelve el sistema
- Tecnologías utilizadas

2 2. Demostración (10 min)

- Login y navegación del sistema
- CRUD de maquinaria en vivo sincronización
- Generación de reportes PDF

3. Aspectos Técnicos (5 min) 3

- Arquitectura del sistema
- Decisiones de diseño justificadas
- Funcionalidad offline y

- Retos superados durante el desarrollo

4 4. Conclusiones (2 min)

- Objetivos alcanzados

- Aprendizajes obtenidos
- Mejoras futuras propuestas

Preguntas Frecuentes de Sustentación

Prepárate para estas preguntas que suele hacer el **Instructor Iván**

Malavé Fierro:

"¿Por qué elegiste SQLite para modo offline en lugar de almacenamiento local del navegador?"

Respuesta: SQLite ofrece consultas SQL complejas, relaciones entre tablas y mejor rendimiento para grandes volúmenes de datos, mientras que localStorage solo almacena texto plano.

"Explica cómo funciona el sistema de alertas automáticas de mantenimiento."

Respuesta: El sistema calcula automáticamente fechas basándose en horas de uso y tiempo transcurrido. Una consulta SQL identifica máquinas que superan umbrales y genera alertas priorizadas por criticidad.

"¿Cómo manejas conflictos cuando dos tablets modifican la misma máquina offline?"

Respuesta: Implementé timestamps de modificación. El sistema detecta conflictos y permite al usuario resolver manualmente qué versión conservar, mostrando ambas versiones lado a lado.

"¿Qué medidas de seguridad implementaste?"

Respuesta: Contraseñas hasheadas con password_hash(), prepared statements contra SQL injection, validación de

sesiones en cada página, y sanitización de inputs del usuario.

Documentación Técnica - Estructura

Tu manual técnico debe incluir estas secciones para obtener máxima puntuación:

1. **Introducción y Objetivos**
   - Descripción del sistema desarrollado
   - Objetivos funcionales y técnicos
   - Alcance del proyecto

2. **Arquitectura del Sistema**
   - Diagrama de arquitectura general
   - Tecnologías utilizadas y justificación
   - Patrón de diseño implementado (MVC, etc.)

3. **Base de Datos**
   - Modelo entidad-relación

▪ Scripts de creación de tablas

▪ Diccionario de datos completo

Módulos y Funcionalidades

▪ Descripción detallada de cada módulo **4**
▪ Diagramas de flujo de procesos críticos ▪ APIs y endpoints documentados

Instalación y Configuración

▪ Requisitos del sistema
5
▪ Pasos de instalación detallados

▪ Configuración de servidor y base de datos

Pruebas y Validación

▪ Casos de prueba ejecutados
6
▪ Resultados de pruebas de rendimiento ▪ Validación de requerimientos funcionales

Manual de Usuario - Guía Práctica El

manual de usuario debe ser comprensible para operadores de campo sin conocimientos técnicos:

Acceso al Sistema

Paso a paso para iniciar sesión, qué hacer si olvida contraseña, y cómo cambiar datos de perfil. Incluye capturas de pantalla de cada paso.

Gestión de Maquinaria

Cómo registrar nuevas máquinas, actualizar horas de uso, cambiar estados (disponible/alquilado), y consultar historial completo.

Control de Mantenimientos

Cómo interpretar alertas, programar servicios, registrar mantenimientos realizados, y generar reportes de cumplimiento.

Registro de Combustible

Proceso para registrar cargas de combustible en campo, cálculo automático de rendimiento, y análisis de consumo anormal.

Uso Offline

Cómo trabajar sin internet, qué datos se guardan localmente, proceso de sincronización automática, y resolución de conflictos.

Generación de Reportes

Tipos de reportes disponibles, filtros de fecha y máquina, interpretación de gráficos, y exportación a PDF para gerencia.

Mejoras Futuras Sugeridas

Demuestra visión técnica proponiendo mejoras realistas para versiones futuras:

Integración GPS

Tracking en tiempo real de ubicación de máquinas con geofencing para alertas cuando salen de zona autorizada

Sensores IoT

Monitoreo automático de horas de uso, temperatura del motor, y nivel de combustible mediante sensores conectados

Inteligencia Artificial

Predicción de fallas mediante machine learning basado en patrones históricos de mantenimiento y uso

App Nativa

Desarrollo de aplicación nativa para Android/iOS con mejor integración de cámara y notificaciones push

Blockchain

Registro inmutable de mantenimientos y ownership para certificación y transferencia de máquinas usadas

Integración ERP

API REST para conectar con sistemas contables y de gestión empresarial existentes

Consejos para el Código Fuente

Asegúrate que tu código cumpla con estándares profesionales:

1 Estructura de Carpetas Clara

Organiza archivos en carpetas lógicas: /includes, /config, /assets, /modules. Usa nombres descriptivos en inglés para archivos y funciones principales.

2 Comentarios Descriptivos

Documenta cada función con propósito, parámetros esperados y valor de retorno. Explica lógica compleja y decisiones de diseño importantes.

3 Constantes de Configuración

Define configuraciones en archivo separado (config.php): URLs, credenciales de DB, limites de paginación, rutas de archivos.

4 Manejo de Errores

Implementa try-catch en operaciones críticas. Registra errores en logs. Muestra mensajes amigables al usuario final.

5 Validación Consistente

Crea funciones reutilizables para validaciones comunes. Sanitiza todos los inputs del usuario. Usa whitelist en lugar de blacklist.

TIP: Incluye un archivo README.md con instrucciones de instalación y descripción breve del proyecto. Los evaluadores lo valorarán positivamente.

Criterios de Evaluación Detallados

Conoce exactamente cómo te evaluará el **Instructor Iván Malavé Fierro**:

Funcionalidad del...

Características Avanzadas

Código Fuente

Manual Técnico

Manual de Usuario

Pruebas de Calidad

Proyecto de Investigación

Presentación y...

