****

**UNIVERSIDADE FEDERAL DE PERNAMBUCO**

**CENTRO DE TECNOLOGIA E GEOCIÊNCIAS**

**DEPARTAMENTO DE ELETRÔNICA E SISTEMAS**

Welington Gabriel Oliveira de Carvalho

Vitor Maciel da Silva

**CONTROLE DE ACESSO ATRAVÉS DE RECONHECIMENTO FACIAL POR IA**

RELATÓRIO DE PROJETO

2025

Recife

**Sumário**

[1. Resumo 2](#_Toc195433357)

[2. Introdução 2](#_Toc195433358)

[3. Objetivo Geral 2](#_Toc195433359)

[3.1. Objetivos Específicos 2](#_Toc195433360)

[4. Tecnologias Utilizadas 2](#_Toc195433361)

[5. Arquitetura Geral e Repositório no GitHub 3](#_Toc195433362)

[5.1. Estrutura do Banco de Dados MySQL 3](#_Toc195433363)

[6. Metodologia 3](#_Toc195433364)

[6.1. Objetivos Específicos 3](#_Toc195433365)

[7. Resultados 3](#_Toc195433366)

[8. Conclusão 3](#_Toc195433367)

[9. Referências Bibliográficas 3](#_Toc195433368)

[4. Arquitetura do Sistema 4](#_Toc195433369)

[5. Detalhamento dos Dados 5](#_Toc195433370)

[6. Tipo de Banco de Dados e Justificativa 5](#_Toc195433371)

[7. Estrutura Inicial do Banco de Dados 5](#_Toc195433372)

[8. Uso Prático no Dia a Dia 6](#_Toc195433373)

[9. Front End 6](#_Toc195433374)

# 1. Resumo

O presente relatório descreve as atividades desenvolvidas para implementação de um sistema de controle de acesso, baseado na estrutura física do Departamento de Engenharia Elétrica (DEE) da UFPE, integrando conhecimentos das disciplinas de Inteligência Artificial (IA) com Banco de Dados (BD). O sistema realiza reconhecimento facial a partir de imagens capturadas por câmera e autoriza ou nega acessos com base em um cadastro prévio de usuários treinados no modelo. A persistência dos dados é realizada via banco MySQL, garantindo registro simples e eficiente dos logs de acesso em tempo real, com baixo custo. A aplicação foi desenvolvida em Python com interface web implementada em Streamlit, permitindo visualização pública de acessos. O protótipo do sistema foi executado localmente em um notebook com webcam.

# 2. Introdução

A segurança e o controle de acesso a espaços acadêmicos são preocupações constantes, visto que medidas de acesso ao Campus são inexistes, comprometendo a segurança dos discentes e docentes, aliado aos altos custos e burocracia de aquisição de equipamento de laboratório. O projeto proposto visa modernizar o controle de acesso ao DEE, substituindo métodos de RFID por reconhecimento facial automatizado. Além de registrar entradas e saídas, o sistema permitirá o monitoramento em tempo real das pessoas presentes no local, com uma interface aberta através do email institucional, útil em situações de emergência ou comodidade de visualização por alunos e professores.

# 3. Objetivo Geral

Demonstrar domínio e conhecimento dos assuntos vistos em sala de aula, nas disciplinas de Inteligência Artificial e Banco de dados, através de um projeto com aplicação prática.

## 3.1. Objetivos Específicos

* Identificar e autorizar pessoas previamente cadastradas;
* Registrar tentativas de acesso não autorizadas, com captura de imagem;
* Monitorar, em tempo real, os registros de acesso ao DEE;

# 4. Tecnologias Utilizadas

* Linguagem: Python 3
* Modelo de IA: OpenCV 2
* Banco de Dados: MySQL
* Interface gráfica: Streamlit
* Hardware: Arduino Uno

# 5. Arquitetura Geral e Repositório no GitHub

Adicionar foto da árvore de arquivos e link do repositório público. Adicionar também trechos de código, caso julgue necessário

## 5.1. Estrutura do Banco de Dados MySQL

Eeeeee

# 6. Metodologia

eeeeeee

## 6.1. Tópico Incrível

Eeeeee

# 7. Resultados

# 8. Conclusão

# 9. Referências Bibliográficas

A segurança e o controle de acesso a espaços acadêmicos são preocupações constantes, visto que medidas de acesso ao Campus são inexistes, comprometendo a segurança dos discentes e docentes, aliado aos altos custos e burocracia de aquisição de equipamento de laboratório. O projeto proposto visa modernizar o controle de acesso ao DEE, substituindo métodos de RFID por reconhecimento facial automatizado. Além de registrar entradas e saídas, o sistema permitirá o monitoramento em tempo real das pessoas presentes no local, com uma interface aberta através do email institucional, útil em situações de emergência ou comodidade de visualização por alunos e professores.

