



# **EyeWeb Reborn**

**Francisco Rafael Carocinho Ribeiro – Nº 2024123**

**José Samuel da Rocha Oliveira – Nº 2024172**

**Tiago Filipe Sousa Carvalho – Nº 2024180**

**Ana Rita da Silva Monteiro – Nº 2024041**

**Vanina Kollen – Nº 2024056**

**2º Ano de CTeSP de Cibersegurança**

**Projeto Final**

Hélder Pinto | Vitor Santos | Vitor Rocha | Ricardo Moura

Porto, 2025

## Resumo

Este projeto consiste na elaboração de uma versão atualizada do *EyeWeb* denominado *EyeWeb Reborn* e para melhor compreender e obter uma visão mais ampla sobre o contexto e as motivações desta nova versão, é necessário conhecer a base a partir da qual o trabalho será desenvolvido.

O *EyeWeb* é uma ferramenta capaz de analisar a segurança de websites, palavras-passe, e-mails e números de telemóvel. Através desta plataforma, é possível verificar a integridade de sites, identificando potenciais ameaças e vulnerabilidades, avaliar a robustez das palavras-passe e confirmar se estas já foram comprometidas em fugas de dados, bem como verificar se determinados números de telemóvel e e-mails foram igualmente afetados. Sempre que uma palavra-passe seja identificada como comprometida, o sistema alerta o utilizador e fornece informações sobre os dados expostos, permitindo assim que tome medidas rápidas para reforçar a segurança.

A funcionalidade de análise de websites permite verificar se um site é malicioso, através da avaliação de fatores como o histórico de ameaças, os certificados de segurança e a existência de ligações suspeitas. A funcionalidade de verificação de palavras-passe permite ao utilizador saber se a sua palavra-passe é segura e se já foi exposta em bases de dados comprometidas. Se sim, o sistema indica quando e onde ocorreu essa exposição e ajuda o utilizador a tomar as medidas necessárias para proteger as suas contas. Por sua vez, a funcionalidade de verificação de números de telemóvel e e-mails permite ao utilizador confirmar se o seu número ou e-mail já foi comprometido. O *software* foi desenvolvido para ser intuitivo e acessível, permitindo que qualquer pessoa, independentemente dos seus conhecimentos técnicos, o utilize para reforçar a sua segurança online.

O *EyeWeb Reborn* constitui a evolução do *EyeWeb*, tendo sido desenvolvido com o objetivo de tornar a navegação e a gestão de websites ainda mais seguras. Trata-se de uma plataforma que combina a análise de websites, a verificação de palavras-passe, e-mails e de números de telemóvel, dois agentes reativos (utilizador e administrador) e APIs externas, de forma a proteger tanto utilizadores como administradores contra ameaças online.

O sistema é capaz de identificar problemas em tempo real, tais como acessos suspeitos ou palavras-passe, números de telemóvel e e-mails comprometidos, permitindo que os responsáveis pelos websites e os próprios utilizadores saibam exatamente o que está a acontecer e possam agir o mais rapidamente possível. Os agentes reativos integrados funcionam como um apoio, ao responderem a dúvidas sem comprometer a privacidade ou a segurança dos dados. Apesar de introduzir novas funcionalidades, esta versão do *EyeWeb*

mantém as funcionalidades tradicionais da plataforma, nomeadamente a análise de websites à procura de *malware* ou *phishing* e a verificação da exposição de palavras-passe, e-mails e números de telemóvel em bases de dados comprometidas, informando sempre o utilizador de forma clara sobre quaisquer riscos detetados.

Em suma, o *EyeWeb Reborn* é uma solução completa de segurança digital, combinando proteção ativa, monitorização constante e apoio aos utilizadores através de agentes, garantindo uma experiência online mais segura e tranquila.

# Índice

EyeWeb Reborn.....	i
Resumo.....	ii
Índice.....	iv
Lista de Acrónimos e Siglas.....	vi
1. Introdução.....	1
1.1. Enquadramento.....	1
1.2. Motivação.....	2
1.3. Objetivos.....	2
1.4. Público-Alvo.....	3
1.5. Critérios de Sucesso.....	3
1.6. Comunicação.....	4
1.7. Requisitos.....	4
1.7.1. Funcionais.....	4
1.7.2. Não Funcionais.....	6
1.8. Análise das Necessidades e Prioridades.....	6
1.8.1. Necessidades.....	6
1.8.2. Prioridades.....	7
1.9. Limitações do Produto.....	8
1.10. Design.....	9
1.10.1. Cenário de Sucesso Principal.....	9
1.11. Medidas a Implementar.....	11
1.11.1. Controles gerais.....	11
1.11.2. Modo Admin.....	11
1.12. Interface Gráfica e Funcionamento.....	12
1.13. Estrutura do Relatório.....	14
1.14. Problema a resolver e Importância do projeto.....	14
1.15. Características do EyeWeb Reborn.....	14
2. Estado da Arte / Soluções Existentes.....	16
3. Desenvolvimento.....	17
3.1. Planeamento do Projeto.....	17

3.2. Análise de Sistemas/SRS.....	18
3.3. Desenvolvimento do Projeto.....	18
3.4. Resultados.....	23
4. Arquitetura.....	40
5. Wireframes / Mockups.....	44
6. Scripts ou Playbooks de Setup.....	55
7. Políticas de Segurança e Isolamento do Ambiente.....	62
8. Manual de Utilização.....	67
9. Conclusão.....	69
9.1. Limitações e Sugestões Futuras.....	69
9.2. Considerações sobre a Inteligência Artificial no Contexto deste Projeto.....	70
Referências.....	71
Diagramas.....	74
Links.....	81

## **Lista de Acrónimos e Siglas**

<b>API</b>	Application Programming Interface – Interface de Programação de Aplicações
<b>DDoS</b>	Distributed Denial of Service – Negação de Serviço Distribuída
<b>HTML</b>	HyperText Markup Language
<b>HTTP</b>	HyperText Transfer Protocol
<b>MFA</b>	Multi-Factor Authentication
<b>IP</b>	Internet Protocol
<b>JWT</b>	JSON Web Token
<b>SSL</b>	Secure Sockets Layer
<b>TLS</b>	Transport Layer Security
<b>Node.js</b>	Ambiente de execução JavaScript baseado no motor Chrome V8
<b>OWASP</b>	Open Web Application Security Project
<b>SQL</b>	Structured Query Language – Linguagem Estruturada de Consulta
<b>SQLite</b>	Structured Query Language Lite – Base de dados leve
<b>UML</b>	Unified Modeling Language – Linguagem de Modelação Unificada
<b>URL</b>	Uniform Resource Locator

<b>VS Code</b>	Visual Studio Code
<b>HTTPS</b>	HyperText Transfer Protocol Secure
<b>IA</b>	Inteligência Artificial
<b>MAC</b>	Media Access Control
<b>PBL</b>	PBL Project Based Learning
<b>RBAC</b>	Role-Based Access Control
<b>SRS</b>	Software Requirements Specification
<b>BD</b>	Base de Dados
<b>FTP</b>	File Transfer Protocol
<b>NoSQL</b>	Not Only SQL

# 1. Introdução

Como anteriormente referido, este projeto consiste na criação de uma versão atualizada do *EyeWeb*, denominada *EyeWeb Reborn*. Esta nova versão tem como objetivo central a proteção dos dados dos utilizadores, oferecendo uma ferramenta acessível, intuitiva e gratuita, num contexto de constante evolução tecnológica, em que os sistemas de segurança precisam de acompanhar o ritmo das ameaças. O projeto pretende apoiar a gestão de riscos, fornecendo uma abordagem estruturada e fundamentada para a tomada de decisões, contribuindo assim para a proteção dos ativos informáticos da organização.

O *EyeWeb Reborn* integra ainda dois Agentes Reativos, que fornecem segurança e suporte contínuos tanto ao administrador como ao utilizador. O agente do administrador monitoriza, em tempo real, as ligações ativas no website, identifica tentativas de acesso suspeitas ou bloqueadas e alerta para vulnerabilidades, como ataques de *SQL injection*, *DDoS* ou *brute force*. Para além disso, propõe medidas de proteção, como o bloqueio de endereços IP suspeitos ou o reforço da *firewall*. Por sua vez, o agente do utilizador explica ao cliente como o website funciona, de que forma deve ser utilizado e quais as boas práticas de segurança a adotar. Este agente fornece ainda informações sobre o projeto e atua dentro de restrições bem definidas, o que garante que apenas responda ao que foi programado, sem comprometer a segurança do sistema nem expor o utilizador a riscos adicionais.

Em suma, esta atualização tem como objetivo proporcionar uma maior proteção ao website e aos seus utilizadores, através da integração de ferramentas como o Agente Reativo e *APIs* externas, garantindo que as implementações estejam alinhadas com os objetivos do projeto e promovam a segurança de forma eficaz.

## 1.1. Enquadramento

O *EyeWeb Reborn* constitui um exemplo de ferramentas e sistemas inseridos no âmbito da cibersegurança, ao proporcionar e garantir uma maior segurança online. Este projeto foi desenvolvido no âmbito da unidade curricular de Projeto Final de Cibersegurança, integrada no 2.º ano do CTeSP de Cibersegurança, com o objetivo de evoluir o projeto base (*EyeWeb*).

Para além de melhorar a versão original, o *EyeWeb Reborn* introduz dois agentes reativos, integra *APIs* externas e reforça os métodos de segurança implementados. Desta forma, o projeto torna-se uma solução essencial para a proteção de dados e da privacidade dos utilizadores, oferecendo um sistema simples de utilizar, seguro e que protege, cumprindo aquilo a que se propõe.

## **1.2. Motivação**

A motivação principal para este projeto surge da crescente necessidade de ferramentas eficazes, intuitivas e automáticas para a proteção digital. A versão original do *EyeWeb* era funcional, mas limitada na resposta a ameaças em tempo real. Assim, o *EyeWeb Reborn* foi criado para prevenir ataques antes que ocorram e auxiliar o administrador e o utilizador com um agente reativo, que simplifica e fortalece a gestão da segurança.

Este projeto também representa uma oportunidade de aplicar, de forma prática, os conhecimentos adquiridos no CTeSP em Cibersegurança, promovendo o trabalho colaborativo e o desenvolvimento de competências técnicas essenciais na área.

## **1.3. Objetivos**

O objetivo do *EyeWeb Reborn* é criar uma plataforma de segurança digital completa, capaz de proteger websites e contas de utilizadores contra ameaças online, oferecendo simultaneamente assistência inteligente. A plataforma pretende ser um aliado tanto para administradores, fornecendo monitorização e alertas em tempo real, como para utilizadores, garantindo que os seus dados se mantêm seguros.

A integração de um agente no site permite que este funcione como agente de segurança, protegendo contra acessos suspeitos e monitorizando vulnerabilidades e como assistente inteligente, interagindo com o utilizador de forma simples e acessível.

O sucesso do projeto será avaliado com base em diversos critérios que assegurem a sua eficácia e impacto, nomeadamente:

- Precisão na identificação de websites maliciosos;
- Avaliação da robustez das palavras-passe;
- Precisão na identificação de números de telemóvel e e-mails comprometidos;
- Facilidade de utilização e acessibilidade para todos os utilizadores;
- Garantia de segurança e privacidade dos dados;
- Desempenho eficiente em tempo real;
- Implementação de um sistema de cache inteligente para otimizar a performance e reduzir custos;

- Garantia de alta disponibilidade, com serviços *cloud* gratuitos;
- Possibilidade de verificação de URLs em tempo real;
- Classificação de URLs como seguras, suspeitas ou perigosas;
- Integração de serviços externos de segurança reconhecidos;
- Fornecimento de explicações claras em português europeu através de IA;
- Aplicação de boas práticas de segurança informática.

Desta forma, o *EyeWeb Reborn* combina proteção, monitorização e assistência inteligente, promovendo uma experiência segura, intuitiva e eficiente para administradores e utilizadores, respondendo às exigências atuais de segurança digital.

#### **1.4. Público-Alvo**

O *EyeWeb Reborn* destina-se a dois grupos principais:

1. **Administradores de websites:** que necessitam de uma ferramenta que lhes permita monitorar a segurança, identificar vulnerabilidades e agir rapidamente em caso de ameaças;
2. **Utilizadores finais:** que procuram uma experiência segura, sabem que os seus dados e palavras-passe são verificados e protegidos.

#### **1.5. Critérios de Sucesso**

O sucesso deste projeto será avaliado com base em vários fatores que garantem a eficácia, a usabilidade e o impacto da ferramenta na segurança digital dos utilizadores. Para que o projeto seja considerado bem-sucedido, devem ser cumpridos os seguintes critérios:

1. **Precisão:** A ferramenta deve identificar corretamente websites maliciosos, avaliar a força das palavras-passe e identificar números de telemóvel ou e-mails comprometidos, incluindo verificações de fugas de dados (Precisão acima de 95% nas deteções);
2. **Usabilidade:** A interface deve ser intuitiva e de fácil utilização, permitindo que qualquer utilizador consiga realizar as análises de forma simples e rápida (Análise de URL em menos de 15 segundos (verificação completa) e resposta em cache em menos de 100ms);

- 3. Segurança e Privacidade:** Nenhuma informação pessoal deve ser armazenada ou partilhada. As análises devem ser seguras e temporárias;
- 4. Desempenho:** A ferramenta deve ter uma resposta rápida, sem falhas, garantindo eficiência na análise de sites, palavras-passe, e-mails e números de telemóvel (disponibilidade 24/7);
- 5. Adoção:** O número de utilizadores e o feedback positivo indicam a utilidade e aceitação da ferramenta;
- 6. Cumprimento de prazos e orçamento:** O projeto deve ser concluído dentro do cronograma e orçamento estabelecidos (Zero custos operacionais (planos gratuitos)).

Esses critérios garantirão que o projeto atenda aos objetivos e tenha um impacto positivo na segurança digital dos utilizadores.

## 1.6. Comunicação

A comunicação entre a equipa e o cliente será feita por e-mail da empresa, através do qual um membro da equipa terá acesso. Assim, o cliente poderá fazer perguntas e solicitar melhorias ou soluções caso o software não funcione como esperado.

## 1.7. Requisitos

### 1.7.1. Funcionais

#### 1. Análise de Websites

O utilizador deve conseguir inserir a URL de um site para obter uma avaliação sobre a sua segurança, identificando se o site é seguro ou se há risco de ser malicioso. O objetivo principal é ajudar a prevenir ataques como *phishing*, *malware* ou websites fraudulentos, fornecendo uma análise clara e comprehensível. Esta funcionalidade combina a verificação da segurança de URLs em tempo real, análise automática com recurso a Inteligência Artificial, integração com serviços de segurança reconhecidos e sistema de cache para melhorar o desempenho. Além de que, foi implementado um sistema de automação que reanalisa automaticamente URLs antigos de forma periódica. Esta tarefa é executada diariamente, garantindo que os resultados apresentados aos utilizadores se mantêm atualizados ao longo do tempo;

## **2. Verificação de Palavras-passe, Números de telemóvel e E-Mails**

A ferramenta deve permitir que o utilizador introduza uma palavra-passe, um endereço de e-mail ou um número de telemóvel para análise. O objetivo é avaliar se a palavra-passe é suficientemente robusta e caso tenha sido comprometida numa violação de dados, indicar onde ocorreu. No que diz respeito tanto à verificação de endereços de e-mail como à verificação de números de telemóvel, a ferramenta deve permitir identificar se ocorreu uma exposição em fugas de dados conhecidas;

## **3. Relatórios Claros**

Depois da análise de um site ou palavra-passe, número de telemóvel ou e-mail, o sistema deverá gerar um relatório simples e direto, com informações sobre os riscos encontrados e sugestões sobre como melhorar a segurança;

## **4. Alertas de Palavras-passe Expostas**

Se a palavra-passe inserida já foi exposta em alguma fuga de dados, a ferramenta deverá alertar o utilizador e informar em qual violação ela foi encontrada, para que ele possa tomar ações, como trocar a senha;

## **5. Interface Simples e Intuitiva**

O objetivo é que o utilizador consiga fazer tudo de maneira rápida e sem complicações. A interface deve ser simples, permitindo que qualquer pessoa, independentemente do seu conhecimento técnico, consiga utilizar a ferramenta com facilidade;

## **6. Integração com Agente**

A integração de dois Agentes tem como objetivo apoiar a segurança, a gestão administrativa do sistema e ajudar o utilizador. O Agente do administrador deve demonstrar as conexões ativas no site, tentativas de acesso suspeitas ou bloqueadas e alertar para vulnerabilidades como *SQL injection*, *DDoS* ou ataques de *brute force*. Além disso, sugere medidas de proteção, como o bloqueio de IPs suspeitos ou o reforço da *firewall*, ou seja, atua como um vigilante contínuo contra potenciais ameaças ou ataques. Paralelamente, o Agente do cliente deve explicar de forma clara como utilizar o site, orientando o utilizador na navegação, funcionalidades disponíveis e boas práticas de uso da plataforma;

## **7. Gestão e Monitorização Administrativa**

O administrador deve ter a capacidade de monitorizar e gerir a segurança do sistema de forma centralizada. Deve poder consultar regtos e relatórios de

atividades, aceder a um painel de administração isolado e seguro, bem como bloquear endereços IP considerados suspeitos.

### **1.7.2. Não Funcionais**

#### **1. Desempenho Rápido**

A ferramenta deve ser ágil. O utilizador não deve esperar muito tempo para ver os resultados das análises, seja para websites, e-mails, palavras-passe ou números de telemóvel;

#### **2. Segurança Garantida**

Nenhuma informação sensível do utilizador, como dados pessoais, deve ser guardada no sistema. A análise deve ser feita de forma temporária e segura, sem riscos para o utilizador;

#### **3. Compatibilidade com Diferentes Navegadores**

A ferramenta deve funcionar bem nos navegadores mais usados (como *Chrome*, *Firefox* e *Edge*), para as pessoas poderem acessá-la de qualquer lugar;

#### **4. Capacidade de Crescer**

O sistema deve conseguir crescer, caso seja necessário. Ou seja, se no futuro mais pessoas começarem a usar ou o sistema precisar de mais funcionalidades, ele deve continuar a funcionar sem problemas;

#### **5. Disponibilidade Constante**

A ferramenta deverá estar disponível sempre, 24/7. O sistema precisa ser estável, com o mínimo de interrupções e manutenção, para o utilizador poder usá-lo sempre que precisar;

#### **6. Privacidade Respeitada**

O foco é a privacidade do utilizador. O sistema não deverá exigir registo de dados pessoais e, de maneira alguma, deverá guardar ou partilhar informações sensíveis sem o consentimento explícito do utilizador.

## **1.8. Análise das Necessidades e Prioridades**

### **1.8.1. Necessidades**

#### **1. Segurança das Palavras-Passe, E-mails, Websites e Números de Telemóvel**

O utilizador precisa de uma ferramenta que consiga analisar palavras-passe, e-mails, websites e números de telemóvel para garantir que estão seguros. É

fundamental verificar se palavras-passe, e-mails e números de telemóvel já foram comprometidos em vazamentos de dados e, se sim, onde aconteceu, para o utilizador poder tomar medidas de proteção. Quanto ao website, é fundamental obter uma avaliação sobre a sua segurança, identificando se o website é seguro ou se há risco de ser malicioso;

## **2. Simples e Fácil de Usar**

A ferramenta deve ser fácil de usar, sem complicações. O objetivo é que qualquer pessoa use a plataforma para verificar palavras-passe, e-mails, websites e números de telemóvel sem precisar de ser um especialista em segurança, tudo de forma clara e prática;

## **3. Verificar Palavras-Passe, E-mails, Websites e Números de Telemóvel de Forma Rápida**

O utilizador quer uma maneira rápida e eficiente de verificar se as suas palavras-passe são fortes, se os e-mails ou números de telemóvel foram comprometidos e se os websites que visita estão seguros. A ferramenta deve demonstrar essas informações claramente, ao ajudar o utilizador a proteger as suas contas e dados;

## **4. Privacidade e Confidencialidade**

As informações dos utilizadores devem ser protegidas a todo o custo. Nenhum dado pessoal deve ser armazenado ou compartilhado. O sistema tem de garantir que a privacidade seja sempre respeitada.