0 8 16 24

Para obtener la máxima calificación, enfócate especialmente en funcionalidad completa y características avanzadas, que suman 35 de los 100 puntos totales.

Recursos Adicionales y Referencias

Enlaces útiles para profundizar en los temas del proyecto:

Documentación Oficial

PHP Manual: php.net/manual

PDO Documentation: php.net/pdo

SQLite Docs: sqlite.org/docs.html

Bootstrap Components: getbootstrap.com

Librerías y Herramientas

Chart.js: chartjs.org

TCPDF: tcpdf.org

PWA Builder: pwabuilder.com

Postman API Testing: postman.com

Herramientas de Desarrollo

Visual Studio Code: code.visualstudio.com

XAMPP: apachefriends.org

HeidiSQL: heidisql.com

Git: git-scm.com

Tutoriales Complementarios

W3Schools PHP: w3schools.com/php

MDN Web Docs: developer.mozilla.org

PHP The Right Way: phptherightway.com Codecademy: codecademy.com

Recomendación del Instructor: Mantén estos recursos marcados durante el desarrollo del proyecto. La documentación oficial siempre es la fuente más confiable para resolver dudas técnicas.

¡Felicitaciones, Futuro Tecnólogo!

¡Lo Lograste!

Has completado la guía más completa para desarrollar tu proyecto de **Sistema de Gestión de Maquinaria Agrícola** con PHP. Ahora posees todos los conocimientos necesarios para:

Desarrollar Sistemas Complejos

Crear aplicaciones web profesionales que resuelven problemas reales del sector agrícola

Crear PWAs Avanzadas

Desarrollar aplicaciones que funcionan sin internet y se instalan como apps nativas

Manejar Bases de Datos

Diseñar, implementar y optimizar bases de datos tanto online como offline

Documentar y Presentar

Comunicar efectivamente tu trabajo técnico a diferentes audiencias

"El conocimiento que has adquirido hoy es la base para construir el futuro tecnológico del país. Como futuro tecnólogo del SENA, llevas contigo la responsabilidad y el orgullo de transformar vidas a través de la tecnología."

– Instructor Iván Malavé Fierro
SENA Mosquera – CBA

¡Éxitos en tu sustentación y en tu carrera como Tecnólogo en Análisis y Desarrollo de Software!

#FuturoTecnólogo #SENACBA #InnovaciónAgrícola

Funciones PHP Avanzadas para Gestión de Maquinaria

1. Función para Calcular Próximo Mantenimiento

```
function calcularProximoMantenimiento($horasActuales, $ultimoMantenimiento) {
 // Mantenimiento cada 250 horas
 $intervaloHoras = 250;
 $horasDesdeUltimo = $horasActuales - $ultimoMantenimiento;
 $horasRestantes = $intervaloHoras - ($horasDesdeUltimo % $intervaloHoras);
```

```
 return [
 'horas_restantes' => $horasRestantes,
 'urgencia' => $horasRestantes <= 25 ? 'URGENTE' : 'NORMAL',
 'fecha_estimada' => date('Y-m-d', strtotime('+' . ($horasRestantes * 8) . ' hours'))
 ];
}

// Ejemplo de uso:
$resultado = calcularProximoMantenimiento(1240, 1000);
echo "Faltan {$resultado['horas_restantes']} horas para mantenimiento";
```

2. Función para Validar Número de Serie

```
function validarNumeroSerie($serie, $tipo) {
 // Patrones por tipo de máquina
 $patrones = [
 'tractor' => '/^TR[0-9]{6}$/', // TR123456
 'cosechadora' => '/^CS[0-9]{6}$/', // CS123456
 'sembradora' => '/^SM[0-9]{6}$/' // SM123456
 ];

 return isset($patrones[$tipo]) && preg_match($patrones[$tipo], $serie);
```

}

Sistema de Notificaciones Push en Tiempo Real

¿Qué es?

Un sistema que envía alertas automáticas cuando ocurren eventos importantes: mantenimiento vencido, combustible bajo, o máquina fuera de zona autorizada.

Implementación Paso a Paso:

1. Configurar Service Worker para Notificaciones

```
// sw-notifications.js
self.addEventListener('push', function(event) {
 const options = {
 body: event.data.text(),
 icon: '/assets/icons/tractor-icon.png',
 badge: '/assets/icons/badge.png',
 vibrate: [200, 100, 200],
 actions: [
 {action: 'ver', title: 'Ver Detalles'},
 {action: 'cerrar', title: 'Cerrar'}
```

```
 ]
 };

 event.waitUntil(
 self.registration.showNotification('Sistema Maquinaria', options)
 );
});
```

2. Función PHP para Enviar Notificaciones

```
function enviarNotificacionPush($mensaje, $tipo = 'info') {
 $payload = json_encode([
 'title' => 'Alerta de Maquinaria',
 'body' => $mensaje,
 'type' => $tipo,
 'timestamp' => time()
 ]);

 // Enviar a todos los dispositivos suscritos
 $subscripciones = obtenerSuscripciones();
 foreach ($subscripciones as $sub) {
 enviarPushADispositivo($sub, $payload);
 }
```

```
}

// Ejemplo de uso:
enviarNotificacionPush("Tractor TR123456 necesita mantenimiento urgente", "warning");
```

Sistema de Geolocalización y Tracking GPS

¿Para qué sirve?

Rastrear la ubicación exacta de cada máquina en tiempo real, crear rutas de trabajo, y recibir alertas si salen de zonas autorizadas.

