

UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERIA ESTADISTICA E INFORMATICA

ESCUELA PROFESIONAL DE INGENIERIA ESTADISTICA E INFORMATICA



PROYECTO FINAL: RECONOCIMINETO FACIAL

CURSO:

LENGUAJE DE PROGRAMACION III

DOCENTE:

DOC .BETO

ESTUDIANTES:

- RONALDHINO JINEZ INCACUTIPA
- VIDMAN RUIS ROQUE MAMANI

SEMESTRE III

PUNO – PERU

2025

1. Introducción

El presente proyecto consiste en el desarrollo de un **Sistema de Control de Asistencias (entrada y salida de estudiantes) mediante Reconocimiento Facial**, orientado al curso de **Gestión de Base de Datos**. El sistema permite registrar automáticamente la asistencia de los estudiantes utilizando técnicas de visión por computadora y almacenar la información en una base de datos estructurada, garantizando integridad, seguridad y facilidad de consulta.

Este proyecto busca reemplazar los métodos tradicionales de control de asistencia (listas manuales o tarjetas) por un sistema más confiable, rápido y difícil de manipular.

2. Objetivos del Proyecto

2.1 Objetivo General

Diseñar e implementar un sistema de control de asistencias basado en reconocimiento facial, gestionado mediante una base de datos relacional.

2.2 Objetivos Específicos

- Registrar estudiantes en una base de datos junto con sus datos biométricos.
- Capturar y entrenar un modelo de reconocimiento facial.
- Identificar automáticamente al estudiante al momento de su ingreso o salida.
- Registrar fecha y hora de entrada y salida en la base de datos.
- Garantizar la correcta gestión, almacenamiento y consulta de los datos.

3. Alcance del Sistema

El sistema está diseñado para:

- Uso académico (instituciones educativas).
- Controlar asistencia diaria de estudiantes.
- Registrar eventos de **entrada y salida**.

No incluye:

- Gestión de pagos o notas académicas.
- Reconocimiento facial en tiempo real a gran escala.

4. Tecnologías Utilizadas

4.1 Lenguaje de Programación

- **Python**: para el desarrollo del sistema de reconocimiento facial y la conexión con la base de datos.

4.2 Base de Datos

- **MySQL**: para el almacenamiento de información de estudiantes y registros de asistencia.

4.3 Librerías Principales

- OpenCV: captura y procesamiento de imágenes.
- OS / NumPy: manejo de archivos y matrices.
- Conector MySQL para Python.

4.4 Entorno de Desarrollo

- Visual Studio Code
- Entorno virtual Python (.venv)

5. Descripción General del Sistema

El sistema funciona en las siguientes etapas:

1. **Registro del estudiante:** se capturan imágenes del rostro y se almacenan como datos de entrenamiento.
2. **Entrenamiento del modelo:** se genera un modelo de reconocimiento facial.
3. **Reconocimiento:** el sistema identifica al estudiante mediante la cámara.
4. **Registro de asistencia:** se almacena en la base de datos la fecha, hora y tipo de evento (entrada o salida).

6. Diseño de la Base de Datos

El sistema implementa una base de datos relacional orientada a garantizar la integridad de la información, la trazabilidad de asistencias y la seguridad de los usuarios administradores.

6.1 Tablas Principales del Sistema

Tabla: personas

Almacena la información básica de cada estudiante registrado en el sistema. Cada registro se asocia directamente con los datos biométricos almacenados en el sistema de archivos.

Campo	Tipo	Descripción
id	INT (PK)	Identificador único del estudiante
Código	VARCHAR	Nombre completo del estudiante
dni	VARCHAR	
nombres	VARCHAR	
apellidos	VARCHAR	
celular	VARCHAR	
correo	VARCHAR	

El campo **id** es utilizado como etiqueta (label) en el entrenamiento y reconocimiento facial, asegurando la correspondencia entre la base de datos y el modelo biométrico.

Tabla: asistencia

Registra la **hora de entrada** del estudiante reconocida por el sistema.

Campo	Tipo	Descripción
id	INT (PK)	Identificador del registro
persona_id	INT (FK)	Referencia a personas(id)
fecha	DATE	Fecha de la asistencia
nombres		
apellidos		
hora_entrada	TIME	Hora de ingreso

Tabla: asistencia_salida

Registra la **hora de salida** del estudiante, complementando el control de asistencia.

Campo	Tipo	Descripción
id	INT (PK)	Identificador del registro
persona_id	INT (FK)	Referencia a personas(id)
nombres		
apellidos		
fecha	DATE	Fecha de salida
hora_salida	TIME	Hora de salida

Tabla: usuarios

Gestiona los usuarios del sistema con privilegios administrativos.

Campo	Tipo	Descripción
id_usuario	INT (PK)	Identificador del usuario
usuario	VARCHAR	Nombre de usuario
contrasena	VARCHAR	Contraseña cifrada (bcrypt)
nombre	VARCHAR	Nombre real
apellido	VARCHAR	Apellido
dni	VARCHAR	Documento de identidad
correo	VARCHAR	Correo electrónico
celular	VARCHAR	Número telefónico

La contraseña se almacena cifrada, garantizando la seguridad de acceso al sistema.

