Sistemas Operativos

Proyecto:

Detección automática de ingreso de la Universidad Católica del Uruguay

Alumnos:

* Brian Morat
* Agustín Negreira
* Alfonso Rodriguez

Profesores:

* Julio Montañes
* Ingrid Machado
* Franyer Malave

2024 – URUGUAY

Implementación de un Sistema de Reconocimiento Facial para Control de Acceso en una Universidad

Para controlar el acceso, hemos integrado la funcionalidad de puertas automáticas que se abren automáticamente para los estudiantes, funcionarios e invitados para que puedan ser reconocidos por el sistema. Los estudiantes y el personal autorizado pueden acceder al campus sin necesidad de interactuar con el personal de seguridad.

En el caso de que una persona no esté registrada en el sistema, nuestro sistema permite al personal de seguridad agregarla manualmente. Esto se logra capturando su rostro y asociándolo con su nombre en la base de datos del sistema.

Nuestro proyecto no solo mejora la seguridad en el campus universitario, sino que también agiliza el proceso de control de acceso y registro de estudiantes y personal. Además, proporciona una solución escalable y eficiente que puede adaptarse a las necesidades de cualquier institución educativa.

Nuestro sistema opera en varias etapas clave:

Detección de Rostros:

Se utilizan hilos para ejecutar de forma paralela la detección de rostros en las imágenes capturadas por las cámaras del sistema.

Reconocimiento Facial:

Hilos separados se encargan de realizar el reconocimiento facial de los rostros detectados, comparándolos con los perfiles previamente registrados en la base de datos.

Control de Acceso:

Mediante el diseño de procesos, se coordina el acceso de los estudiantes y personal autorizado a través de las puertas automáticas. Los semáforos se utilizan para evitar condiciones de carrera y garantizar un acceso seguro y ordenado.

Registro de Nuevos Usuarios:

La captura de rostros y el registro de nuevos usuarios se realizan de manera concurrente utilizando hilos, permitiendo un proceso fluido y eficiente.

La utilización de hilos y semáforos en este proyecto ofrece varias ventajas significativas:

Mayor rendimiento y escalabilidad:

La utilización de Task permite que las solicitudes se procesen de forma asíncrona, aprovechando al máximo los recursos disponibles del sistema. Esto se traduce en un menor tiempo de respuesta y una mayor capacidad para manejar un mayor volumen de solicitudes, especialmente en escenarios de alta carga.

Control de acceso ordenado: Los semáforos garantizan que solo un número específico de hilos pueda acceder a recursos compartidos simultáneamente, evitando condiciones de carrera y asegurando la integridad de los datos. Esto permite un procesamiento ordenado y eficiente, especialmente en situaciones donde se manipulan datos sensibles o se realizan operaciones críticas.

Reducción del tiempo de espera: Al procesar las solicitudes de forma paralela, el tiempo de espera para los usuarios se reduce considerablemente. Esto mejora la experiencia del usuario y aumenta la satisfacción general con el sistema.

Fluidez y eficiencia: La utilización de Task y semáforos permite que el sistema se mantenga fluido y eficiente, incluso en situaciones de alta demanda. Esto evita que el sistema se bloquee o se vuelva lento, lo que es crucial para aplicaciones que requieren una respuesta rápida y un funcionamiento continuo.

Separación de responsabilidades: La implementación de Task y semáforos permite separar las tareas de procesamiento de solicitudes en unidades independientes y manejables. Esto facilita el desarrollo, mantenimiento y escalabilidad del código, ya que permite modificar o reemplazar componentes específicos sin afectar al resto del sistema.

Reutilización de código: Los mecanismos de sincronización y control de acceso implementados con Task y semáforos pueden ser reutilizados en otras partes del proyecto o en diferentes aplicaciones. Esto reduce la duplicación de código y promueve un enfoque de desarrollo más modular y eficiente.

Librerías robustas: C# proporciona librerías robustas y bien documentadas para la gestión de tareas asíncronas y sincronización de hilos, como Task y SemaphoreSlim. Esto facilita el uso y la implementación de estas técnicas en el código del proyecto.

Depuración y análisis simplificados: La utilización de Task y semáforos proporciona mecanismos de seguimiento y control que facilitan la depuración y el análisis del comportamiento del sistema. Esto permite identificar y corregir errores de concurrencia de manera más eficiente.

