Universidade Estácio de Sá

cursos: Ciência da Computação / ADs

UNIDADE NOVA AMÉRICA

**TRABALHO DE PYTHON**

**(DEV. RÁP. DE APLICAÇÕES EM PYTHON)**

Rio de Janeiro – RJ

05/2024

Gabriel Fellipe Venâncio de Oliveira (202303535252)

Jonas Oliveira Silva Campos (202304322741)

Nicolas da Cunha Pinto Rabelo (202304269611)

Railson Albino de Oliveira Vieira (202303468288)

**TRABALHO DE PYTHON**

**(DEV. RÁP. DE APLICAÇÕES EM PYTHON)**

Trabalho de Python apresentado a Universidade Estácio de Sá, como exigência para avaliação na disciplina Dev. Ráp. De Aplicações em Python

Orientador: Prof. Ronaldo Candido dos Santos

SUMÁRIO

[1. INTRODUÇÃO 3](#_Toc166953122)

[1.1 DESCRIÇÃO DO PROBLEMA 3](#_Toc166953123)

[1.2 OBJETIVOS 4](#_Toc166953124)

[2. DESENVOLVIMENTO 5](#_Toc166953125)

[2.1 REQUISITOS FUNCIONAIS E NÃO FUNCIONAIS DO SISTEMA 5](#_Toc166953126)

[REQUISITOS FUNCIONAIS 5](#_Toc166953127)

[• O sistema deve registrar a entrada e a saída dos funcionários, incluindo seu nome, setor e os horários que houve cada registro; 5](#_Toc166953128)

[• O sistema deve ter níveis de acesso, separando o usuário limitado do administrador, que pode incluir, excluir, alterar e consultar os funcionários e seus usuários, locais e setores; 5](#_Toc166953129)

[ REQUISITOS NÃO FUNCIONAIS 5](#_Toc166953130)

[• O sistema deve ser multiplataforma; 5](#_Toc166953131)

[• O sistema deve funcionar offline; 5](#_Toc166953132)

[2.2 DIAGRAMA DE CASO DE USO 5](#_Toc166953133)

[2.3 DIAGRAMA DE CLASSE 6](#_Toc166953134)

[6](#_Toc166953135)

[2.4 ITENS 6](#_Toc166953136)

[2.5 ATUALIZAÇÃO DAS VERSÕES 7](#_Toc166953137)

[3. CONCLUSÃO 8](#_Toc166953138)

[4. REFERÊNCIAS 9](#_Toc166953139)

# INTRODUÇÃO

É com entusiasmo que apresentamos nosso trabalho, que surge da necessidade crescente de otimização e controle dos fluxos de entrada e saída de usuários em ambientes corporativos. Neste contexto, desenvolvemos um programa em Python utilizando a abordagem de Desenvolvimento Rápido de Aplicações (RAD), visando facilitar o registro e monitoramento desses fluxos, fundamentais para a segurança e gestão eficiente de espaços compartilhados.

A escolha deste tema foi motivada pela constatação de uma lacuna significativa nas ferramentas disponíveis para realizar esse tipo de controle. Muitas organizações enfrentam desafios na gestão de acesso de pessoas a diferentes áreas internas, e a falta de soluções eficazes pode resultar em dificuldades operacionais, falhas de segurança e perda de produtividade.

Ao longo desta apresentação, detalharemos o funcionamento e os benefícios do nosso programa, demonstrando como ele pode ser uma ferramenta valiosa para aprimorar a segurança e a eficiência operacional em diversos contextos empresariais.

## DESCRIÇÃO DO PROBLEMA

Uma das principais situações que levaram à realização deste trabalho é a ausência de ferramentas adequadas para registrar e monitorar o movimento de pessoas entre diferentes setores ou áreas dentro de uma empresa. Muitas vezes, os métodos tradicionais de controle, como o uso de registros em papel ou sistemas manuais, mostram-se obsoletos, imprecisos e suscetíveis a erros humanos.

Diante dessas situações e desafios, surge a necessidade premente de um programa que utilize abordagens modernas de desenvolvimento, como o Desenvolvimento Rápido de Aplicações (RAD), para oferecer uma solução ágil, confiável e personalizável para o registro de entrada e saída de usuários.

A gestão eficiente dos fluxos de entrada e saída de usuários em ambientes corporativos é uma demanda crucial para garantir a segurança, a organização e a produtividade dos espaços compartilhados.