6.2 Estructura de Tablas

Tabla: estudiante

Campo	Tipo	Descripción
id_estudiante	INT (PK)	Identificador único
codigo	VARCHAR	Código del estudiante
nombres	VARCHAR	Nombres del estudiante
apellidos	VARCHAR	Apellidos del estudiante
estado	BOOLEAN	Activo / Inactivo

Tabla: asistencia

Campo	Tipo	Descripción
id_asistencia	INT (PK)	Identificador
id_estudiante	INT (FK)	Relación con estudiante
fecha	DATE	Fecha del registro

hora	TIME	Hora del registro
tipo	VARCHAR	Entrada o Salida

7. Gestión de la Base de Datos

7.1 Operaciones CRUD Implementadas

- **Crear:**
 - Registro de estudiantes (tabla personas).
 - Registro de usuarios administradores.
 - Registro automático de asistencias y salidas.
- **Leer:**
 - Consulta de reportes de asistencia por rango de fechas.
 - Listado de estudiantes y usuarios.
- **Actualizar:**
 - Edición de datos de usuarios administradores.
- **Eliminar:**
 - Eliminación lógica y física de estudiantes, incluyendo sus registros biométricos.

7.2 Integridad y Seguridad de Datos

- Uso de claves primarias y foráneas.
- Validación de registros duplicados.
- Protección del usuario administrador principal.
- Cifrado de contraseñas mediante hashing seguro.

8. Descripción de Archivos del Proyecto

- **app_principal.py:** Controla la interfaz gráfica principal del sistema (Tkinter), navegación por pestañas, control de hilos y acceso a los módulos funcionales.
- **database.py:** Capa de acceso a datos. Contiene todas las operaciones CRUD, consultas de reportes, validaciones y conexión a la base de datos MySQL.
- **capturar_rostro.py:** Captura imágenes faciales del estudiante, normaliza las imágenes y las almacena en la carpeta data/ID_persona.
- **entrenar_modelo.py:** Entrena el modelo de reconocimiento facial utilizando EigenFace o FisherFace según la cantidad de estudiantes registrados.
- **ReconocimientoFacial.py:** Ejecuta el proceso de reconocimiento facial para registrar entradas y salidas en la base de datos.
- **tablas.sql:** Script de creación de las tablas del sistema.

- **config.ini:** Almacena la geometría de la ventana para mejorar la experiencia del usuario.

9. Seguridad y Consideraciones Éticas

- Acceso restringido a la base de datos.
- Uso del reconocimiento facial solo con fines académicos.
- Protección de datos personales de los estudiantes.

10. Conclusiones

El sistema desarrollado demuestra la correcta aplicación de los conceptos de **Gestión de Base de Datos**, integrándolos con tecnologías de reconocimiento facial. Permite un control de asistencias automatizado, confiable y eficiente, reduciendo errores humanos y mejorando la gestión académica.

11. Recomendaciones

- Implementar cifrado de datos sensibles.
- Optimizar el reconocimiento facial para mayor precisión.
- Añadir reportes automáticos de asistencia.

12. Bibliografía

- Documentación oficial de Python.
- Manual de MySQL.
- OpenCV Documentation.

8. Aplicación del Sistema y Simulación de Escenarios: Problemas y Soluciones

8.1 Aplicación del Sistema en un Entorno Real

El sistema de control de asistencias por reconocimiento facial se aplica en un entorno institucional (universidades, empresas u organizaciones) donde es necesario registrar de forma automática y confiable las horas de ingreso y salida del personal o estudiantes. La aplicación integra una interfaz gráfica amigable (Tkinter), una base de datos relacional (MySQL) y técnicas de visión por computadora (OpenCV), permitiendo que el proceso de control de asistencia se realice sin contacto físico y con mínima intervención humana.

En la práctica, el flujo de uso es el siguiente:

1. Registro inicial de personas mediante captura de rostros.
2. Entrenamiento del modelo de reconocimiento facial.
3. Reconocimiento en tiempo real para registrar entrada o salida.
4. Almacenamiento automático de los datos en la base de datos.

5. Generación de reportes y exportación de información.

8.2 Simulación de Escenarios: Problemas y Soluciones

A continuación, se describen escenarios comunes que pueden presentarse durante el uso del sistema, junto con los problemas detectados y las soluciones implementadas.

Escenario 1: Usuario no reconocido correctamente

Problema:

El sistema identifica a una persona registrada como “Desconocido” debido a variaciones en iluminación, postura o calidad de la cámara.

Solución:

- Se aplica ecualización del histograma en las imágenes capturadas y en tiempo real.
- Se incrementa el número de imágenes de entrenamiento por persona.
- Se recomienda entrenar nuevamente el modelo después de agregar nuevos registros.

Escenario 2: Registro duplicado de asistencia en un mismo día

Problema:

Una persona puede intentar registrar su asistencia varias veces durante una misma sesión.

Solución:

- Se implementa un conjunto de control (`asistencia_registrada_sesion`) que evita registros duplicados.
- La base de datos valida que solo exista un registro de entrada y una salida por día y persona.

Escenario 3: Intento de eliminación del administrador principal

Problema:

Un usuario con permisos intenta eliminar o modificar al administrador principal del sistema.

Solución:

- Se protege al administrador mediante validación por DNI.
- Los campos críticos (usuario y DNI) se deshabilitan en modo edición.
- El sistema bloquea la eliminación del administrador mostrando un mensaje de error.

Escenario 4: Fallo durante el reconocimiento facial (cierre inesperado)

Problema:

El proceso de reconocimiento puede interrumpirse por cierre de la ventana o problemas con la cámara.