### **1.8.2. Prioridades**

#### **1. Alta Prioridade**

- a. Segurança e Proteção: A prioridade número um é garantir que o sistema analise palavras-passe, e-mails, números de telemóvel e websites de forma segura e eficaz, sem comprometer os dados dos utilizadores;
- a. Facilidade de Uso: A “interface” da ferramenta deve ser simples e intuitiva, para qualquer pessoa conseguir usar sem dificuldades, sem ter de aprender nada complexo;

#### **2. Média Prioridade**

- a. Relatórios Simples e Úteis: Embora os relatórios sobre palavras-passe, e-mails, números de telemóvel e websites sejam importantes, eles não devem ser

- complicados. A ideia é que as informações sejam diretas, com dicas claras sobre como melhorar a segurança;
- b.** Verificação de Vazamentos de Palavras-Passe: Saber se a palavra-passe foi exposta em algum ataque é uma funcionalidade importante, mas a verificação da força da palavra-passe e da segurança do website deve vir em primeiro lugar;
  - c.** Verificação de Vazamentos de Números de Telemóvel e E-mails: Saber se o número de telemóvel ou e-mail foram expostos em alguma base de dados;
  - d.** Verificação de Websites: Concretização de uma avaliação de um url de um website sobre a sua segurança, identificando se o site é seguro ou se há risco de ser malicioso.

### **3. Baixa Prioridade**

- a.** Escalabilidade no Futuro: Embora o sistema deva ser escalável para futuras melhorias, no início, o foco será garantir que ele funcione bem e de forma estável;
- b.** Suporte a Vários Idiomas: A tradução para diferentes idiomas pode ser feita depois, quando o sistema estiver completamente funcional e a sua base de utilizadores já estiver estabelecida.

## **1.9. Limitações do Produto**

### **1. Dependência da Entrada do Utilizador**

O sistema depende das palavras-pases, números de telemóvel, e-mails e sites que o utilizador fornece, ou seja, para funcionar corretamente, é necessário que o utilizador insira as informações certas. Sem essas entradas, a ferramenta não pode fazer a verificação;

### **2. Dependência de Conectividade**

O sistema requer uma conexão contínua à internet para consultar as bases de dados e *APIs* externas;

### **3. Limitações na Verificação de Websites**

A análise de sites para verificar se são maliciosos depende de fontes externas, bases de dados e limites de utilização impostos pelas *APIs*. Se essas fontes não estiverem atualizadas ou não forem acessíveis, a análise pode não ser totalmente precisa;

#### **4. Limitações na Avaliação de Palavras-passe**

Embora a ferramenta verifique se a palavra-passe é forte com base em certos critérios, ela não consegue prever todos os tipos de ataques;

#### **5. Funcionalidades Iniciais**

O foco inicial da ferramenta está em funções básicas, como verificar a força das palavras-passe, exposição dos e-mails, números de telemóvel e analisar a segurança dos websites. Algumas funcionalidades mais avançadas poderão ser adicionadas no futuro;

#### **6. Privacidade e Dados**

O sistema não guarda nenhum dado dos utilizadores. Isso é bom para a privacidade, mas também significa que não podemos criar um histórico das análises feitas;

#### **7. Limitações na Verificação de Vulnerabilidades Desconhecidas (Zero-Day)**

A ferramenta baseia-se em assinaturas e bases de dados de ameaças já conhecidas. Isto significa que vulnerabilidades novas, que ainda não foram reportadas ou registadas nas bases de dados globais, podem não ser detectadas pelo sistema;

#### **8. Limitações de APIs de Terceiros**

O sistema recorre a serviços externos para validação de dados. Existem limites de requisições (*rate limits*) impostos pelas versões gratuitas destas APIs, o que pode restringir o volume de análises simultâneas.

### **1.10. Design**

#### **1.10.1. Cenário de Sucesso Principal**

##### **a. Objetivos do Projeto**

O nosso projeto tem como objetivo criar uma ferramenta simples e eficaz para ajudar os utilizadores a proteger as suas palavras-passe, e-mails e números de telemóvel, bem como verificar se os websites que visitam são seguros. A plataforma será fácil de utilizar e rápida, permitindo que qualquer pessoa, independentemente dos seus conhecimentos em segurança digital, consiga utilizá-la sem dificuldades. O sucesso será medido pela experiência do utilizador: se a ferramenta for prática e contribuir realmente para a melhoria da segurança online. Além disso, a rapidez na análise de palavras-passe, números de telemóvel, e-mails e sites será uma

prioridade. O objetivo final é que os utilizadores se sintam seguros e confiantes ao usar a ferramenta, protegendo as suas informações de forma eficiente.

### b. Análise das Necessidades dos Utilizadores

Para garantir que a ferramenta atenda às necessidades dos utilizadores, fizemos uma análise com base em pesquisas e *feedback* direto. As principais necessidades identificadas foram:

- **Necessidade de Palavras-passe Seguras e Comprovadas:** Os utilizadores precisam de uma forma simples de garantir que as suas palavras-passe são fortes e seguras, além de verificar se já foram expostas em fugas de dados;
- **Verificação de Sites e Segurança Online:** Os utilizadores querem saber se os sites que visitam são seguros, especialmente com o aumento de ataques cibernéticos e fraudes online;
- **Verificação de E-mails e Números de Telemóvel:** Os utilizadores querem saber se os seus e-mails e números de telemóvel foram comprometidos em fugas de dados;
- **Facilidade de Uso e Acessibilidade:** A ferramenta deve ser simples e intuitiva, permitindo que qualquer pessoa verifique as suas palavras-passe, e-mails, números de telemóvel e websites sem conhecimentos técnicos;
- **Privacidade e Confidencialidade:** Os utilizadores querem garantir que as suas informações não sejam armazenadas ou partilhadas com terceiros;
- **Rapidez e Eficiência:** A ferramenta deve ser rápida, processando as informações sem grandes esperas, para garantir uma experiência de uso sem frustrações.

### c. Métricas de Sucesso

As métricas de sucesso definem como vamos avaliar o impacto e o desempenho da nossa ferramenta. Elas são baseadas nas necessidades dos utilizadores e nos objetivos do projeto. As principais métricas são:

1. **Taxa de Adoção da Ferramenta:** Medir o número de utilizadores que começam a usar a plataforma. O sucesso será maior se mais pessoas adotarem a ferramenta;
2. **Satisfação dos Utilizadores:** A satisfação será avaliada por feedback direto e inquéritos. Se os utilizadores acharem a ferramenta útil e fácil de usar, é um bom sinal;

3. **Tempo de Resposta:** A ferramenta deve ser rápida e fornecer resultados em tempo real, garantindo uma experiência sem frustrações;
4. **Taxa de Precisão:** A precisão das análises será medida pela eficácia em identificar palavras-passe fracas, e-mails e números de telemóvel comprometidos e sites inseguros.

## 1.11. Medidas a Implementar

Neste segmento, serão explicadas as medidas que vão ser implementadas no *EyeWeb Reborn* de maneira que a segurança melhore e seja atualizada.

### 1.11.1. Controles gerais

As medidas a implementar a nível geral são:

- **HTTPS obrigatório:** todo o tráfego do site e do *widget* passa por HTTPS, garantindo comunicação segura;
- **Sanitização de inputs:** todas as mensagens enviadas pelos utilizadores são filtradas para evitar códigos ou comandos maliciosos;
- **Limitação de pedidos (Rate Limiting):** limita o número de mensagens que um utilizador pode enviar num determinado período, para evitar abusos;
- **Criação de contas de utilizador:** implementação de um sistema de autenticação que permite identificar utilizadores, enviar mensagens automáticas para eles sobre atualizações, e reforçar a segurança através de permissões e registo de atividade.

### 1.11.2. Modo Admin

As medidas a implementar no modo administrador são:

- **Autenticação em dois fatores (MFA):** todos os administradores precisam de uma segunda confirmação;
- **Utilizadores com permissões limitadas (RBAC):** só o administrador pode executar ações críticas, outros perfis têm acesso limitado à leitura;

- **Registo de atividades (logs):** todas as ações ficam registradas, com quem fez e quando, de forma segura.

## 1.12. Interface Gráfica e Funcionamento

### Interface Gráfica

Relativamente à interface gráfica e ao aspeto visual do website, o utilizador, ao aceder à plataforma, terá acesso a uma página principal com uma animação de entrada representada por um olho. Após clicar nessa animação, o utilizador é redirecionado para a página inicial do sistema.

Nesta página inicial, o nome *EyeWeb* surge em destaque, de forma centralizada. Na zona inferior, existirão campos específicos onde o utilizador poderá introduzir websites, endereços de e-mail, números de telemóvel e palavras-passe, com o objetivo de proceder à respetiva análise de segurança.

Adicionalmente, estará disponível uma secção denominada “About”, que apresentará informações sobre o projeto e os seus integrantes, recorrendo a interações gráficas animadas, de forma a tornar a navegação mais apelativa e intuitiva.

Com a integração da nova versão, *EyeWeb Reborn*, a página inicial passa a disponibilizar um agente reativo para o utilizador, localizado num painel no canto inferior direito do ecrã. Este agente inicia a interação com a mensagem “Seja muito bem-vindo ao EyeWeb! Como posso ajudar?”. Existirá também um agente reativo destinado ao administrador, que permitirá colocar questões como, por exemplo, se um determinado endereço IP foi comprometido, se uma palavra-passe é segura, entre outras funcionalidades relacionadas com a segurança.

A introdução destes agentes contribui para uma gestão mais eficaz, simples e rápida do sistema, uma vez que permitem alertar para vulnerabilidades detetadas e sugerir ações necessárias para reforçar e proteger o website, bem como fornecer informações ao utilizador sobre o funcionamento da plataforma.

### Funcionamento

#### *Funcionamento da Análise de URLs*

O processo de análise de URLs no EyeWeb Reborn decorre da seguinte forma:

- O utilizador insere um URL na plataforma *EyeWeb Reborn*;

- O pedido é enviado para o servidor backend;
- O sistema verifica se o URL já foi analisado recentemente, recorrendo a um sistema de cache;
- Caso não exista um resultado recente, são realizadas novas verificações de segurança;
- O sistema determina o estado final do URL (seguro, suspeito ou perigoso);
- A IA gera uma explicação clara em português europeu sobre o resultado obtido;
- O resultado final é apresentado ao utilizador de forma simples e compreensível.

#### *Funcionamento da Análise de Palavras-Passe*

O funcionamento deste processo é o seguinte:

- O utilizador introduz uma palavra-passe na plataforma *EyeWeb Reborn*;
- A palavra-passe é tratada de forma segura, não sendo armazenada em texto simples;
- O sistema avalia a robustez da palavra-passe, tendo em conta critérios como comprimento, complexidade, utilização de caracteres especiais, números e letras maiúsculas e minúsculas;
- É verificado se a palavra-passe já foi comprometida em fugas de dados conhecidas;
- Caso a palavra-passe tenha sido exposta, o sistema apresenta uma avaliação detalhada, indicando onde ocorreu a fuga de dados, sempre que essa informação esteja disponível;
- O resultado final indica se a palavra-passe é forte, média ou fraca, acompanhado de recomendações para a sua melhoria, caso necessário.

#### *Funcionamento da Análise de Endereços de E-mail*

O processo decorre da seguinte forma:

- O utilizador introduz um e-mail na plataforma *EyeWeb Reborn*;
- O pedido é enviado para o servidor *backend*, onde é processado de forma segura;
- O sistema verifica se o e-mail foi exposto em fugas de dados conhecidas;
- Caso o e-mail tenha sido vazado, o sistema informa o utilizador e sempre que possível, indica a origem da fuga de dados;

- O resultado é apresentado de forma clara, permitindo ao utilizador tomar medidas, como a alteração de palavras-passe associadas a esse email.

#### *Funcionamento da Análise de Números de Telemóvel*

O funcionamento deste processo é o seguinte:

- O utilizador introduz um número de telemóvel na plataforma *EyeWeb Reborn*;
- O pedido é enviado para o servidor *backend*, onde é processado de forma segura;
- O sistema verifica se o número de telemóvel foi exposto em fugas de dados conhecidas;
- Caso o número tenha sido vazado, o sistema informa o utilizador e sempre que possível, indica a origem da fuga de dados;
- O resultado final é apresentado de forma clara, permitindo ao utilizador tomar medidas preventivas, como maior atenção a tentativas de fraude, chamadas suspeitas ou esquemas de *phishing*.

#### **1.13. Estrutura do Relatório**

Neste relatório são apresentados a introdução e o enquadramento geral do projeto descrevendo o que consiste e de que se trata o projeto e assuntos a considerar, como as necessidades, requisitos, etc. (expostos ao princípio deste relatório).

Serão igualmente abordadas as diferentes fases do desenvolvimento do projeto, incluindo a análise dos vários componentes do sistema, o seu funcionamento, conceção, implementação e os testes realizados ao longo do período de desenvolvimento.

Por fim, o relatório apresenta os resultados obtidos, bem como as conclusões finais e possíveis planos de melhoria e evolução futura do projeto.

#### **1.14. Problema a resolver e Importância do projeto**

Atualmente, muitas pessoas utilizam serviços online sem terem conhecimento de que os seus dados pessoais já podem ter sido expostos em fugas de dados. Estas fugas podem comprometer e-mails, palavras-passe ou números de telemóvel, colocando em risco a segurança digital dos utilizadores.

O projeto *EyeWeb Reborn* surge como resposta a este problema, permitindo que qualquer utilizador verifique, de forma simples e segura, se os seus dados foram comprometidos ou se os sites que visita são seguros, sem colocar em risco a sua privacidade.

### **1.15. Características do EyeWeb Reborn**

O *EyeWeb Reborn* diferencia-se de outras soluções por garantir a privacidade total do utilizador, uma vez que não armazena nem transmite dados pessoais em formato legível. O sistema utiliza técnicas de segurança modernas, semelhantes às usadas por plataformas profissionais, garantindo uma utilização segura, confiável e gratuita.

## 2. Estado da Arte / Soluções Existentes

Antes do desenvolvimento, foram analisadas diversas soluções existentes, de modo a compreender as suas funcionalidades e limitações:

- **HaveIBeenPwned** - verifica se uma palavra-passe ou e-mail foram comprometidos em violações de dados, mas não analisa sites nem oferece resposta automática a incidentes;
- **VirusTotal** - faz análise de ficheiros e URLs, porém não possui agentes inteligentes nem relatórios personalizados;
- **Shodan** - fornece informação sobre dispositivos ligados à Internet, mas exige conhecimentos técnicos elevados;
- **Sucuri SiteCheck** - faz análises de segurança em websites, mas não integra dashboards em tempo real.

O *EyeWeb Reborn* diferencia-se por combinar a análise de websites, a verificação de palavras-passe, endereços de e-mail, números de telemóvel e mecanismos de resposta automatizada, tudo numa plataforma acessível e educativa, com a integração de um agente reativo de apoio ao administrador e ao utilizador. Adicionalmente, comparativamente às soluções analisadas, o *EyeWeb Reborn* apresenta uma abordagem focada na privacidade do utilizador, evitando a partilha direta de dados sensíveis com servidores externos. Esta característica torna o projeto mais seguro e adequado à utilização por utilizadores comuns.

## 3. Desenvolvimento

### 3.1. Planeamento do Projeto

Para o desenvolvimento deste projeto, pretendia-se aplicar melhorias e medidas de proteção ao projeto prévio, *EyeWeb*, além de integrar um Agente Reativo que permita uma comunicação com os administradores e com os utilizadores para a resolução de dúvidas.

O desenvolvimento do projeto seguiu uma abordagem de aprendizagem baseada em projetos (*Project-Based Learning*), complementada pela aplicação de princípios e práticas da metodologia *Agile Scrum*.

Esta combinação permitiu estruturar o trabalho de forma eficiente, promovendo a colaboração entre os membros da nossa equipa e a aplicação prática dos conhecimentos adquiridos ao longo do curso.

As principais etapas do processo foram as seguintes:

#### 1. Planeamento e Definição de Requisitos

Numa fase inicial, foi realizada uma análise detalhada para identificar os objetivos e o âmbito do projeto.

Foram definidos os requisitos funcionais e não funcionais, que serviram de base à criação de uma lista organizada e priorizada com todas as funcionalidades, tarefas e melhorias previstas para o sistema.

#### 2. Desenvolvimento Incremental em Sprints

Em cada início de *sprint* era realizada uma reunião de planeamento (*sprint planning*), onde se selecionavam as tarefas a concluir nesse período. Esta abordagem facilitou a divisão do projeto em partes menores e mais controláveis.

#### 3. Testes e Correção Iterativa de Erros

A garantia de qualidade foi tratada como um processo contínuo. Em cada *sprint*, eram realizados testes às funcionalidades desenvolvidas, de modo a detetar falhas, vulnerabilidades ou inconsistências.

O desenvolvimento do projeto foi realizado tanto durante as aulas como em casa. A equipa dividiu as responsabilidades da seguinte forma:

- **Vanina:** elaboração do início do relatório final, criação do *PowerPoint* e desenvolvimento de um agente reativo de apoio ao utilizador com inteligência artificial;
- **José Samuel:** criação do website, implementação da segurança do site, desenvolvimento da área administrativa, integração do agente reativo de apoio ao utilizador e do agente reativo do administrador no website e aplicação de mecanismos de encriptação;
- **Francisco Ribeiro:** criação dos diagramas UML e elaboração da versão base do *PowerPoint*;
- **Tiago Carvalho:** integração do agente reativo com o website (fase inicial);
- **Ana Rita:** desenvolvimento do agente reativo do administrador, criação inicial da interface do website, gestão do repositório no GitHub, elaboração do relatório final do projeto e finalização do *PowerPoint*.

### 3.2. Análise de Sistemas/SRS

Antes de continuarmos com o nosso projeto, fizemos vários diagramas para perceber bem o que precisávamos construir (disponíveis nos anexos do relatório):

- **Diagramas de Casos de Uso** – utilizados para identificar todas as ações possíveis por parte dos utilizadores e administradores;
- **UML** – foram também elaborados diagramas UML, como o Diagrama de Classes, que auxiliaram na definição da estrutura do sistema, no fluxo de interação entre os componentes e na descrição dos processos internos.

### 3.3. Desenvolvimento do Projeto

O desenvolvimento começou com a configuração do ambiente e a criação do Agente reativo do administrador. Um dos principais desafios foi a integração robusta e segura do Agente Reativo, garantindo que este respondia com precisão e não introduzia vulnerabilidades. Outro desafio foi a introdução das defesas no site.

O desenvolvimento do *EyeWeb Reborn* decorreu em várias fases, garantindo que cada componente fosse implementado e testado de forma estruturada:

- **Criação da interface inicial e dashboard:** foi desenvolvida uma interface intuitiva, clara e responsiva, permitindo ao utilizador aceder facilmente às funcionalidades principais e acompanhar a monitorização em tempo real;

- **Integração das APIs externas:** foram incorporadas diversas *APIs*, incluindo serviços de verificação de vulnerabilidades e autenticação segura, assegurando que o sistema pudesse consultar e processar dados em tempo real;
- **Implementação do painel de administração:** o painel administrativo foi construído para permitir a gestão de utilizadores, configuração de alertas, visualização de relatórios e monitorização centralizada de atividades;
- **Criação dos agentes reativos e lógica de monitorização:** foram desenvolvidos dois agentes: o agente do administrador, que atua em tempo real, analisando websites, credenciais e eventos de segurança e o agente do utilizador, que auxilia este a compreender o funcionamento do website;
- **Testes e ajustes finais:** após a integração de todos os módulos, foram realizados ajustes na interface, correção de erros e otimização de desempenho.

## 1. Recursos de Software

- **Front-end (Interface):**
  1. **Next.js 16 & React:** *Framework* principal para uma interface reativa e moderna;
  2. **TypeScript:** Para garantir a robustez do código e evitar erros em tempo de execução;
  3. **Tailwind CSS:** Para um design responsivo e profissional;
  4. **FingerprintJS:** Biblioteca para identificação única de *hardware* (segurança anti-VPN).
- **Back-end (Servidor e Lógica):**
  1. **Python (FastAPI):** Linguagem e *framework* de alto desempenho para gestão de rotas e processamento de dados;
  2. **PostgreSQL (Supabase):** Base de dados para a gestão de utilizadores e logs;
  3. **PyArrow & Pandas:** Para leitura de grandes volumes de dados no formato *Parquet*.
- **Infraestrutura e Alojamento:**
  1. **Vercel:** Alojamento do *Front-end*;

- 2. Render:** Alojamento do *Back-end*;
  - 3. Hugging Face:** Armazenamento de grandes conjuntos de dados (*datasets*) de fugas de informação.
- **Segurança e Controlo de Acesso:**
    - 1. **MFA Localhost (Python):** Script externo para gerar códigos de 6 dígitos sincronizados com o servidor;
    - 2. **HTTPS/SSL:** Encriptação de ponta a ponta para proteção do tráfego.
  - **Ferramentas de Desenvolvimento:**
    - 1. **Visual Studio Code:** Editor de código principal;
    - 2. **GitHub & GitHub Actions:** Controle de versões e automação de fluxos CI/CD (DevOps).

