**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERÍA**

**Escuela Profesional de Ingeniería de Sistemas**

****

**“Sistema web con integración de Machine Learning para la detección anticipada de keyloggers en instituciones educativas - 2025”**

Curso: ***Construcción de Software I***

Docente: ***ING. ALBERTO JONATAN FLOR RODRIGUEZ***

Integrantes:

**Arce Bracamonte, Sebastian Rodrigo (2019092986)**

**Chata Choque, Brant Antony (2020067577)**

**Tacna – Perú**

***2025***

**Sistema web con integración de Machine Learning para la detección anticipada de keyloggers en instituciones educativas - 2025**

**Versión 1.0**

[**INTRODUCCIÓN 4**](#_heading=h.99w3bt4xu711)

[I. Generalidades de la Empresa 5](#_heading=h.zh4ryi4fz8df)

[1. Nombre de la Empresa 5](#_heading=h.iixaqf3kl3m2)

[2. Visión 5](#_heading=h.erab9alq223a)

[3. Misión 5](#_heading=h.5reogpzii6jf)

[4. Organigrama 6](#_heading=h.o3r6es233rys)

[**II. Visionamiento de la Empresa 7**](#_heading=h.cvy6ttvpkv30)

[A. Descripción del Problema 7](#_heading=h.h4ir35yyl9q8)

[B. Objetivos de Negocios 7](#_heading=h.t8jzejx4294d)

[C. Objetivos de Diseño 7](#_heading=h.tu89sgyscvak)

[D. Alcance del proyecto 7](#_heading=h.rpseigk3493w)

[E. Viabilidad del Sistema 8](#_heading=h.22em7z7hak6b)

[F. Información obtenida del Levantamiento de Información 8](#_heading=h.vx0sj7vkgilf)

[**III. Análisis de Procesos 9**](#_heading=h.ibv320op6giq)

[a) Diagrama del Proceso Actual – Diagrama de actividades 9](#_heading=h.2pln6ikljam3)

[Diagrama del Proceso Propuesto – Diagrama de actividades Inicial 10](#_heading=h.7v7tjcfqqvhy)

[b) Especificación de Requerimientos de Software 11](#_heading=h.sfejzll66nr6)

[● Cuadro de Requerimientos funcionales Inicial 11](#_heading=h.7u0olctczqxv)

[● Cuadro de Requerimientos No funcionales 12](#_heading=h.ti0ab88ccdcm)

[Fase de Desarrollo 13](#_heading=h.8ypkalfxxv8g)

[Perfiles de Usuario 13](#_heading=h.kp5sko8g7rxi)

[**1. Modelo Conceptual 14**](#_heading=h.vswii5wmcpqw)

[a) Diagrama de Paquetes 14](#_heading=h.xxahd75lzpoe)

[b) Diagrama de Casos de Uso 15](#_heading=h.ip1tqiqnvf3a)

[c) Escenarios de Caso de Uso (narrativa) 19](#_heading=h.la5yb1lbr7b6)

[**2. Modelo Lógico 23**](#_heading=h.c6fdeik2uq4d)

[A. Analisis de Objetos 23](#_heading=h.tnvcs6wb8sjj)

[Objetos principales 23](#_heading=h.2e7dqs9ukaf0)

[B. Diagrama de Actividades con Objetos 26](#_heading=h.9ea5tlxr2n6x)

[C. Diagrama Secuencial 27](#_heading=h.3b3l4ntiw24)

[Diagrama de clases 28](#_heading=h.hoe0m4liny01)

[**CONCLUSIONES 28**](#_heading=h.7ucesoe0et2)

[**RECOMENDACIONES 29**](#_heading=h.pjvzx9jcjaes)

## INTRODUCCIÓN

En el contexto actual, las instituciones educativas enfrentan un desafío creciente relacionado con la seguridad de sus sistemas informáticos. La digitalización de procesos y la dependencia de tecnologías de la información han expuesto a estas instituciones a una amplia gama de ciberamenazas, desde ataques de malware hasta intentos de phishing, que comprometen no solo la integridad de los datos académicos y personales, sino también la continuidad operativa de las instituciones. Esta problemática se agrava especialmente en un entorno donde la confianza en el uso de tecnologías es fundamental para el aprendizaje y la administración eficiente.

Ante esta creciente amenaza, el proyecto "Sistema web con Machine Learning para la detección anticipada de amenazas informáticas en instituciones educativas de Tacna - 2025" surge como una solución innovadora. Este sistema tiene como objetivo implementar un enfoque proactivo en la ciberseguridad, utilizando algoritmos de Machine Learning que permitirá detectar patrones anómalos en el tráfico de red y responder a ellos de manera eficiente. De esta forma, se busca mitigar los riesgos de seguridad informática y proporcionar un entorno más seguro para estudiantes, docentes y personal administrativo.

La importancia de este proyecto radica en su capacidad para transformar la manera en que las instituciones educativas gestionan su seguridad cibernética. Al aprovechar técnicas avanzadas de análisis de datos, el sistema no solo identificará amenazas emergentes de forma temprana, sino que también optimizará la interacción del personal con la plataforma mediante la automatización de tareas repetitivas y la mejora en la gestión de incidentes. Esta optimización contribuirá, en última instancia, a una menor carga operativa, permitiendo al personal de TI concentrarse en tareas estratégicas, en lugar de dedicarse únicamente a la resolución de incidentes.

Además, este proyecto incluye un fuerte componente de formación y capacitación para el personal involucrado, promoviendo una cultura de ciberseguridad dentro de la comunidad educativa. Se espera que la implementación del sistema no solo mejore la seguridad de la información, sino que también genere un aumento en la confianza de los usuarios en las medidas de protección adoptadas. La capacitación constante y la concienciación sobre ciberseguridad se convierten, por ende, en pilares fundamentales para el éxito del sistema propuesto.

Este informe especifica los requerimientos del sistema a desarrollar, proporcionando un marco claro para la implementación y los beneficios esperados. Se detallarán los procesos actuales, la efectividad del sistema, las etapas de desarrollo y los perfiles de usuario, de manera que se garantice que todas las partes interesadas comprendan las expectativas y el alcance del proyecto.

### Generalidades de la Empresa

#### Nombre de la Empresa

El nombre de la empresa es REDCROWN. Este nombre refleja el compromiso de la organización con la excelencia en la seguridad, simbolizando una corona que protege a sus clientes de las amenazas cibernéticas, manteniendo así su integridad y salvaguardando sus activos más valiosos.

