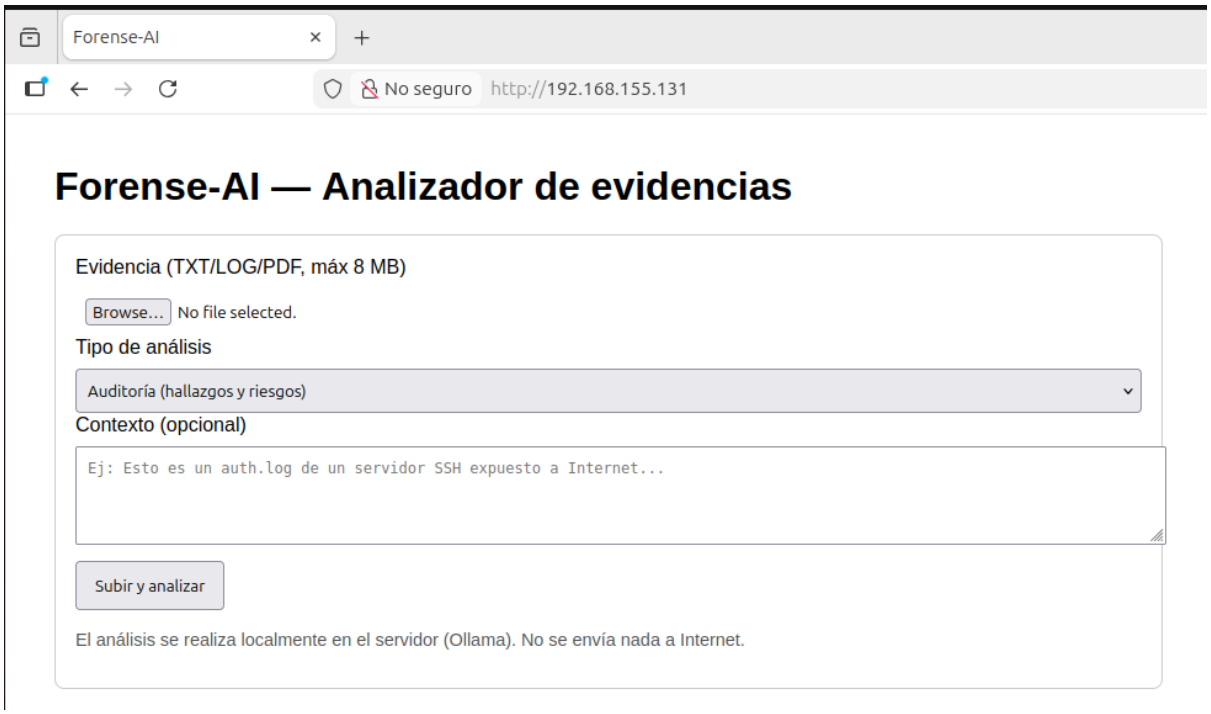
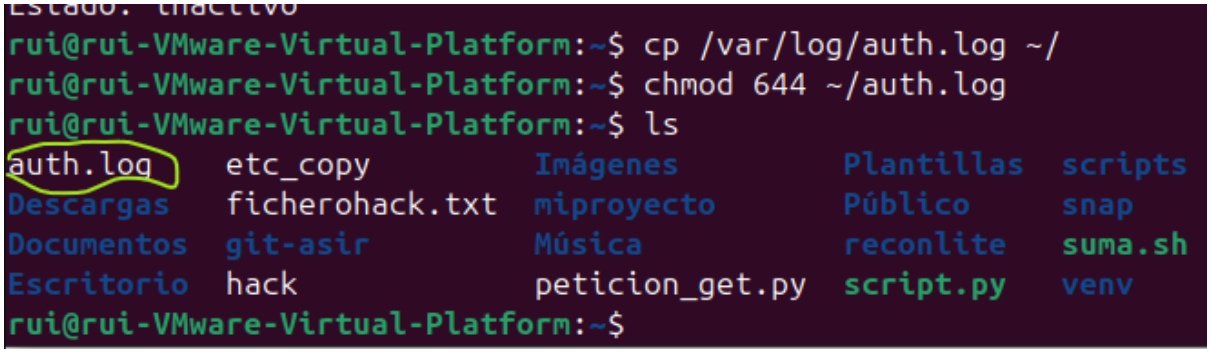