## 2. Recursos de Hardware

Nos recursos de *hardware* só foram necessários os computadores dos respectivos integrantes da equipa.

## 3. Base de Dados e Backend

- **Sistema:** PostgreSQL (Supabase) e Python (Fast API).
- **Segurança e Acesso:** Encriptação de dados sensíveis, políticas de RLS (*Row Level Security*) para isolamento de dados e autenticação administrativa protegida por MFA numérico de 6 dígitos.
- **Estrutura de tabelas:**
  - profiles → utilizadores e roles;
  - blacklist → bans de IP e Hardware;
  - access\_logs → tráfego;
  - audit\_logs → ações admin;
  - numeros\_checks → histórico de verificações de números de telemóvel;
  - api\_status → saúde do sistema.

## 4. Agentes Reativos (Grog/Llama 3)

Um dos nossos agentes, o agente do administrador, é integrado no painel administrativo e utiliza o modelo *Llama 3* (via Groq) para auxiliar o administrador na análise de logs, geração de relatórios de segurança e interpretação de dados de tráfego. O outro, o agente reativo do utilizador, serve para ajudar este a compreender o funcionamento do website. A comunicação de ambos os agentes é restrita ao contexto da base de dados do projeto, garantindo que não existam fugas de informação sensível para o exterior.

Para proteger a privacidade dos utilizadores, os agentes utilizam internamente o protocolo *K-Anonymity* para validar se dados foram comprometidos, sem expor a informação real em texto limpo. O sistema implementa ainda um mecanismo de verificação de comprometimento baseado em *hashing* criptográfico, no qual os dados sensíveis são transformados localmente através do algoritmo SHA-256, sem nunca serem armazenados ou transmitidos em texto limpo. A validação é realizada através da comparação dos primeiros cinco caracteres hexadecimais do hash com os registos existentes na base de dados, permitindo a deteção de possíveis comprometimentos com exposição mínima de informação.

## 5. Ferramentas de Desenvolvimento e Teste

- **Visual Studio Code:** Ambiente de desenvolvimento principal;
- **GitHub:** Controlo de versões e automação de pipelines CI/CD (DevOps);
- **PowerShell:** Validação de *endpoints* da API e testes de conectividade;
- **Vercel e Render:** Plataformas de alojamento para o Frontend (Next.js) e Backend (FastAPI), respetivamente.

## 6. Segurança e Infraestrutura

- **Comunicação Segura:** Protocolo HTTPS com encriptação SSL/TLS em todos os pedidos;
- **Fortaleza MFA:** Autenticação de dois fatores baseada num gerador de códigos de 6 dígitos numéricos executado em ambiente Localhost (*Python EXE*);
- **Proteção Anti-Ataque no admin:** Bloqueio automático de 72 horas após duas tentativas falhadas de MFA, utilizando *Hardware Fingerprinting* para impedir o uso de VPN para entrar;

- **Monitorização de Quotas:** Sistema de controlo de limites de API para garantir a continuidade do serviço no plano gratuito;
- **Gestão de Sessão no admin:** *Auto-Logout* configurado para 45 minutos de inatividade e *Log Rotation* automático para eliminação de dados com mais de 30 dias.

## 7. Soluções Implementadas

Para superar os desafios técnicos e de segurança identificados, foram desenvolvidas as seguintes soluções:

- **Agentes Reativos Especializados:** Para evitar alucinações ou respostas fora do contexto, os agentes foram configurados com um sistema de “confinamento de conhecimento”. Foi utilizado o modelo *Llama 3* e a inteligência artificial foi programada para atuar exclusivamente nas interações com os utilizadores. O agente do administrador serve como assistente técnico, auxiliando na interpretação de métricas e na deteção de anomalias, sem risco de fornecer informações incorretas ou maliciosas. O agente do utilizador tem como função ajudar este a compreender o funcionamento do website de forma segura e intuitiva;
- **Protocolo de Privacidade K-Anonymity:** Na verificação de segurança foi implementado um protocolo de anonimato onde apenas o prefixo do hash dos dados é enviado para o servidor, isto garante que a informação real do utilizador nunca saia do dispositivo;
- **Arquitetura de Segurança "Zero Trust" para Administração:** Para garantir a integridade do painel de gestão, implementou-se uma camada tripla de proteção:
  - 1) **Isolamento de Credenciais:** Bloqueio de logins sociais (*OAuth*) para administradores;
  - 2) **MFA de 6 Dígitos:** Um sistema de autenticação de dois fatores gerado por um executável local, o que elimina interceptações de rede;
  - 3) **Logging e Auditoria:** Um sistema de monitorização contínua que registra cada ação administrativa.

### 3.4. Resultados

O produto final é uma plataforma web totalmente funcional, acessível por meio de um navegador. A interface do utilizador é simples e intuitiva, com uma área central para inserir URLs, número de telemóvel, e-mails e palavras-passe para análise. Os Agentes Reativos estão integrados num widget no canto inferior direito, pronto a interagir com os administradores e com os utilizadores. O painel de administração, acessível somente com credenciais específicas, permite a gestão de dados de *leaks* e sites maliciosos.

#### Testes e Validação

O projeto foi submetido a várias fases de testes:

- **Testes Unitários:** Para verificar componentes individuais do código. Exemplos:
  - Testes no Site: Backend

#### test\_admin\_router.py

```
1 import pytest
2 from app.routers import admin_router
3
4 def test_generate_totp_returns_digits():
5     code = admin_router.generate_totp("testsecret", digits=6)
6     assert code.isdigit()
7     assert len(code) == 6
8
9 def test_is_admin_email_false():
10    assert not admin_router.is_admin_email("notadmin@example.com")
```

**Explicação:** Testa funções do ficheiro admin\_router.py.

- **test\_generate\_totp\_returns\_digits:** Verifica se a função generate\_totp retorna um código numérico com o número correto de dígitos.
- **test\_is\_admin\_email\_false:** Verifica se a função is\_admin\_email retorna *False* para um email que não é do administrador.

### test\_breach\_router.py

```
1 import pytest
2 from app.routers import breach_router
3
4 def test_breach_check_returns_bool():
5     result = breach_router.check_breach("test@example.com")
6     assert isinstance(result, bool)
```

**Explicação:** Testa funções do ficheiro breach\_router.py.

- **test\_breach\_check\_returns\_bool:** Verifica se a função check\_breach retorna um valor booleano ao verificar um email.

### test\_password\_router.py

```
1 import pytest
2 from app.routers import password_router
3
4 def test_password_strength():
5     score = password_router.calculate_password_strength("StrongPass123!")
6     assert 0 <= score <= 100
```

**Explicação:** Testa funções do ficheiro password\_router.py.

- **test\_password\_strength:** Verifica se a função calculate\_password\_strength retorna um score entre 0 e 100 para uma password.

### test\_traffic\_router.py

```
1 from fastapi.testclient import TestClient
2 from backend.app.main import app # Ajuste o import conforme seu ponto de entrada
3
4 def test_block_ip():
5     client = TestClient(app)
6     payload = {"ip": "123.123.123.123", "reason": "Test reason"}
7     response = client.post("/admin/traffic/block-ip", json=payload)
8     assert response.status_code == 200
9     assert response.json()["success"] is True
10    assert "bloqueado" in response.json()["message"]
```

**Explicação:** O teste simula uma requisição POST para o endpoint /admin/traffic/block-ip, enviando um IP e um motivo para bloqueio. Ele verifica se o status da resposta é 200 (requisição

bem-sucedida), se o campo "success" na resposta é *True* (bloqueio realizado) e se a mensagem de resposta contém a palavra "blockaded" (confirmação do bloqueio). Assim, o teste garante que o endpoint está funcionando corretamente ao bloquear um IP e retornar a resposta esperada.

- **Testes de Integração:** Para garantir que todos os módulos funcionem em conjunto.

#### **test\_integration.py**

```
1  from fastapi.testclient import TestClient
2  from app.main import app
3
4  client = TestClient(app)
5
6  def test_admin_mfa_endpoint():
7      response = client.post("/admin/verify-mfa", json={"email": "admin@site.com", "code": "123456"})
8      assert response.status_code in [200, 401]
9
10 def test_breach_endpoint():
11     response = client.post("/breach/check", json={"email": "test@example.com"})
12     assert response.status_code == 200
```

**Explicação:** Testa endpoints da API FastAPI.

- **test\_admin\_mfa\_endpoint:** Faz um pedido POST ao endpoint /admin/verify-mfa e verifica se o status code é válido (200 ou 401).
- **test\_breach\_endpoint:** Faz um pedido POST ao endpoint /breach/check e verifica se o status code é 200.

- Testes no Agente do Utilizador

#### **apiIntegration.test.js**

```
1  const fetchData = async () => {
2      // Simulação de chamada à API
3      return { status: 200, data: { user: 'Alice' } };
4  };
5
6  test('retorna dados do usuário', async () => {
7      const response = await fetchData();
8      expect(response.status).toBe(200);
9      expect(response.data.user).toBe('Alice');
10 })
```

**Explicação:** Este teste simula uma chamada à API e verifica se os dados retornados estão corretos. A função `fetchData()` simula uma resposta de API com status 200 e dados de usuário e verifica se o status é 200 e se o nome do usuário é 'Alice'.

#### **test\_admin.py**

**Explicação:** Testa endpoints de administração, como verificação de MFA (autenticação multifator), validação de email e código.

#### **test\_breaches.py**

**Explicação:** Testa o endpoint de verificação de fugas de dados (breach checker), usando prefixos de hash para K-Anonymity.

#### **test\_health.py**

**Explicação:** Testa se a API está operacional, verificando o endpoint raiz e o endpoint de saúde.

#### **test\_passwords.py**

**Explicação:** Testa o endpoint de verificação de senhas (password checker) e estatísticas, usando prefixos de hash.

#### **test\_url\_checker.py**

**Explicação:** Testa endpoints de análise de URLs, incluindo validação de URLs legítimas e tratamento de erros.

- **Testes de Usabilidade:** Realizados com um pequeno grupo de utilizadores (família e amigos), que confirmaram a interatividade da interface.
  - Testes no Site: Frontend

### **test\_usability.py**

```
1  from playwright.sync_api import sync_playwright
2
3  def test_homepage_loads():
4      with sync_playwright() as p:
5          browser = p.chromium.launch()
6          page = browser.new_page()
7          page.goto("http://localhost:3000")
8          assert page.title() == "EyeWeb"
9          browser.close()
```

**Explicação:** Usa a biblioteca Playwright para abrir um browser (Chromium). Após, acede à página principal da aplicação em http://localhost:3000. Verifica se o título da página é "EyeWeb" e finalmente, fecha o browser no final. Basicamente, serve para garantir que a homepage carrega corretamente e tem o título esperado.

- Testes no Agente do Utilizador

### **usability.test.js**

```
1  import React from 'react';
2  import { render, screen } from '@testing-library/react';
3
4  function Button() {
5      return <button>Enviar</button>;
6  }
7
8  test('botão é exibido', () => {
9      render(<Button />);
10     expect(screen.getByText('Enviar')).toBeInTheDocument();
11 });
```

**Explicação:** Este teste verifica se um botão é exibido na interface, ou seja, renderiza o botão e verifica se ele aparece na tela.

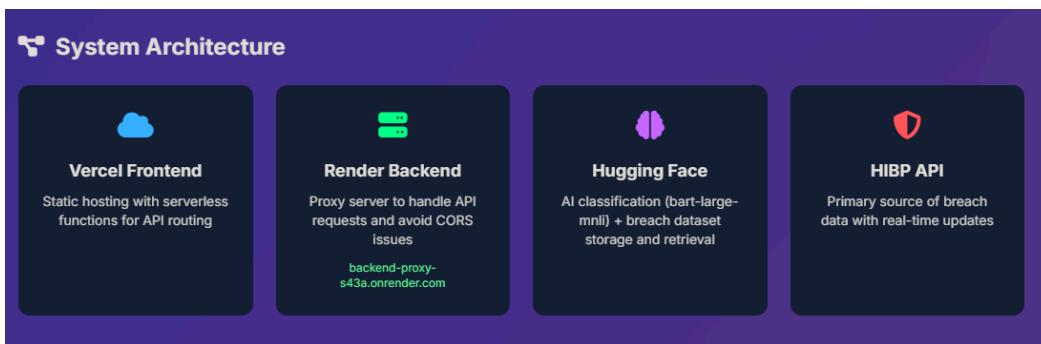
- **Testes de Segurança:** Foram realizados testes de penetração básicos para identificar e corrigir vulnerabilidades comuns, como injection.

- Teste no Site:

- Comando: **bandit -r backend/app/** → Analisa recursivamente o código Python na pasta app à procura de vulnerabilidades de segurança comuns (ex: uso inseguro de funções, má gestão de dados sensíveis, etc.). O

objetivo é detetar problemas de segurança no código fonte antes de ir para produção.

- **Testes de Website:** O website de teste consistiu na integração da *API* do *HIBP*, combinando o alojamento de cada componente em serviços distintos: o Vercel para o *frontend*, o Render para o *backend* e o *Hugging Face* para a classificação e armazenamento do conjunto de dados de fontes de notícias não confirmadas. Estas fontes são pesquisadas em tempo real durante a verificação dos URLs e utilizadas para fornecer uma resposta mais completa, para além das informações sobre violações de dados disponibilizadas pelo *HIBP*.



### Componentes:

Componente	Tecnologia	URL
Frontend	HTML + Tailwind CSS + JavaScript	<a href="https://api-proxy-render.vercel.app">https://api-proxy-render.vercel.app</a>
Backend	Node.js + Express.js (v1.4.0)	<a href="https://backend-proxy-s43a.onrender.com">https://backend-proxy-s43a.onrender.com</a>
API	Have I Been Pwned API v3	<a href="https://haveibeenpwned.com/API/v3">https://haveibeenpwned.com/API/v3</a>

Dataset	Hugging Face (Inference + Hub)	<a href="https://huggingface.co/datasets/Tiago2024180/eyewebdataset">https://huggingface.co/datasets/Tiago2024180/eyewebdataset</a>
---------	--------------------------------	---

## Verificação de Email (HIBP)

- O utilizador insere um email;
- O backend faz proxy do pedido à API HIBP (com API key segura);
- Retorna lista de *breaches* em que o email aparece.

The screenshot shows the 'Breach Results' page from Have I Been Pwned (HIBP). At the top, there's a red circular icon with a white exclamation mark. Below it, the heading 'Vazamentos Encontrados!' is displayed. A message states: 'O email test@example.com foi encontrado em 240 vazamento(s.)'. Below this, two breach entries are listed: 'Adobe' (152 445 165 contas) and 'Stratfor' (859 777 contas). Each entry includes the domain, date, and a snippet of the breach description.

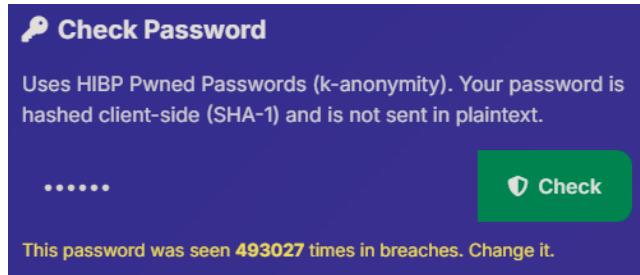
## Verificação de Domínio/URL

- O utilizador insere um domínio (ex: adobe.com, linkedin.com);
- O backend consulta a API HIBP para *breaches* associados ao domínio.

The screenshot shows the 'Check Domain / URL' page from HIBP. It asks the user to check if a domain or URL appears in known breaches. The input field contains 'linkedin.com' and the 'Check' button is highlighted in blue. Below the input fields, it says 'Found 3 breach(es) in HIBP:' followed by a list: LinkedIn (2012-05-05), LinkedInScrape2023 (2023-11-04), and LinkedInScrape (2021-04-08).

## Verificação de Password

- O utilizador insere uma palavra-passe;
- É gerado o *hash SHA-1* no *browser*;
- Utiliza *k-anonymity*: envia apenas os 5 primeiros caracteres do *hash*;
- A API *HIBP* retorna todos os *hashes* que começam assim e o *frontend* compara localmente.



## Dataset no Hugging Face

- Cada pesquisa de domínio que encontra *breaches* grava automaticamente os dados no *Hugging Face*;
- Ficheiro *search\_history.jsonl* com esquema *flat* (1 linha por breach);
- Ficheiros individuais em *.autochecks/{domain}.json*.

## Dataset Explorer (Frontend)

- Tabela interativa no site que carrega o *JSONL* diretamente do *Hugging Face*;
- Mostra: *Domain, Breach, Date, Accounts, Data Exposed, Checked At*;
- Atualiza automaticamente o *dataset* quando novos domínios são pesquisados.

Dataset Explorer — Hugging Face Live Data						
Real-time view of all breach searches recorded in the Hugging Face Dataset. Updated automatically every time a domain is checked.						
Load Dataset		12 records				
Domain	Breach	Date	Accounts	Data Exposed		Checked At
linkedin.com	LinkedIn	2012-05-05	164.6M	Email addresses, Passwords		09/02/2026, 00:30:23
linkedin.com	LinkedIn Scraped and Faked Data (2023)	2023-11-04	19.8M	Email addresses, Genders, Geographic locations...		09/02/2026, 00:30:23
linkedin.com	LinkedIn Scraped Data (2021)	2021-04-08	125.7M	Education levels, Email addresses, Genders, Geo...		09/02/2026, 00:30:23
instagram.com	Instagram	2026-01-07	6.2M	Display names, Email addresses, Geographic loc...		09/02/2026, 00:16:50

## Vercel e Render (front e backend)

The screenshot shows the Vercel interface for the 'api-proxy-render' project. At the top, there are tabs for 'Repository', 'Usage', 'Domains', and 'Visit'. Below that, under 'Production Deployment', there are buttons for 'Build Logs', 'Runtime Logs', and 'Instant Rollback'. The main area displays a preview of the application's front-end, which is a 'Data Breach Monitoring System'. To the right of the preview, deployment details are shown: 'Deployment' (api-proxy-render-64k16jb3l-tiago2024180s-projects.vercelapp), 'Domains' (api-proxy-render.vercel.app), 'Status' (Created, Ready 39m ago by Tiago0612), and 'Source' (main branch, commit 9de492b). Below this, a note says 'Your app is on a manual proxy-activated deployment'. At the bottom, there are links for 'Deployment Settings' and '4 Recommendations'.

The screenshot shows the Vercel interface for the 'backend-proxy' service. It indicates the service is 'Free' and shows a 'Live' status. A note says 'Your free instance will spin down with inactivity, which can delay requests by 50 seconds or more.' with a link to 'Upgrade now'. Below this, a log entry for 'February 9, 2026 at 12:29 AM' shows a successful deployment of commit '9de492b: Add Dataset Explorer table to frontend'. The main part of the screen is a log viewer with a search bar and a 'Live tail' button. The log output shows several entries from the 'wrsx4' worker process, including requests to LinkedIn and HIBP, and internal service detection messages.

## Variáveis de ambiente usadas

- Vercel: BACKEND\_URL (url do render);
- Render: FRONTEND\_URL (url do vercel), HF\_TOKEN (token hugging face), HIBP\_API\_KEY (api do HIBP);
- Github (repositório): HF\_REPO (hugging face), HF\_TOKEN (token hugging face), HIBP\_API\_KEY (api do HIBP), VERCEL\_ORG\_ID (user ID do vercel), VERCEL\_PROJECT\_ID (ID do projeto do vercel), VERCEL\_TOKEN (Token Vercel).

## Segurança Implementada

- *API keys* nunca expostas no frontend — *HIBP key* e *HF token* são variáveis de ambiente no *Render*;
- *k-Anonymity* para palavras-passe — só 5 chars do *SHA-1* enviados, comparação feita localmente;
- *Rate limiting* — 60 requests/minuto por IP;
- *CORS* configurado — só aceita pedidos de origens autorizadas (*Vercel/Render*);
- *Proxy pattern* — o *frontend* nunca comunica diretamente com *APIs* externas;
- *Trust proxy* — headers *X-Forwarded-For* para IP real atrás de *Cloudflare/Vercel*.

### Logs e Evidências de Teste e Métricas

#### Fontes de Log

- Logging Python (stdout): Implementado em *backend/app/main.py*, *routers* e *services* via *logging.basicConfig()*, regista *requests*, respostas, exceções e eventos diretamente para *stdout* (sem ficheiro de log separado);
- Configuração de logging: *backend/app/main.py*, *updater/updater.py*, *updater/password\_updater.py*.