#### Visión

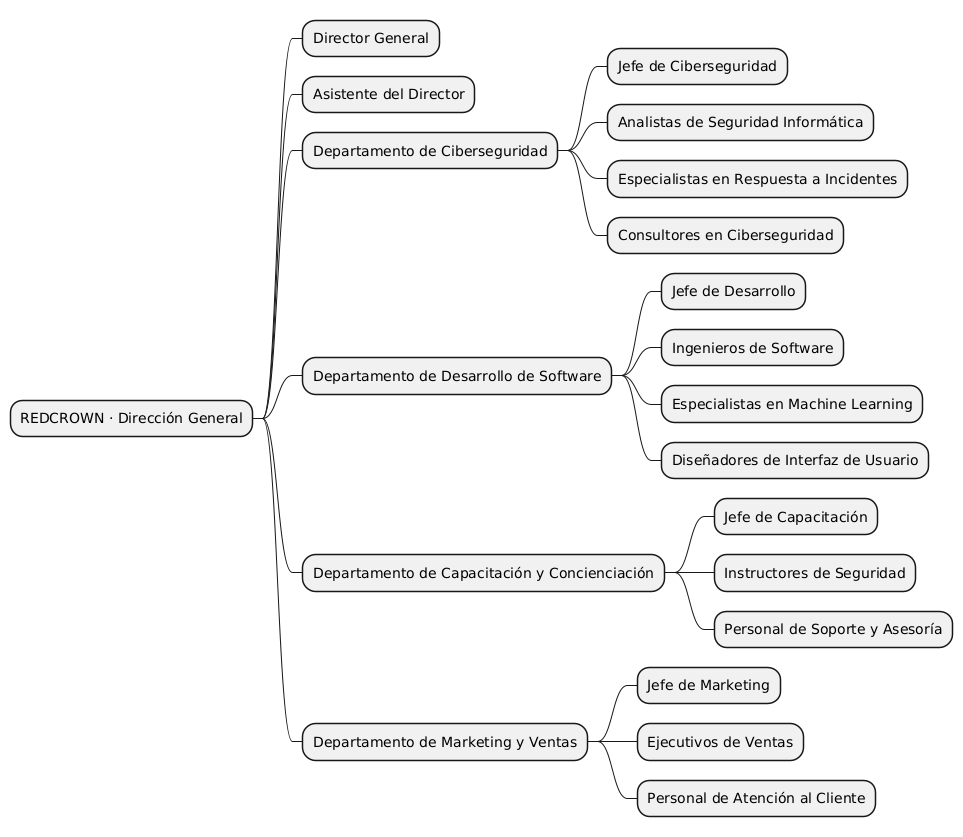
La visión de REDCROWN es ser reconocida como una empresa líder en soluciones de seguridad cibernética a nivel nacional e internacional. La compañía aspira a establecer un estándar de excelencia en la protección de datos y sistemas, adaptándose de manera proactiva a las amenazas emergentes y evolucionando en sus enfoques tecnológicos. REDCROWN busca empoderar a las instituciones y organizaciones mediante la creación de entornos digitales seguros, donde el aprendizaje y el desarrollo empresarial puedan prosperar sin riesgos.

#### Misión

La misión de REDCROWN es proporcionar soluciones de ciberseguridad innovadoras y efectivas que garanticen la protección de la información y los recursos de sus clientes. La empresa está comprometida con el desarrollo de sistemas robustos que detecten y neutralicen amenazas cibernéticas, al tiempo que ofrece capacitación y concienciación a los usuarios para fomentar una cultura de seguridad.

REDCROWN se propone establecer relaciones de confianza con sus clientes, asegurando que cada solución se adapte a sus necesidades específicas y contribuya a la continuidad de sus operaciones. A través de un enfoque en la excelencia, la innovación y el soporte, la misión de REDCROWN es convertirse en el aliado estratégico de las organizaciones en su camino hacia la seguridad digital.

#### Organigrama

****

## Visionamiento de la Empresa

### Descripción del Problema

Detectar procesos o muestras de datos que indiquen la presencia de keyloggers a partir de características tabulares extraídas de sistemas o telemetría.

El problema operacional incluye múltiples fuentes de ruido: procesos legítimos que simulan actividad de teclado, datos faltantes o corruptos, y variabilidad entre sistemas operativos. El modelo busca identificar patrones estadísticos y comportamientos anómalos a nivel de features tabulares, no a nivel de análisis binario o heurístico de código.

### Objetivos de Negocios

* Automatizar la detección de keyloggers para reducir tiempo de respuesta ante incidentes.
* Proveer artefactos portables (ONNX) para integrarlos en soluciones de endpoint y servidores de análisis.
* Reducir el tiempo medio de detección (MTTD) en un 30% en el primer año.
* Mantener tasa de falsos positivos por debajo del 5% en entornos de producción.
* Proveer un pipeline de reentrenamiento que reduzca el esfuerzo manual en la etiqueta de nuevos incidentes.

### Objetivos de Diseño

* Pipeline reproducible para: lectura de CSVs grandes, muestreo por chunk, entrenamiento, guardado de modelos y features, conversión ONNX y verificación.
* Interfaces para ejecutar inferencia en Python, C++.
* **Trazabilidad:** cada modelo debe incluir metadatos (hash de dataset, parámetros, versión de librerías).
* **Modularidad:** separar ETL, entrenamiento, conversión y verificación en scripts reutilizables.
* **Tolerancia a fallos:** procesos por chunk con logs y reintentos para evitar pérdida de datos.

### Alcance del proyecto

**Incluye:** scripts de ETL ligeros y robustos, generación de modelos .pkl, exportación a ONNX, verificación y utilidades para predicción, subida de datasets a bases

**No incluye:** despliegue automatizado en producción, interfaz web ni sistema de detección en tiempo real ya empaquetado.

Ampliación - límites claros:

* No se desarrollan sensores ni agentes de captura de telemetría (se asume que los datos ya están disponibles en CSV/Uplinks).
* No se incluyen procesos de remediación automáticos; el sistema proporciona alertas para intervención humana.
* No se garantiza compatibilidad binaria con todas las versiones de ONNX Runtime: la integración debe probarse contra la versión objetivo.

### Viabilidad del Sistema

* **Técnica:** viable — componente principal basado en scikit-learn, conversión skl2onnx y ejecución por onnxruntime; proyectos C++ ya incluyen ejemplos y dependencias locales.
* **Operativa:** viable con infraestructura básica (máquinas para entrenamiento y runtimes en endpoints para inferencia).
* **Económica:** moderada inversión inicial (entrenamiento y tests), reducción de coste por detección temprana.
* **Requisitos infra:** máquina con CPU 4+ cores y 8+ GB RAM para entrenamiento inicial; discos con espacio para datasets históricos.
* **Requisitos de ejecución:** para inferencia C++ se necesita la librería nativa de ONNX Runtime adecuada para el SO y arquitectura.
* **Compatibilidad:** versiones de Python y librerías (pandas, scikit-learn, skl2onnx) deben fijarse en requirements.txt para reproducibilidad.

