INSTITUTO FEDERAL DE SANTA CATARINA

JOÃO PEDRO MENEGALI SALVAN BITENCOURT

Estrutura de Autenticação e Organização de Dados para o Sistema de Gerenciamento da Plataforma de Laboratório Remoto eLab

São José - SC

fevereiro/2026

ESTRUTURA DE AUTENTICAÇÃO E ORGANIZAÇÃO DE DADOS PARA O SISTEMA DE GERENCIAMENTO DA PLATAFORMA DE LABORATÓRIO REMOTO ELAB

Trabalho de conclusão de curso apresentado à Coordenadoria do Curso de Engenharia de Telecomunicações do campus São José do Instituto Federal de Santa Catarina.

Orientador: Prof. Ederson Torresini, Me.

São José - SC

fevereiro/2026

João Pedro Menegali Salvan Bitencourt

Estrutura de Autenticação e Organização de Dados para o Sistema de Gerenciamento da Plataforma de Laboratório Remoto eLab

Este trabalho foi julgado adequado para obtenção do título de Engenheiro de Telecomunicações, pelo Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, e aprovado na sua forma final pela comissão avaliadora abaixo indicada.

São José - SC, 05 de fevereiro de 2026:

gov.br
Documento assinado digitalmente
EDERSON TORRESINI
Data: 05/03/2026 23:26:36-0300
Verifique em https://validar.iti.gov.br

Prof. Ederson Torresini, Me.
Orientador
Instituto Federal de Santa Catarina

GOV.BR
Documento assinado digitalmente
CLEBER JORGE AMARAL
Data: 06/03/2026 15:56:05-0300
CPF: ***.848.639-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Cleber Jorge Amaral, Dr. Eng.
Instituto Federal de Santa Catarina

GOV.BR
Documento assinado digitalmente
MARCELO MAIA SOBRAL
Data: 09/03/2026 09:45:47-0300
CPF: ***.310.208-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Marcelo Maia Sobral, Dr. Eng.
Instituto Federal de Santa Catarina

RESUMO

O acesso a laboratórios físicos enfrenta limitações de horário, necessidade de deslocamento e custos de manutenção de equipamentos. Neste contexto, laboratórios remotos constituem uma alternativa para superar tais barreiras, permitindo que estudantes acessem recursos através da Internet. Este trabalho apresenta a modelagem do sistema de autenticação e autorização para a plataforma de laboratórios remotos eLab. A arquitetura proposta adota um modelo híbrido de persistência, combinando um banco de dados relacional para entidades estruturadas com um banco de dados não relacional orientado a documentos para sessões e registros de auditoria. O sistema implementa autenticação local e federada, utilizando a infraestrutura CAFe (Comunidade Acadêmica Federada) com protocolo SAML (*Security Assertion Markup Language*) 2.0, permitindo que usuários de instituições participantes acessem recursos compartilhados com credenciais de suas respectivas organizações. A estrutura incorpora o modelo RBAC (*Role-Based Access Control*), no qual permissões são gerenciadas através de grupos associados a papéis. O gerenciamento de sessões utiliza índices TTL (*Time To Live*) para expiração automática, eliminando a necessidade de processos externos para limpeza de dados.

Palavras-chave: Laboratórios remotos; Autenticação federada; Controle de acesso baseado em papéis; Banco de dados híbrido; Gerenciamento de sessões;

ABSTRACT

Access to physical laboratories faces scheduling limitations, commuting requirements, and equipment maintenance costs. In this context, remote laboratories constitute an alternative to overcome such barriers, enabling students to access resources through the Internet. This work presents the authentication and authorization system modeling for the eLab remote laboratory platform. The proposed architecture adopts a hybrid persistence model, combining a relational database for structured entities with a document-oriented non-relational database for sessions and audit logs. The system implements local and federated authentication using the CAFe (Comunidade Acadêmica Federada) infrastructure with SAML (Security Assertion Markup Language) 2.0 protocol, allowing users from participating institutions to access shared resources with their respective organizational credentials. The structure incorporates the RBAC (Role-Based Access Control) model, in which permissions are managed through groups associated with roles. Session management uses TTL (Time To Live) indexes for automatic expiration, eliminating the need for external cleanup processes.

Keywords: Remote laboratories; Federated authentication; Role-based access control; Hybrid database; Session management;

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

ABAC *Attribute-Based Access Control*.

ACID Atomicidade, Consistência, Isolamento e Durabilidade.

API *Application Programming Interface*.

ARBAC *Attribute and Role-Based Access Control*.

CAFe Comunidade Acadêmica Federada.

CAP *Consistency, Availability, Partition Tolerance*.

CAPES Coordenação de Aperfeiçoamento de Pessoal de Nível Superior.

CAS *Central Authentication Service*.

CI Circuito Integrado.

CPU *Central Processing Unit*.

CTEM Ciência, Tecnologia, Engenharia e Matemática.

DAC *Discretionary Access Control*.

DDL *Data Definition Language*.

DML *Data Manipulation Language*.

EER *Extended Entity-Relationship*.

ER Entidade-Relacionamento.

FDBMS *Federated Database Management System*.

FDBS *Federated Database System*.

FIM *Federated Identity Management*.

FPGA *Field Programmable Gate Array*.

GDPR *General Data Protection Regulation*.

GPIB *General Purpose Interface Bus*.

GT-MRE Grupo de Trabalho em Experimentação Remota.

HIPAA *Health Insurance Portability and Accountability Act*.

HTTP *Hypertext Transfer Protocol*.

HTTPS *Hypertext Transfer Protocol Secure*.

IAM *Identity and Access Management*.

IDE Ambiente Integrado de Desenvolvimento.

IdP *Identity Provider*.

IFSC Instituto Federal de Santa Catarina.

JSON *JavaScript Object Notation*.

LDAP *Lightweight Directory Access Protocol*.

MAC *Mandatory Access Control*.

MAS *Multi Agent System*.

NoSQL *Not Only SQL*.

OAuth *Open Authorization*.

OIDC *OpenID Connect*.

OLAP *Online Analytical Processing*.

PKI *Public Key Infrastructure*.

PMI *Privilege Management Infrastructure*.

RAG *Retrieval-Augmented Generation*.

RBAC *Role-Based Access Control*.

RELLE *Remote Labs Learning Environment*.

REST *Representational State Transfer*.

RExLab *Remote Experimentation Laboratory*.

RNP Rede Nacional de Ensino e Pesquisa.

SAML *Security Assertion Markup Language*.

SGBD Sistema de Gerenciamento de Banco de Dados.

SIRTFI *Security Incident Response Trust Framework for Federated Identity*.

SP *Service Provider*.

SQL *Structured Query Language*.

SSO *Single Sign-On*.

TGS *Ticket Granting Service*.

TTL *Time To Live*.

UFSC Universidade Federal de Santa Catarina.

USB *Universal Serial Bus*.

VISIR *Virtual Instrument Systems in Reality*.

WAYF *Where Are You From*.

XACML *eXtensible Access Control Markup Language*.

XML *eXtensible Markup Language*.

SUMÁRIO

1 INTRODUÇÃO

A formação de engenheiros requer o desenvolvimento de habilidades práticas que vão além do conhecimento teórico, demandando experiências com equipamentos reais que permitam a aplicação dos conceitos aprendidos em sala de aula (NENOV; EVSTATIEV; KADIROVA, 2023). No entanto, o acesso a laboratórios físicos tradicionais enfrenta diversos desafios, incluindo limitações de horário, necessidade de deslocamento físico, restrições de capacidade e custos elevados de manutenção e atualização de equipamentos (ARIZA; GALVIS, 2023). Essas barreiras tornam-se ainda mais significativas quando se considera a necessidade de equipamentos especializados, como os utilizados em cursos de engenharia eletrônica e de telecomunicações.

Neste contexto, os laboratórios remotos emergem como uma alternativa viável para superar essas limitações, permitindo que estudantes acessem equipamentos reais através da Internet, sem a necessidade de estar fisicamente presentes no laboratório (ORDUñA et al., 2016). A implementação de tais sistemas, entretanto, apresenta desafios técnicos significativos relacionados à arquitetura de *software*, gerenciamento de recursos, controle de acesso e armazenamento de dados (AITOR et al., 2022). Segundo Aitor et al. (2022), a necessidade de gerenciar múltiplos usuários, controlar filas de acesso, integrar diferentes componentes do sistema e manter informações sobre experimentos e configurações requer uma estrutura de dados robusta e bem projetada.

Como exemplo de equipamento que pode ser utilizado em experimentos remotos, destacam-se os dispositivos *Field Programmable Gate Array* (FPGA), introduzidos em meados dos anos 1980 (PEDRONI, 2010). Esses dispositivos são tipos de Circuito Integrado (CI) contendo blocos de lógica programáveis e interconexões configuráveis, permitindo a engenheiros de projetos realizar uma variedade de tarefas (MAXFIELD, 2004). Entretanto, *kits* de desenvolvimento para FPGA possuem alto custo, sendo necessário investimento significativo por parte do estudante ou da instituição de ensino (BEKASIEWICZ et al., 2021). Ademais, a instalação e manutenção dos Ambientes Integrados de Desenvolvimento (IDEs) utilizados na criação dos projetos e na programação dos mesmos são complexas e podem ser um obstáculo para estudantes iniciantes (LOBO, 2011). A gravação da lógica nesses dispositivos é realizada através de programas como Quartus Prime da Intel (Intel, 2024), Vivado Design Suite da AMD (AMD, 2024), além de outras opções mais genéricas como o GHDL (GHDL, 2017), Icarus Verilog (Stephen Williams, 2019), Matlab/Simulink (MathWorks) quando associado ao toolbox HDL Coder (MathWorks, 2024) e LabVIEW (National Instruments) (National Instruments, 2024).

A disponibilização desses recursos através de uma plataforma em nuvem permite que estudantes acessem tanto o dispositivo físico quanto os ambientes integrados de desenvolvi-

mento a partir de uma página web, acessível por qualquer dispositivo com acesso à Internet e um navegador, eliminando barreiras de compatibilidade e simplificando o processo de configuração do ambiente (FLORES et al., 2021). Além disso, com um sistema de controle de usuário capaz de definir diferentes níveis de acesso, o professor pode monitorar e auxiliar nas atividades de forma facilitada, além de poder propor projetos que outrora seriam inviabilizados pela falta de equipamentos ou *software* especializado (VERA et al., 2024). Ademais, o estudante não ficará limitado à disponibilidade presencial dos recursos, com a possibilidade de exercer seus estudos no horário que mais lhe convier (BAČA; HAULIš; ŠUBA, 2011).