Solución:

- Se ejecuta el reconocimiento en un hilo independiente.

- Se utiliza un evento de control (stop_event) para detener el proceso de forma segura.
- Se liberan correctamente los recursos de la cámara y ventanas de OpenCV.

Escenario 5: Modelo no entrenado o archivo inexistente

Problema:

El usuario intenta iniciar el reconocimiento sin haber entrenado previamente el modelo.

Solución:

- El sistema valida la existencia de los archivos .xml del modelo.
- Se muestra un mensaje de error indicando que debe entrenarse el modelo antes de continuar.

Escenario 6: Eliminación de una persona registrada

Problema:

Al eliminar una persona, pueden quedar residuos de datos o imágenes que afecten futuros entrenamientos.

Solución:

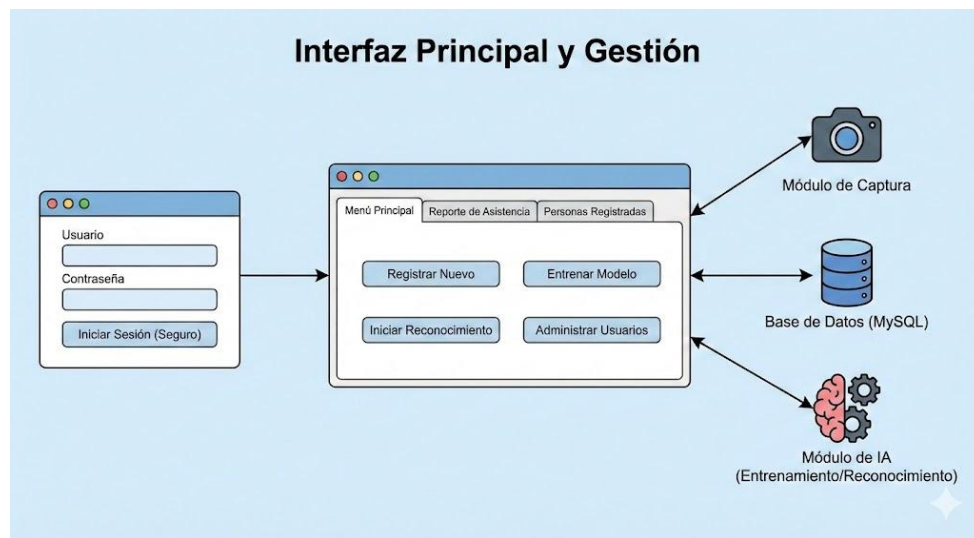
- Se elimina tanto el registro de la base de datos como la carpeta de imágenes asociada.
- Se solicita explícitamente reentrenar el modelo para reflejar los cambios.

8.3 Beneficios de la Simulación de Escenarios

La simulación de escenarios permite anticipar fallos, mejorar la robustez del sistema y garantizar su correcto funcionamiento en un entorno real. Gracias a este enfoque, el sistema presenta mayor confiabilidad, seguridad y facilidad de mantenimiento, cumpliendo con los requisitos funcionales y no funcionales esperados en un proyecto académico y profesional.

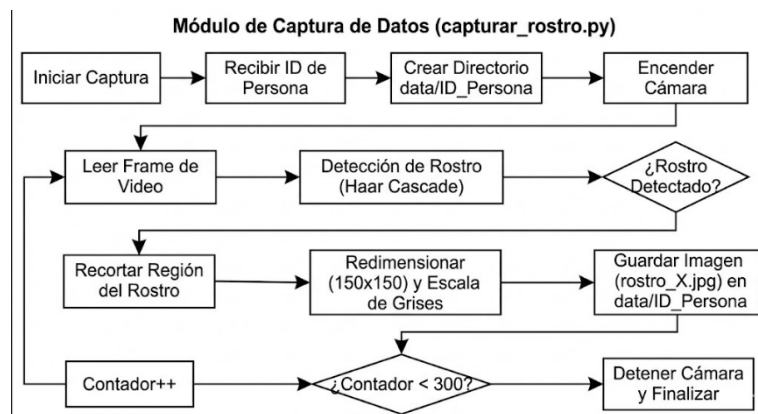
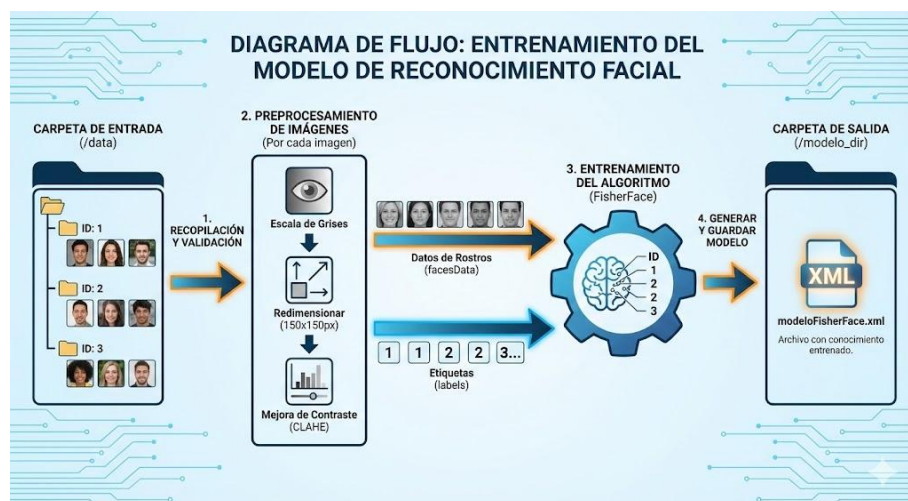
1. Interfaz Principal y Gestión

Gestionando el inicio de sesión seguro y conectando al usuario con los demás módulos del sistema, como la cámara, la base de datos y el motor de inteligencia artificial.