## OBJETIVOS

O principal objetivo deste trabalho foi desenvolver um programa em Python utilizando a abordagem de Desenvolvimento Rápido de Aplicações (RAD), que seja capaz de registrar de forma precisa e automatizada a entrada e saída de usuários de um setor para o outro em ambientes corporativos. Para alcançar esse objetivo, delineamos os seguintes resultados a serem obtidos:

1. Desenvolvemos um programa funcional que permita o registro eficiente das entradas e saídas de usuários, capturando informações como a data do dia e o horário de cada evento.
2. Implementamos uma interface intuitiva e amigável que facilite a utilização do programa por parte dos usuários, garantindo uma experiência de uso fluida e eficaz.
3. Integramos funcionalidades de personalização que permitam adaptar o programa às necessidades específicas de cada organização, incluindo a configuração de políticas de acesso, horários de funcionamento e estrutura de espaços físicos.
4. Garantirmos a segurança e a confiabilidade dos dados registrados, adotando medidas adequadas para proteger as informações sensíveis e prevenir possíveis falhas ou violações de segurança.
5. Avaliamos a eficácia e a usabilidade do programa por meio de testes práticos e feedback dos usuários, buscando identificar possíveis melhorias e refinamentos para futuras versões.

Ao atingir esses resultados, esperamos fornecer uma solução eficiente e robusta para a gestão de acessos em ambientes corporativos, contribuindo para a melhoria da segurança, organização e produtividade desses espaços.

# DESENVOLVIMENTO

## REQUISITOS FUNCIONAIS E NÃO FUNCIONAIS DO SISTEMA

## REQUISITOS FUNCIONAIS

## O sistema deve registrar a entrada e a saída dos funcionários, incluindo seu nome, setor e os horários que houve cada registro;

## O sistema deve ter níveis de acesso, separando o usuário limitado do administrador, que pode incluir, excluir, alterar e consultar os funcionários e seus usuários, locais e setores;

## REQUISITOS NÃO FUNCIONAIS

## O sistema deve ser multiplataforma;

## O sistema deve funcionar offline;

## DIAGRAMA DE CASO DE USO

## DIAGRAMA DE CLASSE

## Diagrama Descrição gerada automaticamente

## MODELO CONCEITUAL

Diagrama

Descrição gerada automaticamente

## MODELO LÓGICO

Diagrama

Descrição gerada automaticamente

## MODELO FÍSICO

## CREATE TABLE IF NOT EXISTS "Setor" (

## "setor\_ID" INTEGER NOT NULL,

## "nome\_setor" VARCHAR(50) NOT NULL UNIQUE,

## "desc\_setor" VARCHAR(255),

## PRIMARY KEY("setor\_ID" AUTOINCREMENT)

## );

## CREATE TABLE IF NOT EXISTS "Funcionario" (

## "funcionario\_ID" INTEGER NOT NULL,

## "nome" VARCHAR(50) NOT NULL,

## "email" VARCHAR(100),

## "setor\_ID" INTEGER NOT NULL,

## PRIMARY KEY("funcionario\_ID" AUTOINCREMENT),

## FOREIGN KEY("setor\_ID") REFERENCES "Setor"("setor\_ID")

## );

## CREATE TABLE IF NOT EXISTS "Usuario" (

## "usuario\_ID" INTEGER NOT NULL,

## "nome\_usuario" VARCHAR(50) NOT NULL UNIQUE,

## "senha" TEXT NOT NULL,

## "funcionario\_ID" INTEGER NOT NULL,

## "admin" TEXT NOT NULL,

## PRIMARY KEY("usuario\_ID" AUTOINCREMENT)

## FOREIGN KEY("funcionario\_ID") REFERENCES "Funcionario"("funcionario\_ID") ON DELETE CASCADE

## );

## CREATE TABLE IF NOT EXISTS "Local" (

## "local\_ID" INTEGER NOT NULL,

## "nome\_local" VARCHAR(50) NOT NULL UNIQUE,

## PRIMARY KEY("local\_ID" AUTOINCREMENT)

## );

## CREATE TABLE IF NOT EXISTS "Acesso" (

## "data\_hora" DATETIME DEFAULT (datetime('now', 'localtime')),

## "tipo" VARCHAR(7) NOT NULL,

## "funcionario\_ID" INTEGER NOT NULL,

## "local\_ID" INTEGER NOT NULL,

## FOREIGN KEY("local\_ID") REFERENCES "Local"("local\_ID"),

## FOREIGN KEY("funcionario\_ID") REFERENCES "Funcionario"("funcionario\_ID"),

## PRIMARY KEY("data\_hora","funcionario\_ID","local\_ID")

## );

## ITENS

Temos aqui alguns itens descritos em tópicos sobre um programa de entrada e saída em um sistema de segurança desenvolvido utilizando a abordagem de Desenvolvimento Rápido de Aplicações (RAD):

1. **Objetivo do Programa:**

- Registrar de forma precisa e automatizada a entrada e saída de usuários em um sistema de segurança.

1. **Funcionalidades Principais:**

- Registro eficiente das entradas e saídas de usuários.

- Captura de informações como data e horário de cada evento.

- Interface intuitiva e amigável para facilitar a utilização do programa pelos usuários.

- Funcionalidades de personalização para adaptar o programa às necessidades específicas de cada organização.

1. **Integração e Personalização:**

- Integração de funcionalidades que permitem configurar políticas de acesso.