La implementación de semáforos facilita la escalabilidad del sistema, ya que permite gestionar eficientemente un mayor número de solicitudes de acceso y procesamiento de datos sin comprometer el rendimiento.

Planificación de criterios

Captura de Imágenes:

Utilización de colas de prioridad: En lugar de utilizar un algoritmo Round Robin simple, se podría implementar una cola de prioridad para la captura de imágenes. Esto permitiría priorizar la captura de imágenes en las cámaras con mayor actividad o con solicitudes más urgentes.

Procesamiento de Imágenes:

Estimación del tiempo de procesamiento: Para mejorar el rendimiento del algoritmo SJF, se podría implementar un mecanismo para estimar el tiempo de procesamiento de cada imagen antes de iniciar su procesamiento. Esto permitiría una mejor priorización de las tareas y reduciría el tiempo de respuesta promedio.

Considerar la disponibilidad de recursos: Además de la longitud de las tareas, se podría considerar la disponibilidad de recursos de hardware, como CPU y memoria, al momento de la priorización. Esto aseguraría que las tareas con mayor demanda de recursos se procesen cuando estos estén disponibles.

Almacenamiento de Datos:

Optimización de acceso a disco: Para mejorar el rendimiento del almacenamiento de datos, se podrían implementar técnicas de optimización de acceso a disco, como la agrupación de datos o el uso de cachés.

Considerar el tipo de datos: Dependiendo del tipo de datos que se almacenen (imágenes, videos, textos, etc.), se podrían utilizar algoritmos de almacenamiento específicos para optimizar el rendimiento.

Monitoreo en Tiempo Real:

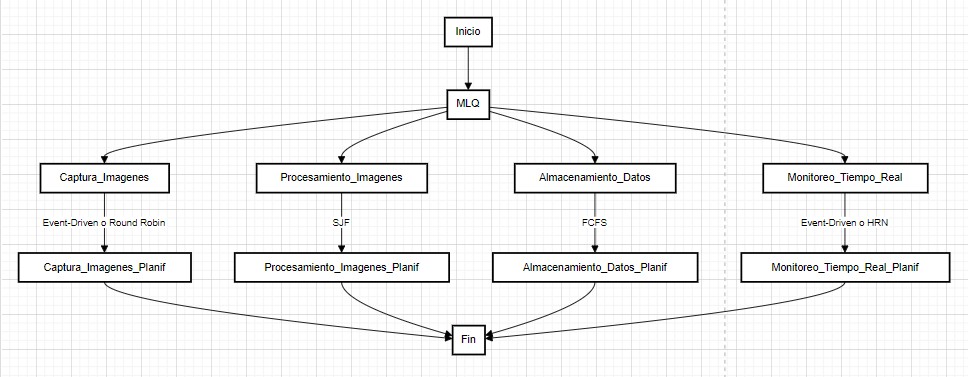
Manejo de múltiples eventos: El código actual parece estar diseñado para manejar un solo evento a la vez. Si se requieren manejar múltiples eventos simultáneamente, se debe implementar un mecanismo para gestionar la concurrencia de manera eficiente.

Notificación de eventos: Se podría implementar un mecanismo para notificar a los usuarios o sistemas externos sobre eventos relevantes en tiempo real.

Consideraciones adicionales:

Manejo de errores: Se recomienda implementar bloques try-catch para manejar adecuadamente las excepciones que puedan ocurrir en las diferentes etapas del proceso.

Registro de eventos: Se podría implementar un sistema de registro de eventos para registrar las actividades del sistema y facilitar la depuración y el análisis del comportamiento.



Una identificación preliminar de los recursos y procesos involucrados en la solución.

Recursos físicos:

Como recursos físicos tenemos a las cámaras de seguridad en primer lugar que se necesitaran colocarlas en las entradas y las cámaras PTZ para los pasillos y espacios comunes.

Para el almacenamiento de datos y el procesamiento de las imágenes necesitaremos una sala de servidores con amplio espacio de almacenamiento y estaciones de trabajo, a raíz de tener este factor deberíamos contar con una sala de contingencia por cualquier catástrofe.

Además del almacenamiento en la sala de servidores y en la sala de contingencia el equipo de almacenamiento de coincidir con los videos (HVRs) que permiten la grabación y gestión de imágenes de cámaras IP y analógicas.