#### Tabelas e Métricas (Supabase PostgreSQL)

- *traffic\_logs*: Regista cada pedido (IP, method, path, status\_code, user\_agent, country, city, is\_vpn, vpn\_provider, fingerprint\_hash, response\_time\_ms, created\_at). Middleware em *backend/app/main.py*;
- *traffic\_suspicious*: Eventos de ameaça detetados automaticamente (rate\_limit, scanner, sql\_injection, path\_traversal, brute\_force, recon\_probe, suspicious\_ua) com IP, severity, details, path, country, city, is\_vpn, fingerprint\_hash;
- *traffic\_blocked\_ips*: IPs bloqueados manualmente ou por associação a dispositivos bloqueados;
- *traffic\_blocked\_devices*: Dispositivos bloqueados por fingerprint\_hash, com associated\_ips, components e reason;
- *traffic\_device\_ips*: Mapa fingerprint\_hash - IPs (com is\_vpn, country, city, last\_seen\_at) - Chave primária composta (fingerprint\_hash, ip);

- `traffic_device_fingerprints`: Componentes do *fingerprint* para *fuzzy matching* contra dispositivos bloqueados;
- `traffic_vpn_cache`: Cache de *lookup VPN* por IP;
- Conexões ativas: *Tracked* em memória no *TrafficService* via *heartbeats* (sem tabela dedicada);
- Visitas: Registadas em `traffic_logs` com `method=PAGE`, identificação por `fingerprint_hash` (não há tabela separada nem campo `device_id`).

## Locais no Código

- backend/app/main.py: *Middleware* de *logging* (`traffic_middleware`), *handler* global de exceções (`global_exception_handler`), inclusão de *routers*;
- backend/app/routers/ (diretório): `admin_router.py`, `auth_router.py`, `breach_router.py`, `chat_router.py`, `password_router.py`, `traffic_router.py`, `url_router.py`, `user_chat_router.py` - endpoints de stats, health-check, breaches, passwords, admin, tráfego, chat;
- backend/app/services/ (diretório): `auth_service.py`, `breach_service.py`, `password_service.py`, `traffic_service.py`, `url_service.py` - lógica de verificação, deteção de ameaças, gestão de bloqueios e fingerprints;
- backend/app/models.py: Schemas Pydantic de request/response (`BreachCheckRequest`, `BreachInfo`, `BreachCheckResponse`, `HealthResponse`, `StatsResponse`, `ErrorResponse`);
- updater/updater.py, password\_updater.py: *Logging*, geração de *datasets*, *hashes*, métricas.

## Dashboard (Painel Administrativo)

- page.tsx: Página principal do painel, navegação, autenticação, *MFA*, perfil, seções menu, e-mails, *health*, *traffic* e *chat*;
- page.tsx: Página de saúde do sistema, exibe *status* geral, categorias, serviços, métricas de online/offline/degraded/unknown, dados da API `/api/admin/health-check`;
- page.tsx: Monitor de tráfego em tempo real, logs detalhados, deteção de ameaças, bloqueio de dispositivos por *fingerprint*, *VPN toggle detection*;
- page.tsx: Gestão de conversas do *chat*;

- page.tsx: Gestão de emails.

### Evidências de Teste

- Logs de acesso em traffic\_logs, deteção de ameaças em traffic\_suspicious, identificação por fingerprint\_hash, heartbeat a cada 20s, relatórios, verificação de email/senha/URL;
- Exemplo: O IP 192.0.2.51 acedeu ao site às 14:02:11. O middleware registou o pedido em traffic\_logs e o TrafficService executou deteção de ameaças;
- SQL: SELECT \* FROM traffic\_logs WHERE ip = 192.0.2.51.

### Resultados

- Primeiro acesso (desktop): OK;
- Heartbeat 20s, last\_seen avança ciclicamente (Online/Offline): OK;
- Gerar relatório: OK;
- Verificar email/senha/URL no agente: OK.

### Locais no Código (Serviços)\*

- backend/app/models.py: Define schemas Pydantic (BreachCheckRequest, HealthResponse, StatsResponse, etc.);
- backend/app/services/url\_service.py: Manipula status de URLs, determina status final, salva e recupera status do cache, registra logs;
- traffic\_service.py: Deteção de ameaças (7 tipos), gestão de fingerprints, bloqueio de dispositivos/IPs, heartbeats, VPN detection;
- backend/app/routers/ (diretório): Manipula requisições, autenticação, erros e logs.

### Dashboard (Painel Administrativo)

- page.tsx: Página principal do painel, navegação, autenticação, MFA, perfil, seções menu, e-mails, health, traffic e chat;
- page.tsx: Página de saúde do sistema, exibe status geral, categorias, serviços, métricas de online/offline/degraded/unknown, dados da API /api/admin/health-check.

### Evidências de Teste

- Logs de acesso em `traffic_logs`, `status` de URLs, identificação por `fingerprint_hash`, `heartbeat`, relatórios e verificação de email/senha/URL.

## Correções e Melhorias

- Relatório expandido - ZIP substituído por TXT único;
- Stack tecnológica: *Backend* em *Python* (*FastAPI*), *Frontend* em *TypeScript* (*Next.js*). Sem PHP.

## Resumo dos Locais

- Logs: `traffic_service.py`, `backend/app/routers/` (diretório), `stdout` via `logging.basicConfig()`;
- Tabelas Supabase: `traffic_logs`, `traffic_suspicious`, `traffic_blocked_ips`, `traffic_blocked_devices`, `traffic_device_ips`, `traffic_device_fingerprints`, `traffic_vpn_cache`;
- Dashboard: `page.tsx`;
- Evidências: Logs em `traffic_logs`, respostas de API, relatórios TXT.

Os resultados dos testes foram positivos, com a plataforma a demonstrar estabilidade, segurança e a cumprir os requisitos de desempenho definidos.

O *EyeWeb Reborn* cumpriu os objetivos propostos, melhorando significativamente a versão original do projeto. Conseguimos aumentar a segurança e o desempenho da plataforma, reduzindo vulnerabilidades críticas que existiam anteriormente. A nova arquitetura e os agentes reativos tornaram o sistema mais robusto e eficiente. O sistema é agora capaz de detetar ameaças de forma muito mais rápida, analisando websites, números de telemóvel, e-mails e palavras-passe, permitindo uma resposta mais eficaz.

A introdução de mecanismos de bloqueio automático constituiu um avanço importante. Estes mecanismos atuam de forma autónoma perante atividades suspeitas, reduzindo drasticamente as falhas de segurança e proporcionando maior tranquilidade aos administradores.

Adicionalmente, dedicamos especial atenção à interface, tornando-a clara e fácil de utilizar. O objetivo foi garantir que qualquer pessoa, mesmo sem conhecimentos técnicos, consiga utilizar a ferramenta sem dificuldades.

Em suma, a nossa análise demonstra que o *EyeWeb Reborn* é uma plataforma viável, funcional e capaz de cumprir a sua missão de melhorar a segurança online de forma prática e eficiente.

## 4. Arquitetura

A arquitetura do EyeWeb Reborn é composta por quatro camadas principais:

### 1. Frontend (Interface Gráfica)

A interface do utilizador foi desenvolvida em *Next.js* (Framework para desenvolvimento da interface web), *TypeScript* (Linguagem com tipagem estática), *Tailwind CSS* (Estilização da interface) e *Vercel* (Plataforma de alojamento do frontend). O nosso site tem um painel central onde os utilizadores podem inserir URLs, e-mails, números de telemóvel e palavras-passe para análise. O painel mostra alertas, estatísticas e recomendações de segurança.

### 2. Backend (Dataset / Servidor / API)

O *backend* foi implementado em *Python* (Linguagem principal), *FastAPI* (Framework para criação da API), *Uvicorn* (Servidor da aplicação) e *Render* (Plataforma de alojamento do *backend*). O *backend* deve comunicar com a *Have I Been Pwned API* e gerir a lógica dos Agentes Reativos. Já os *datasets* guardam os dados de palavras-passe, e-mails e números de telemóvel com encriptação *Sha-256*. O *Huggyface* armazena os *datasets* e o acesso é restrito para não haver ataques.

### 3. Base de Dados e cache

A nossa base de dados principal é *Supabase (PostgreSQL)* que é utilizado para guardar resultados das verificações de links.

### 4. Agentes Inteligentes Reativos

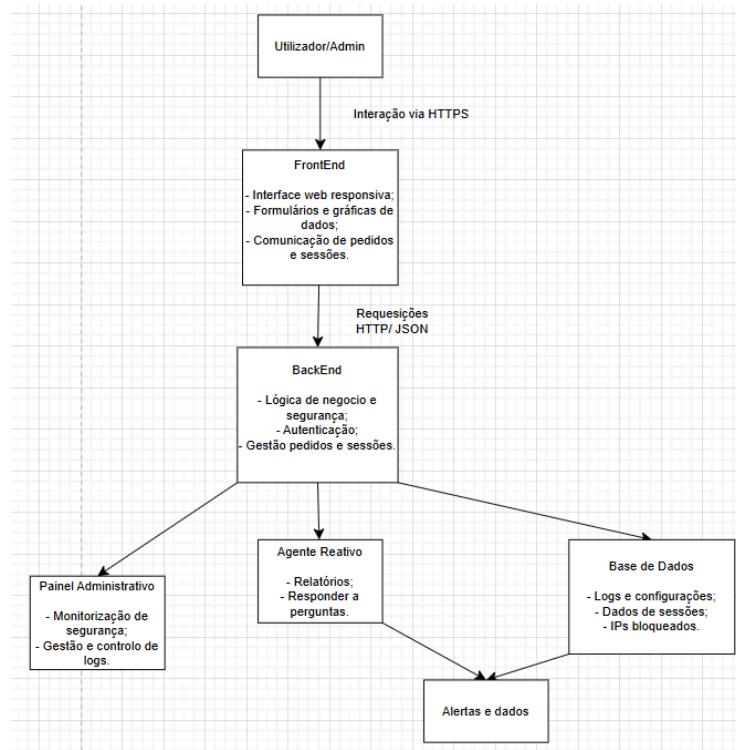
Os agentes são um módulo integrado ao *frontend*, responsável por responder às perguntas/dúvidas do administrador em tempo real (listar IPs bloqueados, gerar relatórios, dizer se o site, e-mail, número de telemóvel são comprometidos, etc.) e do utilizador (auxilia este a compreender o funcionamento do website) . Atuam apenas sobre dados do *EyeWeb Reborn*, o que previne respostas incorretas ou execução de ações indevidas.

#### Comunicação e Segurança

- Toda a comunicação entre *frontend*, *backend*, *dataset*, agentes e base de dados é realizada via HTTPS;

- O painel administrativo está isolado e acessível apenas através de autenticação multifator (MFA);
- O agente utiliza canais seguros e limitados, garantindo que não haja fuga de dados ou execução de comandos não autorizados.

### Diagrama Textual de Alto Nível



### Padrão da Arquitetura – MVC em Camadas

Camada	Função	Componentes
<i>View</i>	Interface com o utilizador.	<i>HTML, CSS e JavaScript</i>
<i>Controller</i>	Gestão de fluxos, autenticação e segurança.	<i>Python, Typescript e Postgresql</i>
<i>Model</i>	Estrutura e armazenamento da informação.	<i>Dataset e Postgresql</i>
Agentes	Análise automática e deteção de ameaças.	Agentes Reativos integrados em <i>Render</i>

### Benefícios desta Arquitetura:

- **Escalabilidade:** permite integrar novos módulos, como aplicação móvel ou novas análises de segurança;
- **Manutenção:** isolamento de erros e atualizações independentes entre *frontend*, *backend* e agentes;
- **Segurança:** controlo de comunicações, permissões e monitorização dos agentes reativos.

### Segurança da Arquitetura:

- Toda a comunicação entre *frontend*, *backend*, agentes e base de dados é via HTTPS;
- O painel administrativo requer autenticação multifator (*MFA*);
- Os logs são armazenados de forma segura;

- Os agentes operam com acesso restrito e tokens temporários, assim só interagem com dados do *EyeWeb Reborn*;
- As chaves das APIs são protegidas através de variáveis de ambiente;
- Os dados sensíveis não são armazenados;
- Os URLs são validados antes de serem processados.

#### APIs externas:

- **Google Safe Browsing:** esta API permite verificar se um URL está associado a *malware*, *phishing* ou *software* malicioso. Caso sejam detetadas ameaças, o URL é classificado como perigoso;
- **URLScan.io:** permite realizar uma análise detalhada de websites, incluindo reputação, categorias e possíveis comportamentos suspeitos;
- **Groq (Llama 3):** utilizada para gerar uma explicação em português, clara e simples, sobre a segurança do URL analisado, com base nos resultados obtidos;
- **Have I Been Pwned:** utilizada para verificar se palavras-pases ou endereços de e-mail foram comprometidos em fugas de informação conhecidas, sem expor os dados reais do utilizador.

#### Sistema de cache:

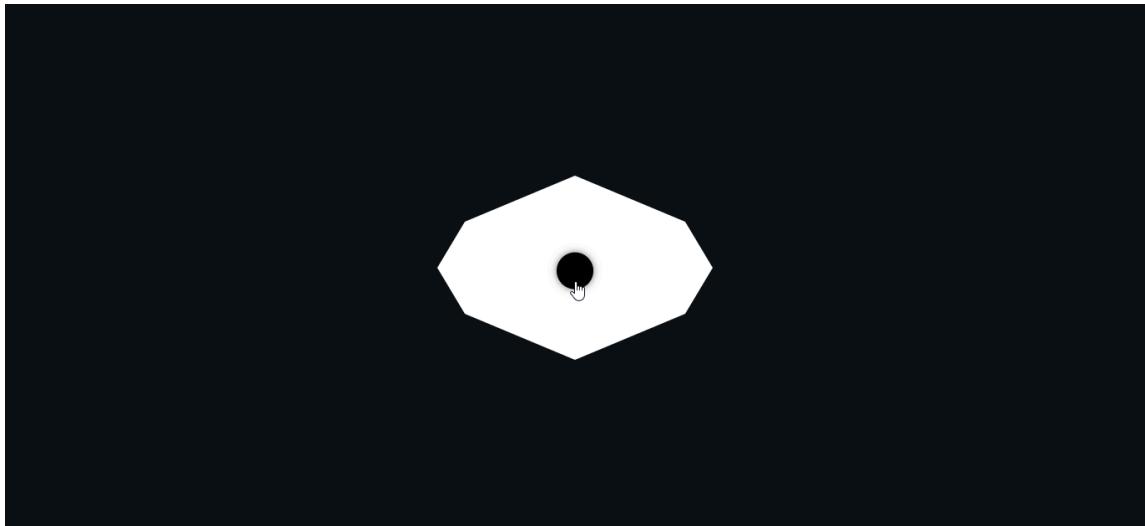
Para melhorar o desempenho e reduzir chamadas excessivas às APIs externas, foi implementado um sistema de cache inteligente:

- URLs analisados recentemente são devolvidos de forma imediata;
- URLs mais antigos são reavaliados automaticamente;
- O sistema continua funcional mesmo em caso de falha temporária de serviços externos.

Este método permite um equilíbrio entre rapidez e atualização da informação.

## 5. Wireframes / Mockups

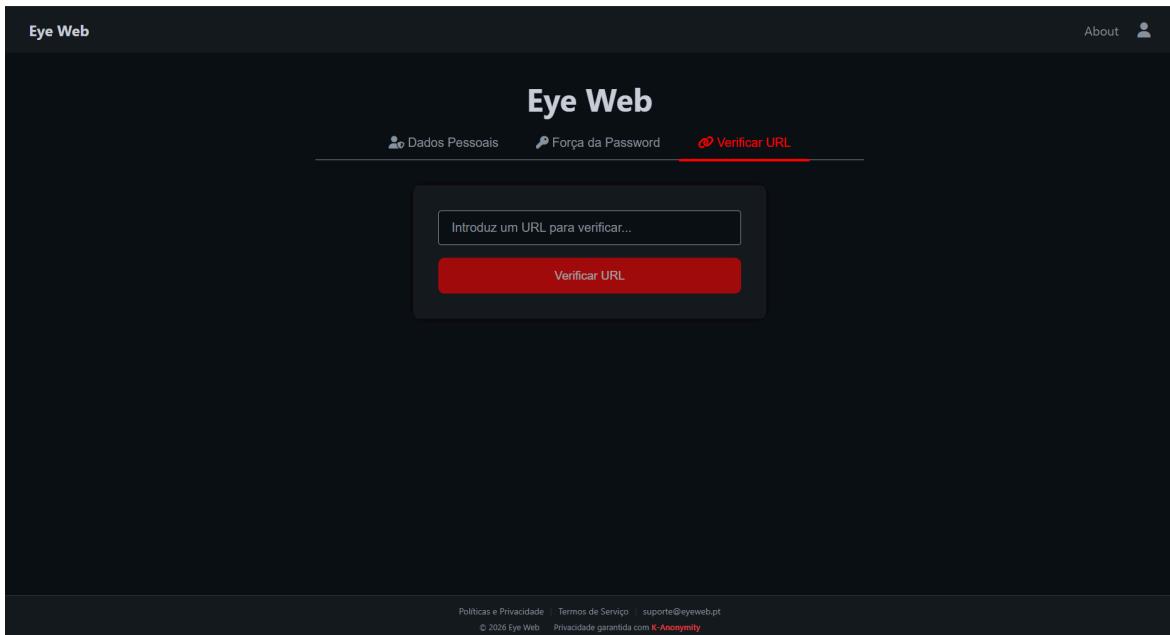
### Entrada do site



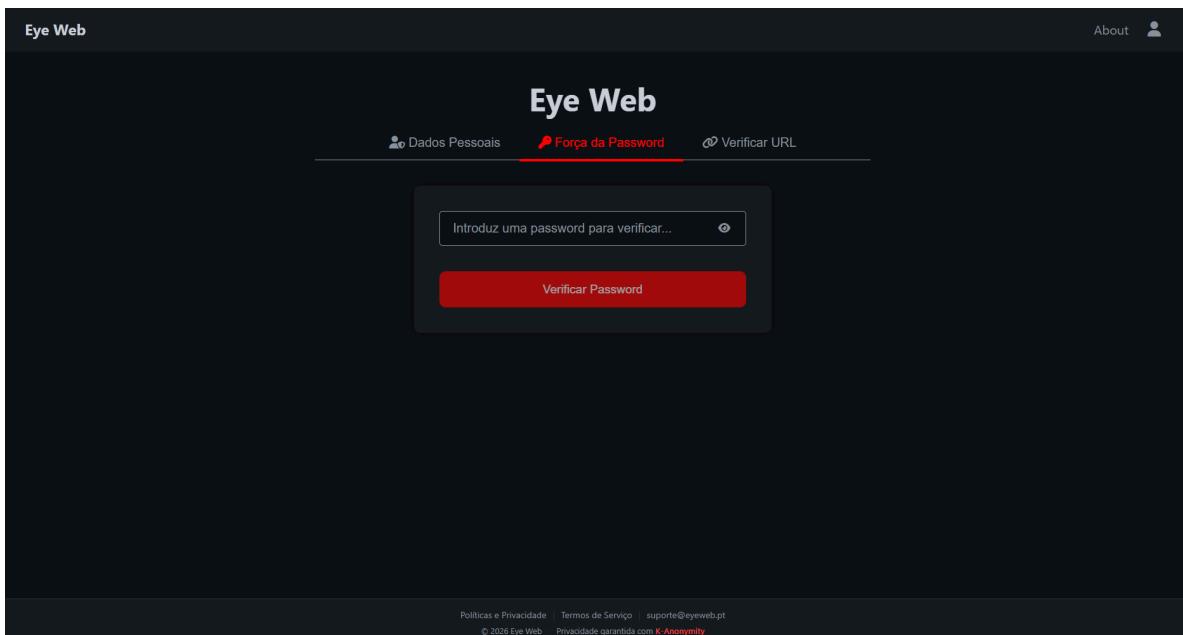
**Personal Data:** Verificação de Emails e números de telemóvel.

A screenshot of the Eye Web website. The header includes the logo "Eye Web", a navigation bar with "About" and a user icon, and three menu items: "Dados Pessoais" (highlighted with a red underline), "Força da Password", and "Verificar URL". Below the header is a search bar with two buttons: "Email" (highlighted with a red border) and "Telefone". A modal window is open, containing an input field with placeholder text "Introduz o teu email..." and a red "Verificar Email" button. At the bottom of the page, there is a footer with links to "Políticas e Privacidade", "Termos de Serviço", and "suporte@eyeweb.pt", along with a note about "Privacidade garantida com K-Anonymity".