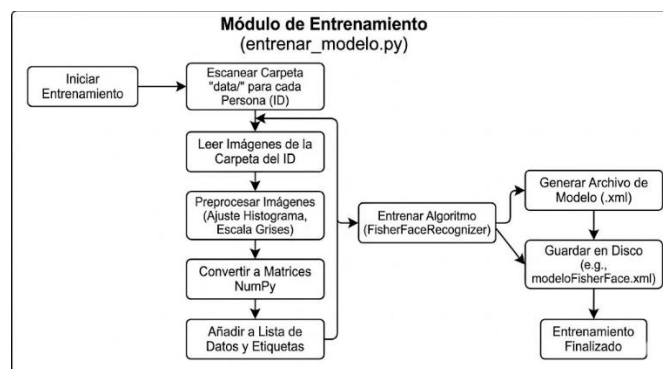
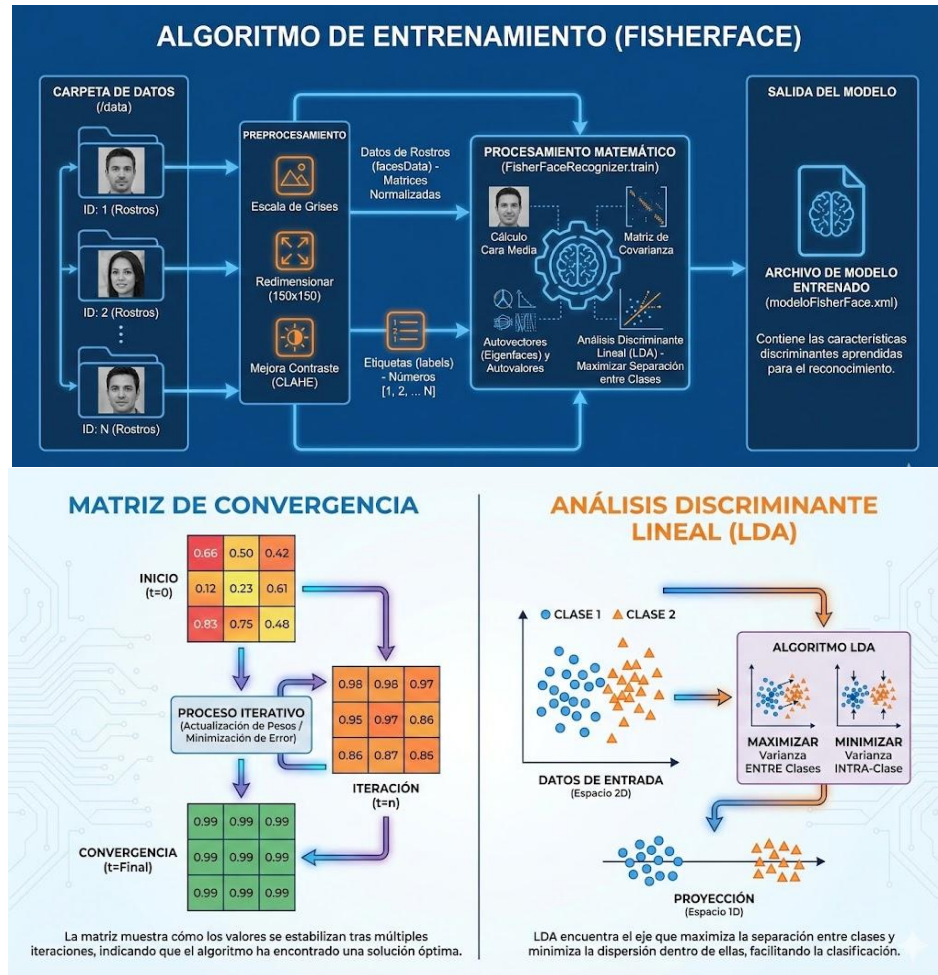
2. Módulo de Captura de Datos

Muestra el proceso de creación del dataset. Muestra cómo el sistema utiliza la cámara para detectar un rostro, lo recorta y lo procesa (convirtiéndolo a escala de grises y normalizando su tamaño) para luego guardar múltiples imágenes en una carpeta específica para esa persona.