A arquitetura de um sistema de gerência de laboratórios remotos é fundamental para o funcionamento adequado da plataforma, sendo responsável por coordenar diferentes componentes, como servidores de *hardware*, sistemas de mídia, ferramentas de desenvolvimento e mecanismos de autenticação (KALENDAR et al., 2023). Neste contexto, o projeto e a modelagem do banco de dados assumem papel central, pois é através dele que serão armazenadas informações críticas sobre usuários, permissões, configurações do sistema, histórico de experimentos e metadados dos laboratórios (CHACóN et al., 2025). Segundo Chacón et al. (2025), a escolha adequada da estrutura de dados e das tecnologias de armazenamento impacta diretamente na escalabilidade, desempenho e manutenibilidade do sistema como um todo.

Considerando que sistemas de laboratórios remotos frequentemente necessitam integrar múltiplas instituições e compartilhar recursos entre diferentes entidades, a implementação de um modelo de autenticação federado torna-se essencial (MAYOZ et al., 2020). Tal modelo permite que usuários de diferentes instituições acessem recursos compartilhados utilizando credenciais de suas respectivas organizações, facilitando a colaboração e o compartilhamento de infraestrutura educacional (VERA et al., 2024). Além disso, a natureza heterogênea dos dados gerados por esses sistemas, que podem incluir informações estruturadas (como dados de usuários e configurações), semiestruturadas (como registros de experimentos) e não estruturadas (como arquivos de projeto), sugere a necessidade de uma abordagem híbrida que combine bancos de dados relacionais e não relacionais (RAMZAN et al., 2019).

Diante desse cenário, identifica-se a necessidade de desenvolver uma estrutura de dados capaz de suportar a gerência de laboratórios remotos que ofereçam acesso a dispositivos reais, como, por exemplo, FPGAs e outros *kits* de desenvolvimento, por meio de uma interface web. Tal estrutura deve contemplar as demandas de um ambiente educacional distribuído, incluindo autenticação federada, escalabilidade e flexibilidade para armazenar diferentes tipos de dados.

1.1 Objetivo Geral

Como objetivo geral, propõe-se a modelagem de um sistema de autenticação e autorização para uma plataforma de laboratórios remotos distribuídos.

1.2 Objetivos Específicos

- Verificar técnicas, metodologias e protocolos para provimento de identidade externa;
- Modelar um sistema de laboratórios remotos distribuídos entre múltiplas instituições de ensino e pesquisa;
- Modelar um banco de dados capaz de armazenar as entidades relacionadas à autenticação e aos laboratórios remotos, considerando tanto entidades internas quanto externas.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são apresentados os conceitos fundamentais para o entendimento do trabalho proposto. A seção 2.1 discute laboratórios remotos, a seção 2.2 aborda os conceitos sobre banco de dados e a seção 2.3 trata do controle de acesso baseado em papéis.

2.1 Laboratórios Remotos

O desenvolvimento da educação na engenharia consiste em uma variedade de habilidades, conhecimentos e competências o qual o engenheiro graduado deve possuir (NENOV; EVSTATIEV; KADIROVA, 2023). Ainda que ferramentas modernas e gratuitas estejam disponíveis para a modelagem digital, no caso de estudantes iniciantes, ter o contato físico é mais recompensador e menos confuso (LOBO, 2011). Ariza e Galvis (2023) explicam que, durante as últimas duas décadas, os laboratórios remotos contribuíram para melhorar o processo de ensino-aprendizagem, principalmente na área de engenharia, e que vários fatores como a crescente quantidade de estudantes em sala, a necessidade de prover ferramentas de alta qualidade tecnológica para aprendizado e experimento e a redução de recursos econômicos destinados à infraestrutura física, levaram a esta situação. Ariza e Galvis (2023) ainda mencionam que várias arquiteturas de laboratórios remotos foram criadas com base nos exemplos bem sucedidos, como o laboratório remoto WebLab-Deusto LabsLand, ou o *Virtual Instrument Systems in Reality* (VISIR), resultado da cooperação de várias universidades. Sobre este último, Alves et al. (2023) afirmam o VISIR é muito mais do que apenas uma ferramenta educacional aprimorada por tecnologia, sendo que, primeiramente, é um laboratório remoto, portanto, um sistema físico com vários tipos de especificações e características operacionais, que podem ser analisadas no escopo da visão técnica. Ainda de acordo com Alves et al. (2023), o VISIR tem sido utilizado como uma ferramenta pedagógica com diferentes abordagens didáticas, abrindo a análise para o escopo da visão pedagógica, de tal forma que agregou uma comunidade de pesquisa considerável, contribuindo para a disseminação contínua de novas metodologias e práticas educacionais, mudando políticas e formadores de opinião.

Segundo Alves et al. (2023), a situação causada pela pandemia de COVID-19, com vários acionistas e formadores de opinião discutindo a necessidade de aprimorar a educação online, evidenciou um interesse sem precedentes no uso de laboratórios remotos para apoiar aulas práticas em cursos relacionados a Ciência, Tecnologia, Engenharia e Matemática (CTEM).

Um Laboratório Remoto Educacional é uma solução de *software* e *hardware* que permite aos estudantes acessar equipamentos reais localizados em sua instituição, como se estivessem em uma sessão de laboratório prática, usando um navegador web padrão (ORDUñA et

al., 2016). Orduña et al. (2016) ainda comenta que um fator chave dos laboratórios remotos é que, uma vez que estão disponíveis através da Internet, seu uso pode ser ampliado e utilizado por estudantes de outras instituições, podendo compartilhar diferentes equipamentos para reduzir custos, exigindo menos equipamentos duplicados, já que estes são tipicamente usados apenas em certas horas do dia e em certos dias do ano. Benmohamed, Leleve e Prevot (2004) explicam que cada laboratório remoto é um conjunto de salas que o usuário acessa através de redes públicas ou privadas. Quando bem planejado e aplicado, os laboratórios remotos podem oferecer resultados similares ou superiores aos laboratórios tradicionais (AITOR et al., 2022)

Aitor et al. (2022) explica que os sistemas de gerência de laboratórios remotos fornecem componentes como gestão de autenticação e filas de usuários e grupos, integração de laboratórios com Sistemas de Gestão de Aprendizagem, compartilhamento de laboratórios entre diferentes entidades por meio de federação, extração de dados e análises de aprendizado, além de fornecerem outras ferramentas para facilitar o desenvolvimento de laboratórios. Ainda segundo Aitor et al. (2022), os desenvolvedores de laboratórios remotos precisam se concentrar apenas nos componentes específicos do seu laboratório remoto, economizando milhares de horas de desenvolvimento.

2.1.1 Plataformas existentes

Uma das plataformas voltadas para laboratórios remotos relacionados à CTEM é a LabsLand, que é uma rede global de laboratórios remotos, que conecta instituições a equipamentos localizados mundialmente, sendo que os equipamentos são reais e não simulações, e podem serem acessados através de um navegador web (LabsLand, 2024). Ainda segundo LabsLand (2024), um laboratório real pode ser um pequeno robô movido a arduino na Espanha, um laboratório de cinemática no Brasil ou um laboratório de testes de radioatividade na Austrália, sendo verdadeiros laboratórios, não simulações: os laboratórios estão lá fisicamente e os estudantes dessas escolas e universidades os acessam.

Ainda explicado por LabsLand (2024), há os laboratórios em tempo real, em que os alunos acessam o equipamento ao mesmo tempo em que as coisas estão acontecendo, e, por exemplo, no caso do robô, os alunos enviam um programa para um robô e eles podem ver como o robô se comporta com o programa em tempo real remotamente, e também há laboratórios ultraconcorrentes, os quais são baseados em um conjunto de experiências pré-gravadas realizadas em um laboratório real e a interface permite ao usuário ter a mesma experiência de um laboratório em tempo real. Sobre esta última modalidade oferecida, Aitor et al. (2022) explicam que por meio desse conjunto de dados, um laboratório ultra-concorrente pode proporcionar uma experiência de laboratório que se assemelha muito ao que o estudante teria com a versão tradicional prática e que estes laboratórios não são simplesmente vídeos; eles são interativos: permitem que o estudante assuma um papel ativo e faça escolhas.

Outra solução é o VISIR+, que é um sistema desenvolvido para realizar experimentos

de eletricidade e eletrônica diretamente a partir do navegador (IFSC, 2024). Ainda de acordo com IFSC (2024), na plataforma, o usuário terá acesso a equipamentos de instrumentação, tais como um osciloscópio, um multímetro, um gerador de funções e uma fonte de alimentação, sendo que, com estes equipamentos e um conjunto de componentes eletrônicos será possível implementar diversos circuitos em uma matriz de contatos. IFSC (2024) ainda menciona que os circuitos são montados com componentes reais e as medidas efetuadas são adquiridas e transmitidas via Internet.

Na Universidade Federal de Santa Catarina (UFSC), há o *Remote Experimentation Laboratory* (RExLab), que surgiu em 1997 e conta com uma rede de doze universidades em cinco países diferentes (UFSC, 2024). Ainda segundo UFSC (2024), um dos objetivos é atender a necessidade de apropriação social da ciência e da tecnologia, popularizando conhecimentos científicos e tecnológicos, estimulando os jovens a inserirem-se nas carreiras científico-tecnológicas e buscar iniciativas que integrem a educação científica ao processo educacional promovendo a melhoria devido à atualização/modernização do ensino em todos os seus níveis, enfatizando ações e atividades que valorizem e estimulem a criatividade, a experimentação e a interdisciplinaridade.

Também provido pela UFSC, há o *Remote Labs Learning Environment* (RELLE), que é um ambiente que permite a manipulação e o gerenciamento de experimentos remotos, sendo desenvolvido pelo Grupo de Trabalho em Experimentação Remota (GT-MRE) do RExLab (RELLE, 2024). RELLE (2024) também explica que todos os experimentos disponíveis estão abertos para uso de qualquer usuário, além da possibilidade deste criar o próprio experimento e disponibilizá-lo na plataforma, desde que as especificações do sistema sejam respeitadas.