Implementación Práctica:

1. Capturar Ubicación con JavaScript

```
// assets/js/gps-tracker.js
class GPSTracker {
 constructor(maquinaId) {
 this.maquinaId = maquinaId;
 this.watchId = null;
 this.ultimaUbicacion = null;
 }

 iniciarRastreo() {
 if (navigator.geolocation) {
 this.watchId = navigator.geolocation.watchPosition(
 (position) => this.actualizarUbicacion(position),
 (error) => this.manejarError(error),
 {
 enableHighAccuracy: true,
 timeout: 10000,
```

```
maximumAge: 60000
}
);
}
}

actualizarUbicacion(position) {
const ubicacion = {
latitud: position.coords.latitude,
longitud: position.coords.longitude,
precision: position.coords.accuracy,
timestamp: new Date().toISOString()
};

// Enviar al servidor
this.enviarUbicacionServidor(ubicacion);
}
}
```

2. Función PHP para Procesar Ubicaciones

```
function procesarUbicacionGPS($maquinaId, $latitud, $longitud, $precision) {
```

```
    global $pdo;

    // Guardar ubicación en base de datos
    $stmt = $pdo->prepare("
    INSERT INTO ubicaciones_gps
    (maquinaria_id, latitud, longitud, precision, fecha_registro)
    VALUES (?, ?, ?, ?, NOW())
    ");
    $stmt->execute([$maquinaId, $latitud, $longitud, $precision]);

    // Verificar si está en zona autorizada
    if (!estaEnZonaAutorizada($latitud, $longitud)) {
    enviarAlertaFueraDeZona($maquinaId, $latitud, $longitud);
    }

    return ['status' => 'success', 'ubicacion_guardada' => true];
}
```

Sistema de Backup Automático y Recuperación

¿Por qué es importante?

Proteger toda la información del sistema contra pérdidas por fallos del servidor, errores humanos o problemas técnicos.

Implementación Completa:

1. Script de Backup Automático

```
// scripts/backup_automatico.php
class BackupManager {
 private $pdo;
 private $backupPath;

 public function __construct($pdo) {
 $this->pdo = $pdo;
 $this->backupPath = __DIR__ . '/../backups/';

 // Crear directorio si no existe
 if (!is_dir($this->backupPath)) {
 mkdir($this->backupPath, 0755, true);
 }
 }

 public function crearBackupCompleto() {
 $fecha = date('Y-m-d_H-i-s');
 $nombreArchivo = "backup_maquinaria_{$fecha}.sql";
 $rutaCompleta = $this->backupPath . $nombreArchivo;
```

```
// Comando mysqldump
$comando = "mysqldump -h localhost -u root maquinaria_agricola > {$rutaCompleta}";
exec($comando, $output, $resultado);

if ($resultado === 0) {
// Comprimir el archivo
$this->comprimirBackup($rutaCompleta);

// Limpiar backups antiguos (mantener solo últimos 30 días)
$this->limpiarBackupsAntiguos();

return ['status' => 'success', 'archivo' => $nombreArchivo];
}

return ['status' => 'error', 'mensaje' => 'Error al crear backup'];
}

private function comprimirBackup($archivo) {
$zip = new ZipArchive();
$archivoZip = str_replace('.sql', '.zip', $archivo);

if ($zip->open($archivoZip, ZipArchive::CREATE) === TRUE) {
$zip->addFile($archivo, basename($archivo));
$zip->close();
```

```
 unlink($archivo); // Eliminar archivo SQL original
 }
 }
}
```

2. Programar Backups con Cron

```
# Agregar al crontab para backup diario a las 2:00 AM
0 2 * * * /usr/bin/php /ruta/proyecto/scripts/backup_automatico.php
```

Sistema de Logs y Auditoría Avanzado

¿Qué registra?

Todas las acciones importantes: quién hizo qué, cuándo, desde dónde, y qué datos cambió. Esencial para seguridad y resolución de

problemas. **Implementación Detallada:**

1. Clase para Manejo de Logs

```
// includes/logger.php
class AuditoriaLogger {
 private $pdo;
 private $usuarioId;
 private $ipAddress;

 public function __construct($pdo, $usuarioId = null) {
 $this->pdo = $pdo;
 $this->usuarioId = $usuarioId ?? $_SESSION['usuario_id'] ?? null;
 $this->ipAddress = $_SERVER['REMOTE_ADDR'] ?? 'unknown';
 }

 public function registrarAccion($accion, $tabla, $registroId, $datosAnteriores = null, $datosNuevos = null) {
 $stmt = $this->pdo->prepare("
 INSERT INTO auditoria_logs
 (usuario_id, accion, tabla_afectada, registro_id, datos_anteriores,
 datos_nuevos, ip_address, user_agent, fecha_accion)
 VALUES (?, ?, ?, ?, ?, ?, ?, ?, NOW())
 ");
```

```
$stmt->execute([
$this->usuarioId,
$accion,
$tabla,
$registroId,
json_encode($datosAnteriores),
json_encode($datosNuevos),
$this->ipAddress,
$_SERVER['HTTP_USER_AGENT'] ?? 'unknown'
]);
}

// Ejemplo de uso específico
public function logCrearMaquina($maquinaId, $datosMaquina) {
$this->registrarAccion('CREATE', 'maquinaria', $maquinaId, null, $datosMaquina);
}

public function logEditarMaquina($maquinaId, $datosAnteriores, $datosNuevos) {
$this->registrarAccion('UPDATE', 'maquinaria', $maquinaId, $datosAnteriores, $datosNuevos);
}
}
```

2. Tabla de Base de Datos para Auditoría

```
CREATE TABLE auditoria_logs (
 id INT PRIMARY KEY AUTO_INCREMENT,
 usuario_id INT,
 accion ENUM('CREATE', 'UPDATE', 'DELETE', 'LOGIN', 'LOGOUT') NOT NULL,
 tabla_afectada VARCHAR(50) NOT NULL,
 registro_id INT,
 datos_anteriores JSON,
 datos_nuevos JSON,
 ip_address VARCHAR(45),
 user_agent TEXT,
 fecha_accion TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 INDEX idx_usuario_fecha (usuario_id, fecha_accion),
 INDEX idx_tabla_registro (tabla_afectada, registro_id)
);
```

Sistema de Carga Masiva de Datos (Excel/CSV) ¿Para qué sirve?