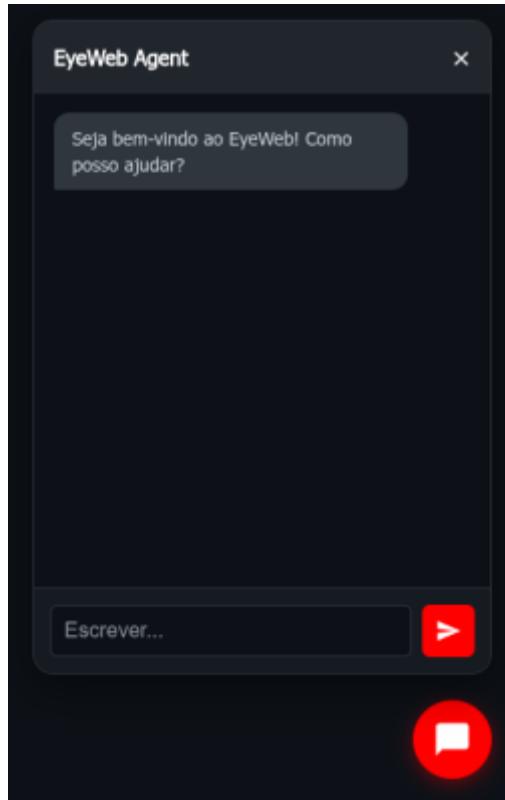
**URL Check:** Verificação de URLs.



**Password Check:** Verificação de Palavras passe.



### Utilizador - Agente



### Sobre Nós

Português

A screenshot of the EyeWeb Reborn homepage. The title "EyeWeb Reborn" is at the top center in a large white font. Below it is a red "Switch to English" button. At the bottom, there is a navigation bar with links: "Motivação: Porquê o EyeWeb Reborn?", "Sobre a Plataforma", "Equipa", and "Infraestrutura e Segurança".

## Motivação

The screenshot shows the 'Motivação' (Motivation) section of the website. At the top, there is a navigation bar with links: 'Motivação: Porquê o EyeWeb Reborn?', 'Sobre a Plataforma', 'Equipa', and 'Infraestrutura e Segurança'. Below the navigation, the title 'Motivação: Porquê o EyeWeb Reborn?' is displayed in bold red text. A large block of text explains the purpose of the platform: "O EyeWeb Reborn surgiu da necessidade de criar ferramentas automáticas e simplificadas para a proteção de dados online. O nosso propósito é democratizar o acesso à cibersegurança, prevenindo ataques através da utilização de agentes reativos inteligentes que auxiliam tanto utilizadores comuns como administradores de sistemas. Acreditamos que a segurança digital deve ser uma norma e não um obstáculo técnico, permitindo que a gestão de dados seja robusta, eficiente e acessível a todos."

## Sobre a Plataforma

The screenshot shows the 'Sobre a Plataforma' (About the Platform) section. The title 'Sobre a Plataforma' is in bold red text. A paragraph of text describes the platform: "O EyeWeb Reborn é uma plataforma inovadora dedicada à análise e proteção de ativos digitais. O sistema permite identificar ameaças e vulnerabilidades de forma intuitiva, fornecendo informações detalhadas sempre que um dado pessoal ou credencial é identificado como comprometido." Below this, a section titled 'Funcionalidades Principais' (Main Features) lists several key features with bullet points:

- Análise de Websites: Verificação de certificados de segurança, histórico de ameaças e identificação de ligações suspeitas.
- Auditoria de Credenciais: Validação de palavras-passe e e-mails através de bases de dados de fugas de dados, garantindo a integridade do utilizador.
- Verificação de Dados: Ferramentas de análise para números de telemóvel e outros identificadores digitais.
- Agentes Reativos: Sistemas inteligentes usados para minimizar riscos e sugerir medidas corretivas imediatas.

## Equipa

The screenshot shows the 'Equipa' (Team) section. The title 'Equipa' is in bold red text. Below it, a list of team members and their GitHub handles is provided:

- Ana Rita da Silva Monteiro — [Galaxiay11](#)
- José Samuel da Rocha Oliveira — [Sam-Ciber-Dev](#)
- Tiago Filipe Sousa Carvalho — [Tiago0612](#)
- Vanina Kollen — [vankol06](#)
- Francisco Rafael Carocinho Ribeiro — [Xico20230](#)

### *Infraestrutura e Segurança*

#### **Infraestrutura e Segurança**

A nossa infraestrutura foi desenhada sob princípios rigorosos de privacidade e defesa para garantir que o utilizador está sempre protegido.

- Criptografia: Utilização de algoritmos SHA-256 para garantir que nenhum dado é processado ou armazenado em texto limpo.
- Anonimato: Implementação de K-Anonymity, permitindo a verificação de segurança sem que as credenciais completas sejam transmitidas.
- Segurança no Transporte: Encriptação HTTPS obrigatória em todo o tráfego de dados.
- Defesa de Infraestrutura: Camadas de sanitização de inputs e mecanismos de rate limiting no backend para prevenir abusos e ataques de injection.

### Inglês

## **EyeWeb Reborn**

[Mudar para Português](#)

[Motivation: Why EyeWeb Reborn?](#)   [About the Platform](#)   [Team](#)   [Infrastructure and Security](#)

### *Motivation*

[Motivation: Why EyeWeb Reborn?](#)   [About the Platform](#)   [Team](#)   [Infrastructure and Security](#)

#### **Motivation: Why EyeWeb Reborn?**

EyeWeb Reborn arose from the need to create automatic and simplified tools for online data protection. Our purpose is to democratize access to cybersecurity, preventing attacks through the use of intelligent reactive agents that help both regular users and system administrators.

We believe that digital security should be a standard, not a technical obstacle, allowing data management to be robust, efficient, and accessible to everyone.

### *About the Platform*

**About the Platform**

EyeWeb Reborn is an innovative platform dedicated to the analysis and protection of digital assets. The system allows intuitive identification of threats and vulnerabilities, providing detailed information whenever personal data or credentials are identified as compromised.

**Main Features**

- Website Analysis: Security certificate verification, threat history, and identification of suspicious links.
- Credential Audit: Validation of passwords and emails through data breach databases, ensuring user integrity.
- Data Verification: Analysis tools for phone numbers and other digital identifiers.
- Reactive Agents: Intelligent systems used to minimize risks and suggest immediate corrective measures.

### *Team*

**Team**

Ana Rita da Silva Monteiro — [Galaxiay11](#)  
José Samuel da Rocha Oliveira — [Sam-Ciber-Dev](#)  
Tiago Filipe Sousa Carvalho — [Tiago0612](#)  
Vanina Kollen — [vankol06](#)  
Francisco Rafael Carocinho Ribeiro — [Xico20230](#)

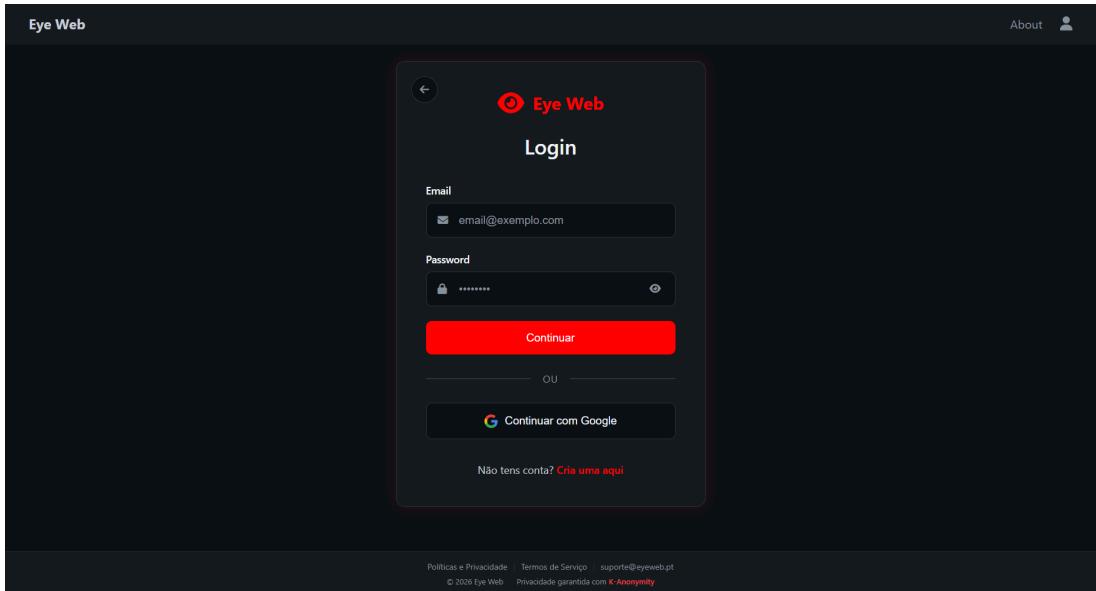
### *Infrastructure and Security*

**Infrastructure and Security**

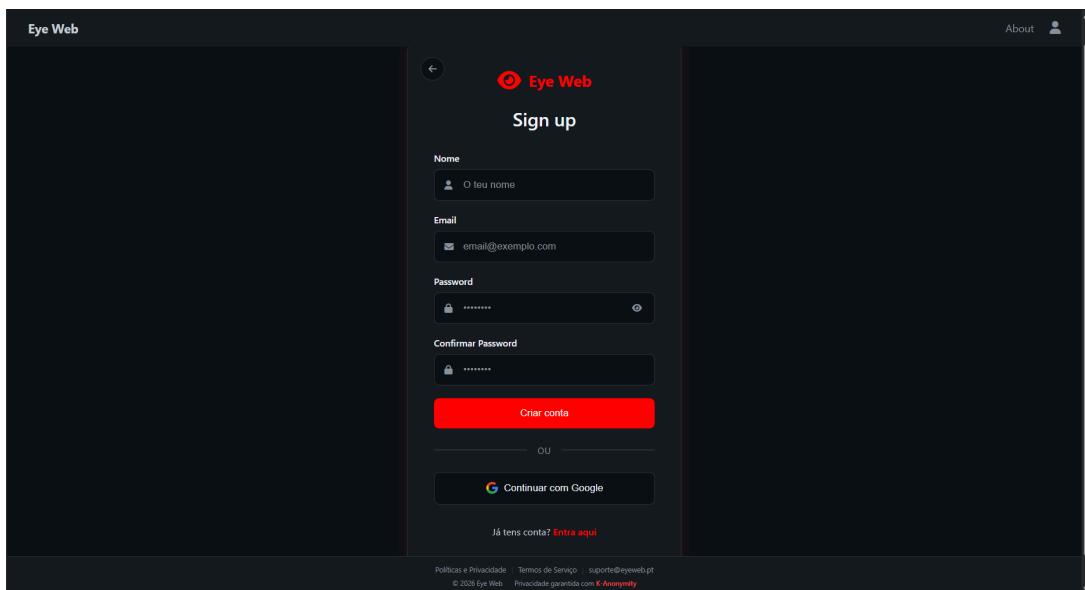
Our infrastructure was designed under strict privacy and defense principles to ensure the user is always protected.

- Encryption: Use of SHA-256 algorithms to ensure no data is processed or stored in plain text.
- Anonymity: Implementation of K-Anonymity, allowing security verification without transmitting complete credentials.
- Transport Security: Mandatory HTTPS encryption for all data traffic.
- Infrastructure Defense: Layers of input sanitization and backend rate limiting mechanisms to prevent abuse and injection attacks.

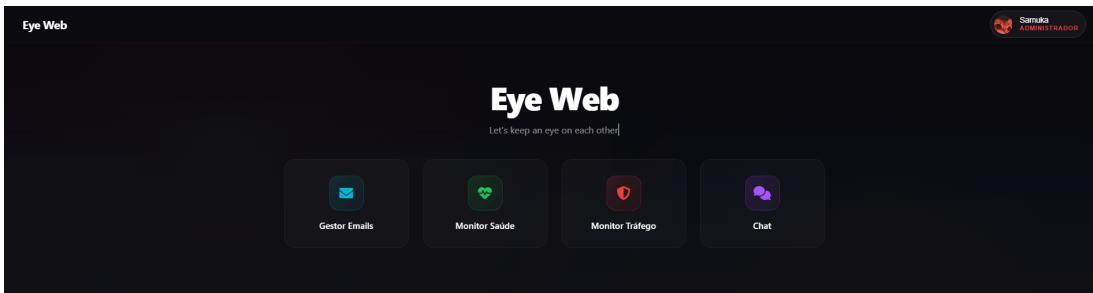
## Painel Login



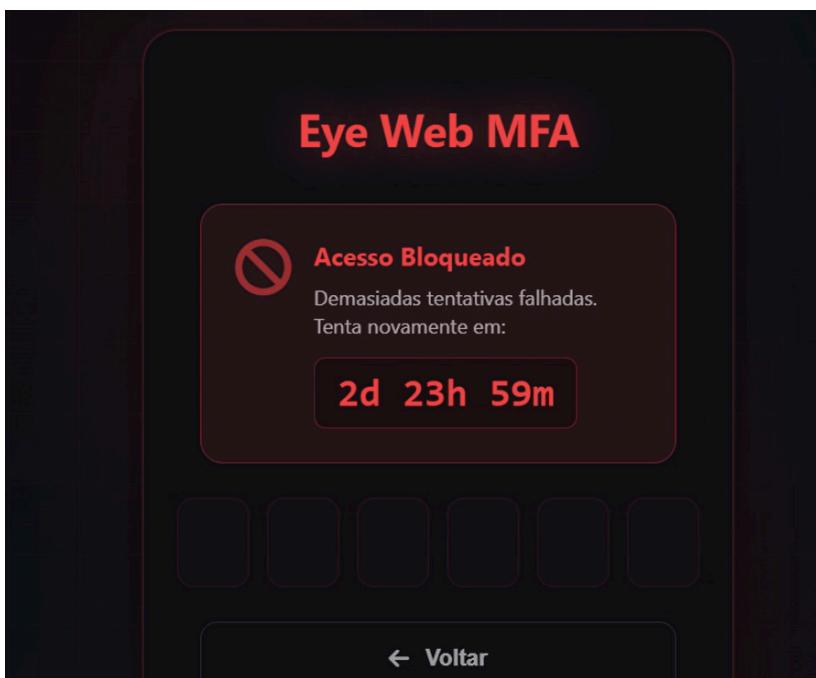
## Painel Sign up



**Painel Admin:**

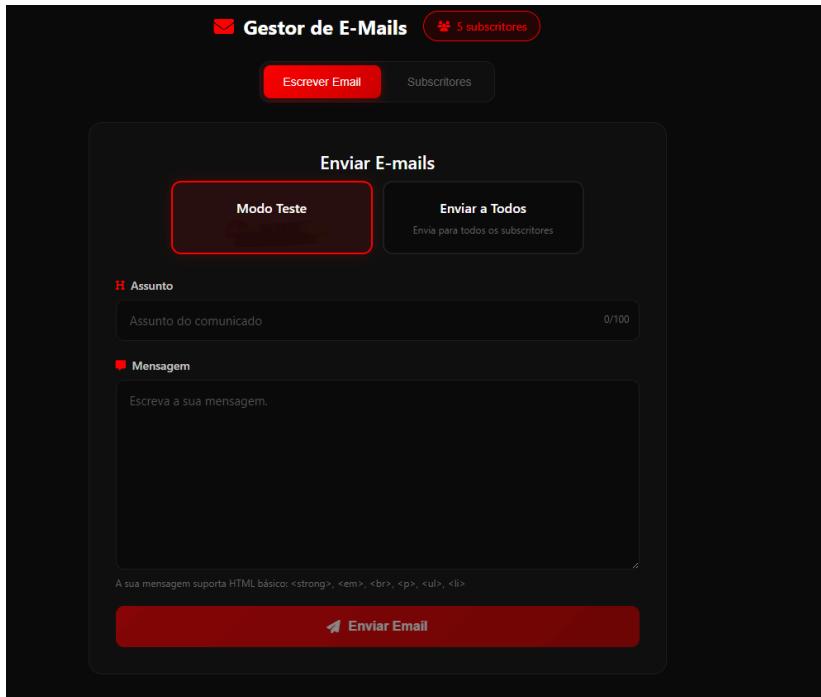


*EyeWeb MFA*

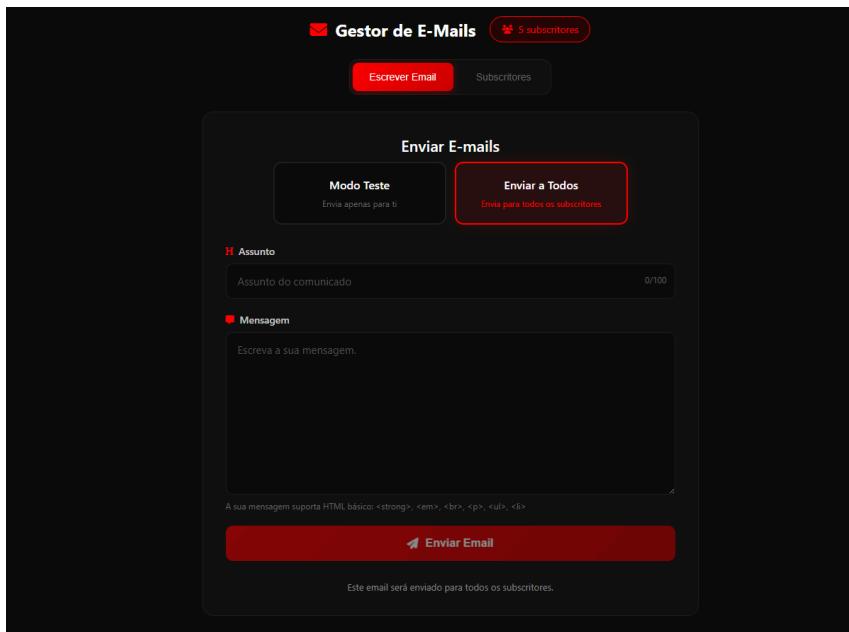


### Gestor de E-mails

#### Modo Teste - O utilizador envia somente para o administrador



#### Enviar para todos - O utilizador envia para todos os subscriptores



## Monitor de Saúde

**Screenshot 1: Backend Services**

Backend API: API a funcionar normalmente  
version: 1.0.0 environment: production

Supabase:

- Supabase - Conexão: Conexão estabelecida
- Supabase - Auth: Serviço de auth disponível
- Supabase - Storage: Storage disponível
- Tabela: profiles: Tabela 'profiles' acessível

**Screenshot 2: External APIs**

Hugging Face:

- Dataset: eye-web-breaches: Dataset acessível  
repo: Samecinho/eye-web-breaches
- Dataset: eye-web-passwords: Dataset acessível  
repo: Samecinho/eye-web-passwords
- Space: eyeweb-n8n: Space a dormir (inativo)  
repo: Samecinho/eyeweb-n8n stage: SLEEPING

APIs Externas:

- Google Safe Browsing: API operacional
- URLScan.io: API operacional
- Groq AI: API operacional  
modelos\_disponíveis: 28 exemplos: ["meta-llama/llama-prompt-guard-2-2m","groq/compound","meta-llama/llama-4-maverick-17b-128e-instruct"]

**Screenshot 3: Infrastructure Services**

Infraestrutura:

- Render (Backend): Render operacional
- Vercel (Frontend): Frontend operacional
- Brevo (Email): API operacional plan: free

## Monitor de Tráfego

**Monitor de Tráfego**

Auto-refresh • LIVE

Requests (24h): 0 | IPs Ativos (5min): 0 | Suspeitos (24h): 0 | Bloqueados: 1

Logs

HORA	IP	LOCALIZAÇÃO	MÉTODO	PATH	STATUS	VPN	TEMPO	AÇÕES
11/02, 09:47:57	176.223.68.65	Portugal, Lisbon	POST	/api/admin/verify-mfa	200	—	899ms	<span>🔗</span>
11/02, 09:47:56	176.223.68.65	Portugal, Lisbon	OPTIONS	/api/admin/verify-mfa	200	—	0ms	<span>🔗</span>
10/02, 11:44:07	35.227.183.65	United States, The Dalles	GET	/	200	∞ VPN	0ms	<span>🔗</span>
10/02, 11:44:02	127.0.0.1	Local	HEAD	/	405	—	0ms	<span>🔗</span>
10/02, 11:41:01	3.101.42.222	United States, San Jose	POST	/api/visit	404	∞ VPN	703ms	<span>🔗</span>
10/02, 11:41:00	143.244.186.37	United States, Santa Clara	POST	/api/visit	404	∞ VPN	2ms	<span>🔗</span>

**Monitor de Tráfego**

Auto-refresh • LIVE

Requests (24h): 0 | IPs Ativos (5min): 0 | Suspeitos (24h): 0 | Bloqueados: 1

Logs

Sem atividade suspeita detetada  
O sistema de defesa está a monitorizar o tráfego