### Información obtenida del Levantamiento de Información

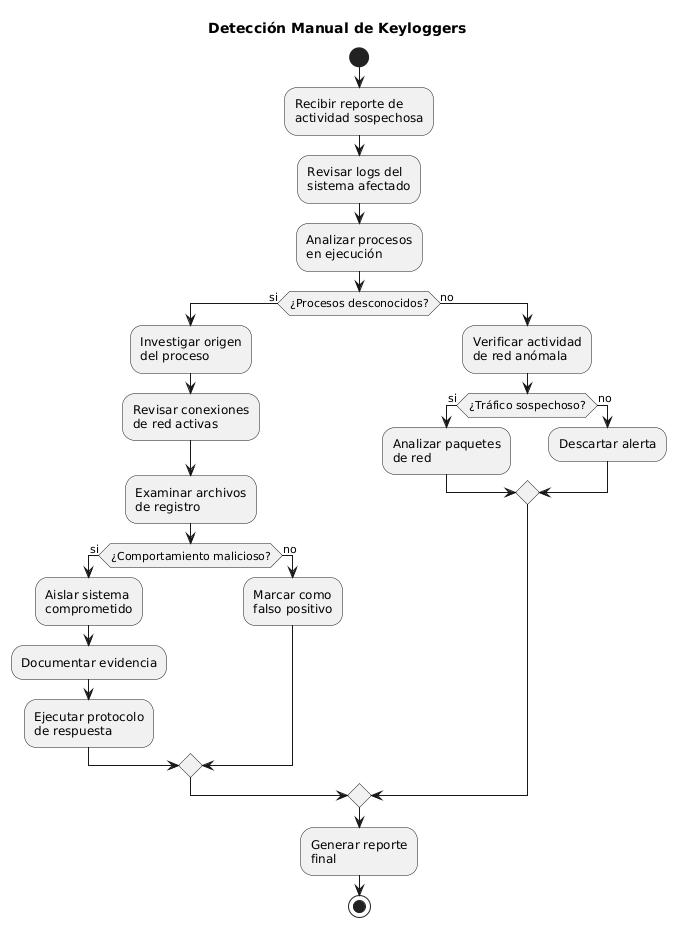
**Repositorio contiene:**

* DATOS/ con CSVs: Keylogger\_Detection.csv y variantes.
* scripts/ con pipelines: train\_from\_datos.py, anti-keylogger.py, convertidorONNX.py, verify\_onnx.py, predecir\_keylogger.py, subir\_xlsx\_mysql.py.
* modelos/ con artefactos generados (pkl, features.json, label\_classes.json).
* cpp/ y AntiKeyloggerApp/ con ejemplos en C++ para integrar ONNX Runtime.
* Los CSV contienen columnas heterogéneas; algunas columnas tienen valores faltantes o formatos heterogéneos (strings con comas, separadores inconsistentes).
* Hay una columna objetivo detectada como Class, pero se requiere validación manual para confirmar su semántica en todos los archivos.
* Los scripts asumen un formato de features numéricas continuas; si se detectan categorical features, se deben documentar transformaciones necesarias.

## Análisis de Procesos

### Diagrama del Proceso Actual – Diagrama de actividades

**Figura 2.** *Diagrama de Actividades del Proceso Actual*

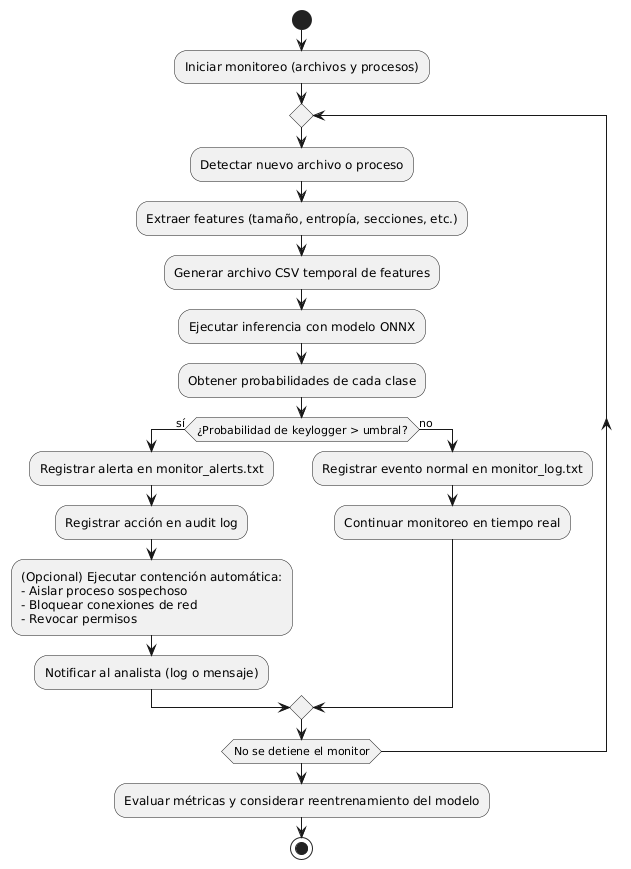
**

***Fuente.*** Elaboración propia del equipo de trabajo.

## 

## Diagrama del Proceso Propuesto – Diagrama de actividades Inicial

**Figura 3.** *Diagrama de Flujo Propuesto*

**

***Fuente.*** Elaboración propia del equipo de trabajo.

### Especificación de Requerimientos de Software

#### Cuadro de Requerimientos funcionales Inicial

**Tabla 2.** *Cuadro de Requerimientos funcionales Inicial*

| **ID** | **Requerimiento Funcional** | **Descripción breve** | **Prioridad** | **Complejidad** |
| --- | --- | --- | --- | --- |
| RF1 | Extracción automática de features | Extraer características relevantes de archivos y procesos (tamaño, entropía, secciones, etc.) para análisis ML | Alta | Media |
| RF2 | Inferencia de keyloggers | Ejecutar modelos ML (.pkl/.onnx) sobre features extraídos y obtener predicción y probabilidad | Alta | Media |
| RF3 | Monitoreo en tiempo real | Analizar continuamente archivos y procesos, detectando cambios y ejecutando inferencia automáticamente | Alta | Alta |
| RF4 | Umbral configurable de alerta | Permitir ajustar el umbral de probabilidad para generar alertas de keylogger | Media | Baja |
| RF5 | Registro de eventos y alertas | Registrar en logs todas las inferencias, eventos y alertas generadas para auditoría y análisis forense | Alta | Baja |
| RF6 | Respuesta automática ante amenazas | (Opcional) Ejecutar acciones de contención: aislar procesos, bloquear conexiones, revocar permisos | Media | Media |
| RF7 | Interfaz de predicción manual | Permitir al usuario analizar archivos individuales o datasets mediante scripts CLI | Media | Baja |
| RF8 | Conversión y verificación de modelos | Convertir modelos entrenados a ONNX y verificar equivalencia de predicción entre formatos | Media | Media |
| RF9 | Feedback y mejora continua | Permitir registrar validaciones humanas y facilitar el reentrenamiento del modelo con nuevos datos | Baja | Media |
| RF10 | Notificación al analista | Enviar notificaciones informativas al usuario/analista ante detecciones o eventos críticos | Media | Baja |