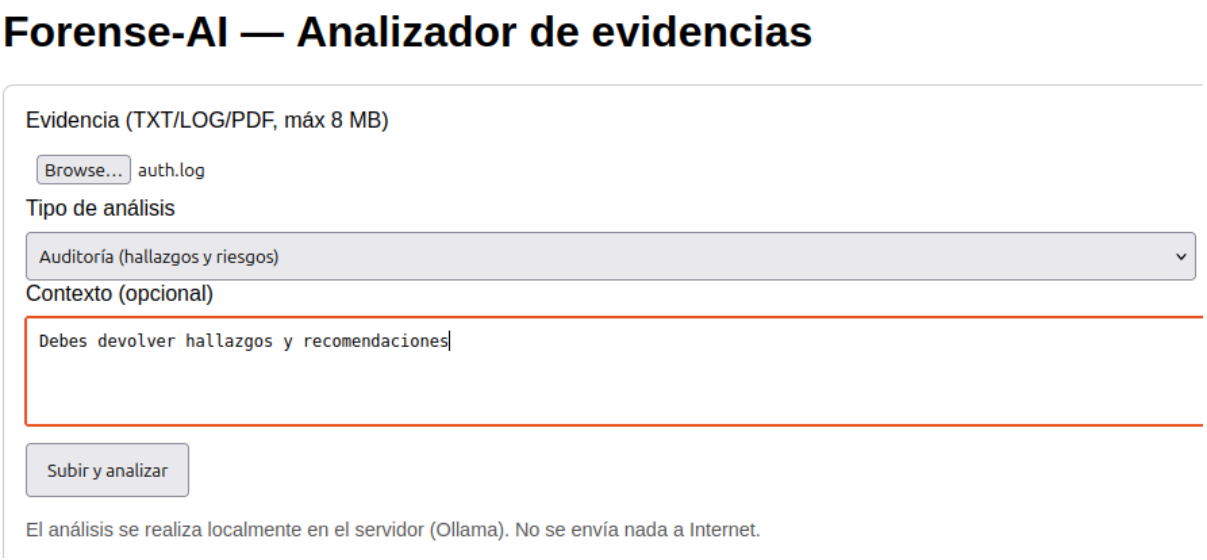
Pruebas obligatorias (para entregar)