**Monitor de Tráfego**

Auto-refresh • LIVE

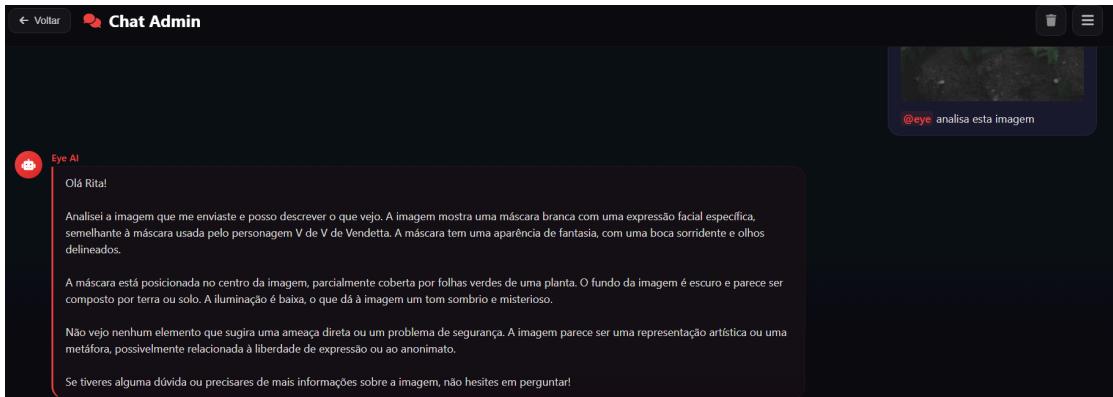
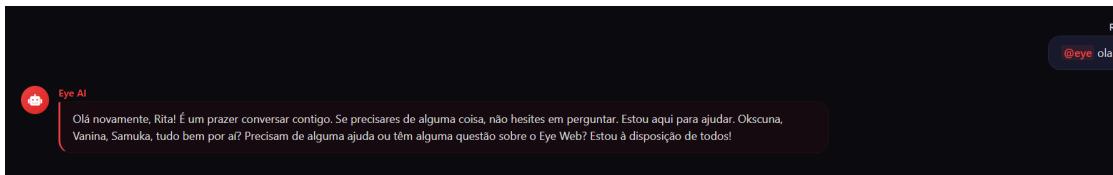
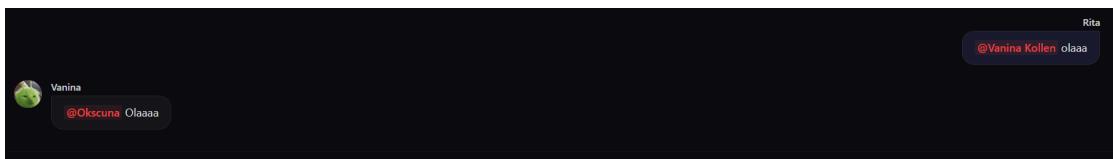
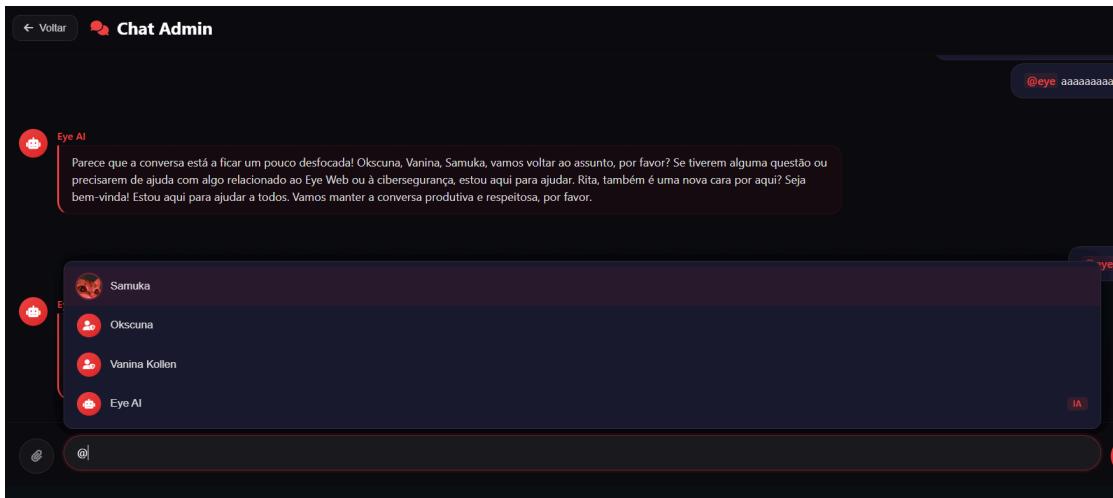
Requests (24h): 0 | IPs Ativos (5min): 0 | Suspeitos (24h): 0 | Bloqueados: 1

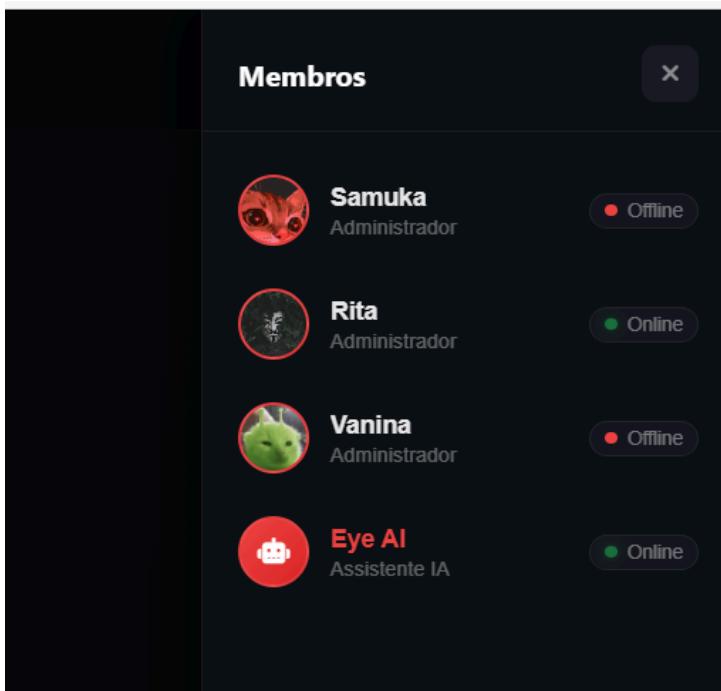
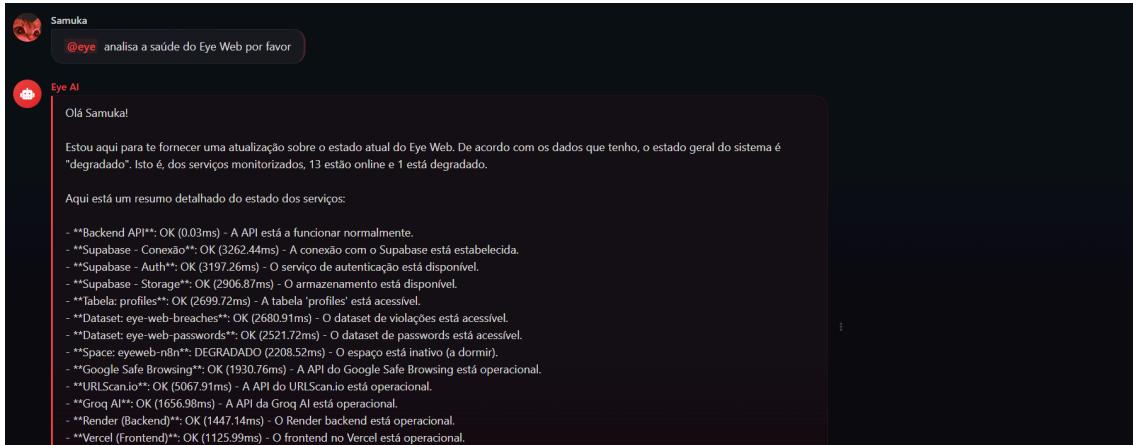
Logs

IPs Bloqueados

IP	PAÍS	MOTIVO	BLOQUEADO POR	REQUESTS	DATA	AÇÕES
78.137.220.137	—	teste	Manual	0	10/02, 11:42:25	<span>🔓</span> <span>📝</span>

### Administrador - Agente





## 6. Scripts ou Playbooks de Setup

### Variáveis de Ambiente

O *EyeWeb Reborn* utiliza variáveis de ambiente para configurar o comportamento de cada módulo (*backend*, *frontend* e *updater*) e estas variáveis são definidas em ficheiros .env.

#### Backend (backend/.env):

```
ENVIRONMENT=development # Define o ambiente (development ou production)
```

```
DEBUG=true # Ativa/desativa modo debug
```

```
HF_DATASET_REPO=Samezinho/eye-web-breaches #Repositório de breaches no Hugging Face
```

```
HF_TOKEN=hf_YOUR_TOKEN_HERE #Token de acesso ao Hugging Face
```

```
SUPABASE_URL=https://YOUR_PROJECT.supabase.co #URL do projeto Supabase (cache de URLs)
```

```
SUPABASE_ANON_KEY=your_anon_key_here #Chave anónima do Supabase
```

```
SUPABASE_SERVICE_KEY=your_service_role_key_here #Chave de serviço do Supabase
```

```
GROQ_API_KEY=gsk_your_groq_key_here #Chave de API para Groq AI (Llama 3)
```

```
GROQ_MODEL=llama-3.3-70b-versatile #Modelo a usar no Groq AI
```

```
GOOGLE_SAFE_BROWSING_KEY=your_google_key_here #Chave de API para Google Safe Browsing
```

```
URLSCAN_API_KEY=your_urlscan_key_here      #Chave de API para URLScan.io

CACHE_MAX_SIZE=100          #Tamanho máximo do cache

CACHE_TTL_SECONDS=3600    #Tempo de vida do cache (segundos)

ADMIN_EMAIL_HASH=your_sha256_email_hash_here #Hash SHA-256 do email admin

MFA_SECRET=your_totp_secret_here # Secret TOTP para MFA (autenticação multifator)
```

```
# --- Ambiente ---
ENVIRONMENT=development
DEBUG=true

# --- Hugging Face (Breach Checker) ---
HF_DATASET_REPO=Samezinho/eye-web-breaches
HF_TOKEN=hf_YOUR_TOKEN_HERE

# --- Supabase (URL Checker Cache) ---
SUPABASE_URL=https://YOUR_PROJECT.supabase.co
SUPABASE_ANON_KEY=your_anon_key_here
SUPABASE_SERVICE_KEY=your_service_role_key_here

# --- Groq AI (Llama 3) ---
GROQ_API_KEY=gsk_your_groq_key_here
GROQ_MODEL=llama-3.3-7b-versatile

# --- Google Safe Browsing ---
GOOGLE_SAFE_BROWSING_KEY=your_google_key_here

# --- URLScan.io ---
URLSCAN_API_KEY=your_urlscan_key_here

# --- Cache Settings ---
CACHE_MAX_SIZE=100
CACHE_TTL_SECONDS=3600

# --- Admin MFA ---
# Hash SHA-256 do email admin
ADMIN_EMAIL_HASH=your_sha256_email_hash_here
# Secret TOTP para MFA (gerar com: python -c "import pyotp; print(pyotp.random_base32())")
MFA_SECRET=your_totp_secret_here
```

### Explicação:

- *ENVIRONMENT* e *DEBUG*: controlam o modo de execução do *backend*;
- *HF\_DATASET\_REPO* e *HF\_TOKEN*: permitem aceder ao *dataset* de *breaches* no *Hugging Face*;
- *GROQ\_API\_KEY* e *GROQ\_MODEL*: integração com *Groq AI (Llama 3)* para análise avançada;
- *GOOGLE\_SAFE\_BROWSING\_KEY*: verifica URLs contra listas de sites maliciosos do *Google*;
- *URLSCAN\_API\_KEY*: permite análise de URLs via *URLScan.io*;

- *CACHE\_MAX\_SIZE* e *CACHE\_TTL\_SECONDS*: definem limites e tempo de vida do cache;
- *ADMIN\_EMAIL\_HASH*: protege o email do admin, evitando exposição direta;
- *MFA\_SECRET*: ativa autenticação multifator para o admin;
- *HF\_DATASET\_REPO* e *HF\_TOKEN*: permitem aceder ao *dataset* de *breaches* no *Hugging Face*;
- As variáveis *Supabase* são usadas para cache de URLs e integração com bases de dados externas.

Frontend (frontend/.env.local):

```
NEXT_PUBLIC_API_URL=http://localhost:8000 #URL do backend (API)
```

```
NEXT_PUBLIC_SUPABASE_URL=your-supabase-url #URL do Supabase (frontend)
```

```
NEXT_PUBLIC_SUPABASE_ANON_KEY=your-supabase-anon-key #Chave anónima do Supabase
```

```
NEXT_PUBLIC_ADMIN_EMAIL_HASH=your-admin-email-sha256-hash #Hash SHA-256 do email do admin
```

```
NEXT_PUBLIC_TURNSTILE_SITE_KEY=your-turnstile-site-key #Site Key do Cloudflare Turnstile (captcha)
```

```
# URL da API Backend  
# Desenvolvimento: http://localhost:8000  
# Produção: https://eye-web-api.onrender.com  
NEXT_PUBLIC_API_URL=http://localhost:8000

# Supabase  
NEXT_PUBLIC_SUPABASE_URL=your-supabase-url  
NEXT_PUBLIC_SUPABASE_ANON_KEY=your-supabase-anon-key

# Admin (hash SHA-256 do email admin)  
NEXT_PUBLIC_ADMIN_EMAIL_HASH=your-admin-email-sha256-hash

# Cloudflare Turnstile (Site Key - visível no frontend)  
NEXT_PUBLIC_TURNSTILE_SITE_KEY=your-turnstile-site-key
```

### Explicação:

- *NEXT\_PUBLIC\_API\_URL*: define o *endpoint* do *backend* para o *frontend*;
- As variáveis *Supabase* permitem integração com bases de dados externas;
- *NEXT\_PUBLIC\_ADMIN\_EMAIL\_HASH*: protege o email do admin;
- *NEXT\_PUBLIC\_TURNSTILE\_SITE\_KEY*: é usada para validação de *captcha*.

### Updater (updater/.env):

```
HF_TOKEN=hf_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx #Token de acesso ao Hugging Face  
(WRITE)
```

```
HF_DATASET_REPO=teu-username/eye-web-breaches #Repositório de breaches no  
Hugging Face
```

```
HF_BRANCH=main #Branch do repositório
```

```
# Token do Hugging Face (obter em: https://huggingface.co/settings/tokens)  
# Necessita de permissões de WRITE  
HF_TOKEN=hf_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
  
# Nome do repositório do dataset no Hugging Face  
# Formato: username/nome-do-repo  
HF_DATASET_REPO=teu-username/eye-web-breaches  
  
# Branch do repositório  
HF_BRANCH=main  
  
# Nível de Logging  
LOG_LEVEL=INFO
```

### Explicação:

- *HF\_TOKEN*: permite *upload* e atualização de *datasets* no *Hugging Face*;
- *HF\_DATASET\_REPO*: define o repositório de destino;
- *HF\_BRANCH*: permite especificar a *branch* (normalmente *main*).

## Deploy

O processo de deploy do *EyeWeb Reborn* está dividido em três componentes principais:

### Backend (FastAPI) — Render

1. Criar um novo serviço Web no *Render*;
2. Ligar ao repositório *GitHub*;
3. Configurar:
  - Diretório raiz: *backend*;
  - Comando de *build*: `pip install -r requirements.txt`;
  - Comando de arranque: `uvicorn app.main:app --host 0.0.0.0 --port $PORT`;
4. Adicionar as variáveis de ambiente necessárias.

#### Frontend (Next.js) — Vercel

1. Importar o projeto para o *Vercel*;
2. Configurar:
  - Diretório raiz: *frontend*;
  - Framework: *Next.js*;
3. Adicionar a variável *NEXT\_PUBLIC\_API\_URL* com o URL do *backend* (*Render*).

#### Updater (Python) — Local ou CI/CD

1. O *updater* pode ser executado localmente ou integrado em *pipelines* CI/CD (ex: *GitHub Actions*);
2. Requer as variáveis de ambiente e permissões para escrever no *Hugging Face Datasets*.

### **Scripts de Setup (Windows e Linux/macOS)**

#### Windows - PowerShell

- 1. Clonar o repositório**
  - `git clone https://github.com/Sam-Ciber-Dev/eyeweb.git`
  - `cd eyeweb`
- 2. Copiar ficheiros de ambiente**
  - `Copy-Item backend\.env.example backend\.env -Force`
  - `Copy-Item frontend\.env.example frontend\.env.local -Force`
  - `Copy-Item updater\.env.example updater\.env -Force`
- 3. Instalar dependências do backend**
  - `cd backend`
  - `pip install -r requirements.txt`
- 4. Iniciar o backend (em nova janela)**

- Start-Process powershell -ArgumentList 'cd """+\$PWD.Path""'; unicorn app.main:app --reload'

## 5. Instalar dependências do frontend

- cd ../frontend
- npm install

## 6. Iniciar o frontend

- Start-Process powershell -ArgumentList 'cd """+\$PWD.Path"""; npm run dev'

## 7. Atualizar datasets

- cd ../updater
- pip install -r requirements.txt
- python updater.py
- python password\_updater.py

## Linux/macOS

### 1. Clonar o repositório

- git clone https://github.com/Sam-Ciber-Dev/eyeweb.git
- cd eyeweb

### 2. Copiar ficheiros de ambiente

- cp backend/.env.example backend/.env
- cp frontend/.env.example frontend/.env.local
- cp updater/.env.example updater/.env

### 3. Instalar dependências do backend

- cd backend
- pip install -r requirements.txt

### 4. Iniciar o backend (em background)

- nohup unicorn app.main:app --reload &

### 5. Instalar dependências do frontend

- cd ../frontend
- npm install

### 6. Iniciar o frontend (em background)

- npm run dev &

## 7. Atualizar datasets

- cd ..../updater
- pip install -r requirements.txt
- python updater.py
- python password\_updater.py

## 7. Políticas de Segurança e Isolamento do Ambiente

### 7.1. Medidas de Segurança Implementadas

#### a) Segurança de Comunicação

##### **HTTPS Obrigatório**

- Todo o tráfego é encriptado via HTTPS → Protege contra ataques *man-in-the-middle*;
- Certificado SSL/TLS ativo para todas as comunicações.

#### b) Defesa Contra Ataques

##### **Proteção contra DDoS**

- Implementação de *rate limiting* no *backend*;
- Limitação de pedidos por IP;
- Monitorização de padrões de tráfego suspeitos.

##### **Proteção contra Injeção (SQL/NoSQL)**

- Sanitização de todos os *inputs* de utilizadores;
- Utilização de parâmetros parametrizados em *queries SQL*;
- Validação de dados tanto no *frontend* como no *backend*.

#### c) Controlo de Acesso Administrativo

##### **Bloqueio por MAC/IP**

- Possibilidade de bloquear usuários pelo *hardware* do dispositivo;
- Bloqueio de IPs suspeitos;
- Lista negra de endereços bloqueados permanentemente.

##### **Credenciais Dinâmicas para Administradores**

- Credenciais geradas por um programa externo localhost de forma aleatória;
- Alteração automática das credenciais a cada 60 segundos;
- Acesso administrativo requer autenticação de dois fatores (MFA).

d) Proteção de Dados e Privacidade

**Política de Não Armazenamento**

- Nenhuma palavra-passe de utilizador é guardada;
- Emails não são armazenados na base de dados;
- Os dados de análise são temporários, apenas durante a sessão;
- Não é necessário registo ou login para utilizadores normais.

**Anonimização de Dados**

- Logs sem informação pessoal identificável;
- Dados de análise são agregados e anonimizados.

**7.2. Arquitetura de Isolamento**

a) Isolamento de Redes

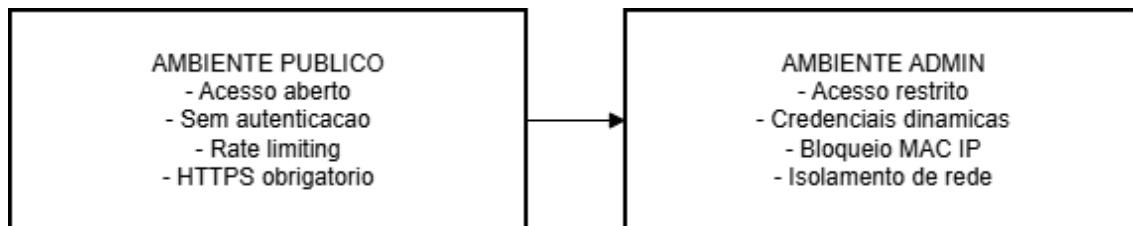
**Ambiente Público**

- Acesso aberto;
- Sem autenticação obrigatória;
- *Rate limiting* aplicado;
- HTTPS obrigatório.