Localizado na Australia, o NetLab é um laboratório remoto interativo e multiusuário, que possui um servidor dedicado conectado à Internet, o qual permite que os usuários acessem a plataforma (NEDIC; MACHOTKA; NAFALSKI, 2008). Nedic, Machotka e Nafalski (2008) ainda explicam que o servidor comunica-se com instrumentos de laboratório programáveis através do barramento *General Purpose Interface Bus* (GPIB), que controla equipamentos como osciloscópios digitais, geradores de função e multímetros digitais. Nedic, Machotka e Nafalski (2008) também mencionam que todos esses instrumentos são conectados a um chaveador de matriz programável com dimensão de 16x16, cujo qual permite ao usuário criar vários circuitos a partir do componentes disponíveis, como resistores, capacitores, indutores e transformadores. Outra característica mencionada por Nedic, Machotka e Nafalski (2008), é que a plataforma possui uma câmera que pode ser controlada pelo usuário e que o sistema limita a quantidade de usuários concorrente para evitar conflitos, sendo que mais de um usuário pode interagir no mesmo experimento simultaneamente.

2.2 Banco de Dados

2.2.1 Bancos de Dados Relacionais

Um banco de dados pode ser definido como uma coleção de dados que, tipicamente, descreve as atividades de uma ou mais organizações relacionadas (RAMAKRISHNAN; GEHRKE, 2008). No contexto dos bancos de dados relacionais, segundo Heuser (2009), a estrutura fundamental é composta por tabelas ou relações, sendo que a terminologia *tabela* é mais comum nos produtos comerciais e na prática, enquanto que a terminologia *relação* foi utilizada na literatura original sobre a abordagem relacional e é mais comum na área acadêmica e nos livros-texto.

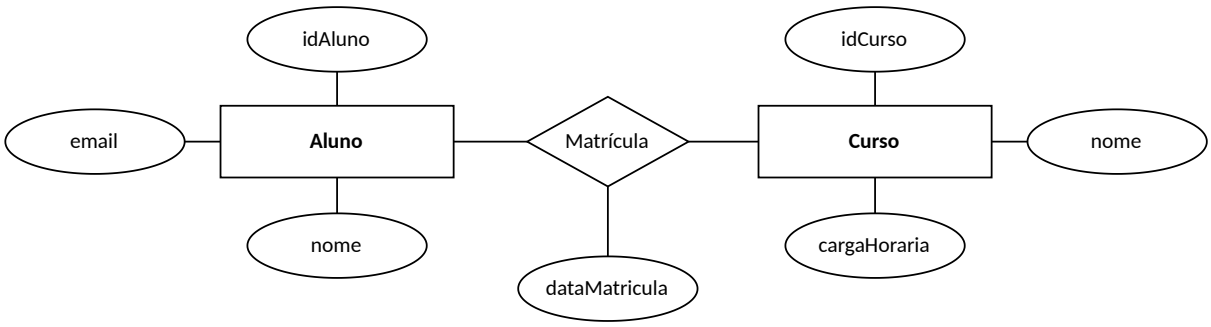
O modelo relacional organiza os dados em estruturas bidimensionais denominadas relações, onde cada relação representa uma entidade do mundo real (ELMASRI; NAVATHE, 2010). Conforme Silberschatz, Korth e Sudarshan (2006), cada relação é composta por um conjunto de atributos que descrevem as propriedades das entidades, e cada linha da relação, denominada tupla, representa uma instância específica da entidade. A estrutura tabular dos bancos de dados relacionais, organizada em tabelas com linhas e colunas, facilita a representação de relacionamentos entre diferentes entidades através de mecanismos de integridade referencial estabelecidos por chaves primárias e estrangeiras (ELMASRI; NAVATHE, 2010).

Adicionalmente, Heuser (2009) explica que em um banco de dados relacional, o conceito de chave é fundamental para estabelecer relações entre as linhas de tabelas, sendo que há três tipos principais de chaves a serem consideradas:

- Chave primária: é uma coluna ou uma combinação de colunas cujos valores distinguem uma linha das demais dentro de uma tabela;
- Chave estrangeira: é uma coluna ou uma combinação de colunas cujos valores aparecem necessariamente na chave primária de uma tabela, sendo este tipo de chave um mecanismo que permite a implementação de relacionamentos em um banco de dados relacional;
- Chave alternativa: ocorre quando mais de uma coluna ou combinações de colunas podem servir para distinguir uma linha das demais, de forma que uma das colunas, ou a combinação de colunas, é escolhida como chave primária e as demais são consideradas chaves alternativas.

A Figura 1 apresenta um exemplo de diagrama Entidade-Relacionamento (ER) que ilustra a estrutura de um banco de dados relacional simples, demonstrando como entidades são representadas e como os relacionamentos entre elas são estabelecidos através de chaves primárias e estrangeiras.

Figura 1 – Exemplo de diagrama entidade-relacionamento.



Fonte: Elaborada pelo autor.

No diagrama ER, cada forma geométrica possui significado próprio. Conforme Heuser (2009), a entidade é representada por um retângulo que contém o nome da mesma. Ainda conforme Heuser (2009), o relacionamento é representado por um losango, ligado por linhas aos retângulos das entidades que participam do relacionamento. Por sua vez, o atributo é representado por uma elipse (RAMAKRISHNAN; GEHRKE, 2008). Também é explicado por Heuser (2009) que relacionamentos podem possuir atributos. Na Figura 1, em conformidade com essa notação, identificam-se as entidades Aluno, com os atributos idAluno, nome e email, e Curso, com os atributos idCurso, nome e cargaHoraria, respectivamente. O losango corresponde ao relacionamento Matrícula, que as associa e uma elipse indica o atributo dataMatricula vinculado a esse relacionamento.

Os bancos de dados relacionais têm sido amplamente utilizados por mais de quatro décadas como modelo predominante para armazenamento, recuperação e gerenciamento de dados (HASSAN, 2021). Ainda segundo Hassan (2021), os sistemas de gerenciamento de banco de dados relacional oferecem garantias de integridade de dados através de propriedades Atomicidade, Consistência, Isolamento e Durabilidade (ACID), além de fornecerem uma linguagem padronizada, a *Structured Query Language* (SQL), para manipulação e consulta de dados. Pokorny (2013) complementa que, embora as propriedades ACID sejam características fundamentais dos bancos de dados relacionais, a implementação completa das mesmas depende de cada solução específica, sendo que algumas aplicações podem requerer transações ACID completas, enquanto outras podem operar com níveis reduzidos de consistência em favor de maior disponibilidade e escalabilidade.

Quanto ao desempenho e às limitações para operar em grandes escalas, os bancos de dados relacionais apresentam características específicas quando submetidos a diferentes cargas de trabalho (CAPRIS et al., 2022). Ainda de acordo com Capris et al. (2022), bancos de dados relacionais como MySQL são adequados para aplicações que requerem transações ACID e estrutura de dados bem definida. A análise comparativa de desempenho entre bancos de dados relacionais e bancos de dados chave-valor demonstra que o desempenho pode variar significativamente dependendo da carga de trabalho aplicada (ALMEIDA et al., 2023). Em con-

trapartida, conforme Chickerur, Goudar e Kinnerkar (2015), os sistemas de gerenciamento de banco de dados relacional convencionais apresentam dificuldades de escalabilidade, não sendo capazes de lidar com crescimento exponencial de dados. Ainda conforme Hassan (2021), os bancos de dados relacionais podem apresentar limitações de desempenho em cenários que requerem processamento de grandes volumes de dados não estruturados ou operações de alta concorrência em larga escala.

2.2.2 Sistemas de Gerenciamento de Banco de Dados

Conforme explicado por Ramakrishnan e Gehrke (2008), um Sistema de Gerenciamento de Banco de Dados (SGBD) é um *software* projetado para auxiliar a manutenção e utilização de vastos conjuntos de dados. Elmasri e Navathe (2010) adicionam que esse sistema facilita o processo de definição, construção, manipulação e compartilhamento de dados entre diversos usuários e aplicações. Cada SGBD necessita de uma linguagem de definição de dados, sendo o termo em inglês *Data Definition Language* (DDL) (ELMASRI; NAVATHE, 2010). Segundo Silberschatz, Korth e Sudarshan (2006), a DDL é utilizada para especificar um esquema, que indicará quais as propriedades adicionais dos dados que serão armazenados e as restrições que devem ser aplicadas a esses dados. O sistema verifica essas restrições toda vez que ocorre uma atualização (SILBERSCHATZ; KORTH; SUDARSHAN, 2006). Além disso, conforme Elmasri e Navathe (2010), o SGBD também oferece um conjunto de operações ou uma linguagem chamada linguagem de manipulação de dados, sendo o termo em inglês *Data Manipulation Language* (DML), que permite realizar operações de recuperação, inserção, exclusão e modificação dos dados. Dessa forma, as linguagens de definição e manipulação de dados não são separadas, mas simplesmente formam partes de uma única linguagem, como a amplamente utilizada SQL (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

2.2.3 Transações e Propriedades ACID

Uma transação é uma execução qualquer de um programa de usuário em um SGBD (RAMAKRISHNAN; GEHRKE, 2008). Segundo Elmasri e Navathe (2010), uma transação executa um acesso logicamente correto a um banco de dados quando ela é executada de forma completa e sem interferência de outras transações. Silberschatz, Korth e Sudarshan (2006) afirma que para garantir a integridade dos dados, é necessário que o sistema de banco de dados tenha as propriedades ACID, que é composta por:

- Atomicidade: todas as operações da transação são refletidas corretamente no banco de dados ou nenhuma delas é refletida;
- Consistência: uma transação isolada preserva a consistência do banco de dados;

- Isolamento: o sistema garante que a execução de uma transação não interfira na execução de outra;
- Durabilidade: garante a persistência dos dados no sistema após a conclusão de uma transação, mesmo que existam falhas no sistema.

