Nome: Vinicius Roberto Gonzales Arado – RGM: 11221100374

Nome: Guilherme da Silva Leon – RGM: 11221103758

Nome: Guilherme Marques Rezende – RGM: 11221104226

Nome: Lucas de Jesus Oliveira – RGM: 11221100971

Nome: João Gabriel Gomes Correia – RGM: 11222201815

**Documento de Requisitos e Arquitetura**

## **1. Introdução**

### 1.1 Objetivo

### Este documento tem como objetivo definir os requisitos funcionais e não funcionais, a modelagem do banco de dados, o fluxo de autenticação e controle de acesso, bem como o formato da API para o módulo de autenticação e gestão de usuários. O módulo será projetado para ser integrado a outros sistemas, garantindo segurança, conformidade com a LGPD e boas práticas de desenvolvimento.

A implementação desse módulo é essencial para garantir a proteção de informações sensíveis dos usuários, assegurando que apenas indivíduos autorizados tenham acesso aos dados. Além disso, a auditoria e o controle de acessos serão registrados de forma detalhada, permitindo que eventuais problemas de segurança sejam identificados e mitigados de maneira eficiente.

Este documento também busca estabelecer padrões de desenvolvimento e boas práticas de segurança para evitar vulnerabilidades comuns, como ataques de injeção de SQL, ataques de força bruta e vazamento de informações sigilosas.

## 1.2 Justificativa

Com o crescente número de aplicações que envolvem dados pessoais sensíveis, torna-se cada vez mais essencial adotar mecanismos robustos de autenticação e controle de acesso. A segurança da informação é um pilar fundamental para a integridade, confidencialidade e disponibilidade de sistemas. Este módulo de autenticação surge como resposta à necessidade de garantir o acesso restrito e controlado a sistemas que armazenam dados sensíveis, oferecendo uma solução escalável, segura e em conformidade com a LGPD.

## 1.3 Público-Alvo

Este sistema foi idealizado para atender equipes de desenvolvimento que precisam integrar um módulo de autenticação seguro em seus projetos, sem ter que desenvolver essa funcionalidade do zero. Também pode ser útil para organizações que buscam adequar seus sistemas à LGPD e que necessitam de mecanismos de auditoria, rastreabilidade e exclusão de dados conforme a legislação.

## **2. Especificação do Sistema**

### 2.1 Requisitos Funcionais

O sistema deverá ser capaz de cadastrar novos usuários, autenticar usuários existentes, e aplicar controle de acesso baseado em papéis (RBAC). Também incluirá funcionalidades para reset de senha, registro de logs de acesso e ações, exclusão de conta conforme diretrizes da LGPD, e mecanismos de anonimização ou pseudoanonimização dos dados. A autenticação será realizada com JWT, incluindo funcionalidades de geração, verificação e revogação de tokens.

* Cadastro de usuários, incluindo a coleta de informações básicas como nome, email e senha.
* Login seguro (autenticação com credenciais protegidas por métodos seguros de hashing).
* Controle de permissões baseado em papéis (RBAC - Role-Based Access Control) para diferenciar acessos administrativos e de usuários comuns.
* Reset de senha seguro, com envio de link de recuperação via email.
* Registro detalhado de logs de acesso e alterações realizadas dentro do sistema.
* Exclusão de conta conforme diretrizes da LGPD, garantindo a remoção segura de dados pessoais.
* Implementação de anonimização ou pseudoanonimização conforme a necessidade de preservação de logs sem identificação direta dos usuários.
* Uso de **JWT (JSON Web Token)** para autenticação e controle de sessão de forma segura e escalável.
* Auditoria de acessos e tentativas de login mal-sucedidas para aumentar a segurança contra acessos não autorizados.

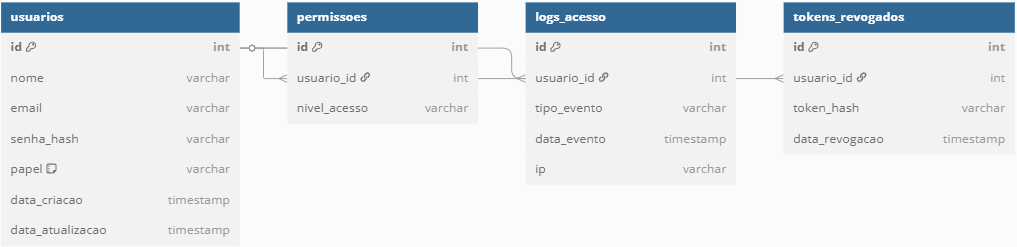
### 2.2 Requisitos Não Funcionais

A aplicação será desenvolvida com foco em segurança, empregando práticas como hash seguro de senhas, prevenção contra injeção SQL, XSS, CSRF e outras ameaças. A modularidade será garantida para facilitar a integração com sistemas externos. As APIs seguirão o padrão RESTful com documentação no formato OpenAPI. A performance será otimizada com técnicas de caching e escalabilidade horizontal.

