

**CENTRO UNIVERSITÁRIO CURITIBA**  
**CIÊNCIA DA COMPUTAÇÃO**

**Projeto Final - Avaliação A3**

**Golpes Contra Prefeituras -**  
**E-mails Maliciosos**

**Versão 1.0**

**CURITIBA**

**2025**

GABRIEL THIAGO CASINI KEPPEM - 172422353

KAWANY VALERA ZELINSKI DOS SANTOS - 172422765

NICOLE CAVAGNINO JARDIM - 172313373

## **GOLPES CONTRA PREFEITURAS - E-MAILS MALICIOSOS**

Relatório técnico apresentado na UC de  
Sistemas Distribuídos e Mobile do Centro  
Universitário Curitiba – Unicuritiba.

Orientador: Profº. Me. Flávio Henrique

CURITIBA

2025

## SUMÁRIO

<b>1. INTRODUÇÃO.....</b>	<b>4</b>
<b>2. DESENVOLVIMENTO.....</b>	<b>5</b>
2.1. Divisão das Tarefas.....	5
2.2. Estrutura do Projeto.....	6
2.2.1. Modelagem do Banco de Dados.....	6
2.2.2. Ambiente para Desenvolvimento.....	7
2.2.3. Bibliotecas.....	7
2.2.4. Classes, Métodos e Objetos.....	7
2.2.4.1. Classes e Métodos (funções internas à classe).....	7
2.2.4.2. Funções (fora da classe).....	8
2.2.4.3. Objetos (instanciados).....	8
2.3. Explicação da Aplicação/Software.....	9
2.3.1. Configuração do Gmail a ser utilizado.....	9
2.3.2. Configurações de Variáveis no arquivo main.py.....	9
2.3.3. Iniciando aplicação.....	10
2.4. Orientações de execução da Aplicação/Software.....	11
2.5. Repositório.....	12
<b>3. CONCLUSÃO.....</b>	<b>13</b>
<b>REFERÊNCIAS.....</b>	<b>14</b>

## 1. INTRODUÇÃO

O relatório em questão visa apresentar as informações necessárias ao desenvolvimento de uma aplicação para solucionar um dos problemas enfrentados pelos bancos no Brasil com relação a golpes. Esse projeto foi proposto pela UC - Sistemas Distribuídos e Mobile juntamente com parceria do banco Bradesco, e foi escolhido através de uma lista de golpes comuns disponível no site do Banco Central do Brasil (BCB) <sup>1</sup>.

De acordo com uma matéria da Folha de São Paulo, um levantamento feito pelo Serasa Experian em 2024, constatou um aumento de 17% nas tentativas de golpe financeiro contra brasileiros, ao todo, foram mais de 11,5 milhões de tentativas, o equivalente a 2,8 golpes por segundo aproximadamente. Dentre os principais e mais comuns, estão o cartão de crédito, pix falsos e *phishing* <sup>2</sup>.

Por essa razão e pelo fato de que os golpes contra as prefeituras miram as casas dos milhões dos cofres públicos<sup>3</sup>, constata-se necessário abordar aquele que se utiliza de *phishing*, especificamente com e-mails, que consiste no uso de engenharia social para fingir ser alguém em quem a vítima confia para enganá-la a compartilhar dados confidenciais, baixar malware ou se expor a crimes cibernéticos de outras formas.

---

<sup>1</sup> FAQs sobre golpes do BCB disponível em: <https://www.bcb.gov.br/meubc/faqs/s/golpes>

<sup>2</sup> Matéria completa disponível em: <https://www1.folha.uol.com.br/mercado/2025/03/brasil-tem-quase-3-tentativas-de-golpe-financeiro-por-segundo-aponta-serasa.shtml>

<sup>3</sup> Exemplo de golpe contra prefeitura disponível em: <https://g1.globo.com/sp/sao-carlos-regiao/noticia/2025/03/31/prefeitura-paga-boleto-falso-de-r-21-milhoes-no-interior-de-sp-justica-nega-bloqueio-de-valor.ghtml>

