

## Sistema De Reconocimiento Facial Para El Control De Asistencia En Aulas Universitarias

Integrantes:

Líder: Ricardo Andrey Flórez Torres

Julián David Merchán Quiroz

Jorge Iván Monroy Martínez

Alejandro David Contreras Parra

Unidades Tecnológicas de Santander

Tecnología en Desarrollo de Sistemas Informáticos

Docente: Ing. Pedro Ramírez

Bucaramanga, Santander

## **1. Misión**

Automatizar la asistencia y la puntualidad en las aulas universitarias a través del reconocimiento facial, brindar a los estudiantes y maestros una herramienta efectiva, objetiva y asequible para monitorear y administrar la asistencia, fomentar la productividad y la responsabilidad del estudiante.

## **Visión**

Efectuar la integración de tecnologías avanzadas en las instituciones educativas. Nuestro sistema será el estándar de la eficiencia y control en la educación superior, todo ambiente orientado a la puntualidad, responsabilidad y claridad.

## **Expectativas de la Implementación del proyecto se espera lo siguiente:**

Los estudiantes deberían ser conscientes de sus faltas y llegadas tarde, por lo tanto, se sentirán presionados.

Los profesores no deberían pasar la asistencia y tendrían más tiempo en la clase.

La universidad obtendrá una mejor cuantificación y cualificación de la asistencia evitando errores humanos, y el entorno del estudio se convierte en más motivador para los alumnos.

## **4. Riesgos y formas de mitigarlos.**

Los riesgos potenciales serían los siguientes:

Riesgo: privacidad y datos. La toma de datos biométricos puede verse como un riesgo para la protección de los mismos.

Solución: establecer algún tipo de protocolos de privacidad y encriptación para que nadie pueda acceder a la información dentro de la organización sin la debida autorización.

Resistentes al cambio: Habrán personas que rechazarán lugares con cámaras.

Solución: educar a las personas sobre las ventajas, los derechos y las nuevas tecnologías para garantizar la privacidad.

Errores en la captura facial: es posible que el sistema no reconozca correctamente a la persona.

Solución: utilizar software especializado con algoritmos precisos y proporcionar algún tipo de copia de seguridad manual. Dificultades técnicas: las cámaras podrían dejar de funcionar y los sistemas operativos podrían bloquearse

## **5. Necesidad de recursos**

### 5.1 Recursión humana:

Desarrollador: Encargado de crear el software de reconocimiento facial

Ingeniero de red : Para instalar las cámaras y asegurarse de que estén correctamente conectadas.

Especialista en ciberseguridad: Responsable de los datos personales y la biometría.

Técnico de soporte: El que se asegurará de que el sistema funcione correctamente

### 5.2 Tecnología de reproducción:

Cámara de alta calidad:

Hikvision: Imagen para el capturador perfecto Incluso en condiciones normales.

Axis: Tiene un sensor que ayuda a la alta calidad en la imagen con tecnología en la calidad de video.

Servidores: Lo mejor para este caso es utilizar servicios de la nube como AWS debido a su escalabilidad

Software de reconocimiento Facial: Como OpenCV para que nos ayuden a solucionar en la nube