22) Caso de prueba A (Auditoría)

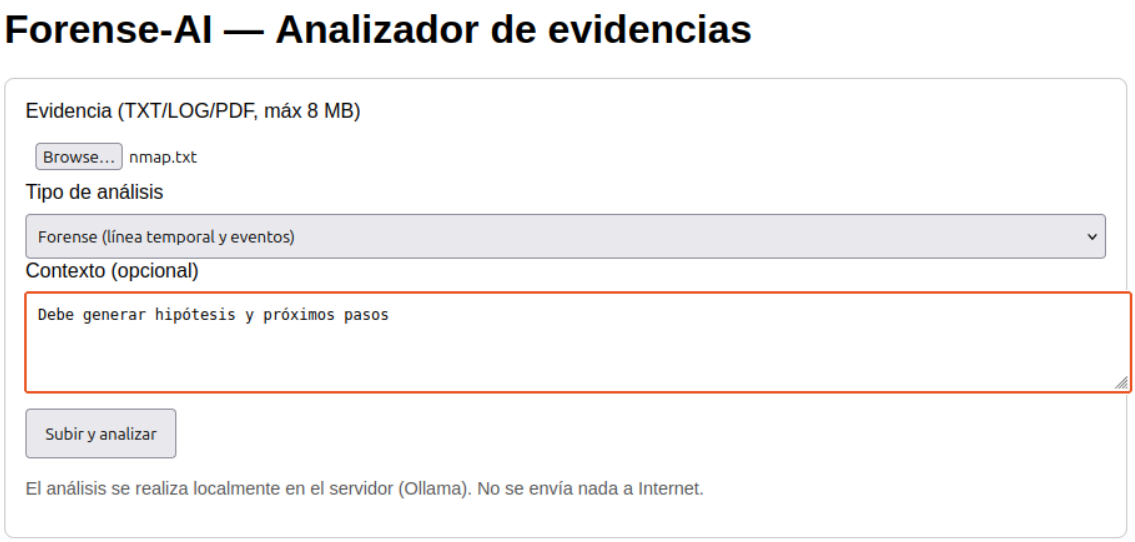


1. Sube un `auth.log` (o un ejemplo creado por vosotros)
2. Modo: Auditoría
3. Debe devolver hallazgos y recomendaciones



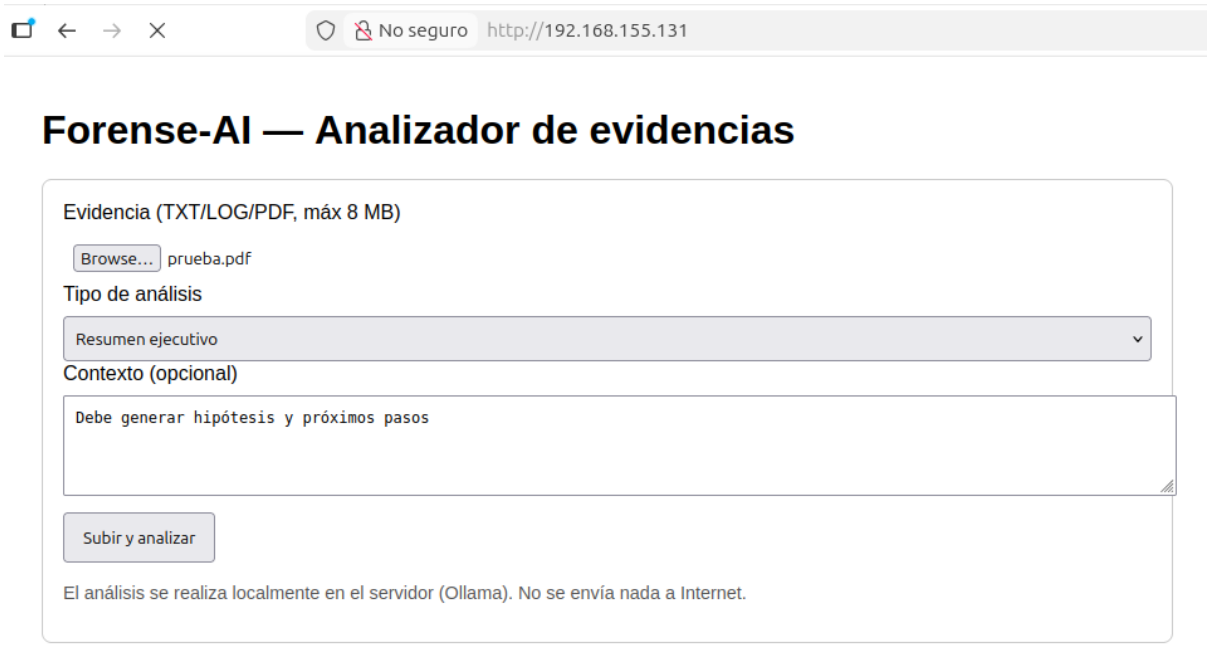
23) Caso de prueba B (Forense)

1. Sube una salida de nmap guardada como TXT
2. Modo: Forense
3. Debe generar hipótesis y próximos pasos



24) Caso de prueba C (PDF)

1. Sube un PDF técnico (manual, informe)
2. Modo: Resumen ejecutivo
3. Debe resumir y extraer puntos clave



En entornos profesionales de administración de sistemas, auditoría y ciberseguridad, el análisis de evidencias no se realiza en el equipo personal del analista.
Los datos —logs, resultados de auditoría, informes técnicos o documentación sensible— se envían a servidores dedicados, diseñados para procesar información de forma controlada, trazable y reproducible.

El proyecto Aletheia plantea la construcción de uno de estos servidores.

Aletheia es una máquina independiente dentro de la red cuya función no es almacenar información ni ofrecer servicios al usuario final, sino transformar evidencias técnicas en conocimiento estructurado.
Para ello, integra un motor de inteligencia artificial local, accesible únicamente desde el propio servidor, que actúa como sistema de análisis y apoyo a la toma de decisiones.