## 2. DESENVOLVIMENTO

A proposta desenvolvida visa, como já abordado na **1. INTRODUÇÃO**, tratar uma forma de prevenção e controle de *phishing* por e-mail. A ideia seria manter um registro em um banco de dados de e-mails seguros para que possa ser monitorado os e-mails recebidos pelos funcionários em seus endereços comerciais e disparado um aviso sobre possíveis golpes e informações não confiáveis, assim como o registro dessas movimentações para disponibilizar relatórios analíticos no futuro.

A seguir nas seções **2.1. Divisão das Tarefas** será identificado as tarefas e responsabilidades de cada membro da equipe, **2.2. Estrutura do Projeto** será abordado especificações do projeto como versões, bibliotecas, classes e até mesmo a modelagem de dados utilizada, **2.3. Explicação da Aplicação/Software** conterá um passo a passo de uso para o usuário, **2.4. Orientações de Execução da Aplicação/Software** se encontra quais as dependências e passo a passo necessário para execução da aplicação e por fim **2.5. Repositório** possui o link para o repositório onde contém o código e banco de dados desenvolvidos.

### 2.1. Divisão das Tarefas

O Gabriel T. C. Keppen e a Nicole C. Jardim ficaram responsáveis pela integração com o banco de dados, lógica e código para verificação de e-mails.

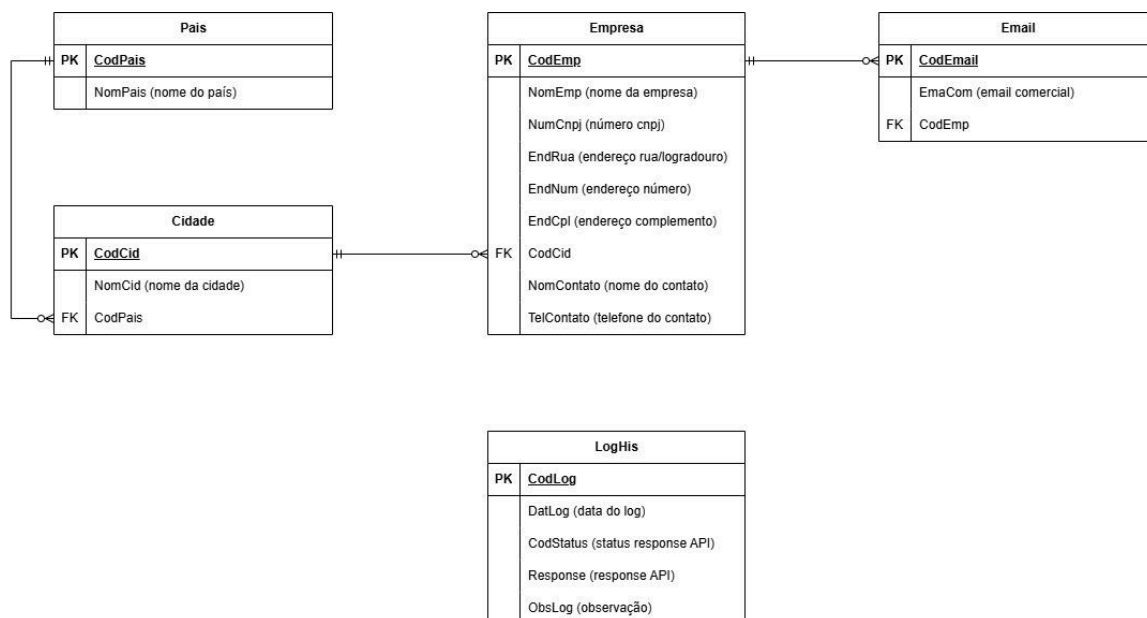
Kawany V. Z. dos Santos foi a responsável pela coordenação e organização da equipe, elaboração do relatório e do banco de dados, assim como auxílio ao desenvolvimento.