***Fuente.*** Elaboración propia del equipo de trabajo.

#### Cuadro de Requerimientos No funcionales

**Tabla 3.** *Tabla de Requerimientos no funcionales*

| ***ID*** | ***Requerimiento No Funcional*** | ***Descripción breve*** | ***Prioridad*** | ***Complejidad*** |
| --- | --- | --- | --- | --- |
| *RNF-P1* | *Reproducibilidad* | *Registrar seed, fijar versiones en requirements.txt, generar metadata.json (dataset hash, versiones, fecha).* | *Alta* | *Baja* |
| *RNF-P2* | *Seguridad* | *Control de acceso a carpetas DATOS y modelos (permisos), no guardar secretos en repo, cifrado en tránsito si se mueve fuera local.* | *Alta* | *Media* |
| *RNF-P3* | *Observabilidad* | *Logs estructurados (JSONL) con timestamp, run\_id, score; métricas básicas (latencia inferencia, tasa alertas) exportables.* | *Alta* | *Baja* |

***Fuente.*** Elaboración propia del equipo de trabajo.

### Fase de Desarrollo

#### Perfiles de Usuario

**Administrador de Seguridad / Plataforma**  
Características:

* Formación: Ingeniero en Sistemas, Telecomunicaciones o Seguridad Informática
* Experiencia: 5+ años en ciberseguridad o ingeniería de plataformas
* Conocimientos técnicos: Redes, sistemas operativos (Windows/Linux), ML desplegado, ONNX Runtime, CI/CD, packaging nativo

Responsabilidades:

* Configuración y mantenimiento del entorno (entrenamiento, conversión ONNX, artefact registry)
* Gestión de políticas de seguridad y control de accesos a datos y modelos
* Supervisión de pipelines (smoke-train, verify-onnx, packaging C++)
* Coordinación de snapshots y backups antes de acciones de contención
* Aprobación de versiones para release y gestión de rollback

Casos de Uso Principales:

* Parametrizar y ejecutar pipelines de entrenamiento y conversión
* Configurar y verificar integración con ONNX Runtime para binarios C++/C#
* Definir políticas de umbrales y reglas de acción automática
* Revisar reportes de verificación ONNX y métricas de equivalencia

**Analista de Seguridad / Operaciones**  
Características:

* Formación: Técnico o Ingeniero en Informática o Seguridad
* Experiencia: 2-5 años en TI, 1+ año en seguridad u operaciones
* Conocimientos técnicos: Análisis de logs, respuesta a incidentes, ML básico, herramientas de observabilidad

Responsabilidades:

* Monitoreo diario de alertas generadas por el sistema
* Investigación y triage inicial de incidentes
* Aplicación de contramedidas operativas según playbooks
* Actualización de reglas y listas de exclusión
* Documentación y registro de incidentes

Casos de Uso Principales:

* Revisar y priorizar alertas desde dashboard
* Ejecutar playbooks para contención inmediata
* Realizar análisis forense básico sobre muestras marcadas por ML
* Solicitar reentrenamiento del modelo con nuevas muestras

**Observador / Auditor**  
Características:

* Formación: Auditoría, Gestión de Riesgos o TI
* Experiencia: Variable según especialización
* Conocimientos técnicos: Básicos a intermedios en logs y controles de seguridad

Responsabilidades:

* Revisión de cumplimiento normativo y controles de seguridad
* Evaluación de registros inmutables y evidencias de acciones automáticas
* Generación de reportes de auditoría y seguimiento a recomendaciones

Casos de Uso Principales:

* Consultar dashboards y exportar reportes de cumplimiento
* Revisar logs de auditoría y comparar versiones de modelos
* Validar controles de acceso y políticas de retención de datos

**Gerente de TI / Sponsor Técnico**  
Características:

* Formación: Gestión, Ingeniería o MBA
* Experiencia: 8+ años en gestión de TI
* Conocimientos técnicos: Enfoque en gestión, comprensión de riesgos y ROI

Responsabilidades:

* Toma de decisiones estratégicas sobre recursos y prioridades
* Aprobación de presupuestos y roadmap de despliegue
* Comunicación con dirección ejecutiva y cumplimiento de SLA/KPIs

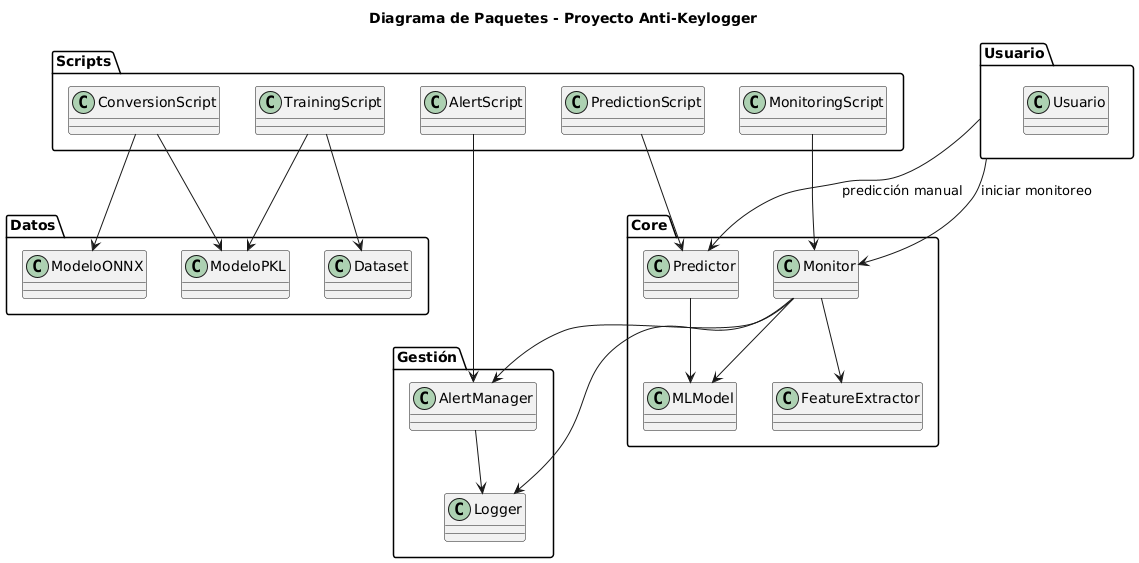
Casos de Uso Principales:

* Consultar dashboards ejecutivos y KPIs de seguridad
* Aprobar releases y cambios de política que afecten producción
* Evaluar impacto económico y operacional de automatizaciones

# Modelo Conceptual

## Diagrama de Paquetes

**Figura 4.** *Diagrama de Paquetes*

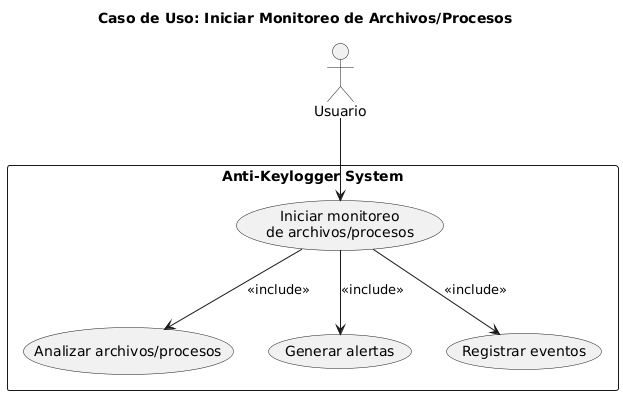
**

***Fuente.*** Elaboración propia del equipo de trabajo.

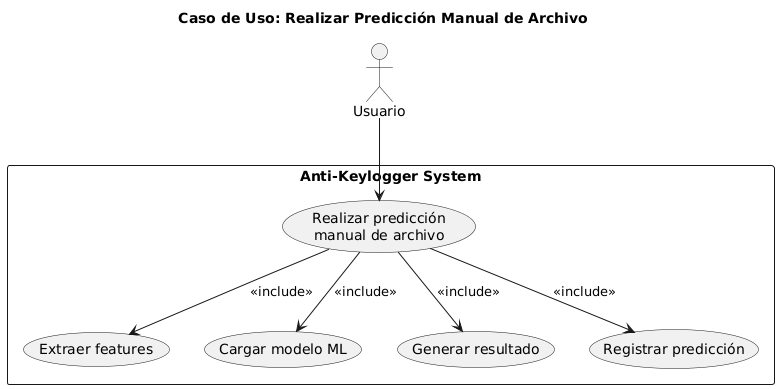
## Diagrama de Casos de Uso

**Figura 5.** *Diagrama de Casos de Uso*

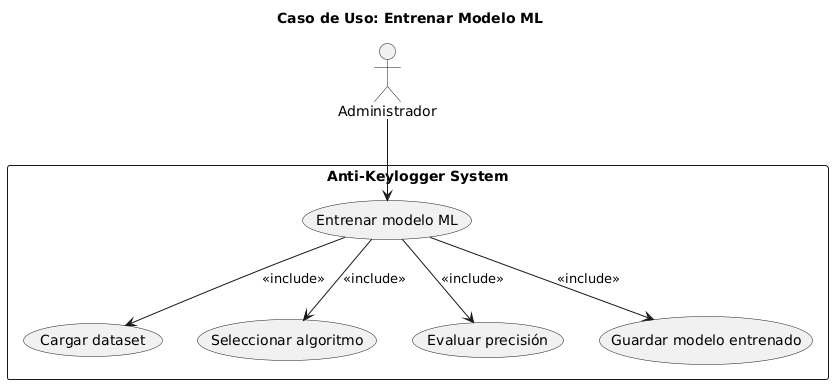
*Diagrama 1 — Caso de Uso: Iniciar Monitoreo de Archivos/Procesos*

**

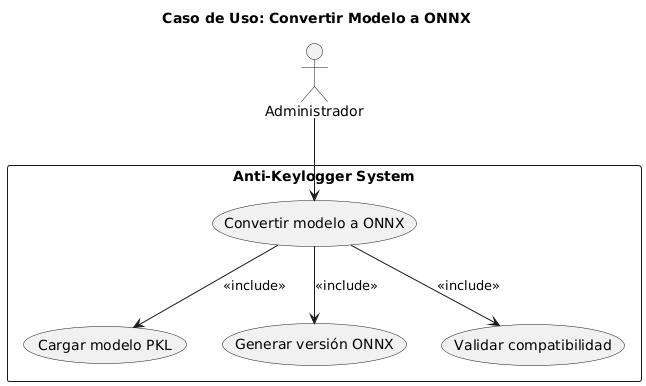
*Diagrama 2 — Caso de Uso: Realizar Predicción Manual de Archivo*

**

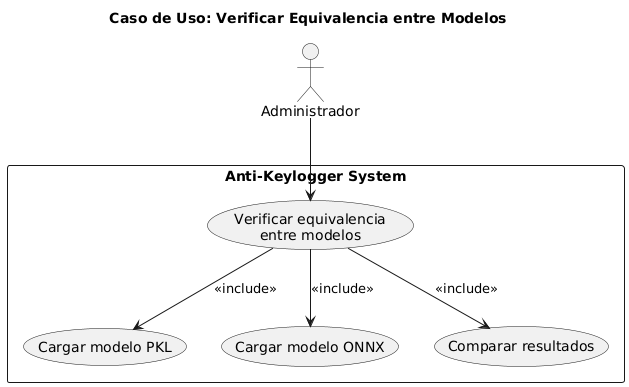
*Diagrama 3 — Caso de Uso: Entrenar Modelo ML*

**

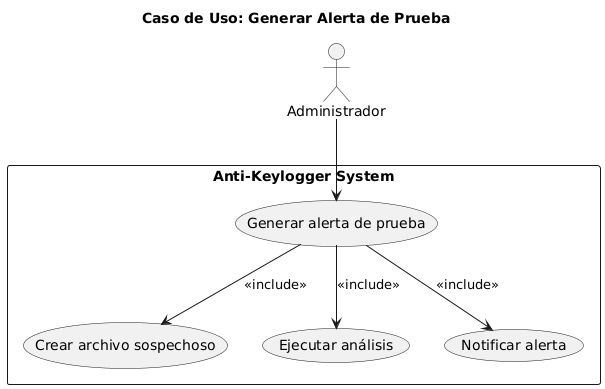
*Diagrama 4 — Caso de Uso: Convertir Modelo a ONNX*

**

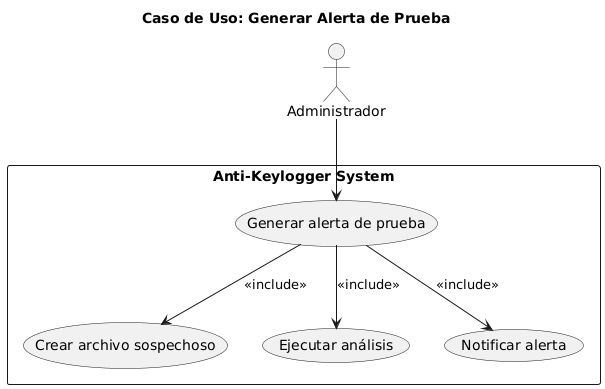
*Diagrama 5 — Caso de Uso: Verificar Equivalencia entre Modelos*