Importar cientos de máquinas desde archivos Excel o CSV de una sola vez, en lugar de registrarlas una por una manualmente.

Implementación Paso a Paso:

1. Procesador de Archivos Excel

```
// includes/importador_excel.php
require_once 'vendor/phpspreadsheet/autoload.php';
use PhpOffice\PhpSpreadsheet\IOFactory;

class ImportadorMaquinaria {
 private $pdo;
 private $errores = [];
 private $procesados = 0;

 public function __construct($pdo) {
 $this->pdo = $pdo;
 }

 public function importarDesdeExcel($rutaArchivo) {
 try {
 $spreadsheet = IOFactory::load($rutaArchivo);
 $worksheet = $spreadsheet->getActiveSheet();
 $filas = $worksheet->toArray();

 // Saltar la primera fila (encabezados)
```

```
array_shift($filas);

foreach ($filas as $numeroFila => $fila) {
$this->procesarFilaMaquina($fila, $numeroFila + 2);
}

return [
'procesados' => $this->procesados,
'errores' => $this->errores,
'total_filas' => count($filas)
];

} catch (Exception $e) {
return ['error' => 'Error al leer archivo: ' . $e->getMessage()];
}
}

private function procesarFilaMaquina($fila, $numeroFila) {
// Validar que tenga los campos mínimos
if (empty($fila[0]) || empty($fila[1]) || empty($fila[2])) {
$this->errores[] = "Fila {$numeroFila}: Faltan campos obligatorios";
return;
}
```

```
try {
$stmt = $this->pdo->prepare("
INSERT INTO maquinaria
(tipo, marca, modelo, año, numero_serie, horas_uso, estado)
VALUES (?, ?, ?, ?, ?, ?, ?)
");

$stmt->execute([
$fila[0], // tipo
$fila[1], // marca
$fila[2], // modelo
$fila[3] ?? date('Y'), // año
$fila[4] ?? 'SIN_SERIE_' . time(), // numero_serie
$fila[5] ?? 0, // horas_uso
$fila[6] ?? 'disponible' // estado
]);

$this->procesados++;

} catch (PDOException $e) {
$this->errores[] = "Fila {$numeroFila}: " . $e->getMessage();
}
}
}
```

Interfaz de Usuario para Importación Formulario de Carga de Archivos:

```
<!-- importar/subir_archivo.php -->
<div class="container mt-4">
 <div class="card">
 <div class="card-header">
 <h3>Importar Maquinaria desde Excel/CSV</h3>
 </div>
 <div class="card-body">
 <form id="formImportar" enctype="multipart/form-data">
 <div class="mb-3">
 <label for="archivo" class="form-label">Seleccionar Archivo</label>
 <input type="file" class="form-control" id="archivo" name="archivo"   accept=".xlsx,.xls,.csv" required>
 <div class="form-text">
 Formatos soportados: Excel (.xlsx, .xls) y CSV (.csv)
 </div>
 </div>

 <div class="mb-3">
 <div class="form-check">
 <input class="form-check-input" type="checkbox" id="validarDuplicados"   name="validar_duplicados" checked>
```

```
<label class="form-check-label" for="validarDuplicados">
Validar números de serie duplicados
</label>
</div>
</div>

<button type="submit" class="btn btn-primary">
<i class="fas fa-upload"></i> Importar Datos
</button>
</form>

<!-- Barra de progreso -->
<div id="progresoImportacion" class="mt-3" style="display: none;">
<div class="progress">
<div class="progress-bar" role="progressbar" style="width: 0%"></div>  </div>
<small class="text-muted">Procesando archivo...</small>
</div>

<!-- Resultados -->
<div id="resultadosImportacion" class="mt-3"></div>
</div>
</div>
</div>
```

JavaScript para Manejo de la Importación:

```
// assets/js/importador.js
document.getElementById('formImportar').addEventListener('submit', function(e) {  e.preventDefault();

 const formData = new FormData(this);
 const progreso = document.getElementById('progresoImportacion');
 const resultados = document.getElementById('resultadosImportacion');

 // Mostrar barra de progreso
 progreso.style.display = 'block';
 resultados.innerHTML = '';

 fetch('api/procesar_importacion.php', {
 method: 'POST',
 body: formData
 })
 .then(response => response.json())
 .then(data => {
 progreso.style.display = 'none';
 mostrarResultados(data);
 })
```

```
 .catch(error => {
 progreso.style.display = 'none';
 mostrarError('Error al procesar archivo: ' + error.message);
 });
});

function mostrarResultados(data) {
 const resultados = document.getElementById('resultadosImportacion');

 if (data.error) {
 resultados.innerHTML = `
 <div class="alert alert-danger">
 <strong>Error:</strong> ${data.error}
 </div>
 `;
 return;
 }

 let html = `
 <div class="alert alert-success">
 <h5>Importación Completada</h5>
 <p><strong>Registros procesados:</strong> ${data.procesados}</p>
 <p><strong>Total de filas:</strong> ${data.total_filas}</p>
 </div>
```

```
`;

 if (data.errores && data.errores.length > 0) {
 html += `
 <div class="alert alert-warning">
 <h6>Errores encontrados:</h6>
 <ul>
 ${data.errores.map(error => `<li>${error}</li>`).join('')}
 </ul>
 </div>
 `;
 }

 resultados.innerHTML = html;
}
```

Sistema de Reportes Avanzados con Gráficos ¿Qué incluye?

Reportes visuales interactivos con gráficos de barras, líneas y tortas que muestran estadísticas importantes del negocio.