## 2.2. Estrutura do Projeto

O projeto foi criado separando na pasta “banco”, o código que criou o banco de dados e suas tabelas (banco.py), o código para popular o banco com dados para testes (dados\_testes.py), o banco (projeto\_bradesco.db) e o DER do mesmo (DER-Golpe\_Prefeitura.jpeg). Assim como uma pasta “documentacao” onde contém o relatório com documentação completa do projeto.

Fora das pastas se encontra o código principal (main.py) com o menu que permite consultar os registros no banco de dados dos Logs e faz a chamada das classes e funções para buscar e-mails e processá-los dentro do arquivo (obj.py).

### 2.2.1. Modelagem do Banco de Dados



O banco de dados criado foi apenas para prova de conceito (POC), onde serão gravadas informações das empresas clientes e fornecedores da prefeitura e também seus respectivos e-mails cadastrados, esses serão entendidos como seguros e auxiliarão na verificação de tentativas de comunicação por e-mails maliciosos. Também foi criada uma tabela para salvar os logs de ações realizadas pelo sistema e movimentações.

### 2.2.2. Ambiente para Desenvolvimento

- Linguagem python versão: 3.11;
- IDE Visual Studio Code versão: 1.100.2;
- Windows 11.

### 2.2.3. Bibliotecas

- Sqlite3 = biblioteca padrão do Python, não é necessário instalá-la. Permite criar, conectar e manipular bancos de dados SQLite diretamente no Python;
- Datetime = biblioteca padrão do Python, não é necessário instalá-la. Manipula datas e horários;
- Smtplib = biblioteca padrão do Python, não é necessário instalá-la. Usada para enviar e-mails via SMTP;
- Imaplib = biblioteca padrão do Python, não é necessário instalá-la. Permite logar em servidores e acessar e-mails via IMAP.
- Email = biblioteca padrão do Python. Fornece ferramentas para criar, interpretar e manipular mensagens de e-mail no formato RFC.
- Parseaddr = Função utilizada para extrair o endereço de e-mail limpo de uma string que pode conter nome e e-mail juntos.
- MIMEText = Classe usada para criar o corpo do e-mail em texto simples (plain text) ou HTML.
- MIMEMultipart = Classe usada para criar e-mails com múltiplas partes, como texto + anexos, ou texto em plain text + HTML.

### 2.2.4. Classes, Métodos e Objetos

#### 2.2.4.1. Classes e Métodos (funções internas à classe)

- GerenciadorEmailRecebido: Lê e processa e-mails não lidos da caixa de entrada (IMAP):
  - `__init__`: Inicializa a classe com e-mails e configurações básicas;
  - `conectar`: Conecta à conta Gmail via IMAP;

- *extrair\_remetente*: Extrai o endereço do remetente de um e-mail recebido;
- *processar\_emails\_nao\_lidos*: Lê e-mails não lidos, extrai o remetente e chama o verificador.
- *VerificadorEmail*: Verifica se o remetente do e-mail está autorizado (consultando o banco):
  - *\_\_init\_\_*: Inicializa o verificador com dados do banco e e-mail;
  - *conectar\_banco*: Cria uma conexão com o banco SQLite.;
  - *verificar\_email*: Verifica se o e-mail do remetente é autorizado e registra no log.
- *AlertaEmailSuspeito*: Envia um e-mail de alerta se o remetente for suspeito:
  - *\_\_init\_\_*: Inicializa com dados do remetente e destino do alerta;
  - *enviar\_alerta*: Envia um e-mail de alerta informando sobre o remetente suspeito.

#### 2.2.4.2. Funções (fora da classe)

- *exibir\_logs()*: Exibe todos os registros da tabela de logs (LogHis);
- *exibir\_logs\_por\_data(data)*: Exibe logs filtrados por uma data específica;
- *exibir\_logs\_por\_remetente(remetente)*: Exibe logs filtrados por remetente;
- *menu\_interativo()*: Interface de menu no terminal para o usuário interagir com o sistema.