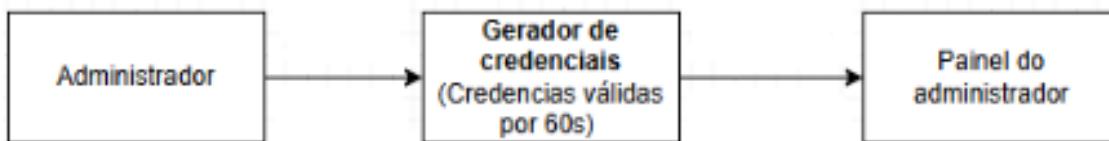
**Ambiente Administrativo**

- Acesso restrito apenas a administradores;

- Credenciais dinâmicas e MFA;
- Bloqueio de MAC/IP;
- Isolamento de rede dedicado.



b) Fluxo de Acesso Administrativo



### 7.3. Políticas Específicas de Implementação

a) Bloqueio Automático de IP/Hardware

#### Critérios de Bloqueio Automático

- Padrões de *scanning* de portas detetados;
- Tráfego identificado como malicioso por *threat intelligence*.

#### Bloqueio Manual pelo Administrador

- IPs específicos via painel administrativo;
- Intervalos de IP conhecidos por ataques ao nosso site.

c) Política de Retenção de Dados

- **Resultados de análise:** mantidos por 24h;
- **IPs bloqueados:** mantidos permanentemente até remoção manual;

- **Cache de verificação:** 24 horas para otimização.

#### 7.4. Procedimentos de Resposta a Incidentes

##### a) Deteção de Ameaças

- Monitorização ativa de múltiplas tentativas de acesso falhadas;
- Notificação imediata de IPs adicionados à lista negra;
- Monitorização de consumo anormal de recursos.

##### b) Resposta a Incidentes

- Bloqueio automático de IPs maliciosos;
- Notificação imediata ao administrador;
- Atualização de regras de *firewall* se necessário.

#### 7.5. Conformidade e Auditoria

##### a) Logs e Auditoria

- Registo de todas as tentativas de acesso administrativo;
- IPs bloqueados e motivo do bloqueio;
- Tentativas de *rate limiting* excedido;
- Erros de autenticação e autorização.

##### b) Revisão de Segurança

- Análise mensal de *logs* de segurança;
- Atualização trimestral de políticas de segurança;
- Teste semestral das medidas de proteção;
- Auditoria anual completa do sistema.

A privacidade dos utilizadores foi uma prioridade no desenvolvimento do *EyeWeb Reborn*. Os dados introduzidos nunca são enviados em formato legível para o servidor e o sistema utiliza técnicas de *hashing* e anonimização, garantindo que a informação pessoal não seja identificada.

- **Hash (SHA-256):** Técnica utilizada para transformar dados num código irreversível, garantindo a proteção da informação do utilizador;
- **K-Anonymity:** Técnica de privacidade que impede a identificação direta do utilizador, enviando apenas parte da informação para o servidor;
- **API:** Interface que permite a comunicação entre diferentes componentes do sistema.

## 8. Manual de Utilização

### **Utilizador:**

Ao aceder à plataforma *EyeWeb Reborn*, o utilizador encontra uma interface centralizada que disponibiliza diversas ferramentas de diagnóstico de segurança sem necessidade de registo prévio. A interação principal permite ao utilizador introduzir os seus dados pessoais, especificamente endereços de e-mail e números de telemóvel, para que o sistema consulte a base de dados interna e verifique se essas informações foram comprometidas em fugas de dados conhecidas.

Paralelamente à verificação de dados pessoais, a plataforma oferece uma ferramenta de análise de credenciais com dupla funcionalidade. O utilizador pode submeter uma palavra-passe para que o sistema avalie a força da senha e, simultaneamente, cruze essa informação com bases de dados de *leaks* para confirmar se a palavra-passe já foi exposta publicamente em incidentes de segurança anteriores. Adicionalmente, para prevenir ataques de *Phishing*, o utilizador dispõe de um campo para inserção de URLs, onde o sistema analisa o *link* em tempo real e se o mesmo é seguro ou malicioso.

Para auxiliar em qualquer uma destas etapas, o site integra um agente inteligente (*Chatbot*) que acompanha a navegação e responde a dúvidas sobre o funcionamento das ferramentas ou boas práticas de cibersegurança. Este agente reativo, disponível para o utilizador, serve como assistente para fornecer aos utilizadores mais informação sobre o *EyeWeb Reborn*, os serviços que oferecemos, além de dicas para ajudar a proteger dados e informações. Está ao mesmo tempo protegido graças a métodos aplicados para a segurança do agente, alinhados com os nossos valores em cibersegurança.

### **Administrador:**

O acesso ao painel de gestão é restrito e requer um procedimento de autenticação reforçada. Ao entrar na área de login, o administrador deve introduzir as suas credenciais e para completar o acesso é necessário executar localmente um *script Python* específico que gera um *token* de Autenticação Multifator (*MFA*). Apenas com a inserção deste código temporário é que o acesso ao painel de administração é libertado.

Uma vez autenticado, o administrador tem ao seu dispor um painel de controlo para a gestão de dados e segurança. Na gestão de informação, encontra-se um *dataset* que atualiza automaticamente os registos de dados vazados, mantendo a plataforma atualizada com novas

ameaças, de forma segura. No que toca à defesa ativa do sistema, o administrador possui uma área de monitorização de tráfego em tempo real, onde pode analisar a origem das requisições e caso detete comportamento suspeito, bloquear imediatamente o endereço IP do atacante, impedindo o seu acesso ao site, este bloqueio é reversível através da gestão da lista de IPs banidos. Por fim, a plataforma inclui um sistema de verificação de vulnerabilidades do próprio Github, que alerta para vulnerabilidades no próprio código do projeto, contando o administrador com o apoio de um agente reativo especializado para sugerir correções e medidas de segurança, como por exemplo, se um determinado endereço IP foi comprometido, se uma palavra-passe é segura, entre outras funcionalidades.

## 9. Conclusão

O projeto *EyeWeb Reborn* consistiu na modernização e no reforço do site *EyeWeb*. Ao longo do desenvolvimento, foi criada uma plataforma que não só mantém as funcionalidades originais de análise de websites, verificação de palavras-passe, e-mails e números de telemóvel, como também as aprimora através da integração de dois Agentes Reativos inteligentes (tanto no utilizador com o administrador) e da implementação de medidas de segurança, tanto para os utilizadores finais como para os administradores.

Os objetivos inicialmente propostos foram atingidos com sucesso. Foi desenvolvida uma plataforma completa de segurança digital que protege os utilizadores, oferece monitorização em tempo real aos administradores e disponibiliza suporte através de dois agentes inteligentes.

### 9.1. Limitações e Sugestões Futuras

A parte mais complexa do projeto foi, sem dúvida, o desenvolvimento do Agente Reativo. Garantir que este compreendesse corretamente às questões colocadas pelos utilizadores/administradores e fornecesse respostas úteis e adequadas, revelou-se mais demorado do que o inicialmente previsto. Adicionalmente, existiram alguns desafios na coordenação das diferentes componentes do projeto entre todos os elementos do grupo.

A ferramenta depende totalmente de uma ligação à Internet e de APIs externas para funcionar, o que a limita em cenários *offline*. Para proteger a privacidade dos utilizadores, optámos por não guardar histórico das análises, o que é positivo para a privacidade. Também identificamos que a falta de suporte a vários idiomas é uma restrição para uma maior internacionalização, algo que gostaríamos de ver implementado futuramente.

Apesar de todas as medidas de segurança que implementámos, há sempre riscos que permanecem. Que são ameaças completamente novas e desconhecidas. Em situações de pico de utilização, existe o risco de o servidor ficar sobrecarregado, afetando a velocidade da plataforma. Por último, há sempre o risco, embora minimizado, de o painel de administração ser comprometido se as credenciais forem roubadas.

#### Ideias para Melhorar no Futuro

Se continuássemos a desenvolver o projeto, gostaríamos de:

1. Adicionar suporte para mais idiomas;

2. Criar uma aplicação móvel;
3. Implementar um sistema de alertas por email quando são detetadas ameaças graves;
4. Adicionar mais tipos de análise de segurança para websites.

Existe sempre margem para melhorias e já foram delineadas várias ideias para desenvolvimentos futuros. Uma das propostas passa pela criação de uma aplicação móvel, permitindo que a monitorização seja realizada a partir de qualquer local. A implementação de notificações automáticas mais personalizadas para eventos críticos constitui outro passo para a evolução do sistema.

Para tornar a plataforma ainda mais inteligente, pretende-se integrar técnicas de *machine learning* que permitam a deteção de padrões de ameaça mais complexos. Por fim, está prevista a expansão contínua da base de dados de ameaças, de forma a manter o sistema sempre atualizado e preparado para lidar com novas ameaças à medida que estas surgem.

## **9.2. Considerações sobre a Inteligência Artificial no Contexto deste Projeto**

O papel da Inteligência Artificial torna-se evidentemente fundamental para a proteção e implementação de melhorias de segurança dentro deste projeto, assim como também como meio de comunicação direta com os utilizadores interessados em verificar e reforçar a sua privacidade. Junto com a sua integração, aplicou-se uma camada de proteção contra possíveis ataques, tendo em conta as existentes vulnerabilidades presentes na IA.

## Referências

1. Blog DNC. (s.d.). *Tipos de cardinalidade: o que é e conheça os tipos*. Disponível em: <https://www.escoladnc.com.br/blog/entendendo-os-tipos-de-cardinalidade-em-modelagem-de-bancos-de-dados/>
2. Cordeiro de Sousa, A. (s.d.). *Explorando os diferentes tipos de chaves em bancos de dados*. LinkedIn. Disponível em: <https://pt.linkedin.com/pulse/explorando-os-diferentes-tipos-de-chaves-em-bancos-cordeiro-de-sousa>
3. EBAC Online. (s.d.). *O que é a normalização de bases de dados e como fazê-la?* Disponível em: <https://ebaconline.com.br/blog/normalizacao-de-bases-de-dados>
4. GitHub. (s.d.). *Documentation – Version control*. Disponível em: <https://docs.github.com/en>
5. Google. (s.d.). *Safe Browsing API – Detection of unsafe web resources*. Disponível em: <https://developers.google.com/safe-browsing>
6. Google. (2024). *Google Authenticator documentation*. Disponível em: <https://github.com/google/google-authenticator>
7. Groq. (s.d.). *Groq platform – Llama 3 inference*. Disponível em: <https://groq.com/>
8. Have I Been Pwned. (2023). *API documentation v3*. Disponível em: <https://haveibeenpwned.com/API/v3>
9. IBM. (s.d.). *O que é integridade de dados?* Disponível em: <https://www.ibm.com/br-pt/topics/data-integrity>
10. JWT.io. (s.d.). *Introduction to JSON Web Tokens*. Disponível em: <https://jwt.io/introduction>
11. Machado, D. B. (s.d.). *Normalização em bancos de dados*. Medium. Disponível em: <https://medium.com/@diegobmachado/normaliza%C3%A7%C3%A3o-em-banco-de-dados-5647cdf84a12>
12. Mozilla Developer Network. (2024). *HTTP – Hypertext Transfer Protocol*. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTTP>
13. Neon Tech. (s.d.). *PostgreSQL INSERT*. Disponível em: <https://neon.tech/postgresql/postgresql-tutorial/postgresql-insert>
14. NIST. (2023). *Digital Identity Guidelines: Authentication and Lifecycle Management (SP 800-63-3)*. Disponível em: <https://pages.nist.gov/800-63-3/>

15. Node.js. (2024). *Node.js official documentation*. Disponível em:  
<https://nodejs.org/en/docs/>
16. OWASP. (s.d.). *Cheat Sheet Series – Security best practices*. Disponível em:  
<https://cheatsheetseries.owasp.org/>
17. Paes de Souza, H. (s.d.). *Chaves em bancos de dados relacionais*. LinkedIn. Disponível em:  
<https://pt.linkedin.com/pulse/chaves-em-bancos-de-dados-relacionais-henrique-paes-de-souza>
18. Planez Diniz, P. (s.d.). *Integridade em banco de dados*. LinkedIn. Disponível em:  
<https://pt.linkedin.com/pulse/integridade-em-banco-de-dados-paulo-planez-diniz>
19. PostgreSQL. (s.d.). *CREATE VIEW – SQL command*. Disponível em:  
<https://www.postgresql.org/docs/current/sql-createview.html>
20. PostgreSQL. (s.d.). *Generated columns*. Disponível em:  
<https://www.postgresql.org/docs/current/ddl-generated-columns.html>
21. PostgreSQL. (s.d.). *INSERT – SQL command*. Disponível em:  
<https://www.postgresql.org/docs/current/sql-insert.html>
22. PostgreSQL. (s.d.). *Security – Best practices*. Disponível em:  
<https://www.postgresql.org/docs/current/security.html>
23. PostgreSQL. (2024). *PostgreSQL 16.2 documentation*. Disponível em:  
<https://www.postgresql.org/docs/16/index.html>
24. React. (2024). *Getting started with React*. Disponível em:  
<https://reactjs.org/docs/getting-started.html>
25. Security Headers. (s.d.). *Security headers analysis*. Disponível em:  
<https://securityheaders.com/>
26. Shodan. (s.d.). *Search engine for Internet-connected devices*. Disponível em:  
<https://www.shodan.io/>
27. Sucuri. (s.d.). *Sucuri SiteCheck – Website security scanner*. Disponível em:  
<https://sitecheck.sucuri.net/>
28. URLScan.io. (s.d.). *Website analysis and threat detection*. Disponível em:  
<https://urlscan.io/>
29. VirusTotal. (s.d.). *API documentation – File and URL analysis*. Disponível em:  
<https://developers.virustotal.com/reference/overview>
30. Visual Studio Code. (s.d.). *Documentation – Development environment*. Disponível em:  
<https://code.visualstudio.com/docs>
31. W3Schools. (s.d.). *SQL CREATE TABLE statement*. Disponível em:  
[https://www.w3schools.com/sql/sql\\_create\\_table.asp](https://www.w3schools.com/sql/sql_create_table.asp)
32. W3Schools. (s.d.). *SQL constraints*. Disponível em:  
[https://www.w3schools.com/sql/sql\\_constraints.asp](https://www.w3schools.com/sql/sql_constraints.asp)

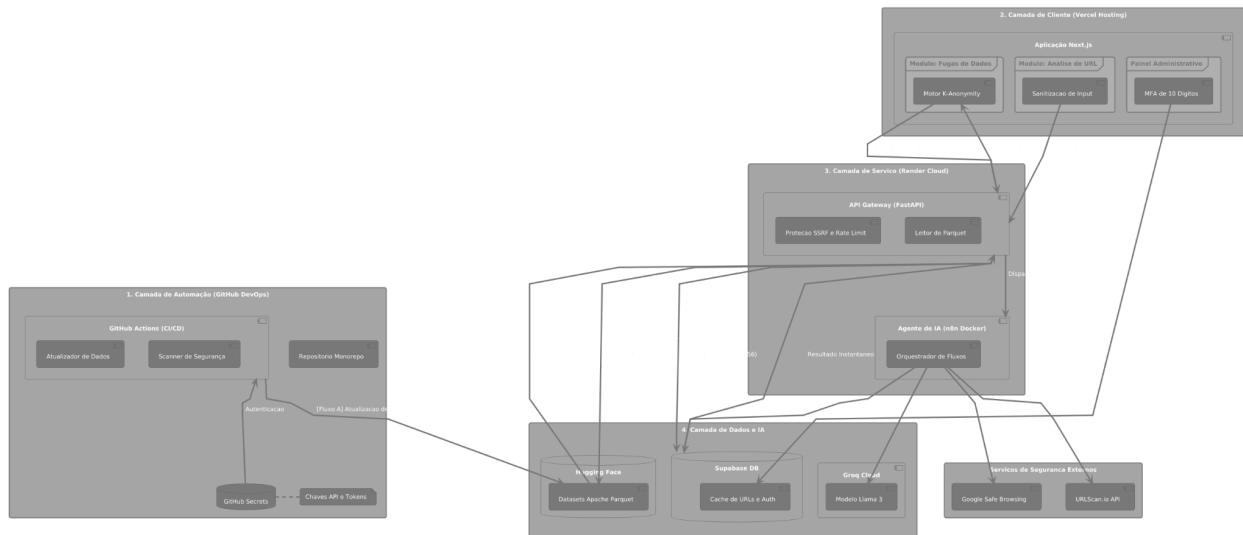
33. Wikipedia. (s.d.). *K-anonymity*. Disponível em:

<https://en.wikipedia.org/wiki/K-anonymity>

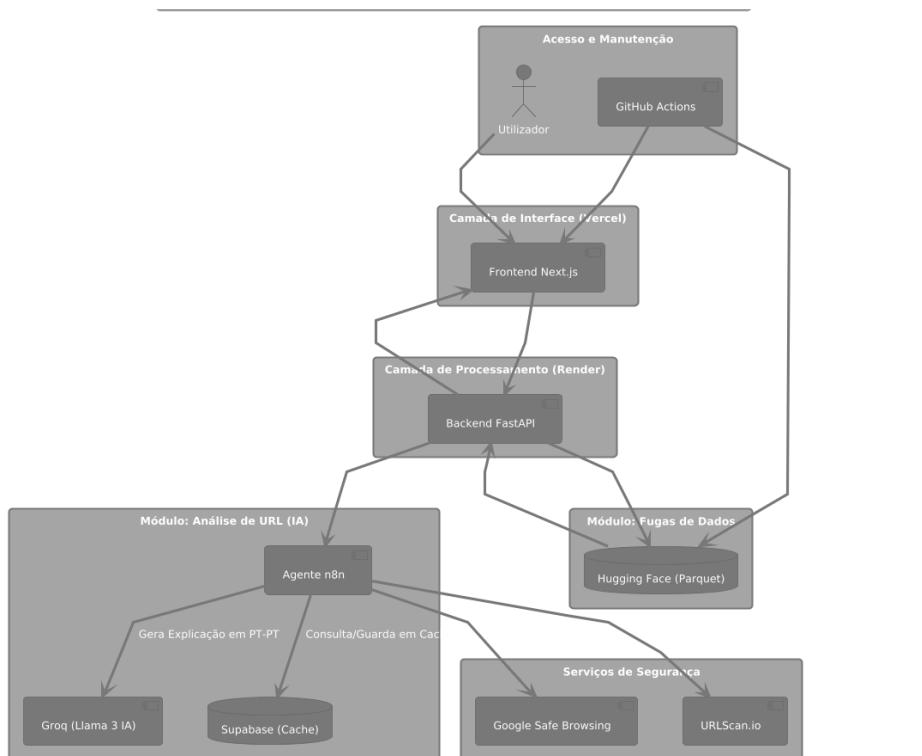
## Diagramas

### 1.- Arquitetura do Sistema

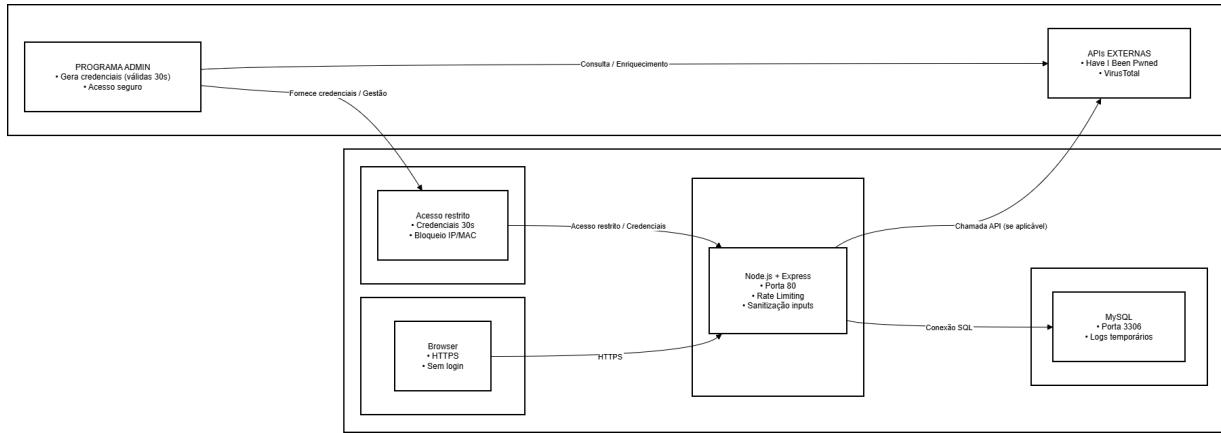
- Diagrama de Arquitetura Integrada do Sistema do Site