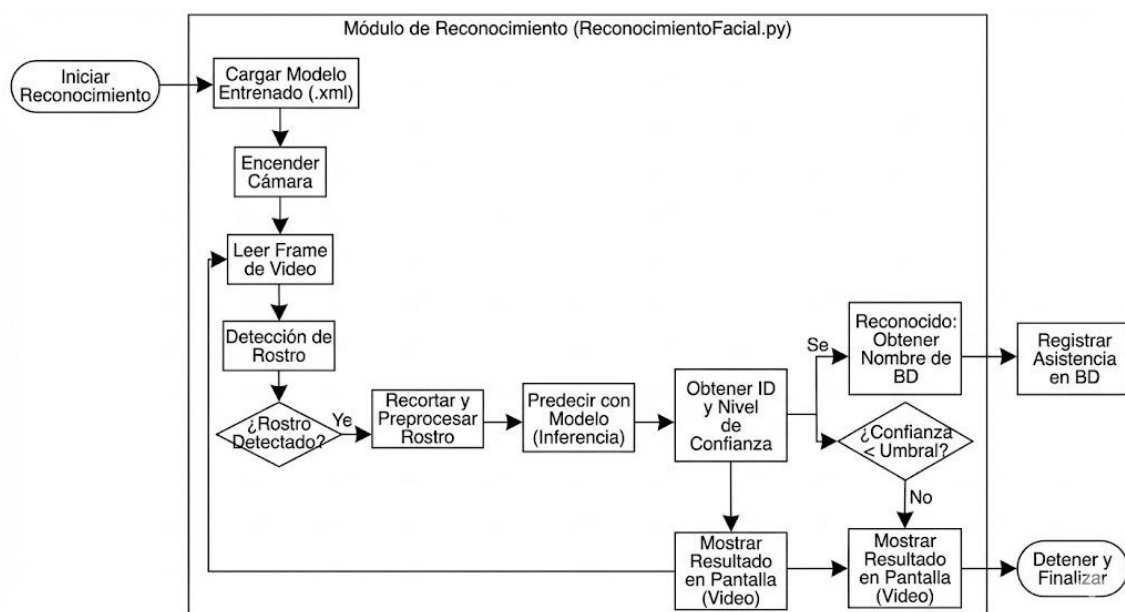
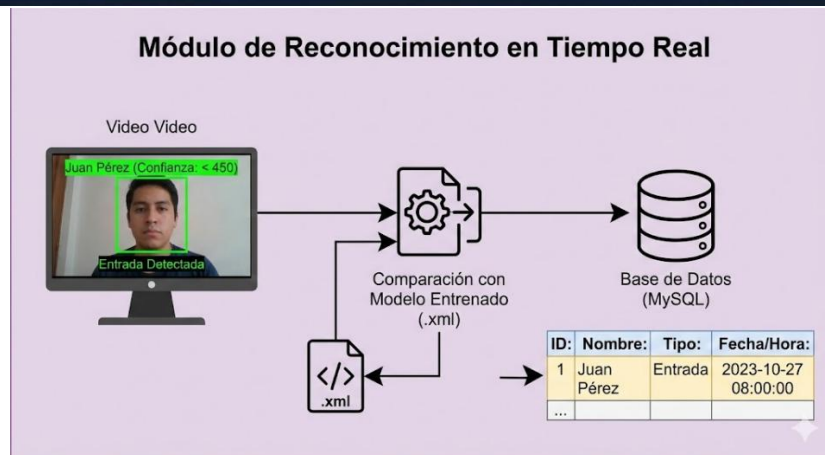
3. Módulo de Entrenamiento del

Esta imagen explica cómo se "enseña" al sistema a reconocer las caras. Muestra el proceso de tomar las carpetas con las fotos capturadas, pasarlas por el algoritmo de aprendizaje automático (FisherFace) y generar como resultado un archivo de modelo matemático (.xml) que contiene los patrones faciales aprendidos.

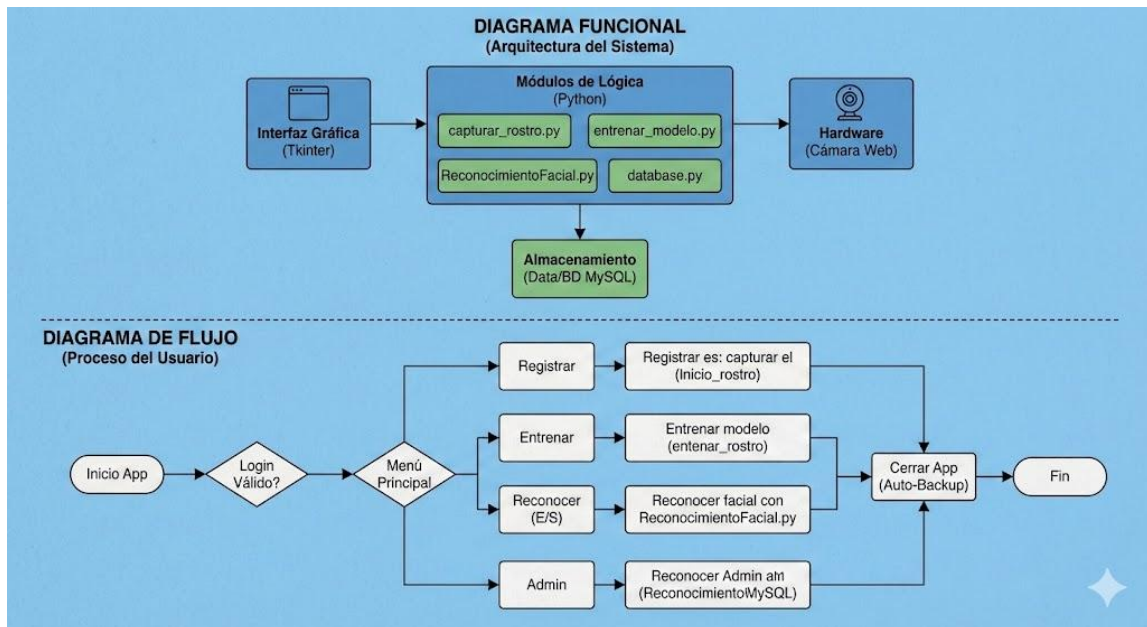
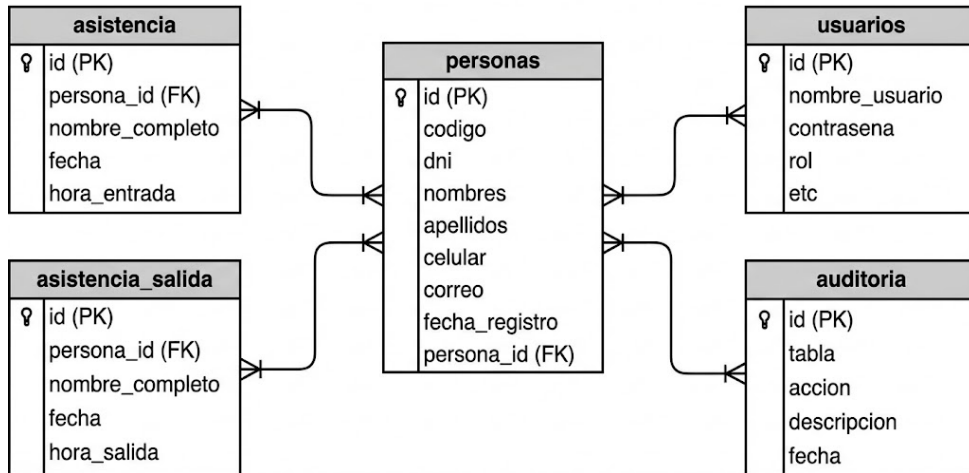


