|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| E.I.S.”P.D.M.” CARRERA INFORMÁTICA INDUSTRIAL | LABORATORIO No \_\_ | | | | | | C |
| APELLIDOS Y NOMBRE(S): CHAMBI NINA ARMANDO JUAN | | | | | | |
| MATERIA: SISTEMAS DE  INFORMACION II | | SIGLA:  SII-500 | | GESTION:  I/2025 | | FECHA:\_\_11\_/\_05\_\_  \_/\_\_\_2025\_ |
| DOCENTE: LIC. JULIA ZENTENO ZENTENO | | | | | / | SELLO DOCENTE |
| **TEMA:** REPORTES GRAFICOS | | | NO  MAQUINA | |
| INICIAL APELLIDO |

# 1. OBJETIVO

Desarrollar e implementar un panel de reportes gráficos para la plataforma 'Tienda Scripts', permitiendo visualizar estadísticas relevantes del sistema y exportarlas en formato PDF, todo ello mediante tecnologías web modernas como PHP, JavaScript y Chart.js.

# 2. TEORÍA (RESUMEN)

Un panel de reportes es una herramienta visual que permite interpretar grandes volúmenes de datos de forma sencilla. En este caso, se utilizan librerías como Chart.js para representar gráficamente información sobre usuarios, solicitudes y pagos. jsPDF permite la exportación de estos gráficos a documentos PDF. AJAX facilita la comunicación entre el frontend y el backend sin recargar la página.

# 3. APLICACIÓN – PROCEDIMIENTO

## 3.1 Condiciones Previas (Material y Equipo)

- Navegador Web  
- Servidor local con PHP (ej. XAMPP, Apache en Termux)  
- Editor de código (Visual Studio Code, Neovim, etc.)  
- Conexión a base de datos MySQL  
- Librerías: Chart.js, jsPDF, jQuery

## 3.2 Implementación - Pasos

Paso 1: Crear la interfaz base con HTML y Bootstrap

Paso 2: Obtener estadísticas desde el backend usando PHP y MySQL

Paso 3: Cargar los datos mediante AJAX

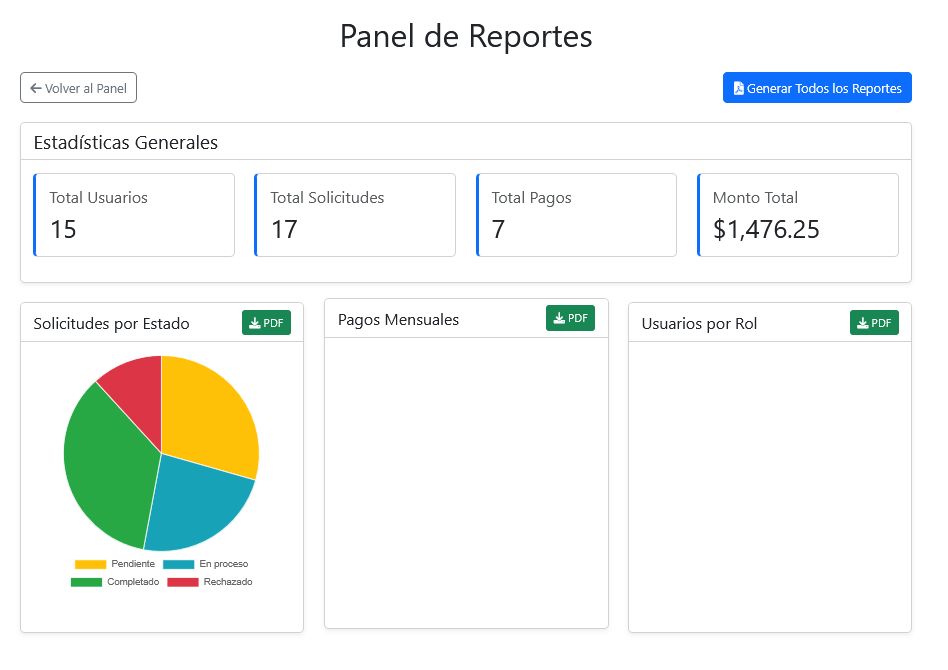
Paso 4: Representar los datos con gráficos Chart.js