2.2.4 Bancos de Dados NoSQL

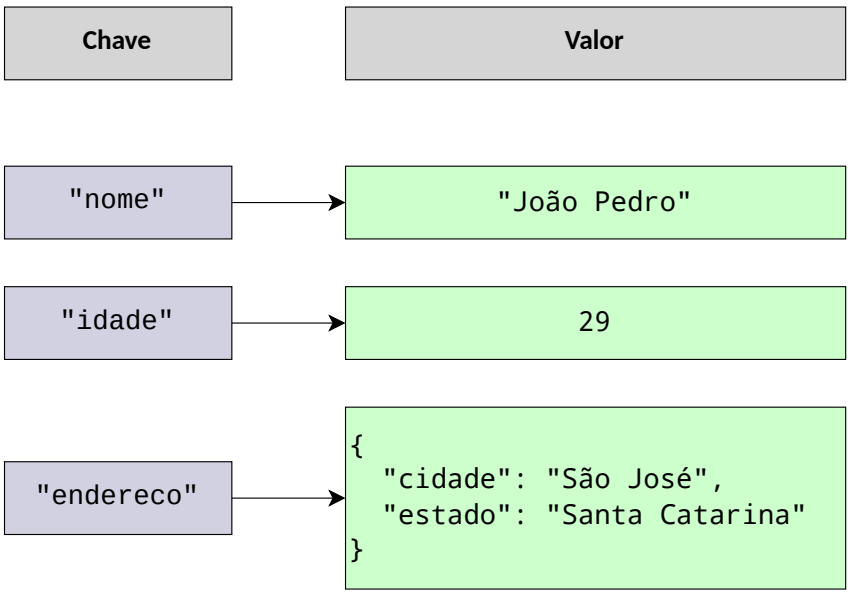
Embora os bancos de dados relacionais tenham sido utilizados confortavelmente por muitos anos, já que a quantidade de entrada de dados era pequena e de forma estruturada, com o surgimento da era *big data*, a complexidade para lidar com a quantidade e velocidade crescente na variedade de dados que não estão estruturados ficou maior (RAMZAN et al., 2019). Neste contexto, Ramzan et al. (2019) ainda afirmam que bases de dados *Not Only SQL* (NoSQL) são uma escolha melhor por possuírem vantagens como lidar com vários tipos de dados, sejam estes estruturados, semiestruturados ou não estruturados, além de serem flexíveis, distribuídos e escaláveis.

O termo NoSQL, que significa ”Not Only SQL”(Não Apenas SQL), refere-se a uma classe de sistemas de gerenciamento de banco de dados que se afastam do modelo relacional tradicional (HASSAN, 2021). Segundo Hassan (2021), os bancos de dados NoSQL são caracterizados por serem não relacionais, de código aberto, distribuídos, horizontalmente escaláveis, sem esquema fixo, com suporte a replicação e com *Application Programming Interface* (API) simples. Kaur, Singla e Khawas (2023) complementam que os bancos de dados NoSQL são projetados para lidar com grandes volumes de dados não estruturados e semiestruturados, oferecendo escalabilidade horizontal através da distribuição de dados em múltiplos servidores.

Em relação à escalabilidade, os bancos de dados NoSQL são projetados para escalabilidade horizontal (*scale-out*), permitindo a distribuição de dados e carga de trabalho através de múltiplos servidores (POKORNY, 2013). Ainda é explicado por Pokorny (2013) que, diferentemente dos bancos de dados relacionais que tradicionalmente utilizam escalabilidade vertical (*scale-up*), os bancos de dados NoSQL podem adicionar novos nós ao *cluster* para aumentar a capacidade de processamento e armazenamento. Chickerur, Goudar e Kinnerkar (2015) destacam que essa capacidade de escalabilidade horizontal é uma das principais vantagens dos bancos de dados NoSQL para aplicações de *big data*.

Uma das características fundamentais dos bancos de dados NoSQL é a flexibilidade de esquema, também conhecida como esquema-livre (*schema-less*) ou esquema flexível (CHICKERUR; GOUDAR; KINNERKAR, 2015). Ainda segundo Chickerur, Goudar e Kinnerkar (2015), essa característica permite que os dados sejam armazenados sem a necessidade de um esquema pré-definido, facilitando o desenvolvimento ágil e a adaptação a mudanças nos requisitos de dados. Lee, Tang e Choi (2013) acrescentam que o armazenamento no formato chave-valor é uma técnica comum em bancos de dados NoSQL sem esquema e que proporciona flexibilidade.

Figura 2 – Exemplo de estrutura de armazenamento chave-valor.



Fonte: Elaborada pelo autor.

Os bancos de dados NoSQL podem ser classificados em diferentes modelos de dados, sendo os principais: armazenamento chave-valor (*key-value store*), orientado a documentos (*document-oriented*), orientado a colunas (*column-oriented* ou *wide columnar*) e orientado a grafos (*graph-oriented*) (HASSAN, 2021; AGGOUNE; NAMOUNE, 2020).

2.2.4.1 Armazenamento Chave-Valor

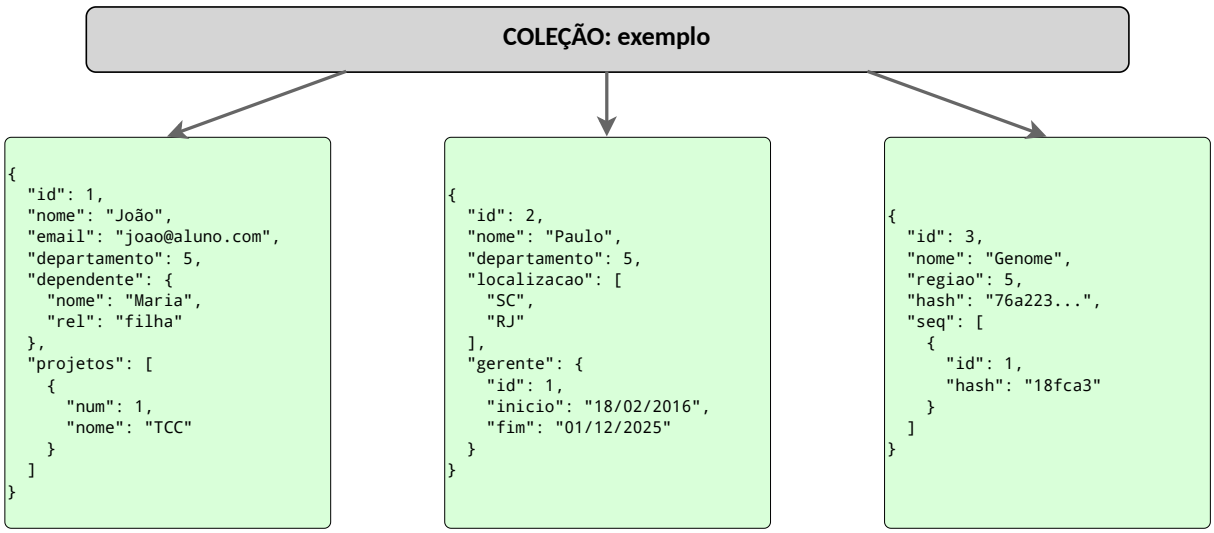
No modelo chave-valor, os dados são armazenados como pares chave-valor únicos, onde cada chave é associada a um valor que pode ser uma *string*, número ou objeto complexo (HASSAN, 2021). Este modelo é adequado para aplicações que requerem acesso rápido a dados simples e é amplamente utilizado em sistemas de cache e armazenamento de sessões (KAUR; SINGLA; KHAWAS, 2023).

A Figura 2 demonstra três exemplos de dados armazenados utilizando a estrutura de chave-valor: uma chave com valor do tipo *string*, outra com valor numérico inteiro, e uma terceira cujo valor é um objeto aninhado contendo múltiplos pares chave-valor, evidenciando a flexibilidade em armazenar dados estruturados e não estruturados (RAMZAN et al., 2019).

2.2.4.2 Orientado a Documentos

Já no modelo orientado a documentos, os dados são armazenados em formato de documentos, tipicamente em JSON ou XML, permitindo estruturas hierárquicas e aninhadas (CHICKERUR; GOUDAR; KINNERKAR, 2015). Ainda segundo Chickerur, Goudar e Kinnerkar (2015), esta estruturação é particularmente adequada para aplicações de *big data*, pois permite armazenar dados complexos e semiestruturados de forma eficiente. Sharma e Kaur (2021) mencionam que a natureza sem esquema dos bancos de dados orientados a documentos, como

Figura 3 – Exemplo de estrutura de banco de dados orientado a documentos em formato JSON.



Fonte: Elaborada pelo autor.

o MongoDB, permite que diferentes documentos na mesma coleção tenham estruturas variadas, o que é especialmente útil em sistemas *multitenancy*, onde diferentes clientes podem ter requisitos de dados distintos.

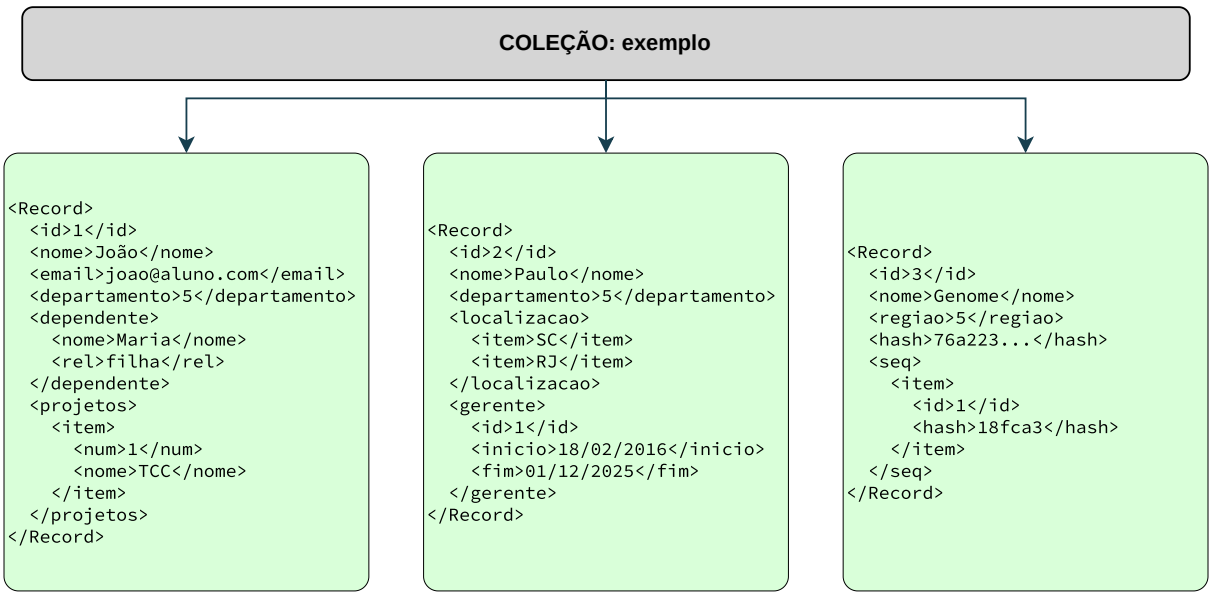
A Figura 3 apresenta três documentos em formato JSON armazenados em uma coleção, onde cada documento possui estrutura diferente, com chaves e valores diferentes. O formato JSON é utilizado em bancos de dados orientados a documentos com esquema dinâmico, permitindo que diferentes documentos na mesma coleção tenham diferentes números de campos (HASSAN, 2021). Segundo Hamouda (2023), cada documento contém dados no modelo chave-valor, onde os valores podem ser de qualquer tipo de dado, incluindo objetos aninhados e arranjos.