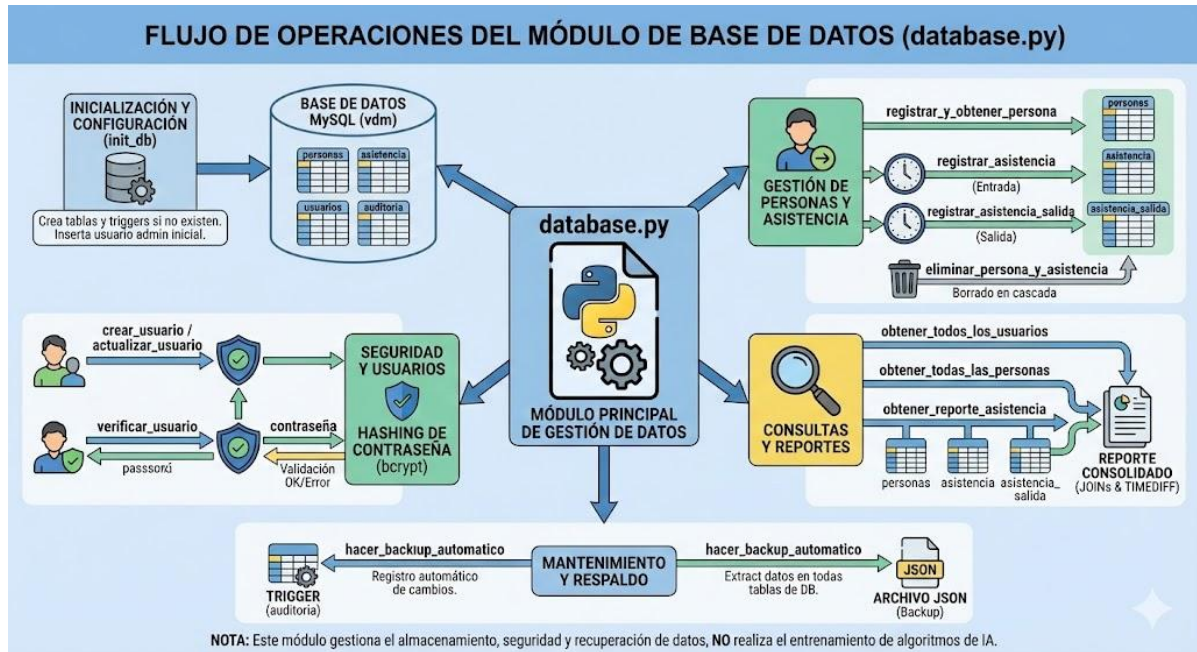
4. Módulo de Reconocimiento en Tiempo Real

Esta última imagen muestra el sistema en funcionamiento. Ilustra cómo se captura el vídeo en vivo, se detecta una cara, se compara con el modelo entrenado y, si hay una coincidencia, se registra automáticamente la asistencia (entrada o salida) en la base de datos con la fecha y hora exactas.



Módulo de Base de Datos (database.py)





¿POR QUE OPTAMOS EN USAR EL METODO DE FISHER Y NO OTROS METODOS COMO TRIANGULO GEOMETRICO EN EL RECONOCIMIENTO?

1. Robustez frente a variaciones reales

Triángulo geométrico (ojos–nariz)

- Depende de detectar **puntos faciales exactos**.
- Falla fácilmente con:
 - cambios de iluminación
 - rotación leve del rostro
 - expresiones faciales
 - gafas, mascarilla, barba
- Un pequeño error en la detección de un punto altera toda la geometría.

EigenFace / FisherFace

- Analiza el **rostro completo** como una unidad.
- No depende de puntos específicos.
- Tolera mejor:
 - variaciones de expresión
 - cambios leves de pose
 - ruido en la imagen

Conclusión: tu método es más estable en condiciones reales de aula.