- Possibilidade de definir horários de funcionamento específicos.

1. **Segurança e Confiabilidade:**

- Adoção de medidas para garantir a segurança dos dados registrados.

- Implementação de protocolos de segurança para proteger informações sensíveis.

- Prevenção de possíveis falhas ou violações de segurança por meio de medidas adequadas.

1. **Avaliação e Melhorias Contínuas:**

- Realização de testes práticos para avaliar a eficácia do programa.

- Coleta de feedback dos usuários para identificar áreas de melhoria.

- Busca constante por refinamentos e atualizações para futuras versões do programa.

## ATUALIZAÇÃO DAS VERSÕES

- Foi inserido no projeto as funcionalidades de INSERIR e LISTAR as entradas e saídas. (Versão 1).

- Foi criada a classe ACESSO.PY para substituir os objetos ENTRADA e SAÍDA. (Versão 2).

- Foi implementada a versão gráfica da classe ACESSO.PY usando a biblioteca do Python TKINTER. (Versão 3).

- Foi implementada o banco de dados e a interface gráfica de LOCAL, criando a classe LOCAL.PY. (Versão 4).

- Foi implementado o sistema de Login para aumentar a segurança do projeto, criando a rotina LOGIN.PY.

- Foi implementado o banco de dados e a interface gráfica de SETOR, criando a classe SETOR.PY.

- Foi implementado o banco de dados e a interface gráfica de FUNCIONARIO, criando a classe FUNCIONARIO.PY.

- Foi implementado o banco de dados e a interface gráfica de USUARIO, criando a classe USUARIO.PY (Versão 5).

- Foi inserido níveis de acesso para melhorar a segurança do projeto, criando a rotina MENUADMIN.PY, página que tem acesso e controle para inserir, excluir, alterar e consultar as classes SETOR, FUNCIONARIO, USUARIO, LOCAL e ACESSO. (Versão 6).

- Foi inserido a funcionalidade de dar as opções de Administrador para usuários que até então não tinham este poder.

- Alteração e embelezamento da interface gráfica do projeto, usando a biblioteca CUSTOM TKINTER para o auxiliar nas mudanças. (Versão 7).

# CONCLUSÃO

A implementação bem-sucedida do programa em Python utilizando a abordagem de Desenvolvimento Rápido de Aplicações (RAD) alcançou os objetivos delineados de forma satisfatória. O programa desenvolvido permite o registro preciso e automatizado das entradas e saídas de usuários em ambientes corporativos, capturando informações essenciais como data e horário. Além disso, a interface intuitiva e amigável, juntamente com funcionalidades de personalização, garante uma experiência de uso eficaz e adaptável às necessidades específicas de cada organização. A segurança e confiabilidade dos dados foram priorizadas, com medidas adequadas implementadas para proteger as informações sensíveis. Por meio de testes práticos e feedback dos usuários, a eficácia e usabilidade do programa foram avaliadas, buscando constantemente melhorias para futuras versões. No geral, essa solução oferece uma gestão de acessos eficiente e robusta, contribuindo significativamente para a segurança, organização e produtividade dos ambientes corporativos.

# REFERÊNCIAS

brModelo - Ferramenta de Ensino: Modelagem de Dados (MER). Disponível em: <http://www.sis4.com/brModelo/download.html>. Acesso em: 26 mar. 2024.

CUSTOMTKINTER DOCUMENTATION: Documentação oficial do CustomTkinter. Disponível em: <https://customtkinter.tomschimansky.com/documentation/>. Acesso em: 1 mai. 2024.

GITHUB: Plataforma líder mundial de desenvolvimento de software, colaboração e segurança. Disponível em: GitHub. Acesso em: 10 mar. 2024.

PROGRAMIZ PYTHON TUTORIAL: Tutorial de Python strftime() - datetime para string. Disponível em: <https://www.programiz.com/python-programming/datetime/strftime>. Acesso em: 1 abr. 2024.

PYTHON DOCUMENTATION: Documentação do hashlib — Hashes seguros e resumos de mensagens — Python 3.12.3. Disponível em: <https://docs.python.org/3/library/hashlib.html>. Acesso em: 16 abr. 2024.

PYTHON DOCUMENTATION: Documentação do tkinter.ttk — Widgets temáticos Tk. Disponível em: <https://docs.python.org/3/library/tkinter.ttk.html>. Acesso em: 12 abr. 2024.

PYTHON DOCUMENTATION: sqlite3 — DB-API 2.0 interface for SQLite databases. Disponível em: <https://docs.python.org/3/library/sqlite3.html>. Acesso em: 19 mar. 2024.

VISUAL STUDIO CODE: Editor de código Visual Studio Code. Disponível em: <https://code.visualstudio.com/Download>. Acesso em: 10 mar. 2024.

W3SCHOOLS PYTHON TUTORIAL: Tutorial de Python. Disponível em: <https://www.w3schools.com/python/>. Acesso em: 15 mar. 2024.