#### 2.2.4.3. Objetos (instanciados)

- *gerenciador*: da classe *GerenciadorEmailRecebido*, usado para buscar e processar e-mails não lidos:



- *self.verificador*: da classe *VerificadorEmail*, usado para verificar remetentes:
  - *self.alerta*: da classe *AlertaEmailSuspeito*, usado para enviar e-mail de alerta.

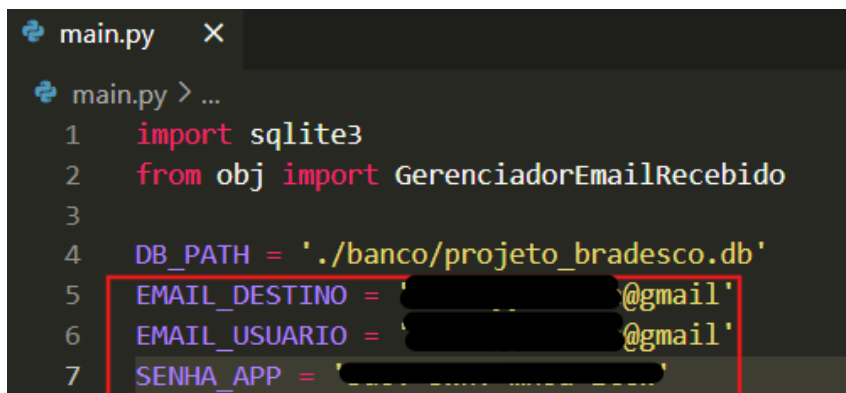
## 2.3. Explicação da Aplicação/Software

### 2.3.1. Configuração do Gmail a ser utilizado

- Para usar a aplicação é necessário estar com a sua conta do e-mail (requisito ser uma do Gmail para o funcionamento eficiente), logada e com as verificações de duas etapas ativada e já feita, segue o FAQ da Google: <https://support.google.com/accounts/answer/185839?hl=pt-BR&co=GENIE.Platform%3DDesktop>;
- Crie uma senha de app, segue o FAQ da Google para criá-la: <https://support.google.com/accounts/answer/185833?hl=pt-B>.

### 2.3.2. Configurações de Variáveis no arquivo main.py

- *EMAIL\_DESTINO*: Preencher e-mail que irá receber alertas sobre recebimentos suspeitos;
- *EMAIL\_USUARIO*: Preencher e-mail que será monitorado a caixa de entrada;
- *SENHA\_APP*: Preencher com a senha de app criada para a aplicação.



```
main.py X
main.py > ...
1 import sqlite3
2 from obj import GerenciadorEmailRecebido
3
4 DB_PATH = './banco/projeto_bradesco.db'
5 EMAIL_DESTINO = '[redacted]@gmail'
6 EMAIL_USUARIO = '[redacted]@gmail'
7 SENHA_APP = '[redacted]'
```

### 2.3.3. Iniciando aplicação

- Digitar no terminal ***python main.py*** (verificar se está no diretório que contém esse arquivo), irá aparecer um menu:

```

--- MENU ---
1. Verificar novos e-mails
2. Exibir todos os logs
3. Filtrar logs por data
4. Filtrar logs por remetente
5. Sair
Escolha uma opção: █

```

1. Para verificar e-mails na caixa de entrada do *EMAIL\_USUARIO*, digitar 1 e apertar a tecla Enter:

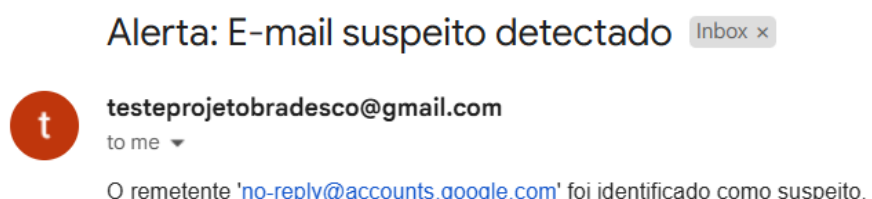
```

Escolha uma opção: 1
Verificando remetente: no-reply@accounts.google.com
[ALERTA] Email suspeito: no-reply@accounts.google.com
Alerta enviado para testeprojetobradesco@gmail.com sobre 'no-reply@accounts.google.com'