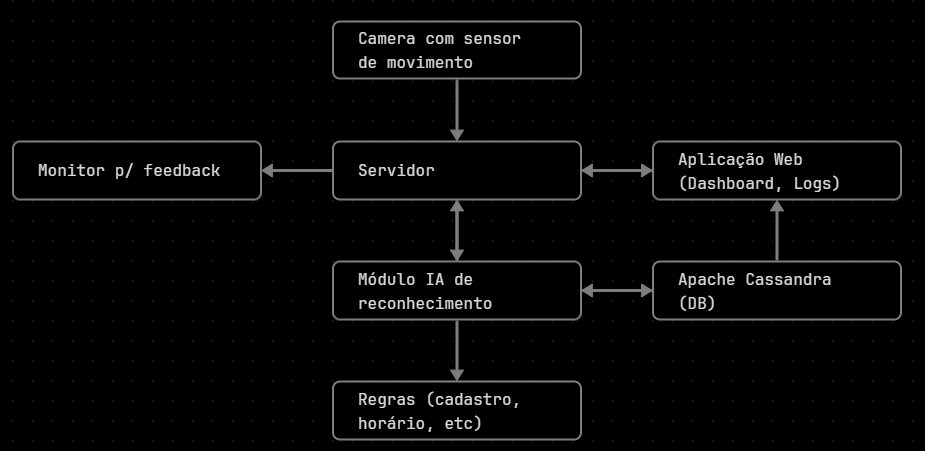
A coleta dos dados é feita através da câmera de um notebook, que atua como sensor de entrada. A captura da imagem é ativada por detecção de movimento. A imagem é então enviada para o módulo de IA, que verifica se o rosto é autorizado. Para atualização do perfil, o sistema armazena uma nova imagem por dia, e também registra imagens de tentativas não autorizadas com timestamp.

# 4. Arquitetura do Sistema

O sistema é composto por quatro módulos principais:

* Módulo de Captura (Notebook): Detecta movimento e captura a imagem do rosto.
* Módulo de IA (Python): Realiza a classificação do rosto com base em modelos treinados via Tensor Flow (Google Teachable Machine).
* Módulo de Banco de Dados (Cassandra): Registra e consulta os rostos, autorizações e logs.
* Interface Web (Python): Dashboard para visualização de acessos, alertas e ocupação em tempo real.

**Figura 1: Arquitetura do Sistema**



# 5. Detalhamento dos Dados

Os dados coletados e manipulados incluem:

* Imagem facial (em base64 ou referência a arquivo)
* Nome do usuário
* Data e hora do acesso
* Status da tentativa (autorizada ou não)
* Logs de entrada e saída
* Registro de renovação de permissão (cadastro válido por 5 anos)

# 6. Tipo de Banco de Dados e Justificativa

O banco de dados utilizado é o Apache Cassandra, um banco NOSQL orientado a colunas. A escolha se deu pelas seguintes razões:

* Alta performance em escrita e leitura: Ideal para sistemas com grande volume de logs de acesso.
* Escalabilidade horizontal: Para uso futuro em outros departamentos.
* Tolerância a falhas: Importante para um sistema de segurança.
* Flexibilidade: Permitindo facilmente atualizar a estrutura dos dados conforme o projeto evolui.

O Apache Cassandra utilizando uma estrutura chamada CQL, similar a uma estrutura SQL, com algumas penas diferenças. Atualmente, não há suporte ao Windows desde a atualização para sua versão 4, em julho de 2021. Sendo assim, o Cassandra é popularmente utilizado em Linux OS, via Docker, ou então utilizando WSL 2 (Windows Subsystem for Linux).

Importante mencionar a existência de uma pesquisa entre os membros da equipe, procurando validar a aplicabilidade de um banco de dados substituto ao Cassandra, visto que o mesmo é aplicado inteiramente em sistemas Linux, enquanto o restante do sistema será baseado em Windows. A lista inclui o MySQL (banco relacional) e Google BigQuery (similar ao Cassandra).

# 7. Estrutura Inicial do Banco de Dados

Tabela: usuarios

* id\_usuario (User ID)
* nome
* imagem\_ref
* data\_cadastro
* validade\_cadastro (timestamp)

Tabela: acessos

* id\_acesso (User ID)
* id\_usuario
* timestamp
* tipo\_acesso (entrada/saida)
* status (autorizado/negado)
* imagem\_registro

Tabela: logs\_nao\_autorizados

* id\_log
* imagem\_ref
* timestamp
* motivo (ex: rosto não reconhecido)

# 8. Uso Prático no Dia a Dia

Fluxograma de utilização na prática:

1. Ao detectar movimento, a câmera captura a imagem.
2. A IA identifica se o rosto é conhecido e autorizado.
3. Se autorizado e dentro do horário permitido (06h às 23h59), o acesso é liberado.
4. Um log de entrada é gerado, com imagem e horário.
5. Ao sair, a câmera externa registra a saída automaticamente.
6. O dashboard exibe em tempo real quem está no local, e mantém um histórico de acessos.
7. Cada novo dia, uma nova imagem é adicionada ao perfil para tornar o modelo mais robusto.
8. A cada 5 anos, o cadastro do usuário deve ser renovado manualmente.

# Front End

Para interface front-end, duas possibilidades estão sendo consideradas: Gradio e CustomTkinter.

* 1. Gradio

Uma biblioteca Python desenvolvido para criar interfaces web interativas, voltadas para modelos de machine learning, com possibilidade de rodar localmente ou hospedada em um servidor. A biblioteca oferece suporte para upload de mídias (imagens, vídeos, áudios e textos), visualização em tempo real e criação de dashboards sem necessidade de front-end avançado.

* 1. CustomTkinter

Biblioteca do Python voltada para criação de interfaces com estilo moderno, sendo uma extensão do Tkinter, com possibilidade de estilização via programação em CSS.

Uma breve comparação entre ambas as opções:

|  |  |  |
| --- | --- | --- |
| **Critério** | **Gradio** | **CustomTkinter** |
| Tipo de interface | Web (browser) | Desktop (ideal para MVP) |
| Integração com IA | Foco principal da biblioteca | Manual, mas possível |
| Facilidade de uso | Simplificado | Moderado |
| Estética | Minimalista e limpa | Moderna e personalizável |
| Principal aplicação do projeto | Verificação facial e Dashboard de segurança | Painel de administração local |
| Conexão com banco de dados | Via back-end Python | Via back-end Python |
| Hospedagem | Local ou remoto | Local no próprio computador |
| Atualizações automáticas | Em tempo real | Sim, em ciclos pré-definidos |