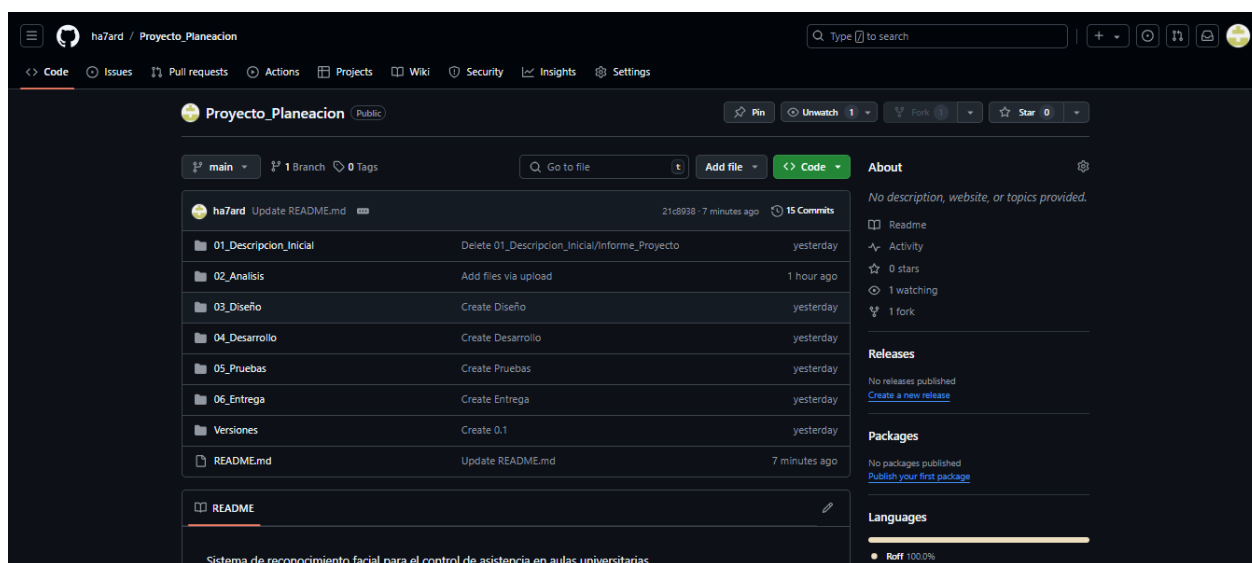
con mayor precisión.

### 5.3 Otros Recursos:

Red de infraestructura: para mantener una conexión estable de todas las cámaras y datos.

Base de datos robusta: necesitaría el almacenamiento de toda la información sobre los estudiantes y su tiempo de asistencia a la universidad.

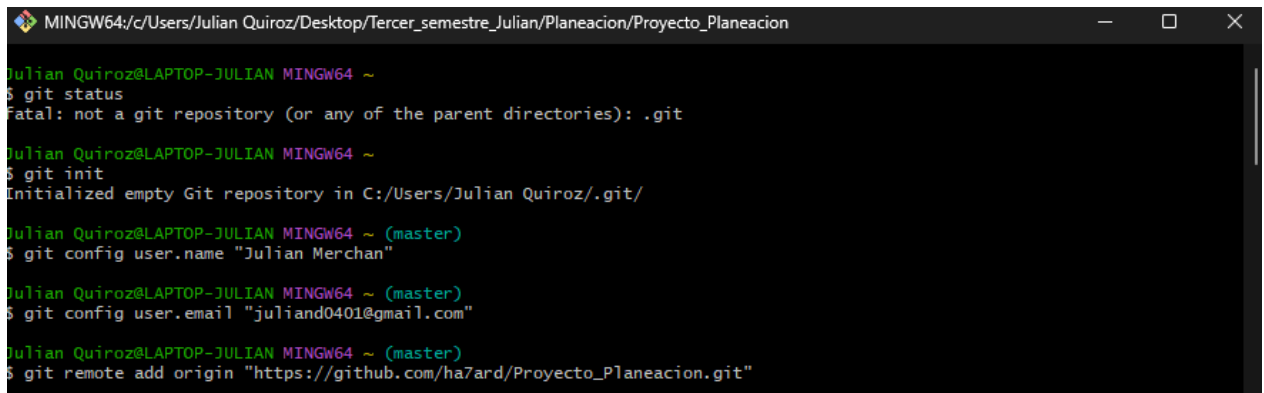
## Evidencias del proceso



- Para la creación del repositorio fue mediante la página GitHub
- Las configuraciones del repositorio realizadas son que al momento de la creación fuera en base al branche de main, con visibilidad publica y que automáticamente generara el archivo readme.md.
- Para la creación de los directorios nos ubicamos en el apartado Add file, de ahí nos movemos al nombre del archivo, para poder crear el fichero se proporciona el nombre

del directorio y seguidamente slash. Ejemplo: “01\_Descripcion\_Inicial/”. Así con el resto de los directorios.