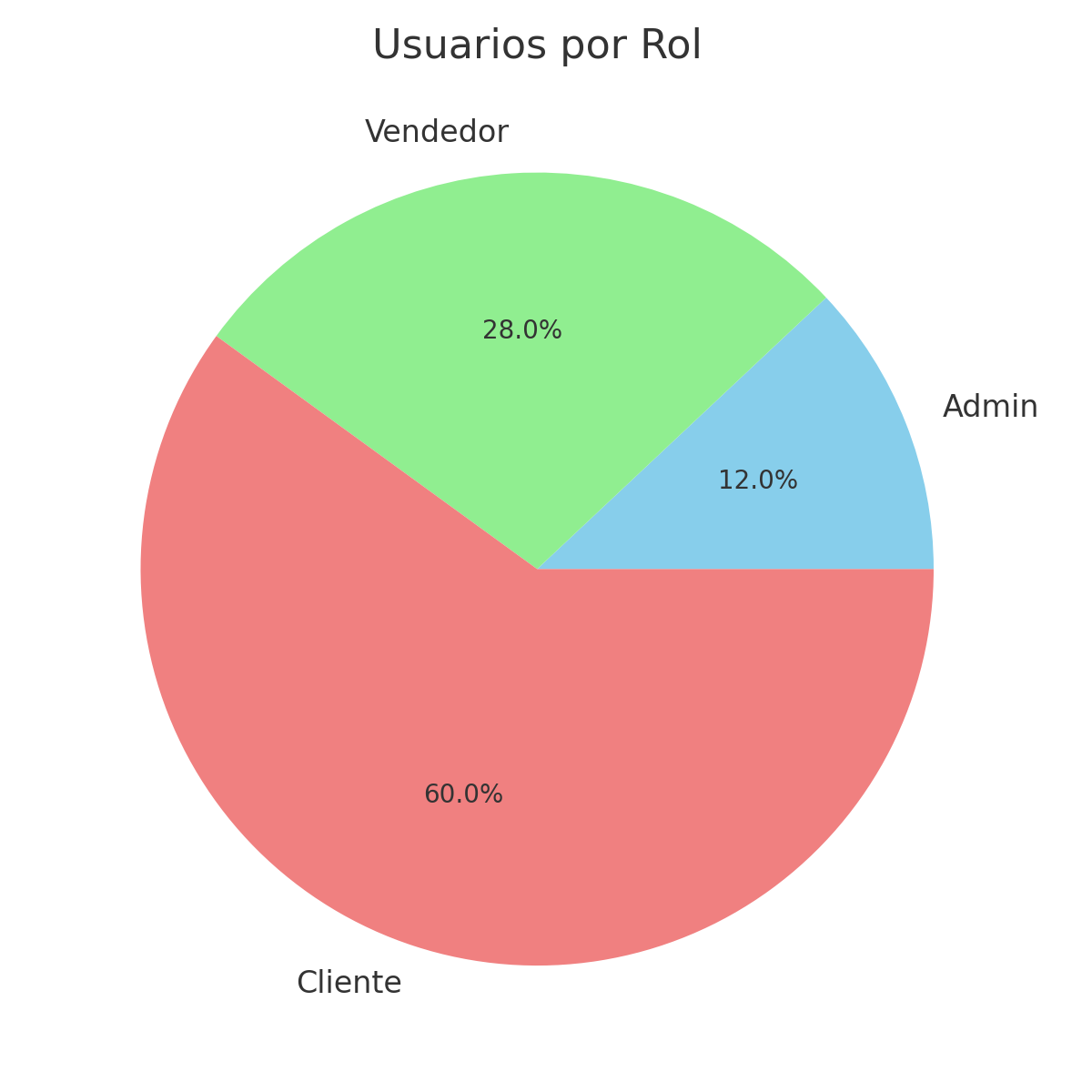
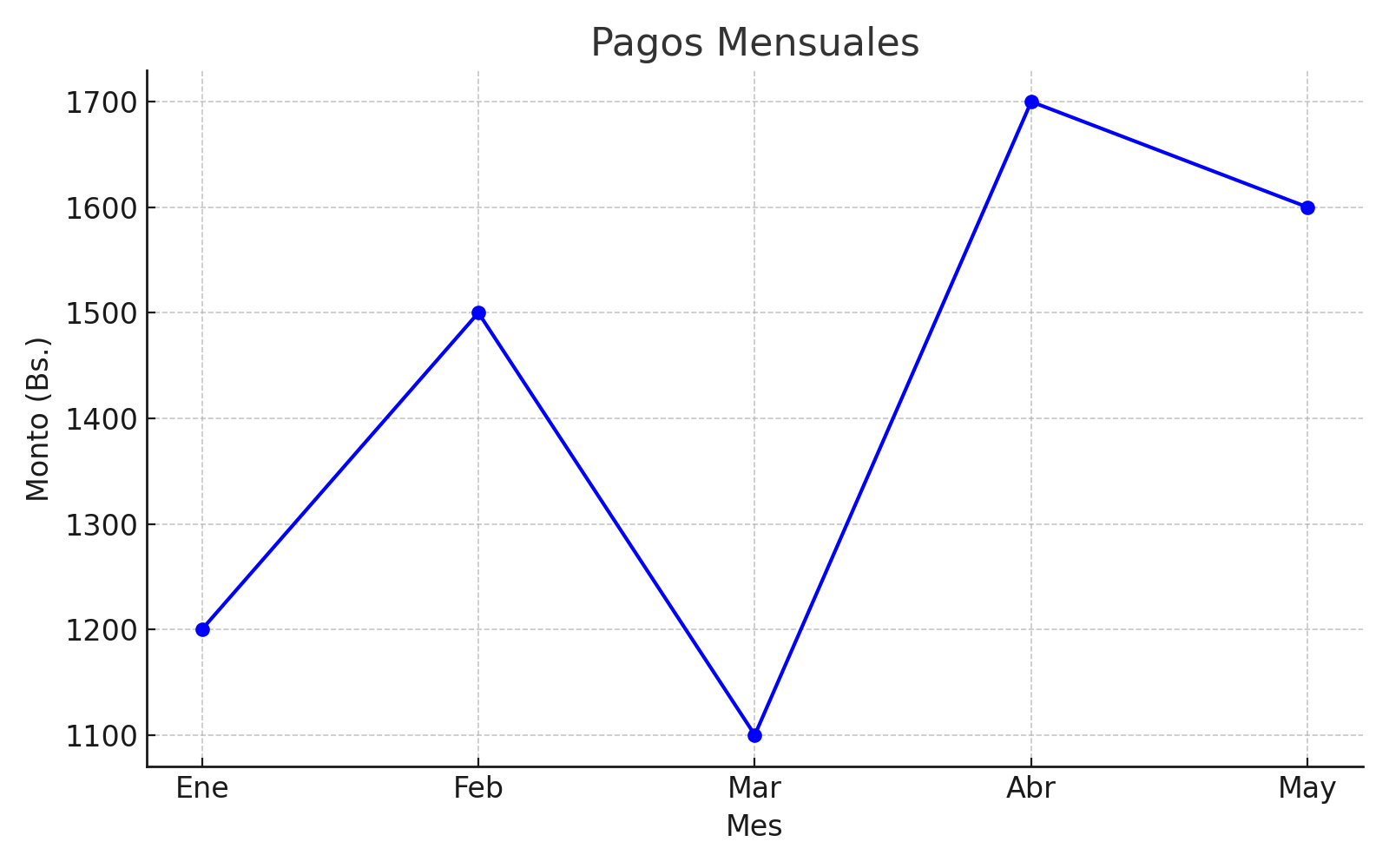
Paso 5: Agregar funcionalidad para exportar los gráficos a PDF con jsPDF