Já a Figura 4 apresenta os mesmos três documentos da Figura 3 em formato XML, que nativamente suporta armazenamento de dados hierarquicamente através de elementos aninhados (KARNITIS; ARNICANS, 2015). Conforme AGGOUNE e NAMOUNE (2020), esse tipo de linguagem de marcação foi projetada para codificar documentos eletronicamente e representar dados de forma estruturada, sendo utilizada como formato padrão para encapsular documentos em bancos de dados orientados a documentos.

2.2.4.3 Orientado a Colunas

O modelo orientado a colunas, também denominado *wide columnar* ou *column-store*, diferencia-se fundamentalmente dos sistemas orientados a linhas pela forma como organiza fisicamente os dados (KANADE; GOPAL, 2013). Enquanto bancos de dados relacionais tradicionais armazenam registros completos sequencialmente, os sistemas orientados a colunas particionam verticalmente os dados, armazenando cada atributo de forma separada e contígua

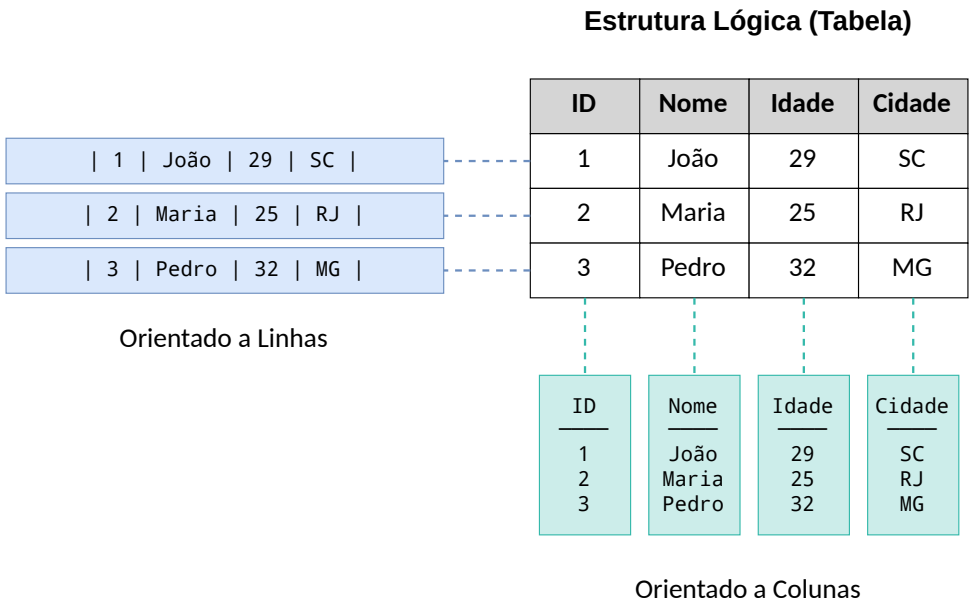
Figura 4 – Exemplo de estrutura de banco de dados orientado a documentos em formato XML.



Fonte: Elaborada pelo autor.

(QIYUE, 2015). A Figura 5 apresenta uma comparação visual entre essas duas abordagens de armazenamento.

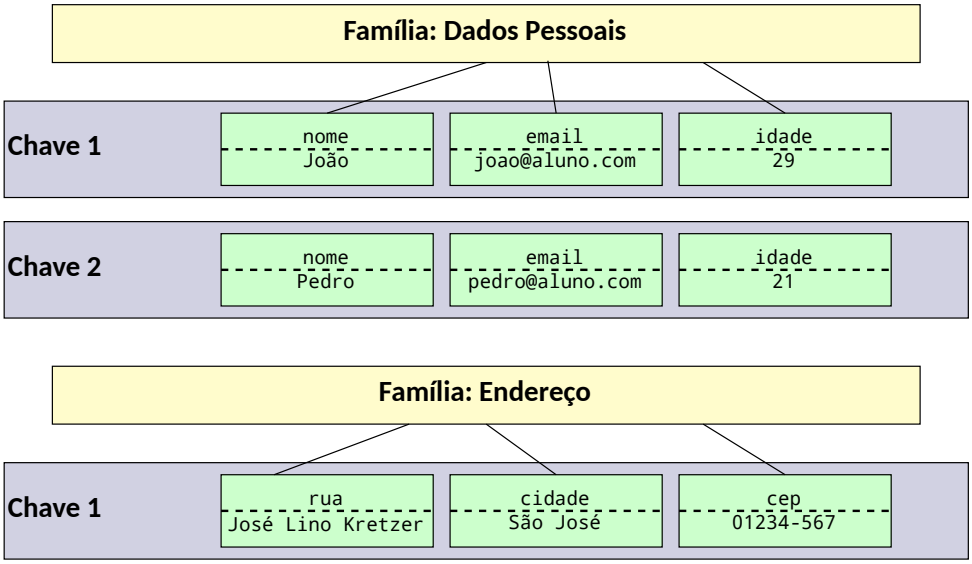
Figura 5 – Comparação da estrutura física de armazenamento: orientado a linhas vs orientado a colunas.



Fonte: Elaborada pelo autor.

A organização física por colunas oferece vantagens significativas para consultas analíticas que processam grandes volumes de dados (QIYUE, 2015). Ainda segundo Qiyue (2015), como apenas os atributos necessários são lidos do disco, em vez de linhas completas, reduz-se

Figura 6 – Exemplo que ilustra a estrutura lógica de um banco de dados orientado a colunas NoSQL.



Fonte: Elaborada pelo autor.

o tráfego de I/O e melhora-se a utilização da largura de banda de memória ao transferir dados para os registradores da *Central Processing Unit* (CPU). Além disso, valores de uma mesma coluna apresentam maior localidade de dados, o que favorece técnicas de compressão mais eficientes (KANADE; GOPAL, 2013). Por conseguinte, ainda segundo Kanade e Gopal (2013), esse modelo é especialmente adequado para aplicações que escaneiam grandes frações de tabelas e calculam agregações ou estatísticas, como sistemas de *data warehousing*, processamento analítico online (*Online Analytical Processing* (OLAP)) e aplicações de inteligência de negócios.

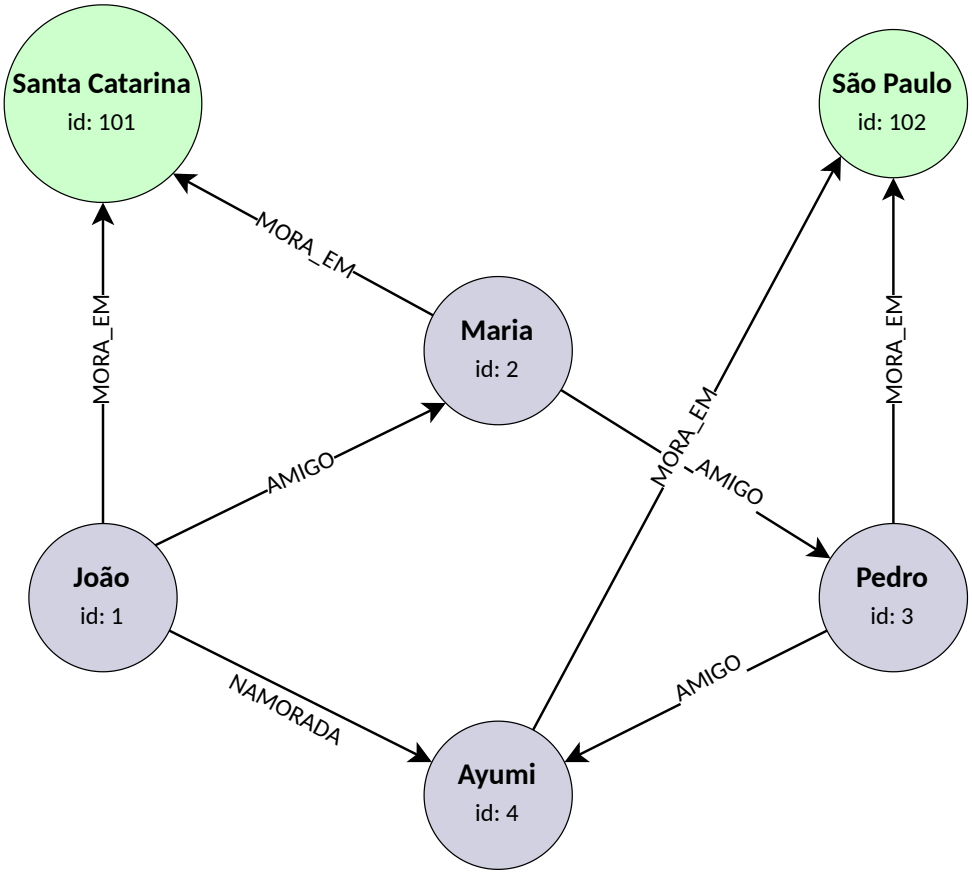
Em bancos de dados orientados a colunas do tipo NoSQL, a estrutura lógica organiza-se em famílias de colunas, onde cada família contém qualificadores de coluna e seus respectivos valores no formato ”Família: Qualificador = Valor”(CHEN; LEE, 2020). A estrutura inclui uma chave de linha (*Row Key*) que identifica unicamente cada registro, funcionando de forma análoga à chave primária em bancos de dados relacionais (SIREGAR; AZIZAH, 2023). Nessa perspectiva, a modelagem de dados requer considerações específicas sobre a distribuição dos atributos entre as famílias de colunas, sendo que metodologias de modelagem consideram elementos do diagrama entidade-relacionamento estendido (*Extended Entity-Relationship* (EER)) para definir a estrutura física (PRAKOSO; AZIZAH, 2023).

A migração de bancos de dados relacionais para sistemas orientados a colunas NoSQL envolve processos de conversão de esquema e transformação de dados, onde tabelas relacionais são convertidas em famílias de colunas (SIREGAR; AZIZAH, 2023). Na Figura 6 é mostrado um exemplo que ilustra a estrutura lógica de um banco de dados orientado a colunas NoSQL, mostrando as chaves de linha e as famílias de colunas.