A diferencia de soluciones basadas en la nube, este servidor:

no envía información a terceros,

no depende de conectividad externa,

y permite mantener control total sobre los datos analizados.

Durante el desarrollo del proyecto, el alumnado diseñará y desplegará un servicio capaz de:

recibir archivos y evidencias técnicas,

procesarlos de forma segura y estática,

analizarlos desde una perspectiva de auditoría o forense,

y generar informes claros, coherentes y reutilizables.

El objetivo no es “usar IA”, sino integrarla como parte de una arquitectura real de sistemas, entendiendo su papel como herramienta de análisis, no como sustituto del criterio técnico.

Este proyecto reproduce un escenario habitual en organizaciones reales, donde la inteligencia no reside en el usuario final, sino en la infraestructura que da soporte al análisis.

Fase 1 — Puesta a punto del servidor

Requisitos mínimos de la VM

1. Crea una VM en VirtualBox:
    a. Ubuntu Server 22.04/24.04 (64-bit)
    b. 4 vCPU
    c. 8 GB RAM
    d. 40 GB disco
    e. Red: **Adaptador puente**

Instala Ubuntu Server y crea un usuario (no root), por ejemplo: `alumno`.

1) Actualiza el sistema

```
sudo apt update && sudo apt -y upgrade
sudo reboot
```

2) Instala herramientas base

Instalar utilidades básicas:

- curl
- unzip
- jq
- python3

```
sudo apt -y install curl unzip jq ca-certificates gnupg
python3 python3-venv python3-pip
```

3) Configura hostname (opcional pero recomendado)

```
sudo hostnamectl set-hostname forense-ai
```

Comprueba:

```
hostnamectl
```

4) Comprueba IP del servidor

```
ip a
```

Fase 2 — Instalar y probar Ollama

5) Instala Ollama

```
curl -fsSL https://ollama.com/install.sh | sh
```

6) Comprueba que el servicio está activo

```
sudo systemctl status ollama --no-pager
```

7) Descarga un modelo ligero (recomendado para VM)

```
ollama pull phi3
```

8) Prueba que responde

```
ollama run phi3 "Resume en 5 líneas qué es una auditoría de sistemas."
```

Si esto funciona, ya tienes “motor IA” listo.

Fase 3 — Instalar Apache + PHP (interfaz web)

9) Instala Apache y PHP

```
sudo apt -y install apache2 php libapache2-mod-php
```

10) Habilita y arranca Apache

```
sudo systemctl enable --now apache2
```

11) Abre el firewall (solo si lo estás usando)

Si tienes UFW activo:

```
sudo ufw allow 'Apache'
sudo ufw status
```

12) Comprueba acceso web

Desde tu PC (host), abre:

- http://IP_DEL_SERVIDOR/

Debes ver la página por defecto de Apache.

Fase 4 — Crear la estructura del servicio “Forense-AI”

13) Crea carpetas del proyecto

```
sudo mkdir -p /var/www/forense-
ai/{uploads,results,scripts,logs}
sudo chown -R www-data:www-data /var/www/forense-ai
sudo chmod -R 750 /var/www/forense-ai
```

14) Crea un VirtualHost

```
sudo nano /etc/apache2/sites-available/forense-ai.conf
```

Pega esto:

```
<VirtualHost *:80>
    ServerName forense-ai.local
    DocumentRoot /var/www/forense-ai

    <Directory /var/www/forense-ai>
        Options -Indexes
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/forense-ai_error.log
    CustomLog ${APACHE_LOG_DIR}/forense-ai_access.log
combined
</VirtualHost>
```

Activa el sitio:

```
sudo a2ensite forense-ai
sudo a2dissite 000-default
sudo systemctl reload apache2
```

Fase 5 — Servicio web: formulario de subida + análisis

15) Instala herramientas para extraer texto de PDF

```
sudo apt -y install poppler-utils
```

(trae `pdftotext`)

16) Crea el formulario web `index.php`

```
sudo nano /var/www/forense-ai/index.php
```