2. Complejidad computacional y rendimiento

Triángulo geométrico

- Requiere:
 - detección de landmarks faciales
 - cálculos geométricos
 - normalización de escala y rotación
- Mayor carga computacional.
- Más propenso a errores en tiempo real.

EigenFace / FisherFace

- Pipeline simple:
 - detección del rostro
 - proyección matemática
 - comparación

- Menor latencia.
- Ideal para **reconocimiento en tiempo real** con webcam estándar.

Conclusión: tu enfoque es más rápido y eficiente.

3. Integración con base de datos

Triángulo geométrico

- Produce vectores geométricos pequeños.
- Difícil de:
 - versionar
 - validar
 - relacionar directamente con registros históricos

EigenFace / FisherFace

- Cada rostro está asociado a un **ID entero** (persona_id).
- Relación directa:
 - carpeta de imágenes
 - modelo entrenado
 - tabla de personas
 - tabla de asistencias
- Encaja perfectamente con un **modelo relacional**.

Conclusión: tu método se integra mejor con la base de datos.

4. Escalabilidad del sistema

Triángulo geométrico

- A mayor número de personas:
 - aumenta la confusión entre rostros similares
- Poco discriminativo en grupos grandes.

FisherFace

Maximiza la separación entre clases.

- Funciona mejor cuando hay **muchos estudiantes**.
- Diseñado específicamente para clasificación supervisada.

Conclusión: tu sistema escala mejor para uso institucional.

5. Mantenibilidad y simplicidad del código

Triángulo geométrico

- Código más largo y frágil.
- Dependencia de librerías externas (dlib, mediapipe).
- Más puntos de fallo.

Tu implementación

- OpenCV puro.
- Código claro y modular.
- Fácil de explicar, depurar y mantener.

Conclusión: es más sostenible a largo plazo.

POR QUE SE USO BASE DE DATOS RELACIONAL Y NO LA NO RELACIONAL?

El sistema de control de asistencias por reconocimiento facial utiliza una **base de datos relacional** debido a la naturaleza estructurada de la información gestionada y a la necesidad de mantener **integridad, consistencia y trazabilidad** de los datos.

En este sistema, las entidades principales —estudiantes, asistencias, usuarios y auditoría— presentan relaciones bien definidas entre sí, las cuales requieren el uso de **claves primarias y foráneas**, así como restricciones que garantizan la coherencia de la información almacenada. La base de datos relacional permite modelar estas relaciones de manera explícita y controlada.

Asimismo, los registros de asistencia constituyen información **crítica y verificable**, por lo que es indispensable asegurar propiedades **ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad)**, las cuales son propias de los sistemas de gestión de bases de datos relacionales. Esto garantiza que cada marcación de entrada o salida se almacene de forma correcta, evitando duplicidades o inconsistencias.

Desde el punto de vista del rendimiento y la inteligencia artificial en tiempo real, el sistema separa el almacenamiento de imágenes faciales, que se gestionan en el sistema de archivos, de los datos estructurados que se almacenan en la base de datos. Este enfoque reduce la carga de la base de datos y permite consultas eficientes y rápidas, características que se gestionan de manera óptima en un modelo relacional.

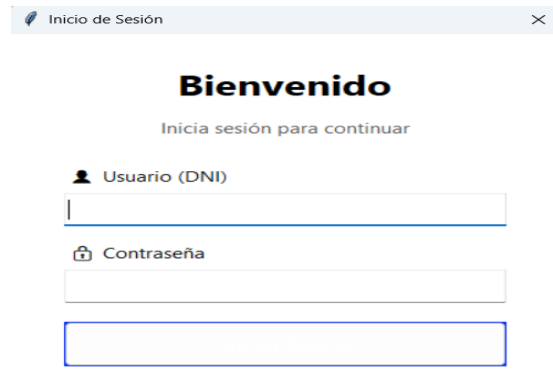
Por otro lado, el uso de una base de datos no relacional no resulta adecuado para este proyecto, ya que dichas bases están orientadas principalmente a datos no estructurados o semiestructurados y a escenarios donde la consistencia fuerte no es prioritaria. En un sistema académico de control de asistencias, la confiabilidad de los datos y la trazabilidad histórica son requisitos fundamentales.

En consecuencia, se optó por una base de datos relacional al ser la alternativa que mejor se ajusta a los requerimientos funcionales, de integridad y de control del sistema.