```

**Atenção:** Possível erro - “ Erro ao conectar: b'[AUTHENTICATIONFAILED] Invalid credentials (Failure)' ”, revisar configurações da senha de app, informada na variável *SENHA\_APP*, do seu gmail.

Se não houver erro, será salvo a resposta da verificação na tabela LogHis do banco de dados e um e-mail será enviado ao *EMAIL\_DESTINO* setado no item anterior:



2. Para exibir todos os logs pressione 2 e aperte a tecla enter;
3. Para filtrar os logs por data pressione 3 e aperte a tecla enter, depois digite no formato indicado no terminal a data desejada:

```
Escolha uma opção: 3
Digite a data no formato AAAA-MM-DD: 2025-06-06

[LOGS - DATA 2025-06-06]

Data: 2025-06-06 00:18:21 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
Data: 2025-06-06 00:18:17 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
Data: 2025-06-06 00:18:13 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
```

4. Para filtrar logs por remetente pressione 4 e aperte a tecla enter, depois digite o remetente desejado:

```
Escolha uma opção: 4
Digite o e-mail do remetente: no-reply@accounts.google.com

[LOGS - REMETENTE no-reply@accounts.google.com]

Data: 2025-06-06 00:18:21 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
Data: 2025-06-06 00:18:17 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
Data: 2025-06-06 00:18:13 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
Data: 2025-06-05 04:56:10 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
Data: 2025-06-05 04:56:09 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
Data: 2025-06-05 04:41:11 | Status: 403 | Resposta: None | Obs: E-mail no-reply@accounts.google.com não reconhecido.
```

5. Para finalizar a aplicação digite 5 e aperte a tecla enter.

## 2.4. Orientações de execução da Aplicação/Software

1. Baixar o projeto na sua máquina local através do link do item **2.5. Repositório**;
2. Abrir a pasta **ProjetoUni\_GolpeEmail** pelo Visual Studio Code;
3. Para monitorar as verificações feitas, baixar a extensão *SQLite Viewer by Florian Klampfer* no Visual Studio Code e abrir o arquivo **banco\projeto\_bradesco.db** pelo SQLite Viewer:
  - a. **Atenção:** Caso seja necessária a criação do banco de dados novamente, rodar o código pelo comando **python banco\banco.py** no terminal;
  - b. Rodar no terminal **python banco\dados\_testes.py**, caso necessite popular o banco manualmente;
  - c. A tabela LogHis apresenta os e-mails verificados e o resultado da verificação dos mesmos, nas tabelas pais e cidade se encontram os países e cidades cadastrados no banco de dados, na empresa as empresas cadastradas e na tabela email os e-mails associados as empresas cadastradas ao qual estão classificados como seguros na verificação do nosso projeto.

4. Todas verificações e alimentação do Log do banco dados são feitas através do arquivo python ***main.py***, o mesmo arquivo apresenta um menu para filtro e exibição de Logs anteriores salvos no banco. Digitar no terminal ***python main.py***.

## 2.5. Repositório

Disponível em: [https://github.com/kwvalera/ProjetoUni\\_GolpeEmail](https://github.com/kwvalera/ProjetoUni_GolpeEmail)

### 3. CONCLUSÃO

*Phishing* tem se tornado um problema que merece atenção e tem necessidade de ações para a prevenção e combate não só em órgãos públicos como a prefeitura, mas também em empresas privadas que visam proteger seu patrimônio e informações sigilosas. É um desafio grande, já que não se trata somente de um ataque cibernético onde pode-se criar barreiras e proteções, e sim de artifícios para se utilizar do erro humano que pode ser facilmente manipulado e enganado.