Paso 6: Estilizar la vista para una presentación amigable

# 4. RESULTADOS - LISTADOS

Se obtuvo un panel que presenta:  
- Total de usuarios, solicitudes y pagos  
- Gráfico de pastel: solicitudes por estado  
- Gráfico de barras y líneas: pagos por mes  
- Gráfico de dona: usuarios por rol





# CODIGO REPORTES EN PHP

Function cargarDatos() {

// Solicitar los datos para los reportes

$.ajax({

url: ‘process.php’,

type: ‘POST’,

data: {

action: ‘getReportData’

},

dataType: ‘json’,

success: function(response) {

if (response.success) {

crearGraficoSolicitudes(response.solicitudes);

crearGraficoPagos(response.pagos);

crearGraficoUsuarios(response.usuarios);

} else {

Alert(‘Error al cargar los datos: ‘ + response.message);

}

},

Error: function() {

Alert(‘Error de conexión al cargar los datos.’);

}

});

}

// Crear gráfico de solicitudes

Function crearGraficoSolicitudes(datos) {

Const ctx = document.getElementById(‘solicitudesChart’).getContext(‘2d’);

Const labels = datos.map(item => {

// Capitalizar primera letra

Return item.estado.charAt(0).toUpperCase() + item.estado.slice(1);

});

Const valores = datos.map(item => item.cantidad);

// Colores para los diferentes estados

Const colores = {

‘pendiente’: ‘#FFC107’,

‘en proceso’: ‘#17A2B8’,

‘completado’: ‘#28A745’,

‘rechazado’: ‘#DC3545’

};

Const backgroundColors = datos.map(item => colores[item.estado] || ‘#6C757D’);

New Chart(ctx, {

Type: ‘pie’,

Data: {

Labels: labels,

Datasets: [{

Data: valores,

backgroundColor: backgroundColors,

borderWidth: 1

}]

},

Options: {

Responsive: true,

maintainAspectRatio: false,

plugins: {

legend: {

position: ‘bottom’

},

Tooltip: {

Callbacks: {

Label: function(context) {

Const label = context.label || ‘’;

Const value = context.raw || 0;

Const total = context.dataset.data.reduce((a, b) => a + b, 0);

Const percentage = Math.round((value / total) \* 100);

Return `${label}: ${value} (${percentage}%)`;

}

}

}

}

}

});

}

// Crear gráfico de pagos mensuales

Function crearGraficoPagos(datos) {

Const ctx = document.getElementById(‘pagosChart’).getContext(‘2d’);

Const labels = datos.map(item => item.nombre\_mes);

Const valores = datos.map(item => parseFloat(item.total\_pagos));

Const cantidades = datos.map(item => parseInt(item.cantidad\_pagos));

New Chart(ctx, {

Type: ‘bar’,

Data: {

Labels: labels,

Datasets: [{

Label: ‘Monto Total ($)’,

Data: valores,

backgroundColor: ‘rgba(54, 162, 235, 0.5)’,

borderColor: ‘rgba(54, 162, 235, 1)’,

borderWidth: 1,

yAxisID: ‘y’

},

{

Label: ‘Cantidad de Pagos’,

Data: cantidades,

Type: ‘line’,

Fill: false,

backgroundColor: ‘rgba(255, 99, 132, 0.5)’,

borderColor: ‘rgba(255, 99, 132, 1)’,

borderWidth: 2,

tension: 0.1,

yAxisID: ‘y1’

}

]

},

Options: {

Responsive: true,

maintainAspectRatio: false,

scales: {

y: {

beginAtZero: true,

position: ‘left’,

title: {

display: true,

text: ‘Monto ($)’

}

},

Y1: {

beginAtZero: true,

position: ‘right’,

grid: {

drawOnChartArea: false

},

Title: {

Display: true,

Text: ‘Cantidad’

}

}

},

Plugins: {

Legend: {

Position: ‘bottom’

}

}

}

});

}

# 5. CONCLUSIONES

La implementación del panel de reportes permite a los administradores tomar decisiones informadas basadas en datos. El uso de herramientas como Chart.js y jsPDF ofrece interactividad y portabilidad. Este tipo de soluciones es vital para el monitoreo y mejora continua de plataformas como 'Tienda Scripts'.