2.2.4.4 Orientado a Grafos

O modelo orientado a grafos representa os dados como nós e arestas, onde os nós representam entidades e as arestas representam relacionamentos entre essas entidades (HASSAN, 2021). Este modelo é adequado para aplicações que lidam com dados altamente interconectados, como redes sociais, sistemas de recomendação e análise de relacionamentos complexos (KAUR; SINGLA; KHAWAS, 2023). A Figura 7 apresenta um exemplo que ilustra a estrutura de um banco de dados orientado a grafos, indicando o relacionamento entre os nós de pessoas e os estados onde residem.

Figura 7 – Exemplo que ilustra a estrutura de um banco de dados orientado a grafos.



Fonte: Elaborada pelo autor.

2.2.4.5 Vantagens e limitações

Os bancos de dados NoSQL oferecem escalabilidade horizontal e flexibilidade de esquema, sendo adequados para aplicações que lidam com grandes volumes de dados não estruturados (SAMARTA; GUNAWAN; SYAHPUTRA, 2024). Ainda segundo Samarta, Gunawan e Syahputra (2024), esses sistemas priorizam disponibilidade e escalabilidade sobre consistência forte, frequentemente adotando modelos de consistência eventual em detrimento das propriedades ACID. Krishan, Gupta e Bhathal (2024) explicam que o teorema *Consistency, Availability, Partition Tolerance* (CAP) fundamenta as compensações realizadas pelos bancos de dados

NoSQL, onde geralmente se prioriza disponibilidade e tolerância a partições. Em contrapartida, Sakib et al. (2025) destacam que bancos orientados a documentos apresentam limitações em suporte a transações ACID, sendo que alguns sistemas oferecem apenas suporte limitado a transações multi-documento. De forma complementar, Matholia e Adedayo (2025) identificam vulnerabilidades de segurança, como ataques de injeção NoSQL, que podem permitir acesso não autorizado e manipulação de dados quando não há validação adequada de entrada. Por conseguinte, Samarta, Gunawan e Syahputra (2024) mencionam que a falta de padronização nas linguagens de consulta e a complexidade na programação manual de consultas constituem dificuldades adicionais para os bancos de dados NoSQL.

2.2.5 Bancos de Dados Federados

À medida que sistemas de informação crescem em escala e complexidade, surge a necessidade de integrar fontes de dados distribuídas e heterogêneas, mantendo a autonomia de cada componente (SHETH; LARSON, 1990). Um Sistema de Banco de Dados Federado (*Federated Database System* (FDBS)) é um conjunto de sistemas de bancos de dados cooperativos e autônomos que participam de uma federação para permitir o compartilhamento parcial e controlado de seus dados (PINO; RAVIDá; SCIBILIA, 2013). Conforme Sheth e Larson (1990), o Sistema Gerenciador de Bancos de Dados Federados (*Federated Database Management System* (FDBMS)) é o *software* responsável por gerenciar essa federação, fornecendo acesso transparente e controlado aos múltiplos bancos de dados componentes.

Sob esse viés, Dharmasiri e Goonetillake (2013) complementam que a federação de bancos de dados possibilita o acesso unificado a múltiplas fontes de dados sem a necessidade de centralização física dos mesmos. Em contrapartida, Pino, Ravidá e Scibilia (2013) ressaltam que sistemas de bancos de dados federados diferem de sistemas distribuídos tradicionais por não assumirem homogeneidade entre os componentes, permitindo que cada fonte de dados mantenha sua autonomia operacional. Nesse sentido, Trinquet et al. (2025) demonstram que a federação de bancos de dados viabiliza consultas abrangentes em conjuntos de dados descentralizados, como no caso de pesquisas científicas que requerem acesso a repositórios mantidos por diferentes instituições.

2.2.5.1 Características Fundamentais

Os sistemas de bancos de dados podem ser caracterizados segundo três dimensões principais: distribuição, heterogeneidade e autonomia (SHETH; LARSON, 1990). A distribuição refere-se à dispersão física dos dados em múltiplos locais conectados por uma rede de computadores (PINO; RAVIDá; SCIBILIA, 2013). Segundo Sheth e Larson (1990), a heterogeneidade manifesta-se em diferentes níveis, desde diferenças nos SGBDs utilizados até divergências semânticas na representação dos dados. Por conseguinte, Dharmasiri e Goonetillake (2013) observam que sistemas NoSQL amplificam essa heterogeneidade, visto que diferentes modelos

de dados, como orientado a documentos, orientado a colunas e chave-valor, apresentam linguagens de consulta e estruturas distintas.

A autonomia constitui a terceira dimensão e subdivide-se em autonomia de projeto, comunicação, execução e associação (SHETH; LARSON, 1990). Ela permite que cada banco de dados componente seja desenvolvido independentemente dos demais, escolhendo seu próprio modelo de dados, linguagem de consulta e restrições de integridade (PINO; RAVIDá; SCIBILIA, 2013). Conforme Sheth e Larson (1990), a autonomia de comunicação concede ao sistema componente a decisão de quando e como responder a requisições externas. Adicionalmente, a autonomia de execução garante que cada componente controle a ordem de execução de operações locais e externas, enquanto a autonomia de associação permite que sistemas decidam sobre sua participação na federação (SHETH; LARSON, 1990).

2.2.5.2 Arquitetura de Referência

A arquitetura de referência para sistemas de bancos de dados federados fundamenta-se em uma estrutura de cinco níveis de esquemas (SHETH; LARSON, 1990). O esquema local representa a estrutura de dados conforme definida pelo SGBD componente em seu modelo nativo (DHARMASIRI; GOONETILLAKE, 2013). Em seguida, o esquema componente traduz o esquema local para um modelo de dados comum, denominado Modelo Canônico de Dados (*Canonical Data Model* – CDM), eliminando diferenças de representação entre os sistemas participantes (SHETH; LARSON, 1990).

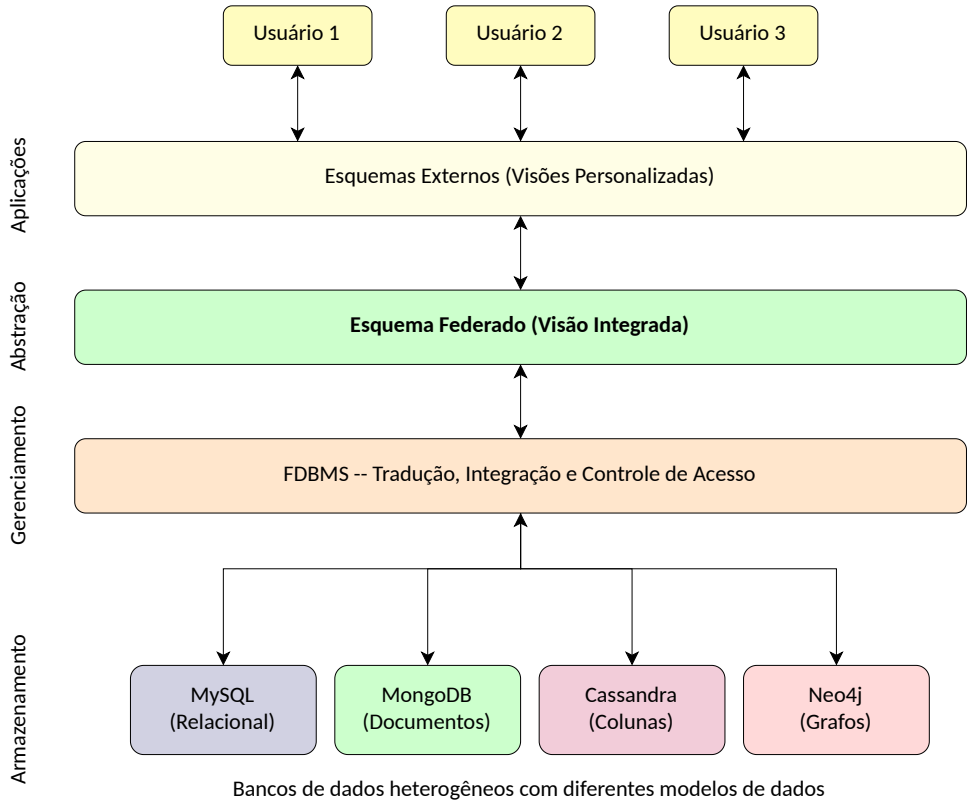
O esquema de exportação define o subconjunto de dados que cada componente disponibiliza para a federação, implementando controle de acesso e restrições de compartilhamento (SHETH; LARSON, 1990). Conforme Pino, Ravidá e Scibilia (2013), esse nível permite que administradores locais determinem quais informações serão visíveis aos demais participantes da federação. Por conseguinte, o esquema federado integra os múltiplos esquemas de exportação em uma visão unificada, resolvendo conflitos semânticos e estruturais (SHETH; LARSON, 1990). Por fim, o esquema externo fornece visões personalizadas do esquema federado para grupos específicos de usuários ou aplicações (DHARMASIRI; GOONETILLAKE, 2013).

Além dos esquemas, a arquitetura de referência inclui diferentes tipos de processadores (SHETH; LARSON, 1990). Processadores de transformação convertem dados e comandos entre diferentes representações, enquanto processadores de filtragem implementam controle de acesso e restrições de integridade (PINO; RAVIDá; SCIBILIA, 2013). Dentro dessa lógica, Sheth e Larson (1990) complementam que processadores construtores são responsáveis pela integração de esquemas e pelo gerenciamento de transações globais.

A Figura 8 ilustra os níveis de abstração em um sistema de banco de dados federado. Na base, encontram-se os bancos de dados heterogêneos com diferentes modelos de dados, como relacional, orientado a documentos, orientado a colunas e orientado a grafos. O FDBMS atua como camada intermediária, responsável pela tradução entre formatos, integração de es-

quemas e controle de acesso. O esquema federado oferece uma visão integrada dos dados distribuídos, enquanto os esquemas externos fornecem visões personalizadas para cada usuário ou aplicação.

Figura 8 – Níveis de abstração em um sistema de banco de dados federado.



Fonte: Elaborada pelo autor.

2.2.5.3 Tipos de Acoplamento

Os sistemas de bancos de dados federados podem ser classificados quanto ao grau de acoplamento entre seus componentes (SHETH; LARSON, 1990). Em federações de acoplamento frouxo (*loosely coupled*), não existe um esquema federado centralizado, e os usuários são responsáveis por construir suas próprias visões federadas a partir dos esquemas de exportação disponíveis (PINO; RAVIDá; SCIBILIA, 2013). Segundo Sheth e Larson (1990), esse modelo preserva maior autonomia dos componentes, porém transfere a complexidade de integração para os usuários finais.