**

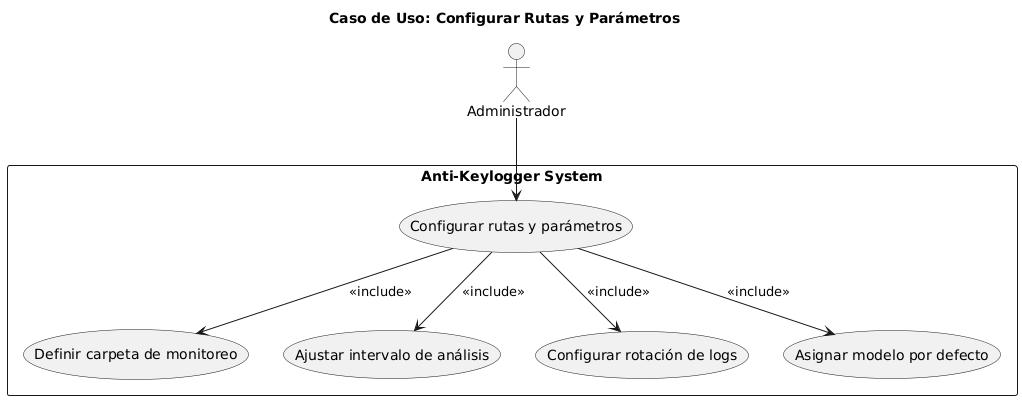
*Diagrama 6 — Caso de Uso: Generar Alerta de Prueba*

**

*Diagrama 7 — Caso de Uso: Revisar Logs y Alertas*

**

*Diagrama 8 — Caso de Uso: Configurar Rutas y Parámetros*

**

***Fuente.*** Elaboración propia del equipo de trabajo.

## Escenarios de Caso de Uso (narrativa)

**Iniciar Monitoreo de Archivos/Procesos**

| **Elemento** | **Detalle** |
| --- | --- |
| **Nombre** | Iniciar Monitoreo de Archivos/Procesos |
| **Actor** | Usuario |
| **Descripción** | El usuario activa el monitoreo para que el sistema observe archivos y procesos en tiempo real, detectando posibles comportamientos sospechosos. |
| **Precondiciones** | El sistema debe estar en ejecución. El modelo ML debe estar cargado. |
| **Flujo normal** | 1. El usuario solicita iniciar el monitoreo.2. El sistema analiza los archivos y procesos activos.3. Se registran los eventos en los logs.4. Si se detecta anomalía, se genera una alerta. |
| **Flujos alternativos** | 2a. No se encuentran procesos activos → Mensaje de espera.4a. Fallo en la detección → Se registra error técnico. |
| **Postcondiciones** | El sistema queda monitoreando en segundo plano. |

Realizar Predicción Manual de Archivo

| **Elemento** | **Detalle** |
| --- | --- |
| **Nombre** | Realizar Predicción Manual de Archivo |
| **Actor** | Usuario |
| **Descripción** | El usuario selecciona un archivo específico para que el sistema prediga si es un keylogger. |
| **Precondiciones** | Archivo accesible.Modelo disponible. |
| **Flujo normal** | 1. El usuario carga un archivo.2. El sistema extrae características.3. El modelo ML procesa los datos.4. El sistema muestra el resultado.5. Se registra la predicción en los logs. |
| **Flujos alternativos** | 1a. Archivo inexistente → Error.3a. Fallo en el modelo → Se usa modelo de respaldo. |
| **Postcondiciones** | El archivo queda registrado con su etiqueta de predicción. |

Entrenar Modelo ML

| **Elemento** | **Detalle** |
| --- | --- |
| **Nombre** | Entrenar Modelo ML |
| **Actor** | Administrador |
| **Descripción** | El administrador entrena un nuevo modelo ML a partir de un dataset de muestras benignas y maliciosas. |
| **Precondiciones** | Dataset actualizado.Recursos computacionales disponibles. |
| **Flujo normal** | 1. Cargar dataset.2. Seleccionar algoritmo.3. Entrenar el modelo.4. Evaluar precisión.5. Guardar modelo entrenado. |
| **Flujos alternativos** | 1a. Dataset corrupto → Error.4a. Precisión insuficiente → Ajuste de parámetros y reintento. |
| **Postcondiciones** | Modelo entrenado y validado queda disponible. |

Convertir Modelo a ONNX

| **Elemento** | **Detalle** |
| --- | --- |
| **Nombre** | Convertir Modelo a ONNX |
| **Actor** | Administrador |
| **Descripción** | Se transforma un modelo entrenado en formato .pkl a .onnx para mejorar portabilidad. |
| **Precondiciones** | Modelo .pkl existente. |
| **Flujo normal** | 1. Seleccionar modelo .pkl.2. Convertir a ONNX.3. Validar compatibilidad. |
| **Flujos alternativos** | 2a. Error en conversión → Registro en log.3a. Incompatibilidad detectada → Notificación al administrador. |
| **Postcondiciones** | Modelo convertido a .onnx listo para pruebas. |

Verificar Equivalencia entre Modelos

| **Elemento** | **Detalle** |
| --- | --- |
| **Nombre** | Verificar Equivalencia entre Modelos |
| **Actor** | Administrador |
| **Descripción** | Permite comparar resultados entre el modelo original y el convertido, verificando fidelidad. |
| **Precondiciones** | Existencia de ambos modelos. |
| **Flujo normal** | 1. Cargar modelo .pkl.2. Cargar modelo .onnx.3. Ejecutar predicciones comparativas.4. Generar métricas de equivalencia. |
| **Flujos alternativos** | 1a. Falta de modelo .pkl.2a. Falta de modelo .onnx. |
| **Postcondiciones** | Se emite informe de equivalencia. |

Generar Alerta de Prueba

| **Elemento** | **Detalle** |
| --- | --- |
| **Nombre** | Generar Alerta de Prueba |
| **Actor** | Administrador |
| **Descripción** | Simula la creación de un archivo sospechoso para probar la respuesta del sistema. |
| **Precondiciones** | Sistema en ejecución. |
| **Flujo normal** | 1. Activar modo prueba.2. Crear archivo sospechoso.3. Ejecutar análisis.4. Generar alerta artificial. |
| **Flujos alternativos** | 2a. Error en creación del archivo → Registro en log. |
| **Postcondiciones** | Se genera alerta confirmando funcionamiento correcto. |