MANUAL DE USO

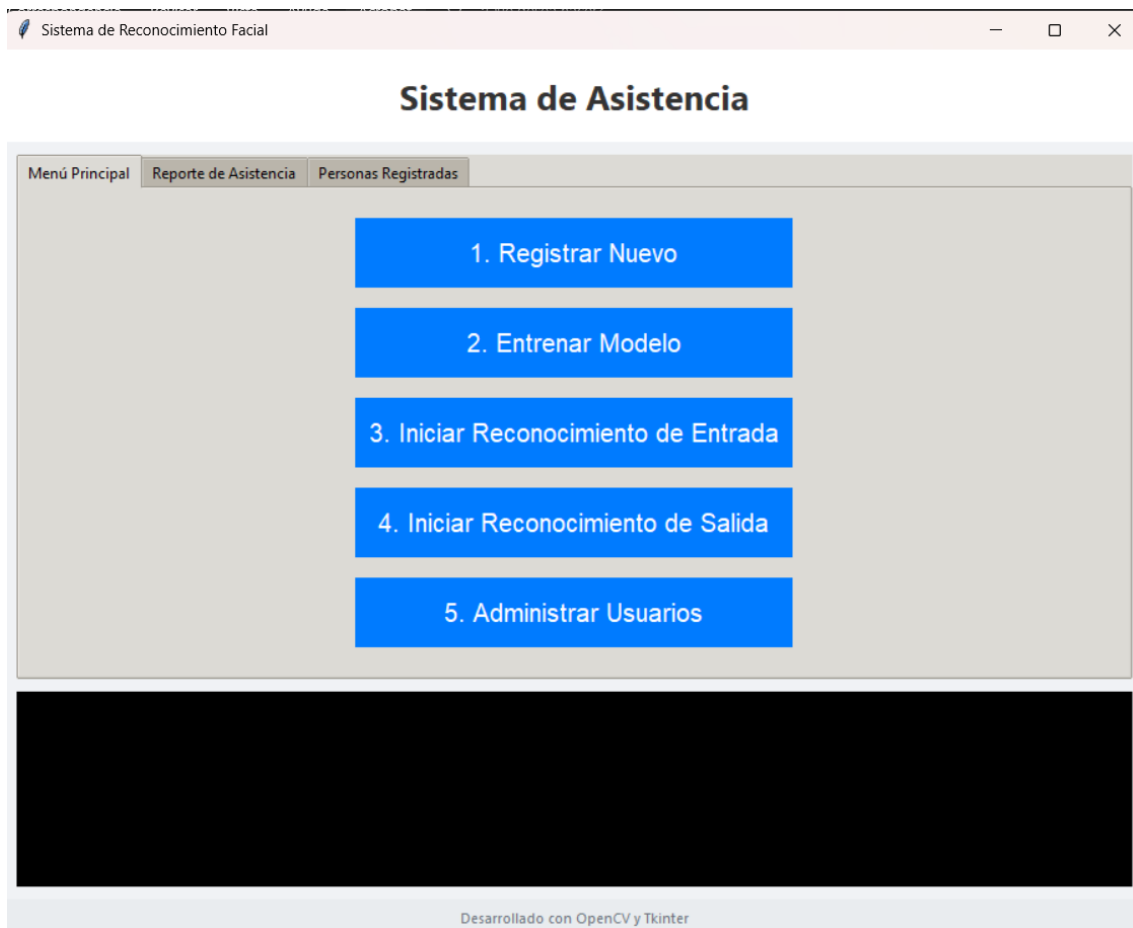
Corremos el .py en una versión de Python no superior de 3.12

Nos dará un inicio de sesion



The screenshot shows a window titled "Inicio de Sesión" with a close button. Inside, it says "Bienvenido" and "Inicia sesión para continuar". There are two input fields: "Usuario (DNI)" and "Contraseña". Below the password field is a button.

Nos dará la interfaz principal con el entorno de consola



The screenshot shows a window titled "Sistema de Reconocimiento Facial" with a close button. Inside, it says "Sistema de Asistencia". There are three tabs: "Menú Principal", "Reporte de Asistencia", and "Personas Registradas". The "Menú Principal" tab is active, showing five blue buttons: "1. Registrar Nuevo", "2. Entrenar Modelo", "3. Iniciar Reconocimiento de Entrada", "4. Iniciar Reconocimiento de Salida", and "5. Administrar Usuarios". Below the buttons is a large black rectangle. At the bottom, it says "Desarrollado con OpenCV y Tkinter".

Tenemos 3 secciones: Menu Principal, Reporte de asistencia y Personas Registradas

En el menú principal tenemos 5 puntos:

Registrar Nuevo: Aquí se registra un usuario nuevo, nombre completo y toma 300 capturas fotográficas de la persona registrada.

Entrenar el modelo: se entrena el modelo con todos los usuarios para que se optimice y tenga reconocimientos mas precisos

Iniciar Reconocimiento de Entrada: se registra con reconocimiento la hora de entrada del usuario

Iniciar Reconocimiento de Salida: Se registra con el reconocimiento la hora de salida del usuario

Administrar Usuario: Aquí solo tiene el permiso el super usuario para agregar eliminar usuarios

EN LA SEGUNDA SECCION

Sistema de Reconocimiento Facial

Sistema de Asistencia

Menú Principal Reporte de Asistencia Personas Registradas

Fecha Inicio: 2025-12-09 Fecha Fin: 2025-12-23 [Filtrar](#) [Exportar a CSV](#)

ID	Nombre Completo	Fecha	Hora Entrada	Hora Salida	Horas Asistidas
3	jhosimar daniel	2025-12-23	8:10:45	8:17:32	0:06:47
1	ronal	2025-12-17	11:26:31	11:27:17	0:00:46

Aquí se puede filtrar por fechas: Inicio y Fin, se visualiza el usuario, fecha, hora entrada y salida y las horas asistidas, También se puede exportar a csv.

En la 3ra sección

Sistema de Asistencia

Menú Principal

Reporte de Asistencia

Personas Registradas

Actualizar Lista

Eliminar Persona Seleccionada

ID	Nombre Completo	Fecha de Registro
3	jhosimar daniel	2025-12-23 08:02:10
1	ronal	2025-11-27 13:18:09

Aquí tenemos las opciones de Actualizar Lista y Eliminar persona seleccionada, se visualiza la fecha de registro