Implementación con Chart.js:

1. Reporte de Rendimiento por Tipo de Máquina

```
// reportes/rendimiento_maquinas.php
function obtenerDatosRendimiento($pdo, $fechaInicio, $fechaFin) {
 $stmt = $pdo->prepare("
 SELECT
 m.tipo,
 COUNT(m.id) as total_maquinas,
 AVG(c.cantidad_litros / NULLIF((c.horas_uso_fin - c.horas_uso_inicio), 0)) as consumo_promedio,
 SUM(c.costo_total) as costo_total_combustible,
 AVG(DATEDIFF(CURDATE(), COALESCE(MAX(man.fecha_realizado), m.fecha_registro))) as dias_sin_mantenimiento  FROM maquinaria m
 LEFT JOIN combustible c ON m.id = c.maquinaria_id
 AND c.fecha BETWEEN ? AND ?
 LEFT JOIN mantenimientos man ON m.id = man.maquinaria_id
 GROUP BY m.tipo
 ORDER BY total_maquinas DESC
 ");

 $stmt->execute([$fechaInicio, $fechaFin]);
 return $stmt->fetchAll(PDO::FETCH_ASSOC);
```

```
}

// API endpoint para los gráficos
if ($_GET['action'] === 'datos_grafico') {
 header('Content-Type: application/json');

 $fechaInicio = $_GET['fecha_inicio'] ?? date('Y-m-01');
 $fechaFin = $_GET['fecha_fin'] ?? date('Y-m-t');

 $datos = obtenerDatosRendimiento($pdo, $fechaInicio, $fechaFin);

 // Formatear datos para Chart.js
 $response = [
 'labels' => array_column($datos, 'tipo'),
 'datasets' => [
 [
 'label' => 'Total Máquinas',
 'data' => array_column($datos, 'total_maquinas'),
 'backgroundColor' => ['#FF6384', '#36A2EB', '#FFCE56', '#4BC0C0']
 ],
 [
 'label' => 'Costo Combustible',
 'data' => array_column($datos, 'costo_total_combustible'),
 'backgroundColor' => '#FF9F40',
```

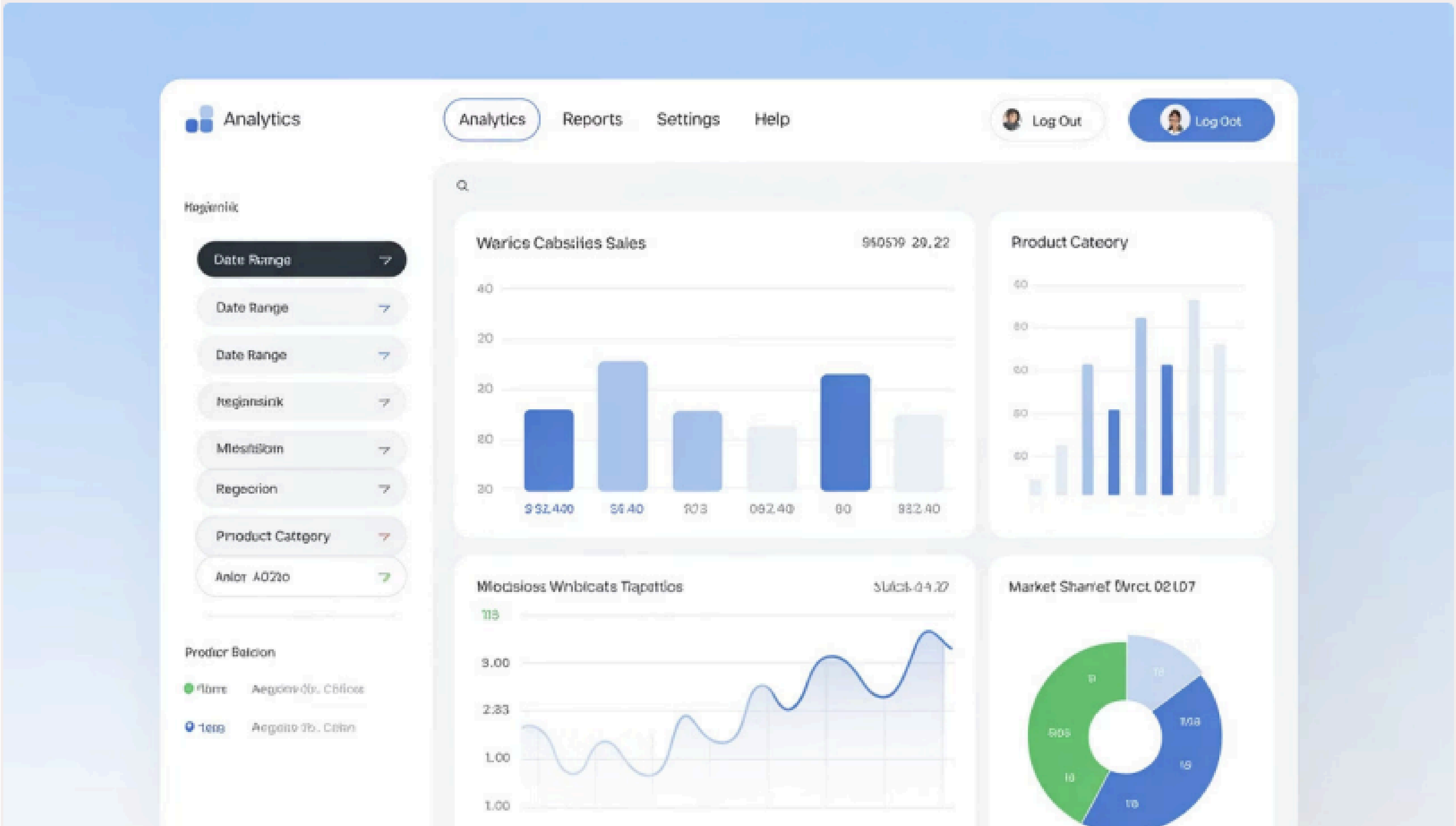
```
 'type' => 'line',
 'yAxisID' => 'y1'
 ]
 ]
 ];

 echo json_encode($response);
 exit;
}
```

2. HTML para Mostrar Gráficos

Distribución por Tipo de Máquina

Consumo de Combustible Mensual

Alertas de Mantenimiento por Mes


Analytics
Analytics
Reports
Settings
Help
Log Out
Date Range
Date Range
Date Range
Regecion
Product Category
Warics Cabsilies Sales
Product Cateory
Market Sharet

JavaScript para Renderizar Gráficos Interactivos 3.