* **Segurança:** O sistema deve adotar hashing seguro de senhas, como bcrypt ou Argon2, garantindo que mesmo em caso de vazamento de banco de dados, as credenciais permaneçam protegidas. Deve-se utilizar proteção contra ataques comuns, incluindo SQL Injection, XSS, CSRF e ataques de força bruta.
* **Modularidade:** O sistema deverá ser desenvolvido de forma modular, permitindo que seja facilmente integrado a outras aplicações que necessitem de um sistema de autenticação robusto.
* **APIs bem documentadas:** A documentação da API deverá seguir padrões como OpenAPI/Swagger, garantindo que novos desenvolvedores consigam utilizar a autenticação sem dificuldades.
* **Escalabilidade e performance:** A solução deverá ser capaz de lidar com um grande número de usuários simultâneos, utilizando técnicas de caching e otimização para garantir alto desempenho.

## **3. Modelagem do Banco de Dados**

3.1 Diagrama Entidade-Relacionamento (DER)

O banco de dados será composto por quatro tabelas principais: usuários, permissões, logs de acesso e tokens revogados. Essas entidades refletem a estrutura mínima necessária para suportar a autenticação segura com JWT, rastreamento de atividades e controle de permissões.

### 3.2 Estrutura das Principais Tabelas

* **usuarios** (id, nome, email, senha\_hash, papel, data\_criacao, data\_atualizacao)
* **permissoes** (id, usuario\_id, nivel\_acesso)
* **logs\_acesso** (id, usuario\_id, tipo\_evento, data\_evento, ip)
* **tokens\_revogados** (id, usuario\_id, token\_hash, data\_revogacao) (Para blacklist de tokens JWT revogados)

## **4. Fluxo de Autenticação e Controle de Acesso**

O fluxo de autenticação será baseado em JWT, garantindo eficiência e segurança. O funcionamento seguirá as seguintes etapas:

1. O usuário se cadastra informando email e senha forte, que são armazenados no banco com hash seguro.
2. No login, as credenciais são validadas e um token **JWT** é gerado.
3. O token JWT inclui informações sobre o usuário e suas permissões de acesso.
4. O token deve ser enviado no cabeçalho das requisições para acessar recursos protegidos.
5. A API valida o token a cada requisição, verificando sua validade e permissões associadas.
6. Logs detalhados são gerados para auditoria e segurança.
7. O usuário pode revogar seu token JWT a qualquer momento para realizar logout seguro.
8. Tokens revogados são armazenados para prevenir reutilização indevida.

## **5. Formato da API**

## 5.1 Padrões de Desenvolvimento

A API seguirá o padrão REST, utilizando os métodos HTTP adequadamente (GET, POST, PUT, DELETE). Todos os endpoints protegidos exigirão token JWT válido. A comunicação será feita via JSON, e os códigos de status HTTP serão utilizados para indicar o resultado das operações. A documentação será disponibilizada no formato Swagger, facilitando testes e integração.

### 5.2 Exemplos de Endpoints

* X POST /auth/register - Cadastro de usuários.
* X POST /auth/login - Autenticação e geração de token JWT.
* POST /auth/reset-password - Recuperação de senha.
* X GET /users/{id} - Consulta de usuários (restrito por RBAC).
* X DELETE /users/{id} - Exclusão de conta conforme LGPD.
* X POST /auth/logout - Revogação do token JWT.
* X GET /logs/access - Consulta de logs de acessos.

## **6. Considerações Finais**

O presente documento define a estrutura inicial e os fundamentos técnicos do módulo de autenticação e gestão de usuários. Ao aplicar princípios de segurança da informação, conformidade legal e boas práticas de engenharia de software, o projeto visa criar um componente confiável, reutilizável e fácil de integrar em diferentes aplicações. A próxima etapa consiste na implementação do módulo com base nos requisitos aqui apresentados.