DOCUMENTAÇÃO TÉCNICA DO PROJETO – API AUDITORIA E CONTROLE INTERNO

Sistema de Auditoria e Controle Interno

Disciplina de Desenvolvimento de Software Visual

Grupo: Arthur Soares, Rebecca Beccari e João Bender
DUPLA: Apenas Arthur Soares

17/11/2025

1. Objetivo do Projeto

O objetivo deste projeto é desenvolver uma API voltada à gestão de controle interno organizacional, utilizando tecnologias de desenvolvimento web. A aplicação foi concebida para o funcionamento de um sistema de auditoria e controle interno, possibilitando operações como consulta, cadastro e exclusão de políticas, permissões, logs de acesso e trilhas de auditoria. Esse modelo é aplicável em contextos corporativos nos quais é necessário garantir rastreabilidade, segurança e conformidade nas operações realizadas pelos usuários responsáveis pelo controle organizacional.

2. Funcionalidades implementadas:

2.1 Modelagem de Dados

A modelagem da aplicação foi centrada em quatro entidades principais, definidas com base nos requisitos de um sistema de controle interno. As entidades Politica, Permissao, LogAcesso e TrilhaAuditoria possuem atributos selecionados para representar informações para identificação, rastreamento e controle de acesso. Essa modelagem foi transformada em classes na camada de domínio, servindo como base para a construção do banco de dados e definição das regras de negócio da aplicação.

2.2 Integração da API com o front-end

A API foi estruturada para atuar como a interface de comunicação entre o sistema e os dados armazenados. Cada rota foi implementada para representar uma funcionalidade específica da aplicação, como consulta geral, busca por ID, cadastro e exclusão de registros. Essas implementações estão presentes em todos os campos da interface WEB, como nas políticas, trilhas de auditoria, permissões e logs de acesso. Dentro do arquivo index.html, estão presentes funções que fazem a conexão dos pedidos da API com a interface gráfica do projeto.

2.3 Banco de dados auto populado

O banco de dados, caso ausente, será automaticamente populado pelo arquivo SeedData.cs, que adiciona dados fictícios para avaliação do sistema.

2.4 Verificação de erros e avisos

O projeto contém verificações de erros em todos seus campos, como obrigatoriedade de preenchimento de campos e escolha de usuário que fará a alteração. Além disso, ele adiciona automaticamente data e IP de origem do usuário a cada alteração, permitindo auditoria real dos dados que não podem ser falseados.

3. Descrição técnica das funcionalidades:
O projeto foi dividido em componentes com responsabilidades definidas:

Models/Politica.cs -- Define a estrutura da entidade de políticas de controle interno.

Models/Permissao.cs -- Define a estrutura da entidade de permissões de acesso.

Models/LogAcesso.cs -- Define a estrutura da entidade de logs deacesso.

Models/TrilhaAuditoria.cs -- Define a estrutura da entidade de trilhas de auditoria.

Data/ControleInternoContext.cs -- Responsável pela configuração da base de dados e integração com o Entity Framework.

Data/SeedData.cs -- Responsável pela população inicial do banco de dados.

Rotas/GetRoutes.cs -- Contém os métodos de listagem e consulta por ID.

Rotas/PostRoutes.cs -- Responsável pelo cadastro de novos registros.

Rotas/DeleteRoutes.cs -- Gerencia a remoção de registros existentes.

Program.cs -- Arquivo de inicialização da aplicação, onde são registrados os serviços e rotas.

AuditoriaService.cs – arquivo responsável para criação automática de trilhas de auditoria após o usuário executar alguma operação no sistema, como criação de uma nova política ou permissão.

wwwroot/index.html – Todo o front-end, contendo tanto o código HTML quanto a estilização feita com CSS. Além disso, ele também contém funções para permitir a conexão com a API, como **showTab** para alternar entre as abas da API; **showMessage** para exibir ao usuário mensagens de aviso, de erro e sucesso; **carregarUsuarios** para exibir a tabela de usuários cadastrados; **preencherSelectsUsuarios** que exibe um menu dropdown para exibir um funcionário especifico; **apiRequest** para traduzir JSON para exibição no front. Outras funções estão presentes no arquivo, sendo elas responsáveis, em geral, por exibir conteúdo de endpoints ou fazer alterações neles. Dessa forma, a interface gráfica está totalmente integrada a API, com todas suas funções e endpoints funcionais

4. Endpoints da API

A aplicação foi estruturada com rotas separadas para cada tipo de operação. Abaixo estão as rotas implementadas:

Método	Rota	Descrição
GET	/api/politicas	Retorna todas as políticas
GET	/api/politicas/{id}	Retorna uma política por ID
POST	/api/politicas	Adiciona uma nova política
DELETE	/api/politicas/{id}	Remove uma política por ID
GET	/api/permissoes	Retorna todas as permissões
GET	/api/permissoes/{id}	Retorna uma permissão por ID
POST	/api/permissoes	Adiciona uma nova permissão
DELETE	/api/permissoes/{id}	Remove uma permissão por ID
GET	/api/logsacesso	Retorna todos os logs de acesso
GET	/api/logsacesso/{id}	Retorna um log de acesso por ID
POST	/api/logsacesso	Adiciona um novo log de acesso
DELETE	/api/logsacesso/{id}	Remove um log de acesso por ID
GET	/api/trilhasauditoria	Retorna todas as trilhas de auditoria
GET	/api/trilhasauditoria/{id}	Retorna uma trilha de auditoria por ID
POST	/api/trilhasauditoria	Adiciona uma nova trilha de auditoria
DELETE	/api/trilhasauditoria/{id}	Remove uma trilha de auditoria por ID

5. Justificativa Técnica

A modelagem das quatro entidades principais foi feita com base nos atributos para rastreamento, controle de acesso e auditoria em ambientes organizacionais. As entidades Politica e Permissao representam as regras e níveis de acesso do sistema, enquanto LogAcesso e TrilhaAuditoria garantem rastreabilidade das ações realizadas. Essa modelagem orientou a arquitetura da aplicação, incluindo a configuração da base de dados, a definição das rotas e a estrutura do código.