Com essa implantação de controle de e-mails criada nesse projeto pode-se desenvolver uma ferramenta para prevenção de entrega de e-mails falsos e mal intencionados, enquanto dificulta que os golpistas consigam contato com os funcionários e cria avisos para gerar desconfiança nas solicitações falsificadas.

Recomenda-se que o software criado e apresentado nesse relatório seja usado em conjunto com outras formas de prevenção e conscientização, como campanhas com palestras e simulações, como por exemplo, e-mails spam planejadamente enviados de propósito para monitorar e analisar o comportamento dos usuários quanto a eles. E mesmo com todas essas medidas não teria como impedir completamente de que esses ataques sejam efetivos, porém irá diminuir significativamente as chances do acontecimento e da perda de bens e informações privadas através deles.

Diante do exposto, ter um procedimento e ferramenta para controlar e monitorar as tentativas de golpes como esse são estritamente essenciais para a segurança da informação em uma organização importante que se preocupa com a segurança de dados e de seu patrimônio.

## REFERÊNCIAS

AGÊNCIA CNM de Notícias. CNM. Prefeitos questionam Banco do Brasil sobre golpe em contas e pedem ressarcimento. Disponível em: <<https://cnm.org.br/comunicacao/noticias/prefeitos-questionam-banco-do-brasil-sobre-golpe-em-contas-e-pedem-ressarcimento>>. Acesso em: 25 mai. 2025.

FAMURS. Banco Central alerta prefeituras contra golpe telefônico. Disponível em: <<https://famurs.com.br/noticia/257>>. Acesso em: 25 mai. 2025.

GALVÃO, Júlia. Folha de São Paulo. Brasil tem quase 3 tentativas de golpe financeiro por segundo, aponta Serasa. Disponível em: <<https://www1.folha.uol.com.br/mercado/2025/03/brasil-tem-quase-3-tentativas-de-golpe-financeiro-por-segundo-aponta-serasa.shtml>>. Acesso em: 25 mai. 2025.

G1 São Carlos e Araraquara. Prefeitura cai em golpe e paga R\$ 2,1 milhões no interior de SP; liminar bloqueia valor. Disponível em: <<https://g1.globo.com/sp/sao-carlos-regiao/noticia/2025/03/31/prefeitura-paga-boleto-falso-de-r-21-milhoes-no-interior-de-sp-justica-nega-bloqueio-de-valor.ghtml>>. Acesso em: 25 mai. 2025.

KOSINSKI, Matthew. IBM. Phishing. Disponível em: <<https://www.ibm.com/br-pt/think/topics/phishing>>. Acesso em: 25 mai. 2025.

SENADO NOTÍCIAS. Golpes digitais atingem 24% da população brasileira, revela DataSenado. Disponível em: <<https://www12.senado.leg.br/noticias/materias/2024/10/01/golpes-digitais-atingem-24-da-populacao-brasileira-revela-datasenado>>. Acesso em: 25 mai. 2025.

VAYANSKY, I.; KUMAR, S. Phishing – challenges and solutions. Computer Fraud & Security, p. 15–20, 01 jan. 2018. Disponível em: <[https://www.researchgate.net/profile/Sathish-Kumar-26/publication/322823383\\_Phishing\\_-\\_challenges\\_and\\_solutions/links/5acde9fa4585154f3f420911/Phishing-challenges-and-solutions.pdf](https://www.researchgate.net/profile/Sathish-Kumar-26/publication/322823383_Phishing_-_challenges_and_solutions/links/5acde9fa4585154f3f420911/Phishing-challenges-and-solutions.pdf)>. Acesso em: 25 mai. 2025.

VENV - Criação de ambientes virtuais. Python 3.11. Disponível em: <https://docs.python.org/pt-br/3.11/library/venv.html>. Acesso em: 25 mai. 2025.