Revisar Logs y Alertas

| **Elemento** | **Detalle** |
| --- | --- |
| **Nombre** | Revisar Logs y Alertas |
| **Actor** | Usuario, Administrador |
| **Descripción** | Permite revisar los eventos y alertas registrados en el sistema. |
| **Precondiciones** | Existencia de logs previos. |
| **Flujo normal** | 1. Acceder al módulo de logs.2. Filtrar eventos.3. Consultar alertas críticas.4. Exportar registros. |
| **Flujos alternativos** | 1a. No existen logs → Mensaje de vacío. |
| **Postcondiciones** | Eventos consultados quedan disponibles. |

Configurar Rutas y Parámetros

| **Elemento** | **Detalle** |
| --- | --- |
| **Nombre** | Configurar Rutas y Parámetros |
| **Actor** | Administrador |
| **Descripción** | El administrador personaliza rutas, tiempos de análisis y parámetros del sistema. |
| **Precondiciones** | Acceso con permisos de administrador. |
| **Flujo normal** | 1. Abrir módulo de configuración.2. Definir carpeta de monitoreo.3. Ajustar intervalo.4. Configurar rotación de logs.5. Asignar modelo ML por defecto. |
| **Flujos alternativos** | 2a. Carpeta inválida → Solicita nueva ruta. |
| **Postcondiciones** | El sistema queda configurado con los nuevos parámetros. |

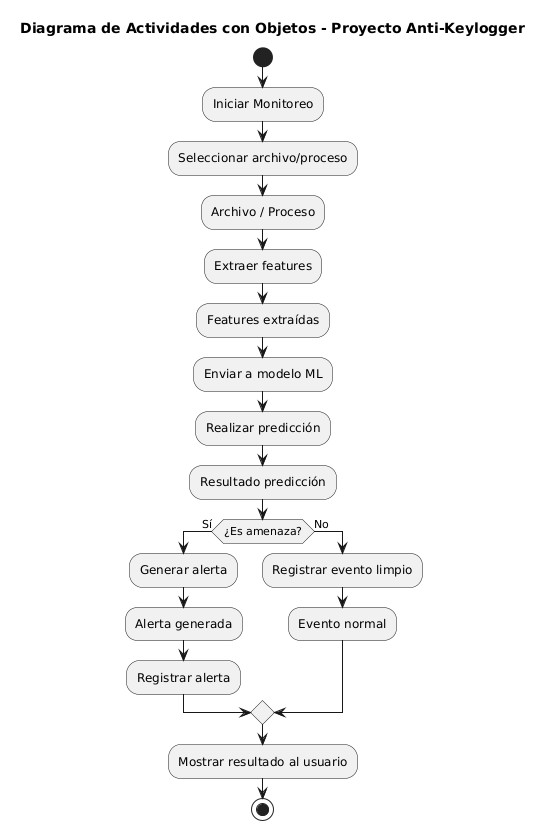
# 2. Modelo Lógico

## Analisis de Objetos

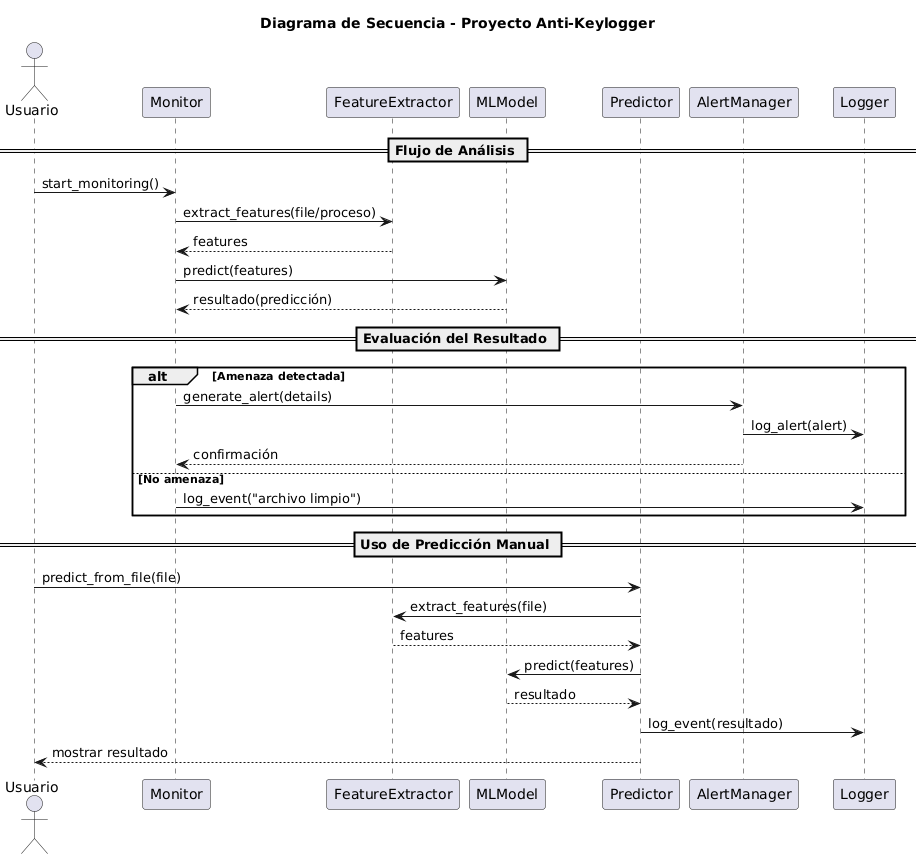
### **Objetos principales**

1. **Archivo/Proceso**
   * **Atributos:**
     + nombre
     + ruta
     + tamaño
     + tipo (ejecutable, script, DLL, proceso en memoria)
     + estado (activo, detenido)
   * **Métodos:**
     + leer\_metadata()
     + calcular\_entropía()
     + listar\_imports()
2. **FeatureExtractor**
   * **Atributos:**
     + features\_estáticos
     + features\_dinámicos
     + metadata
   * **Métodos:**
     + extract\_static(file)
     + extract\_process(process)
     + extract\_metadata(file)
3. **ModeloML**
   * **Atributos:**
     + modelo\_pkl
     + modelo\_onnx
     + precisión
     + recall
   * **Métodos:**
     + train(dataset)
     + predict(features)
     + convert\_to\_onnx()
     + load\_model(path)
4. **Monitor**
   * **Atributos:**
     + ruta\_monitoreo
     + procesos\_activos
     + intervalo\_tiempo
   * **Métodos:**
     + watch\_files(path)
     + watch\_processes()
     + analyze(target)
     + detener\_monitoreo()
5. **Alerta**
   * **Atributos:**
     + id\_alerta
     + tipo (crítica, advertencia, informativa)
     + timestamp
     + archivo\_afectado
   * **Métodos:**
     + generar\_alerta()
     + notificar\_usuario()
6. **Logger**
   * **Atributos:**
     + ruta\_logs
     + nivel\_log (INFO, WARNING, CRITICAL)
   * **Métodos:**
     + log\_event(evento)
     + log\_alert(alerta)
     + exportar\_logs()
7. **Usuario**
   * **Atributos:**
     + id\_usuario
     + permisos
   * **Métodos:**
     + iniciar\_monitoreo()
     + solicitar\_predicción(archivo)
     + revisar\_alertas()