Configuración de Chart.js

El siguiente código JavaScript define la clase `DashboardGraficos`, encargada de inicializar y gestionar los diferentes gráficos interactivos del dashboard utilizando la librería Chart.js. Incluye métodos para cargar datos de la API y configurar los gráficos de tipo rosquilla (doughnut) y línea, así como la lógica para la inicialización y actualización periódica.

```
// assets/js/graficos-dashboard.js
class DashboardGraficos {
 constructor() {
 this.charts = {};
 this.colores = {
 primary: '#007bff',
 success: '#28a745',
 warning: '#ffc107',
 danger: '#dc3545',
 info: '#17a2b8'
 };
 }

 async inicializar() {
 await this.cargarGraficoTipoMaquinas();
```

```
 await this.cargarGraficoCombustible();
 await this.cargarGraficoMantenimientos();
 }
}
```

Gráfico de Tipos de Máquinas

Este método carga los datos de tipos de máquinas y configura un gráfico de tipo rosquilla (doughnut) para visualizar la distribución. Incluye una leyenda en la parte inferior y un tooltip personalizado para mostrar porcentajes.

```
 async cargarGraficoTipoMaquinas() {
 try {
 const response = await fetch('api/datos_graficos.php? tipo=maquinas_por_tipo');
 const data = await response.json();

 const ctx =
document.getElementById('graficoTipoMaquinas').getContext('2d');

 this.charts.tipoMaquinas = new Chart(ctx, {
 type: 'doughnut',
 data: {
```

```
labels: data.labels,
datasets: [{
data: data.valores,
backgroundColor: [
this.colores.primary,
this.colores.success,
this.colores.warning,
this.colores.danger,
this.colores.info
],
borderWidth: 2,
borderColor: '#fff'
}]
},
options: {
responsive: true,
plugins: {
legend: {
position: 'bottom'
},
tooltip: {
callbacks: {
label: function(context) {
const total = context.dataset.data.reduce((a, b)  => a + b, 0);
```

```
const percentage = ((context.parsed / total) *  100).toFixed(1);
return `${context.label}: ${context.parsed}  (${percentage}%)`;
}
}
}
}
}
});
} catch (error) {
console.error('Error cargando gráfico de tipos:', error);  }
}
```

Inicialización y Actualización
Gráfico de Consumo de Combustible

Este método obtiene los datos de consumo de combustible mensual y los representa en un gráfico de línea. Utiliza dos ejes Y para mostrar simultáneamente los litros consumidos y el costo total.

```
async cargarGraficoCombustible() {
try {
const response = await fetch('api/datos_graficos.php? tipo=combustible_mensual');
const data = await response.json();
```

```
 const ctx =
document.getElementById('graficoCombustible').getContext('2d');

 this.charts.combustible = new Chart(ctx, {
 type: 'line',
 data: {
 labels: data.meses,
 datasets: [{
 label: 'Litros Consumidos',
 data: data.litros,
 borderColor: this.colores.primary,
 backgroundColor: this.colores.primary + '20',  fill: true,
 tension: 0.4
 }, {
 label: 'Costo Total ($)',
 data: data.costos,
 borderColor: this.colores.danger,
 backgroundColor: this.colores.danger + '20',  fill: false,
 yAxisID: 'y1'
 }]
 },
 options: {
 responsive: true,
```

```
scales: {
y: {
type: 'linear',
display: true,
position: 'left',
title: {
display: true,
text: 'Litros'
}
},
y1: {
type: 'linear',
display: true,
position: 'right',
title: {
display: true,
text: 'Costo ($)'
},
grid: {
drawOnChartArea: false,
},
}
}
}
```

```
});
} catch (error) {
console.error('Error cargando gráfico de combustible:',  error);
}
}
```

El siguiente fragmento de código asegura que los gráficos se carguen cuando la página esté completamente cargada y se actualicen automáticamente cada 5 minutos para mostrar los datos más recientes.

```
// Inicializar cuando cargue la página
document.addEventListener('DOMContentLoaded', function() {
 const dashboard = new DashboardGraficos();
 dashboard.inicializar();

 // Actualizar cada 5 minutos
 setInterval(() => {
 dashboard.inicializar();
 }, 300000);
});
```


Dashboard
Vehicles
Reports
Settings
Log Out
Fleet Venricloie Performance
202
Sedan
Truck
SUV
Fuel Consumption
Maintenance Alerts
Average Driven
MPG
Total Miles
MPG
Total Drives
Log Out

Sistema de Configuración Avanzada

¿Para qué sirve?

Permitir que los administradores personalicen el comportamiento del sistema sin tocar código: intervalos de mantenimiento, tipos de combustible, zonas geográficas, etc.