Procesos:

Captura de imágenes: Proceso de captura de imágenes faciales por parte de las cámaras instaladas en las entradas, pasillos y espacios comunes.

Procesamiento de imágenes: Análisis de las imágenes capturadas para el reconocimiento facial, que implica comparar las características faciales con una base de datos de personas autorizadas.

Identificación y verificación: Proceso de identificación de personas autorizadas y verificación de su acceso al edificio.

Monitoreo en tiempo real: Supervisión continua de las imágenes de las cámaras y las identificaciones faciales en el centro de monitoreo.

Almacenamiento y gestión de datos: Grabación y almacenamiento de las imágenes capturadas para su posterior revisión y análisis.

Respuesta a eventos: Acciones tomadas en respuesta a eventos detectados, como alertar al personal de seguridad o tomar medidas preventivas.

Priorización de personas embarazadas y/o discapacidades: Acceso VIP al edificio para aquellas personas ya registradas en la base de datos del sistema como personas con acceso exclusivo.

Una descripción de los distintos criterios de optimización que podrían usarse

Tiempo de Respuesta:

Es crucial garantizar que el sistema responda rápidamente a las solicitudes de acceso para proporcionar una experiencia fluida a los usuarios y minimizar los tiempos de espera en la entrada.

Precisión del Reconocimiento Facial:

La precisión en la identificación de estudiantes y personal autorizado es fundamental para garantizar la seguridad del campus y evitar accesos no autorizados.

Eficiencia en el Uso de Recursos:

Se busca optimizar el uso de recursos del sistema, como la CPU y la memoria, para garantizar un rendimiento óptimo y reducir los costos operativos asociados.

Priorización de eventos y tipos de público:

Implica dar prioridad al procesamiento de ciertos eventos o tipos de público sobre otros. Los criterios de optimización podrían incluir la asignación de recursos adicionales para el reconocimiento de personas VIP o con necesidades especiales, así como la reducción del tiempo de espera para estos grupos

Optimización del uso de recursos:

Distribución equitativa de recursos entre áreas de alto tráfico y bajo tráfico, así como la minimización del consumo de energía.

Seguridad y privacidad de datos:

Implementación de medidas de seguridad robustas, como el cifrado de datos y el control de acceso a los sistemas.

Un ordenamiento justificado de los criterios y una selección de los primeros que el equipo decida optimizar.

Las prioridades de optimización se determinarán en función de su relevancia directa con los objetivos del proyecto. Por lo tanto, daremos prioridad a aquellos aspectos cuya importancia es fundamental para el funcionamiento eficaz del sistema. Dependiendo de su relevancia, estos aspectos ocuparán los primeros lugares en nuestra lista de optimización.

Seguridad y privacidad de datos:

Este criterio es fundamental para proteger la integridad y la confidencialidad de la información personal recopilada por el sistema. La implementación de medidas robustas de seguridad y privacidad es crucial para evitar posibles violaciones. Esto incluiría la encriptación de datos, el control de acceso adecuado y la adopción de prácticas de seguridad cibernética.

Algoritmo de reconocimiento facial:

Antes de abordar la eficiencia operativa del sistema, es esencial garantizar que el algoritmo de reconocimiento facial sea preciso y confiable. Esto implica realizar pruebas exhaustivas del algoritmo en diversas condiciones y ajustarlo según sea necesario para minimizar los errores de identificación y mejorar la precisión general del sistema.

Optimización del uso de recursos:

Una distribución equitativa de recursos y la minimización del consumo de energía son importantes para garantizar la eficiencia operativa del sistema y reducir los costos a largo plazo.

La descripción de los distintos escenarios de prueba de la simulación y las mediciones que se realizarán.

Escenario de Prueba 1 - Precisión del Reconocimiento Facial:

* Utilizando un conjunto de datos diversos que incluya imágenes de estudiantes y personal autorizado, así como de posibles intrusos, se evaluará la precisión del sistema al identificar correctamente a las personas autorizadas y rechazar a los no autorizados.