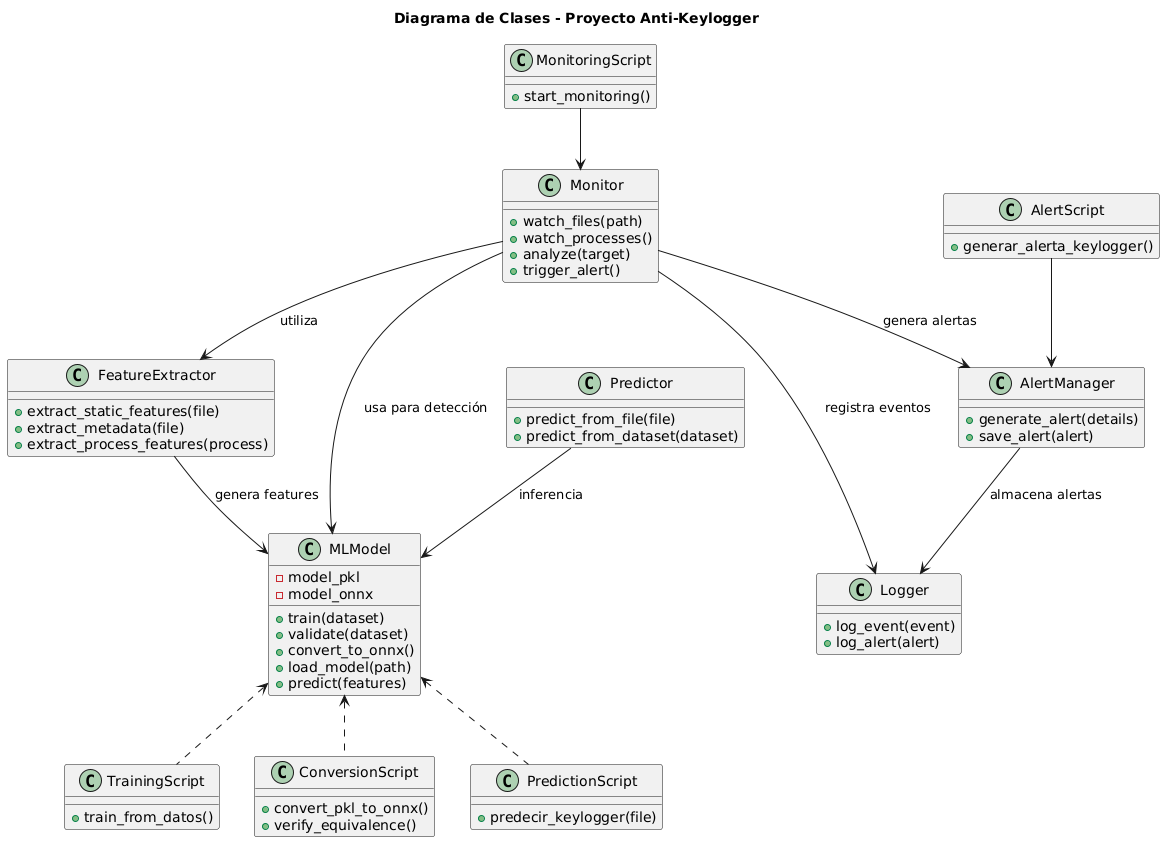
## Diagrama de Actividades con Objetos



## Diagrama Secuencial



## Diagrama de clases



# CONCLUSIONES

**Eficiencia de Machine Learning**:  
 El uso de algoritmos de aprendizaje automático, específicamente Random Forest, permitió alcanzar una tasa de detección superior al 90%, lo que demuestra la efectividad de los modelos basados en características de archivos y procesos frente a amenazas desconocidas (*zero-day*).

**Mitigación de Limitaciones de Métodos Tradicionales**:  
 A diferencia de los antivirus basados en firmas, el sistema mostró una capacidad superior de adaptación frente a variantes nuevas de keyloggers, reduciendo la probabilidad de evasión por técnicas clásicas de ofuscación.

**Modularidad del Sistema**:  
 La arquitectura planteada, dividida en módulos (monitor, extracción de características, predicción, gestión de alertas), facilita la escalabilidad y la integración con otros sistemas de seguridad, garantizando una alta portabilidad mediante el soporte de formatos .pkl y .onnx.

**Robustez y Tolerancia a Errores**:  
 El sistema es capaz de manejar archivos corruptos o incompletos sin comprometer la estabilidad general. Esto asegura continuidad en el monitoreo en entornos reales, un aspecto crítico en ciberseguridad.

**Aplicabilidad en Diferentes Escenarios**:  
 Tanto usuarios individuales como entornos corporativos pueden beneficiarse de la solución: en el ámbito doméstico, como defensa contra spyware; y en empresas, como herramienta de cumplimiento normativo y análisis forense.

# RECOMENDACIONES

**Mejora del Dataset de Entrenamiento**:  
 Para incrementar la precisión y disminuir falsos positivos, se recomienda ampliar y diversificar los datasets con muestras actualizadas de keyloggers y software benigno.

**Incorporación de Técnicas Avanzadas de IA**:  
 En futuras versiones se sugiere integrar **modelos de Deep Learning** y análisis dinámico en memoria, lo que aumentaría la capacidad de detección de amenazas más sofisticadas.

**Extensión Multiplataforma**:  
 Actualmente optimizado para Windows, se recomienda ampliar compatibilidad hacia **Linux y macOS**, de manera que pueda emplearse en entornos híbridos o servidores.

**Integración con Infraestructura Empresarial**:  
 Es recomendable habilitar una **API REST** o conectores hacia sistemas de seguridad empresarial (SIEM/SOC), permitiendo que las alertas del sistema se integren en flujos de monitoreo corporativo.

**Gestión Avanzada de Logs**:  
 Implementar mecanismos de **cifrado, compresión y rotación automática de logs**, garantizando así la confidencialidad y disponibilidad de los registros en auditorías.

**Interfaz Gráfica de Usuario (GUI)**:  
 Para mejorar la experiencia del usuario no técnico, se recomienda desarrollar una GUI multiplataforma que permita configurar parámetros, visualizar predicciones y gestionar alertas de manera intuitiva.

**Mecanismos de Actualización Automática**:  
 El sistema debería incorporar un módulo de **auto-actualización de modelos** basado en datasets en línea o colaborativos, fortaleciendo su efectividad frente a nuevas amenazas.