Em contrapartida, federações de acoplamento apertado (*tightly coupled*) mantêm um esquema federado gerenciado centralmente por um administrador (DHARMASIRI; GOONETILLAKE, 2013). Conforme Sheth e Larson (1990), esse modelo facilita o acesso aos dados pelos usuários, que utilizam o esquema federado pré-definido, porém requer maior coordenação entre os administradores dos sistemas componentes. Por conseguinte, Dharmasiri e Goonetillake (2013) optam pelo modelo de acoplamento apertado para federar repositórios NoSQL hetero-

gêneos, argumentando que uma interface de consulta unificada simplifica o desenvolvimento de aplicações.

2.2.5.4 Processamento de Consultas

O processamento de consultas em sistemas federados apresenta desafios adicionais em comparação com sistemas centralizados (GAO; WEN; ZHANG, 2024). Conforme Sheth e Larson (1990), uma consulta submetida ao sistema federado deve ser decomposta em subconsultas direcionadas aos componentes relevantes, cujos resultados são posteriormente integrados. Seguindo essa linha, Pino, Ravidá e Scibilia (2013) observam que o desempenho pode ser comprometido quando consultas envolvem campos não indexados nos sistemas componentes, resultando em varreduras completas das tabelas remotas.

A otimização de consultas federadas requer a coleta de informações estatísticas sobre os dados distribuídos (GAO; WEN; ZHANG, 2024). Segundo Gao, Wen e Zhang (2024), técnicas de *computational push-down* permitem que operações de filtragem e agregação sejam executadas nos sistemas componentes, reduzindo a quantidade de dados transmitidos pela rede. Além disso, Dharmasiri e Goonetillake (2013) demonstram que a federação de repositórios NoSQL introduz sobrecarga de desempenho devido à tradução de consultas e à consolidação de resultados, porém essa penalidade pode ser aceitável considerando os benefícios de acesso unificado.

Em contextos que exigem privacidade e segurança, como sistemas de Geração Aumentada por Recuperação (*Retrieval-Augmented Generation* (RAG)), Zhao (2024) propõe técnicas criptográficas para executar buscas aproximadas de k vizinhos mais próximos sobre dados cifrados em bancos de dados vetoriais federados. Essa abordagem permite que múltiplas partes colaborem sem revelar seus dados individuais, atendendo a regulamentações de privacidade como *General Data Protection Regulation* (GDPR) e *Health Insurance Portability and Accountability Act* (HIPAA) (ZHAO, 2024).

2.2.5.5 Casos de Uso

Os sistemas de bancos de dados federados encontram aplicação em diversos domínios que requerem integração de fontes heterogêneas (SHETH; LARSON, 1990). No contexto de governo eletrônico, Pino, Ravidá e Scibilia (2013) avaliam a viabilidade de federar bases de dados distribuídas entre diferentes órgãos públicos, permitindo consultas unificadas sem centralização dos dados. Os autores comparam soluções como MySQL Federated e OpenLink Virtuoso, concluindo que a escolha da tecnologia depende dos requisitos específicos de desempenho e dos tipos de consultas predominantes (PINO; RAVIDá; SCIBILIA, 2013).

Na área de descoberta científica, Trinquet et al. (2025) utilizam bancos de dados federados através da API OPTIMADE para acessar repositórios de materiais mantidos por diferentes instituições. Ainda conforme Trinquet et al. (2025), essa federação abrange mais de 20

provedores de dados e oferece acesso a informações sobre mais de 30 milhões de estruturas cristalinas, viabilizando a aplicação de técnicas de aprendizado de máquina para descoberta de novos materiais ópticos.

Por fim, Dharmasiri e Goonetillake (2013) demonstram a federação de repositórios NoSQL heterogêneos, incluindo Cassandra, MongoDB e CouchDB, através de uma interface de consulta unificada baseada em SQL. Essa abordagem permite que aplicações acessem dados armazenados em diferentes modelos NoSQL sem conhecimento específico de cada tecnologia, abstraindo a heterogeneidade subjacente (DHARMASIRI; GOONETILLAKE, 2013).

A arquitetura de sistemas de bancos de dados federados inclui processadores de filtragem responsáveis pelo controle de acesso, gerenciando como os usuários são identificados e agrupados para fins de segurança (SHETH; LARSON, 1990). Em ambientes heterogêneos distribuídos, o Gerenciamento de Identidade Federado (*Federated Identity Management* (FIM)) estabelece relações de confiança entre múltiplas aplicações e organizações por meio de provedores de terceiros, viabilizando o acesso a diferentes serviços sem a necessidade de fornecer credenciais a cada provedor individualmente (SHARMA; SHARMA; DAVE, 2015). Nesse contexto, à medida que sistemas de tecnologia da informação proliferam para suportar processos de negócio, os usuários tipicamente precisam autenticar-se em múltiplos lugares, tornando mecanismos de autenticação única componentes fundamentais para sistemas que integram fontes heterogêneas (ZHAO; ZHENG; CHEN, 2004).

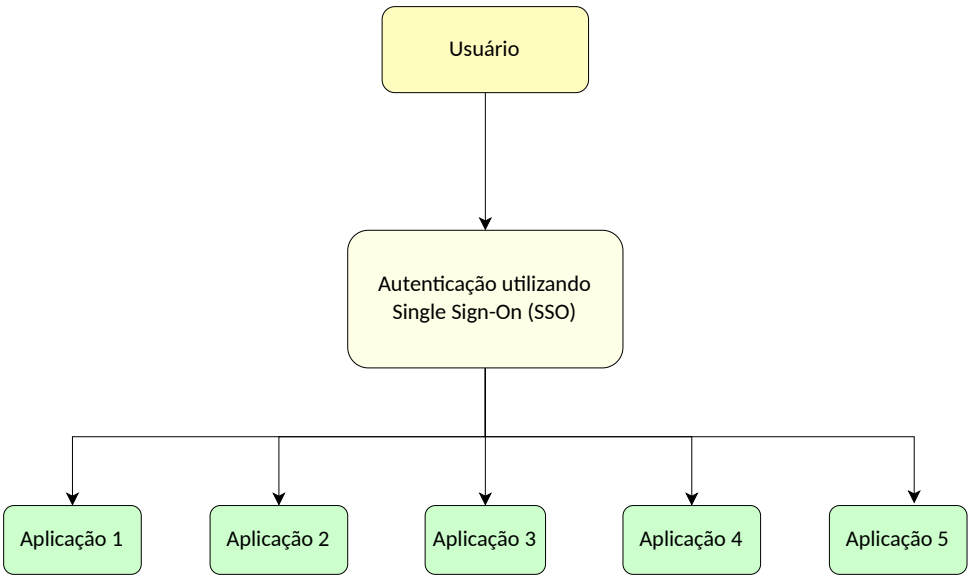
2.2.6 Single Sign-On (SSO)

O *Single Sign-On* (SSO) constitui um mecanismo de autenticação que permite aos usuários acessar múltiplas aplicações e sistemas utilizando um único conjunto de credenciais, eliminando a necessidade de autenticação repetida (SHAIKH; KASAT; JADHAV, 2022). Ainda de acordo com Sharma, Sharma e Dave (2015), essa tecnologia representa um subconjunto do Gerenciamento de Identidade e Acesso (*Identity and Access Management* (IAM)) e está intrinsecamente relacionada ao Gerenciamento de Identidade Federado (FIM), sendo que o FIM proporciona capacidades de SSO, enquanto o SSO isolado não necessariamente oferece funcionalidades de federação de identidade. A partir dessa premissa, ainda conforme Shaikh, Kasat e Jadhav (2022), o SSO armazena de forma segura múltiplas informações de *login* em uma única conta, gerando informações de autenticação aceitas por diferentes aplicações e sistemas após a validação inicial das credenciais.

2.2.6.1 Conceito e Fundamentação

O SSO foi desenvolvido para reduzir o número de autenticações necessárias entre diversos sistemas (KARUNANITHI; KIRUTHIKA, 2011). Segundo Zhao, Zheng e Chen (2004), à medida que sistemas de tecnologia da informação proliferam para suportar processos de negócio, usuários tipicamente precisam autenticar-se em múltiplos lugares, exigindo um número equi-

Figura 9 – Diagrama conceitual do propósito do SSO.



Fonte: Elaborada pelo autor.

valente de diálogos de autenticação, cada um podendo envolver diferentes nomes de usuário e informações de autorização. Por conseguinte, Shaikh, Kasat e Jadhav (2022) destacam que o SSO minimiza os riscos de segurança relacionados ao gerenciamento de usuários com o auxílio de um administrador, permitindo que os mesmos acessem diversas aplicações após uma única autenticação.

Em contrapartida, Shaikh, Kasat e Jadhav (2022) esclarecem que o sistema SSO não mescla informações de contas de usuários de diferentes serviços, aplicações e sistemas. Em acréscimo, Sharma, Sharma e Dave (2015) complementam que o SSO elimina a necessidade de lembrar múltiplos identificadores e senhas, sendo relevante em ambientes de computação heterogêneos que requerem práticas adequadas para garantir a segurança dos recursos.

2.2.6.2 Tipos de SSO

Os sistemas de SSO podem ser classificados em diferentes categorias baseadas em seu escopo de implantação (SHAIKH; KASAT; JADHAV, 2022; SERGEY, 2023). Segundo Sergey (2023), as principais categorias são SSO empresarial (*Enterprise SSO*) e SSO baseado em web (*Web-based SSO*).

O SSO empresarial, também denominado SSO de intranet, permite que múltiplos sistemas conectem-se ao mesmo ambiente (SHAIKH; KASAT; JADHAV, 2022). Conforme Sergey (2023), esse tipo envolve a instalação de um agente nas estações de trabalho dos funcionários que automaticamente insere o *login* e a senha do funcionário em formulários de autenticação de aplicações. Por conseguinte, ainda conforme Shaikh, Kasat e Jadhav (2022), esse modelo reduz o número de credenciais de *login* necessárias para acessar diferentes *softwares*, fornecendo *tokens* únicos em máquina com a finalidade de autenticação.