Escenario de Prueba 2 - Tiempo de Respuesta:

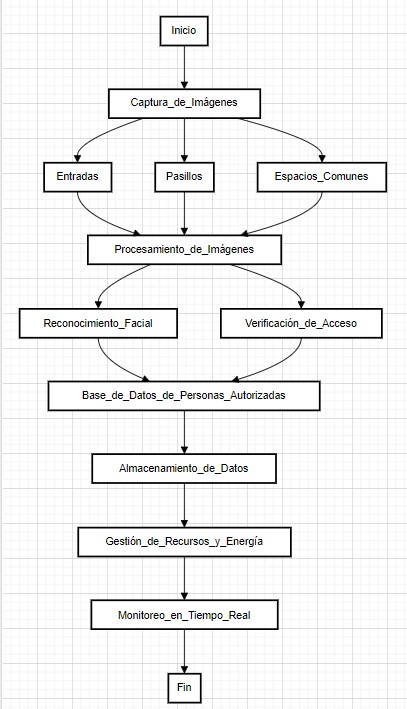
* Se simulará el acceso de múltiples usuarios al sistema en un período de tiempo determinado.
* Se medirá el tiempo que tarda el sistema en procesar cada solicitud de acceso, desde la detección facial hasta la apertura de la puerta automática.

Escenario de Prueba 3 - Eficiencia en el Uso de Recursos:

* Se realizarán pruebas de carga para evaluar el uso de recursos del sistema bajo diferentes niveles de demanda.
* Se medirá el consumo de CPU y memoria durante las pruebas para garantizar un uso eficiente de los recursos del sistema.

Un bosquejo de la simulación que se efectuará. Arquitectura de la solución,

identificando hilos y principales procesos involucrados.



Conclusión.

En resumen, nuestro código funciona simula un sistema robusto de control de acceso para la universidad, utilizando múltiples cámaras de seguridad que gestionan solicitudes de acceso de diversas personas. Cada solicitud es representada por la clase SolicitudAcceso, la cual incluye propiedades como prioridad, persona asociada, tiempo de llegada y análisis, además de la cámara y puerta por la cual se realiza la solicitud. Estas solicitudes son gestionadas mediante la estructura de datos PriorityQueue, la cual permite mantener las solicitudes ordenadas por prioridad.

Para garantizar la integridad de los datos y evitar problemas de concurrencia, se utiliza un mecanismo de semáforos (SemaphoreSlim) en cada instancia de la clase CamaraSeguridad. Este semáforo controla el acceso a la cola de solicitudes, asegurando que solo un hilo pueda modificar la cola en un momento dado. Esto es escencial para evitar condiciones de carrera donde múltiples hilos intenten acceder o modificar la misma estructura de datos concurrentemente.

El proceso de solicitud comienza cuando se crea una instancia de SolicitudAcceso y se agrega a la cola de prioridad de la cámara correspondiente utilizando el método AgregarSolicitud. Entonces se actualiza el tiempo de llegada de la solicitud y se adquiere el semáforo para garantizar que la modificación de la cola de prioridad sea segura y consistente. Una vez agregada la solicitud, se libera el semáforo para permitir que otros hilos accedan y agreguen nuevas solicitudes.

El procesamiento real de las solicitudes se lleva a cabo de manera asíncrona en el método ProcesarSolicitudesAsync. Cada cámara ejecuta este método, y continúa procesando solicitudes mientras haya solicitudes pendientes en la cola de prioridad. Dentro de este método, se vuelve a adquirir el semáforo para intentar desencolar la solicitud con mayor prioridad mediante TryDequeue. Este método asegura que la solicitud sea removida de la cola de manera segura y atómica.

Una vez obtenida la solicitud, se verifica si la persona asociada está autorizada para el acceso comparándola con un conjunto de personas autorizadas (personasAutorizadas). Dependiendo de la autorización, se imprime un mensaje indicando si el acceso está permitido o si se requiere una autorización adicional.

Además, el sistema permite aplicar "tratos especiales" a personas según su tipo, reduciendo el tiempo de análisis requerido si cumplen con ciertos criterios. Esto se implementa utilizando un diccionario (tratosEspeciales) que mapea tipos de personas a ajustes en el tiempo de análisis.

Para monitorear y evaluar el desempeño del sistema, se registran métricas como el tiempo total de retorno de cada tipo de persona en un diccionario (metricas). Esto permite a los administradores y desarrolladores analizar y optimizar el rendimiento del sistema, identificando posibles cuellos de botella o áreas de mejora en el manejo de las solicitudes.