Contenido:

```
<?php
$maxSize = 8 * 1024 * 1024; // 8 MB
$allowed = ['txt','log','pdf'];

function safe_name($name) {
  $name = preg_replace('/[^a-zA-Z0-9._-]/', '_', $name);
  return $name;
}

?>
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Forense-AI</title>
  <style>
    body { font-family: Arial, sans-serif; margin: 40px;
```

```
max-width: 900px; }
    .box { padding: 16px; border: 1px solid #ccc; border-
radius: 8px; }
    input, textarea, select { width: 100%; padding: 8px;
margin-top: 8px; }
    button { padding: 10px 14px; margin-top: 12px; }
    .small { color: #555; font-size: 0.9em; }
  </style>
</head>
<body>
  <h1>Forense-AI — Analizador de evidencias</h1>
  <div class="box">
    <form action="process.php" method="post"
enctype="multipart/form-data">
      <label>Evidencia (TXT/LOG/PDF, máx 8 MB)</label>
      <input type="file" name="evidence" required>

      <label>Tipo de análisis</label>
      <select name="mode">
        <option value="audit">Auditoría (hallazgos y
riesgos)</option>
        <option value="forensic">Forense (línea temporal y
eventos)</option>
        <option value="summary">Resumen ejecutivo</option>
      </select>

      <label>Contexto (opcional)</label>
      <textarea name="context" rows="4" placeholder="Ej:
Esto es un auth.log de un servidor SSH expuesto a
Internet..."></textarea>

      <button type="submit">Subir y analizar</button>
      <p class="small">El análisis se realiza localmente en
el servidor (Ollama). No se envía nada a Internet.</p>
    </form>
  </div>
</body>
</html>
```

17) Crea el script de procesamiento `process.php`

```
sudo nano /var/www/forense-ai/process.php
```

Contenido:

```
<?php
$uploadDir = __DIR__ . "/uploads/";
$resultDir = __DIR__ . "/results/";
$logDir    = __DIR__ . "/logs/";

$maxSize = 8 * 1024 * 1024;
$allowed = ['txt','log','pdf'];

function safe_name($name) {
  return preg_replace('/[^a-zA-Z0-9._-]/', '_', $name);
}

function ext($name) {
  $p = pathinfo($name);
  return strtolower($p['extension'] ?? '');
}

function run_cmd($cmd) {
  $output = [];
  $ret = 0;
  exec($cmd . " 2>&1", $output, $ret);
  return [$ret, implode("\n", $output)];
}

if (!isset($_FILES['evidence'])) {
  http_response_code(400);
  exit("No file uploaded");
```

```
}

$f = $_FILES['evidence'];
if ($f['error'] !== UPLOAD_ERR_OK) {
  http_response_code(400);
  exit("Upload error");
}

if ($f['size'] > $maxSize) {
  http_response_code(400);
  exit("File too large");
}

$original = $f['name'];
$extension = ext($original);

if (!in_array($extension, $allowed, true)) {
  http_response_code(400);
  exit("Invalid extension");
}

$mode = $_POST['mode'] ?? 'audit';
$context = trim($_POST['context'] ?? '');

$base = date("Ymd_His") . "_" . safe_name($original);
$dest = $uploadDir . $base;

if (!move_uploaded_file($f['tmp_name'], $dest)) {
  http_response_code(500);
  exit("Failed to save upload");
}

// Extraer texto
$textFile = $dest . ".txt";
if ($extension === 'pdf') {
  // pdftotext
  [$ret, $out] = run_cmd("pdftotext " .
escapeshellarg($dest) . " " . escapeshellarg($textFile));
```

```
  if ($ret !== 0) {
    http_response_code(500);
    exit("pdftotext failed:\n" . htmlspecialchars($out));
  }
} else {
  // Copiar tal cual
  copy($dest, $textFile);
}

// Leer texto (limitamos por tamaño para no petar memoria)
$raw = file_get_contents($textFile);
if ($raw === false) {
  http_response_code(500);
  exit("Cannot read extracted text");
}
$raw = substr($raw, 0, 50000); // 50k chars (suficiente para
práctica)

$promptBase = "Eres un analista de ciberseguridad. Responde
en español con formato claro.\n";
if ($mode === 'audit') {
  $task = "Analiza la evidencia como auditoría. Devuelve: 1)
Resumen, 2) Hallazgos (con evidencias), 3) Riesgos, 4)
Recomendaciones priorizadas.";
} elseif ($mode === 'forensic') {
  $task = "Analiza la evidencia con enfoque forense.
Devuelve: 1) Resumen, 2) Línea temporal aproximada, 3)
Eventos relevantes, 4) Hipótesis, 5) Próximos pasos.";
} else {
  $task = "Resume la evidencia para un responsable no
técnico. Devuelve un resumen ejecutivo y 5 puntos clave.";
}

$contextPart = $context ? "\nContexto aportado por el
usuario:\n" . $context . "\n" : "";

$prompt = $promptBase . $task . $contextPart . "\nEVIDENCIA
(texto):\n" . $raw . "\n";
```

```
// Llamar a Ollama
$model = "phi3";
$cmd = "ollama run " . escapeshellarg($model) . " " .
escapeshellarg($prompt);

[$ret, $analysis] = run_cmd($cmd);
if ($ret !== 0) {
  http_response_code(500);
  exit("Ollama failed:\n" . htmlspecialchars($analysis));
}

$reportName = basename($dest) . "_informe.md";
$reportPath = $resultDir . $reportName;

$report = "# Informe Forense-AI\n\n"
        . "- Archivo: **" . htmlspecialchars($original) .
"**\n"
        . "- Fecha: **" . date("Y-m-d H:i:s") . "**\n"
        . "- Modo: **" . htmlspecialchars($mode) . "**\n\n"
        . "---\n\n"
        . $analysis . "\n";

file_put_contents($reportPath, $report);

// Log básico
file_put_contents($logDir . "activity.log",
  date("c") . " file=" . $base . " mode=" . $mode . " ip=" .
($_SERVER['REMOTE_ADDR'] ?? 'unknown') . "\n",
  FILE_APPEND
);

header("Location: result.php?f=" . urlencode($reportName));
```