Implementación Completa:

1. Tabla de Configuraciones

```
-- Tabla para almacenar configuraciones del sistema
CREATE TABLE configuraciones (
 id INT PRIMARY KEY AUTO_INCREMENT,
 categoria VARCHAR(50) NOT NULL,
 clave VARCHAR(100) NOT NULL,
 valor TEXT NOT NULL,
 tipo_dato ENUM('string', 'integer', 'float', 'boolean', 'json') DEFAULT 'string',
 descripcion TEXT,
 valor_por_defecto TEXT,
 fecha_modificacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
 modificado_por INT,
```

```
 UNIQUE KEY unique_config (categoria, clave),
 INDEX idx_categoria (categoria)
);

-- Insertar configuraciones por defecto
INSERT INTO configuraciones (categoria, clave, valor, tipo_dato, descripcion, valor_por_defecto) VALUES
('mantenimiento', 'intervalo_horas_menor', '250', 'integer', 'Horas entre mantenimientos menores', '250'),
('mantenimiento', 'intervalo_horas_mayor', '500', 'integer', 'Horas entre mantenimientos mayores', '500'),
('mantenimiento', 'dias_alerta_previa', '7', 'integer', 'Días de anticipación para alertas', '7'),
('combustible', 'precio_por_defecto', '4500', 'float', 'Precio por litro por defecto (COP)', '4500'),
('combustible', 'tipos_combustible', '["Diesel", "Gasolina", "ACPM", "Biodiesel"]', 'json', 'Tipos de combustible disponibles', '["Diesel", "ACPM"]'),
('sistema', 'nombre_empresa', 'Empresa Agrícola Demo', 'string', 'Nombre de la empresa', 'Mi Empresa'),
('sistema', 'backup_automatico', 'true', 'boolean', 'Activar backup automático diario', 'true'),
('notificaciones', 'email_administrador', 'admin@empresa.com', 'string', 'Email del administrador', ''),
('gps', 'radio_zona_trabajo', '5000', 'integer', 'Radio en metros de zona de trabajo autorizada', '5000');
```

2. Clase para Manejo de Configuraciones

```
// includes/configuracion.php
class ConfiguracionSistema {
 private $pdo;
 private $cache = [];
```

```
public function __construct($pdo) {
$this->pdo = $pdo;
}

public function obtener($categoria, $clave, $valorPorDefecto = null) {
$claveCompleta = "{$categoria}.{$clave}";

// Verificar cache primero
if (isset($this->cache[$claveCompleta])) {
return $this->cache[$claveCompleta];
}

$stmt = $this->pdo->prepare("
SELECT valor, tipo_dato, valor_por_defecto
FROM configuraciones
WHERE categoria = ? AND clave = ?
");
$stmt->execute([$categoria, $clave]);
$config = $stmt->fetch();

if (!$config) {
return $valorPorDefecto;
}
```

```
$valor = $config['valor'] ?? $config['valor_por_defecto'];

// Convertir según el tipo de dato
switch ($config['tipo_dato']) {
case 'integer':
$valor = (int) $valor;
break;
case 'float':
$valor = (float) $valor;
break;
case 'boolean':
$valor = filter_var($valor, FILTER_VALIDATE_BOOLEAN);
break;
case 'json':
$valor = json_decode($valor, true);
break;
}

// Guardar en cache
$this->cache[$claveCompleta] = $valor;

return $valor;
}
```

```
public function establecer($categoria, $clave, $valor, $usuarioId = null) {
// Convertir valor a string para almacenamiento
if (is_array($valor) || is_object($valor)) {
$valorString = json_encode($valor);
$tipoDato = 'json';
} elseif (is_bool($valor)) {
$valorString = $valor ? 'true' : 'false';
$tipoDato = 'boolean';
} elseif (is_int($valor)) {
$valorString = (string) $valor;
$tipoDato = 'integer';
} elseif (is_float($valor)) {
$valorString = (string) $valor;
$tipoDato = 'float';
} else {
$valorString = (string) $valor;
$tipoDato = 'string';
}

$stmt = $this->pdo->prepare("
INSERT INTO configuraciones (categoria, clave, valor, tipo_dato, modificado_por)
VALUES (?, ?, ?, ?, ?)
ON DUPLICATE KEY UPDATE
valor = VALUES(valor),
```

```
tipo_dato = VALUES(tipo_dato),
modificado_por = VALUES(modificado_por),
fecha_modificacion = CURRENT_TIMESTAMP
");

$resultado = $stmt->execute([$categoria, $clave, $valorString, $tipoDato, $usuarioId]);

// Limpiar cache
$claveCompleta = "{$categoria}.{$clave}";
unset($this->cache[$claveCompleta]);

return $resultado;
}

public function obtenerPorCategoria($categoria) {
$stmt = $this->pdo->prepare("
SELECT clave, valor, tipo_dato, descripcion, valor_por_defecto
FROM configuraciones
WHERE categoria = ?
ORDER BY clave
");
$stmt->execute([$categoria]);

$configuraciones = [];
```

```
while ($row = $stmt->fetch()) {
$valor = $row['valor'] ?? $row['valor_por_defecto'];

// Convertir según tipo
switch ($row['tipo_dato']) {
case 'integer':
$valor = (int) $valor;
break;
case 'float':
$valor = (float) $valor;
break;
case 'boolean':
$valor = filter_var($valor, FILTER_VALIDATE_BOOLEAN);
break;
case 'json':
$valor = json_decode($valor, true);
break;
}

$configuraciones[$row['clave']] = [
'valor' => $valor,
'descripcion' => $row['descripcion'],
'tipo_dato' => $row['tipo_dato']
];
```

```
 }

 return $configuraciones;
 }
}

// Función global para acceso rápido
function config($categoria, $clave, $valorPorDefecto = null) {
 global $pdo;
 static $configuracion = null;

 if ($configuracion === null) {
 $configuracion = new ConfiguracionSistema($pdo);
 }

 return $configuracion->obtener($categoria, $clave, $valorPorDefecto);
}
```

Interfaz de Administración de Configuraciones 3.