- Diagrama de Arquitetura do Site e a sua Visão Geral



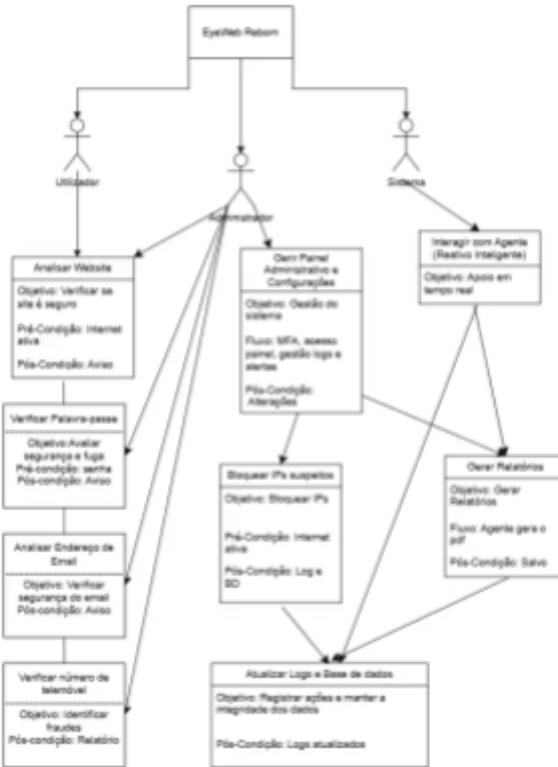
- Diagrama de Deployment



## 2.- Modelação de Requisitos

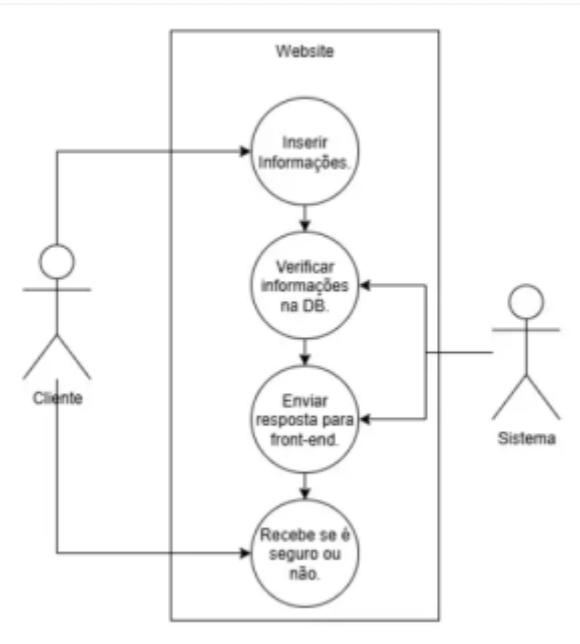
- Diagrama de Caso de Uso - Tudo

<https://drive.google.com/file/d/1ABX1rmWdiJy0YHJvgEohAeirGnBIFlq/view?usp=sharing>



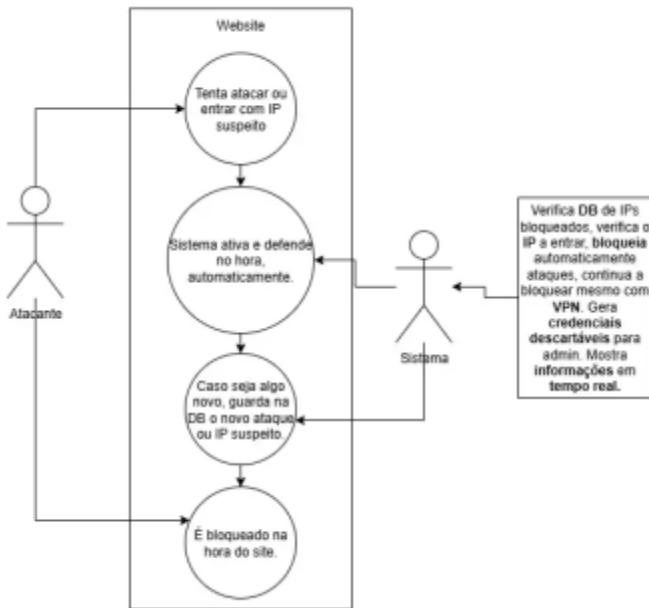
- Diagrama de Caso de Uso - Site

[https://drive.google.com/file/d/1CMwkKZ4ZRjNX6T3iPjE8X1--J6wDTF\\_J/view?usp=sharing](https://drive.google.com/file/d/1CMwkKZ4ZRjNX6T3iPjE8X1--J6wDTF_J/view?usp=sharing)



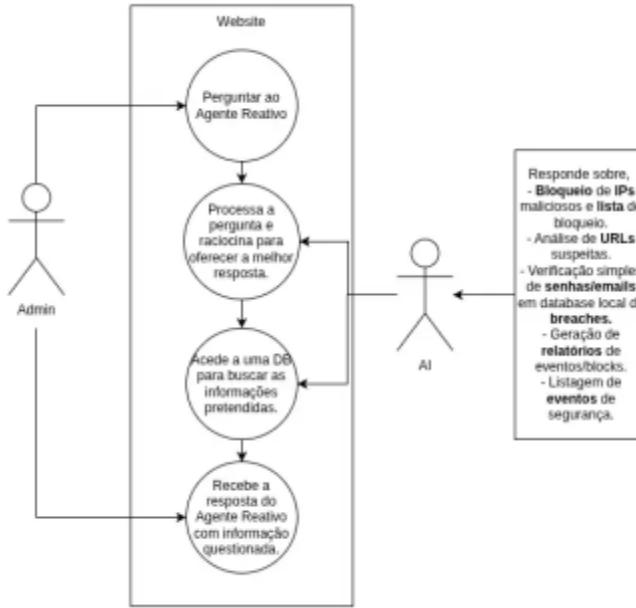
- Diagrama de Caso de Uso - Segurança

<https://drive.google.com/file/d/1BoFUfILnGEfGL7TVmxqltj-889mWtGNV/view?usp=sharing>

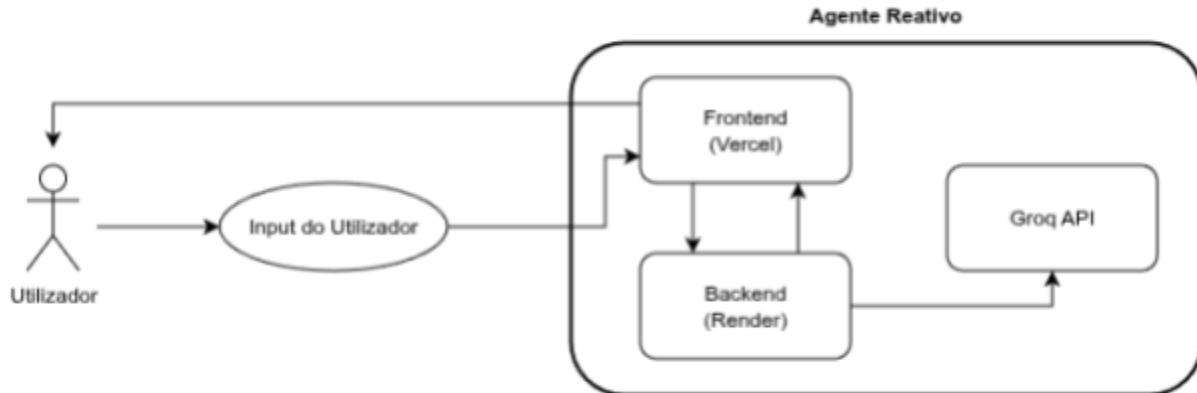


- Diagrama de Caso de Uso - Agente Reativo Administrador

[https://drive.google.com/file/d/19rcz1aZ6j\\_vJdvmY7foEzyAYzTO2wfNa/view?usp=sharing](https://drive.google.com/file/d/19rcz1aZ6j_vJdvmY7foEzyAYzTO2wfNa/view?usp=sharing)



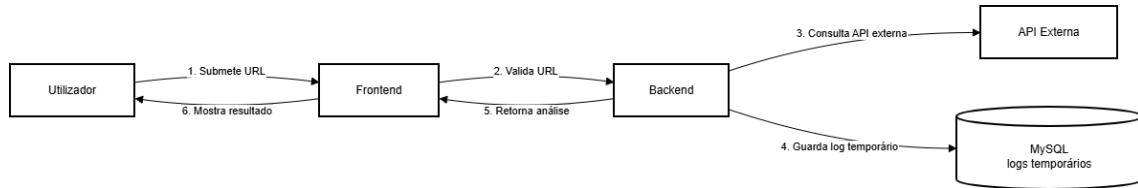
- Diagrama de Caso de Uso - Agente Reativo Utilizador



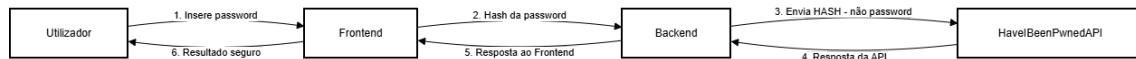
### 3.- Modelação de Comportamento e Processo

- Diagramas de Fluxo de Dados

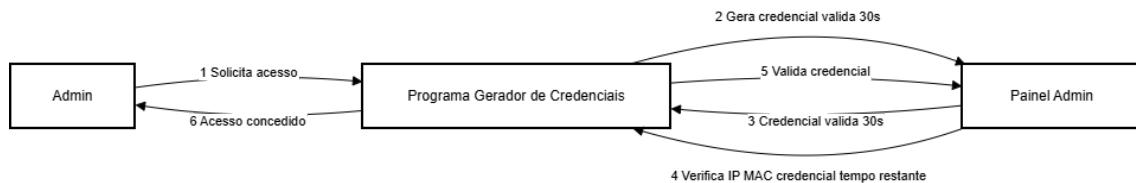
- Fluxo 1: Análise de Website (Utilizador Normal)



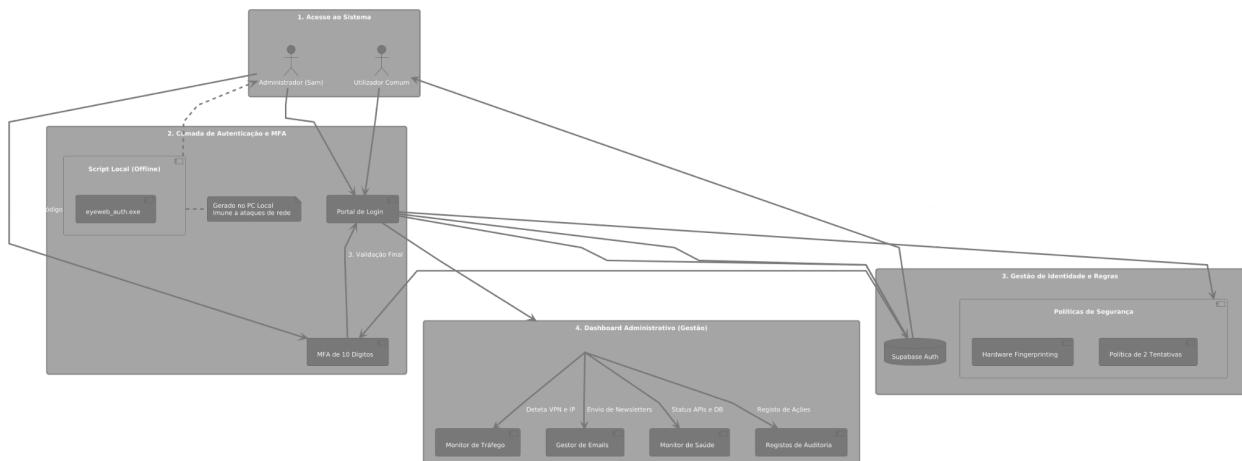
- Fluxo 2: Verificação de Password (Privacidade Total)



- Fluxo 3: Acesso Administrativo (Segurança Máxima)

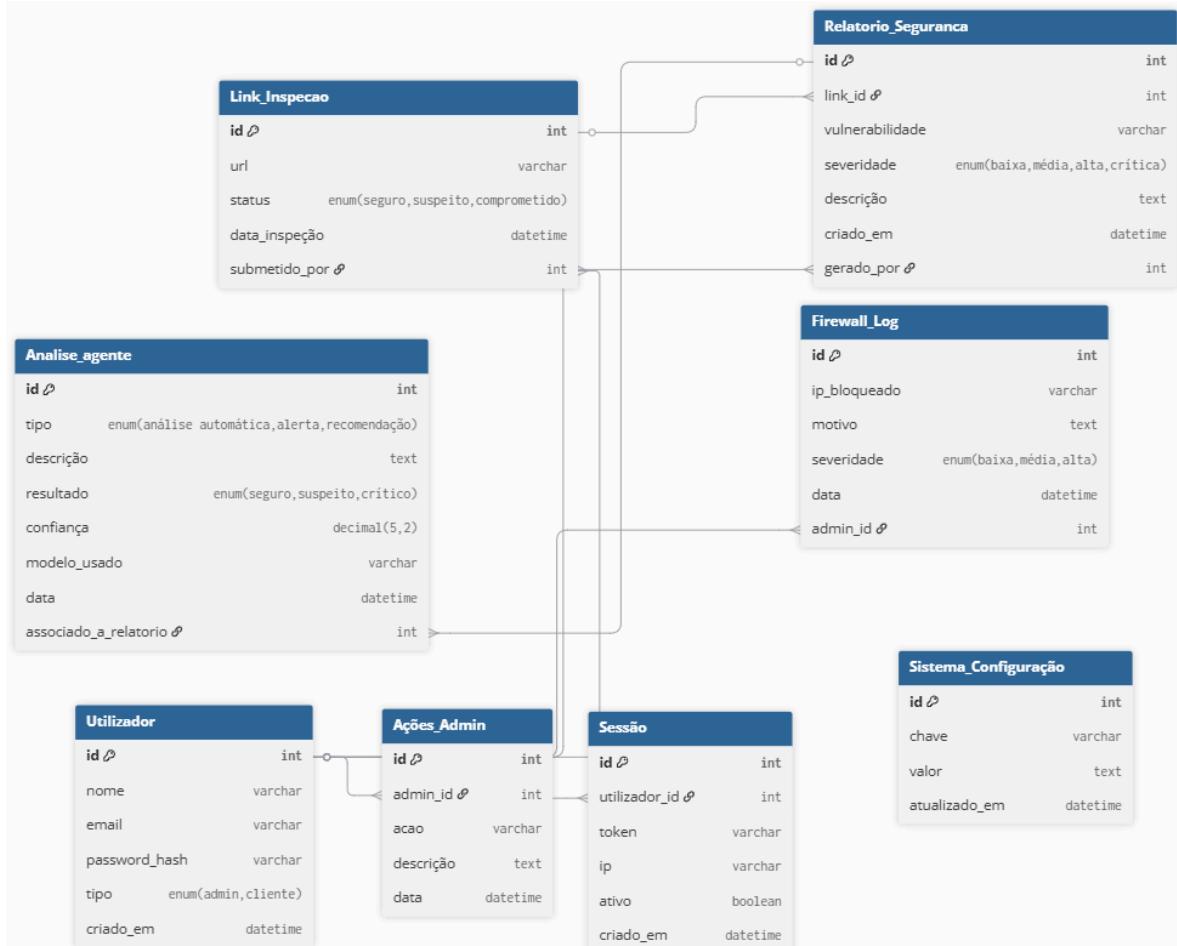


- Diagrama de Verificação de Segurança do Painel De Administrador



- Diagrama de Sequências

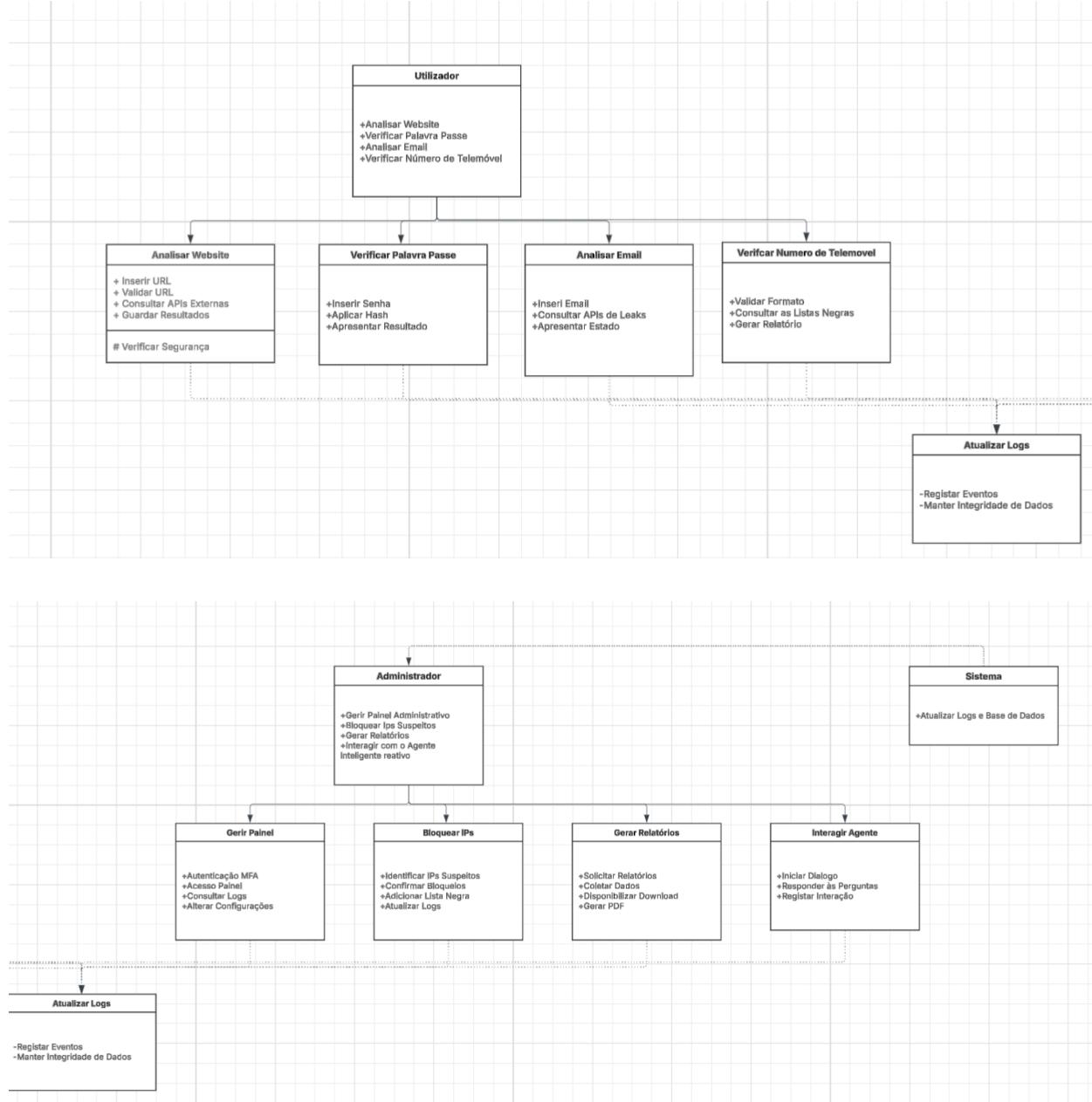
<https://dbdiagram.io/d/Diagrama-projeto-3-68f89d9c2e68d21b41acf1b7>



#### 4.- Estrutura de Código/Dados

- Diagrama UML

[https://lucid.app/lucidchart/6787c3cb-e520-45da-b24b-d4c3060b7ece/edit?viewport\\_loc=-2852%2C-1284%2C4189%2C1965%2C0\\_0&invitationId=inv\\_0d95522a-bbb3-446b-9bee-44f16e3a41da](https://lucid.app/lucidchart/6787c3cb-e520-45da-b24b-d4c3060b7ece/edit?viewport_loc=-2852%2C-1284%2C4189%2C1965%2C0_0&invitationId=inv_0d95522a-bbb3-446b-9bee-44f16e3a41da)



## Links

- Link do site

<https://eyeweb.vercel.app/>

- Link do repositório e documentação do GitHub

<https://github.com/Galaxiay11/EYEWEB/tree/Eyeweb-Reborn>