- El historial de cambios se encuentra en el apartado de “Commits” junto con el nombre de cambios que se han ido realizando en tiempo real.



```

MINGW64:/c:/Users/Julian Quiroz/Desktop/Tercer_semestre_Julian/Planeacion/Proyecto_Planeacion
Julian Quiroz@LAPTOP-JULIAN MINGW64 ~
$ git status
fatal: not a git repository (or any of the parent directories): .git

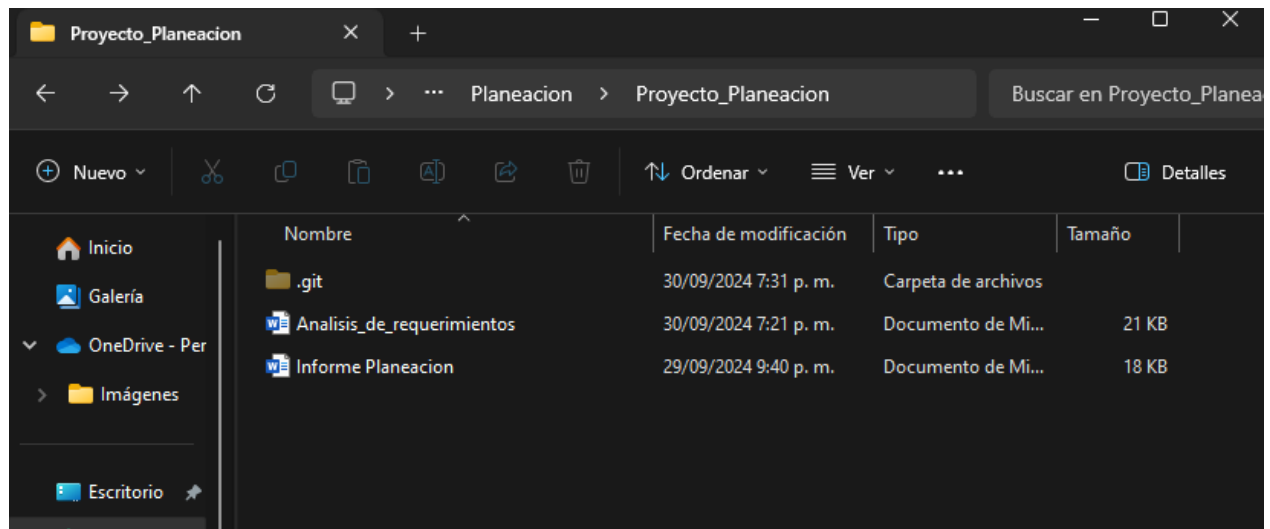
Julian Quiroz@LAPTOP-JULIAN MINGW64 ~
$ git init
Initialized empty Git repository in C:/Users/Julian Quiroz/.git/

Julian Quiroz@LAPTOP-JULIAN MINGW64 ~ (master)
$ git config user.name "Julian Merchan"

Julian Quiroz@LAPTOP-JULIAN MINGW64 ~ (master)
$ git config user.email "juliand0401@gmail.com"

Julian Quiroz@LAPTOP-JULIAN MINGW64 ~ (master)
$ git remote add origin "https://github.com/ha7ard/Proyecto_Planeacion.git"
  
```

- Para poder conectar el repositorio virtual con el local mediante la aplicación Git, con el terminal de GitBash, podemos ubicarnos y poder unir un repositorio local con el virtual con los comandos realizados en la captura.
- Esto con el fin de facilitar el manejo de archivos en el repositorio



- Al momento de inicializar el repositorio local nos crea una carpeta de archivos ocultos con el nombre de .git.
- Esto nos ayuda para la transferencia de archivos locales al repositorio virtual.
- Mediante los comandos “git add (nombre del archivo con su extensión)” podemos subir archivos si nos encontramos conectados al repositorio virtual

**PARA CONSULTAR EL REPOSITORIO INGRESE AL SIGUIENTE ENLACE:**

**[https://github.com/ha7ard/Proyecto\\_Planeacion.git](https://github.com/ha7ard/Proyecto_Planeacion.git)**