Panel de Configuración para Administradores

Categorías

Mantenimiento Combustible Sistema
Notificaciones GPS y Ubicación

Configuraciones de Mantenimiento

4. JavaScript para Manejo Dinámico

```
// assets/js/configuraciones.js
class AdminConfiguraciones {
 constructor() {
 this.categoriaActual = 'mantenimiento';
 this.configuracionesOriginales = {};
 this.configuracionesModificadas = {};
 }

 async inicializar() {
```

```
this.configurarEventos();
await this.cargarCategoria('mantenimiento');
}

configurarEventos() {
// Cambio de categoría
document.querySelectorAll('[data-categoria]').forEach(enlace => {
enlace.addEventListener('click', async (e) => {
e.preventDefault();
const categoria = e.target.closest('[data-categoria]').dataset.categoria;
await this.cambiarCategoria(categoria);
});
});

// Guardar cambios
document.getElementById('btnGuardarTodo').addEventListener('click', () => {
this.guardarConfiguraciones();
});

// Detectar cambios en formularios
document.getElementById('formConfiguraciones').addEventListener('input', (e) => {
this.marcarCambio(e.target);
});
}
```

```
async cambiarCategoria(categoria) {
// Actualizar menú activo
document.querySelectorAll('[data-categoria]').forEach(item => {
item.classList.remove('active');
});
document.querySelector(`[data-categoria="${categoria}"]`).classList.add('active');

// Cargar configuraciones
await this.cargarCategoria(categoria);
}

async cargarCategoria(categoria) {
try {
const response = await fetch(`api/configuraciones.php?categoria=${categoria}`);
const configuraciones = await response.json();

this.categoriaActual = categoria;
this.configuracionesOriginales[categoria] = configuraciones;

this.renderizarFormulario(configuraciones);
this.actualizarTitulo(categoria);

} catch (error) {
```

```
console.error('Error cargando configuraciones:', error);
this.mostrarError('Error al cargar configuraciones');
}
}

renderizarFormulario(configuraciones) {
const contenedor = document.getElementById('contenidoConfiguraciones');
let html = '';

for (const [clave, config] of Object.entries(configuraciones)) {
html += this.generarCampoConfiguracion(clave, config);
}

contenedor.innerHTML = html;
}

generarCampoConfiguracion(clave, config) {
const id = `config_${clave}`;
let inputHtml = '';

switch (config.tipo_dato) {
case 'boolean':
inputHtml = `
```

Activado

`; break; case 'integer': inputHtml = ` `; break; case 'float': inputHtml = ` `; break; case 'json': if (Array.isArray(config.valor)) { inputHtml = ` ${config.valor.join('\n')} `; } else { inputHtml = ` ${JSON.stringify(config.valor, null, 2)} `; } break; default: // string inputHtml = ` `; } return `

${this.formatearNombreClave(clave)} ${inputHtml}

${config.descripcion}

`; } formatearNombreClave(clave) { return clave.replace(/_/g, ' ') .replace(/\b\w/g, l => l.toUpperCase()); } marcarCambio(elemento) { const categoria = this.categoriaActual; const clave = elemento.name; if (!this.configuracionesModificadas[categoria]) { this.configuracionesModificadas[categoria] = {}; } let valor = elemento.value; // Convertir según el tipo if (elemento.type === 'checkbox') { valor = elemento.checked; } else if (elemento.type === 'number') { valor = elemento.step === '1' ? parseInt(valor) : parseFloat(valor); } this.configuracionesModificadas[categoria][clave] = valor; // Marcar visualmente que hay cambios elemento.classList.add('border-warning'); document.getElementById('btnGuardarTodo').classList.add('btn-warning'); document.getElementById('btnGuardarTodo').classList.remove('btn-success'); } async guardarConfiguraciones() { if (Object.keys(this.configuracionesModificadas).length === 0) { this.mostrarInfo('No hay cambios para guardar'); return; } try { const response = await fetch('api/guardar_configuraciones.php', { method: 'POST', headers: { 'Content-Type': 'application/json' }, body: JSON.stringify(this.configuracionesModificadas) }); const resultado = await response.json(); if (resultado.success) { this.mostrarExito('Configuraciones guardadas correctamente'); this.configuracionesModificadas = {}; // Resetear estilos document.querySelectorAll('.border-warning').forEach(el => { el.classList.remove('border-warning'); }); document.getElementById('btnGuardarTodo').classList.remove('btn-warning'); document.getElementById('btnGuardarTodo').classList.add('btn success'); } else { this.mostrarError('Error al guardar: ' + resultado.mensaje); } } catch (error) { console.error('Error guardando configuraciones:', error); this.mostrarError('Error de conexión al guardar'); } } actualizarTitulo(categoria) { const titulos = { 'mantenimiento': 'Configuraciones de

Mantenimiento', 'combustible': 'Configuraciones de Combustible', 'sistema': 'Configuraciones del Sistema', 'notificaciones': 'Configuraciones de Notificaciones', 'gps': 'Configuraciones de GPS y Ubicación' }; document.getElementById('tituloCategoria').textContent = titulos[categoria] || 'Configuraciones'; } mostrarExito(mensaje) { // Implementar notificación de éxito this.mostrarNotificacion(mensaje, 'success'); } mostrarError(mensaje) { // Implementar notificación de error this.mostrarNotificacion(mensaje, 'danger'); } mostrarInfo(mensaje) { // Implementar notificación informativa this.mostrarNotificacion(mensaje, 'info'); } mostrarNotificacion(mensaje, tipo) { // Crear toast o alert temporal const alert = document.createElement('div'); alert.className = `alert alert-${tipo} alert-dismissible fade show position-fixed`; alert.style.cssText = 'top: 20px; right: 20px; z-index: 9999; min-width: 300px;'; alert.innerHTML = ` ${mensaje}

Ejemplos de Uso en el Sistema:

```
// Usar configuraciones en el código
$intervaloMantenimiento = config('mantenimiento', 'intervalo_horas_menor', 250);
$precioDefecto = config('combustible', 'precio_por_defecto', 4500);
$tiposCombustible = config('combustible', 'tipos_combustible', ['Diesel', 'ACPM']);
$backupAutomatico = config('sistema', 'backup_automatico', true);

// En el cálculo de alertas de mantenimiento
if ($horasUso >= $intervaloMantenimiento) {
 enviarAlertaMantenimiento($maquinaId);
```

```
}
```