18) Crea la página de resultados `result.php`

```
sudo nano /var/www/forense-ai/result.php
```

Contenido:

```
<?php
$resultDir = __DIR__ . "/results/";
$f = $_GET['f'] ?? '';
$f = basename($f); // evita traversal
$path = $resultDir . $f;

if (!$f || !file_exists($path)) {
  http_response_code(404);
  exit("Report not found");
}

$txt = file_get_contents($path);
?>
<!doctype html>
<html>
<head>
  <meta charset="utf-8">
  <title>Resultado — Forense-AI</title>
  <style>
    body { font-family: Arial, sans-serif; margin: 40px;
max-width: 900px; }
    pre { white-space: pre-wrap; border: 1px solid #ccc;
padding: 16px; border-radius: 8px; }
    a { display:inline-block; margin-top: 12px; }
  </style>
</head>
<body>
  <h1>Informe generado</h1>
  <a href="download.php?f=<?php echo urlencode($f);
?>">Descargar informe</a>
```

```
  <pre><?php echo htmlspecialchars($txt); ?></pre>
  <a href="index.php">⬅ Volver</a>
</body>
</html>
```

19) Crea el descargador `download.php`

```
sudo nano /var/www/forense-ai/download.php
```

Contenido:

```
<?php
$resultDir = __DIR__ . "/results/";
$f = $_GET['f'] ?? '';
$f = basename($f);
$path = $resultDir . $f;

if (!$f || !file_exists($path)) {
  http_response_code(404);
  exit("Not found");
}

header('Content-Type: text/markdown; charset=utf-8');
header('Content-Disposition: attachment; filename="' . $f .
'"');
readfile($path);
```

Fase 6 — Permisos y seguridad mínima del servicio

20) Permisos finales (muy importante)

```
sudo chown -R www-data:www-data /var/www/forense-ai
sudo find /var/www/forense-ai -type d -exec chmod 750 {} \;
sudo find /var/www/forense-ai -type f -exec chmod 640 {} \;
sudo systemctl reload apache2
```

21) Importante: Ollama solo local

Comprueba que Ollama no está expuesto:

- El servicio web llama a ollama run localmente.
- No hace falta abrir puertos de Ollama.