O SSO baseado em web, também conhecido como SSO de internet, é um mecanismo acessado através do navegador que fornece acesso dentro de uma única autenticação às aplicações no navegador web (SHAIKH; KASAT; JADHAV, 2022). Segundo Sergey (2023), essa tecnologia fornece uma função de autenticação única para aplicações web, com suporte a protocolos como *Security Assertion Markup Language* (SAML), *Open Authorization* (OAuth) e *OpenID Connect* (OIDC). Nesse contexto, Shaikh, Kasat e Jadhav (2022) explicam que quando um cliente autorizado digita suas credenciais de *login*, o mesmo obtém acesso a todos os recursos e precisa autenticar-se apenas uma vez, sem necessidade de múltiplas autenticações.

2.2.6.3 Protocolos e Tecnologias

Diversos protocolos e tecnologias são empregados na implementação de sistemas SSO, cada um adequado a diferentes cenários e requisitos de segurança (SHARMA; SHARMA; DAVE, 2015; SERGEY, 2023).

O protocolo Kerberos fornece um servidor que autentica funções para autenticar usuários ao servidor (SHAIKH; KASAT; JADHAV, 2022). Ainda segundo Shaikh, Kasat e Jadhav (2022), para verificação de cliente, o Kerberos utiliza o servidor e o banco de dados, sendo os principais componentes o Servidor de Autenticação, que realiza a autenticação e fornece *tickets* para seus serviços, e o *Ticket Granting Service* (TGS), através do qual é possível coletá-los para acessar o servidor. Por conseguinte, Jian (2009) mencionam que o Kerberos previne ataques de repetição utilizando *timestamps* nos *tickets*, porém requer sincronização de relógio entre os servidores e clientes correspondentes.

O SAML é um padrão baseado em XML para trocar dados de autenticação e autorização entre participantes, em particular entre um provedor de identidade e um provedor de serviço (SERGEY, 2023). Conforme Sharma, Sharma e Dave (2015), o SAML V2.0 é amplamente utilizado por empresas, habilitando SSO na Internet, eliminando a manutenção de múltiplas credenciais, aumentando a segurança e reduzindo incidentes de *phishing*. Somado a isso, Shaikh, Kasat e Jadhav (2022) explicam que o SAML possui grande flexibilidade e recursos, melhorando a experiência do usuário e aumentando a segurança ao fornecer autenticação e transferir informações para o provedor de serviço.

O OIDC é um padrão aberto para um sistema de autenticação descentralizado que fornece ao usuário a capacidade de criar uma única conta para autenticação em múltiplos recursos web não relacionados usando serviços de terceiros (SERGEY, 2023). Segundo Sharma, Sharma e Dave (2015), o OIDC possui três partes principais: o Identificador OIDC, a Parte Confiante (*Relying Party*) e o Provedor OIDC, sendo que o provedor armazena, emite e gerencia o usuário OIDC. Por conseguinte, Kamra e Shekhawat (2025) cita vários provedores OIDC, incluindo Google, Microsoft e Apple.

O OAuth é um *framework* de autorização que utiliza *tokens* únicos que permitem ao cliente fazer requisições em nome do usuário ou do proprietário do recurso (SHARMA; SHARMA;

DAVE, 2015). Ainda conforme Sharma, Sharma e Dave (2015), o OAuth 2.0 possui principalmente quatro participantes: Cliente, Proprietário do Recurso, Servidor de Recurso e Servidor de Autorização, sendo que este último verifica as credenciais do recurso e, na maioria dos casos, é o próprio servidor de recurso. Adicionalmente, Hossain et al. (2018) propõem um *framework* denominado OAuth-SSO para proteger o serviço SSO baseado em OAuth para aplicações web empacotadas, demonstrando a aplicação prática desse protocolo em contextos de SSO.

O *Central Authentication Service* (CAS) é um sistema de SSO de código aberto que utiliza um protocolo baseado em *tickets* projetado para fornecer autenticação e autorização de usuários (SERGEY, 2023). Ainda segundo Sergey (2023), a arquitetura do protocolo baseia-se na interação entre cliente e servidor, onde este autentica usuários e concede acesso a aplicações. Por conseguinte, Hu, Sun e Chen (2010) demonstram à aplicação do CAS em ambientes de campus digital, utilizando *Lightweight Directory Access Protocol* (LDAP) para estabelecer um banco de dados unificado de pessoal e criar grupos dinâmicos para gerenciar elementos do banco de dados.

O Shibboleth é um sistema de controle de acesso baseado em SAML e de código aberto (SERGEY, 2023). Conforme Gueye et al. (2014), ele foi projetado para implementar um protocolo baseado em padrões abertos para transferir com segurança atributos de usuário entre sites colaboradores, sendo uma extensão do SAML que expande suas funcionalidades para uma federação de entidades. Além disso, Nishioka e Okabe (2020) investigam o controle centralizado de migração de contas no SSO em Shibboleth, propondo um protocolo para migrar contas de um usuário em múltiplos *Service Providers* (SPs) de uma vez usando um provedor de atributos no ambiente SSO.

Além dos protocolos tradicionais, existem abordagens alternativas para implementação de SSO. Jun, Zhishu e Yanyan (2008) propõem uma arquitetura de segurança descentralizada baseada em JSON para comércio eletrônico, demonstrando que esse formato de dados oferece ganhos nos quesitos de eficiência de análise em relação aos seus antecessores. Por conseguinte, ainda conforme Jun, Zhishu e Yanyan (2008), é apresentado um modelo de segurança SSO baseado em JSON que facilita a integração de sistemas legados e sistemas recém-desenvolvidos em cenários de negócio para negócio (B2B).

Em relação a dispositivos com recursos limitados, Sharma e Sihag (2016) propõem um protocolo SSO híbrido que combina a arquitetura de protocolo do Kerberos com a funcionalidade do SAML, melhorando a segurança das asserções deste ao herdar a funcionalidade do Kerberos. Ainda segundo Sharma e Sihag (2016), essa abordagem implementa autenticação bidirecional, onde tanto o cliente quanto o provedor de serviço são autenticados, utilizando criptografia de chave simétrica e evitando a dependência de protocolos de camada de transporte para segurança.

2.2.6.4 *Single Logout* e Outras Funcionalidades

Além do mecanismo de autenticação única, os sistemas SSO podem incluir funcionalidades de *logout* único (*Single Logout*) (KARUNANITHI; KIRUTHIKA, 2011). Ainda conforme Karunanithi e Kiruthika (2011), esse protocolo define um mecanismo para permitir a saída, quase simultânea, de sessões ativas associadas a um principal, sendo que o *logout* pode ser iniciado diretamente pelo usuário, por um *Identity Provider* (IdP) ou SP, devido ao tempo limite de sessão, comando de administrador, entre outros. Por conseguinte, também é explicado por Karunanithi e Kiruthika (2011) que, uma vez que o SSO tenha sido alcançado, várias sessões individuais com provedores de serviço compartilham um único contexto de autenticação, e o *logout* único permite o encerramento da sessão quase em tempo real de um usuário de todas as quais participa.

2.2.6.5 Casos de Uso

Os sistemas SSO encontram aplicação em diversos contextos organizacionais e acadêmicos (HU; SUN; CHEN, 2010; HUANG; GUO, 2021; SERRANO; OñATE, 2021). No contexto de campus digitais, Hu, Sun e Chen (2010) demonstram a integração de aplicações de arquitetura B/S (*Browser/Server*) com o sistema SSO, estabelecendo um banco de dados unificado de pessoal através de um servidor LDAP, utilizando CAS e tecnologia de formulários para alcançar a integração de aplicações de rede interna do campus e outros sistemas de caixa preta com SSO.

Em plataformas de informação de administração educacional, Huang e Guo (2021) pesquisam a tecnologia SSO para plataformas de serviços de informação de administração educacional, utilizando OAuth 2.0 para gerenciamento de direitos. Por conseguinte, Serrano e Oñate (2021) demonstram a integração de API *Representational State Transfer* (REST) com sistemas de informação estudantil para compartilhamento seguro de dados, permitindo acesso unificado a múltiplos serviços acadêmicos.

Em ambientes de organização virtual baseados em computação em nuvem, Gueye et al. (2014) propõem um método de autenticação descentralizada para acessar recursos pedagógicos, incluindo Shibboleth para separar a autenticação de usuários da organização virtual. Nesse contexto, Zhang, Chen e Shen (2012) desenvolvem um modelo de autorização de SSO para ambiente distribuído, utilizando *eXtensible Access Control Markup Language* (XACML) para alcançar controle de acesso baseado em atributos (*Attribute-Based Access Control* (ABAC)).

Em portais municipais, Hu e Guo (2013) aplicam SSO entre domínios utilizando CAS para permitir que usuários acessem múltiplos serviços governamentais com uma única autenticação. Alinhado a isso, Wang, Wen e Zhang (2010) propõem um esquema de SSO para aplicações web usando criptografia baseada em identidade, permitindo que *tickets* possam transitar de um domínio de SSO para outro.

2.2.6.6 Desafios e Limitações

Apesar dos benefícios proporcionados pelos sistemas SSO, existem desafios e limitações que devem ser considerados durante a implementação (SHAIKH; KASAT; JADHAV, 2022; SERGEY, 2023). Segundo Shaikh, Kasat e Jadhav (2022), um dos principais desafios é que, se um atacante comprometer a conta SSO, todas as outras conectadas estarão em risco. Por conseguinte, Sergey (2023) observam que a principal desvantagem da tecnologia SSO é o uso de uma única senha, que se comprometida, torna-se possível acessar todos os recursos os quais possui acesso.

Da mesma forma, Shaikh, Kasat e Jadhav (2022) mencionam que, se o servidor SSO for desligado ou ficar lento, os sites ou aplicações conectados a ele SSO interrompem o funcionamento. Em contrapartida, ainda conforme Shaikh, Kasat e Jadhav (2022), quando se adiciona um sistema SSO, ele requer um longo procedimento de configuração, motivo pelo qual pequenas empresas não utilizam essa tecnologia por conta do custo desse processo.

Adicionalmente, Shaikh, Kasat e Jadhav (2022) acrescentam que o SSO apresenta riscos para computadores multiusuários, sendo que, em pequenas organizações, um computador é utilizado por diferentes funcionários em turnos rotativos, o que pode comprometer a segurança das sessões autenticadas. Em relação aos requisitos técnicos de sincronização, Shi, Yan e Li (2010) propõem um protocolo SSO seguro sem sincronização de relógio, demonstrando que é possível prevenir ataques de repetição sem exigir sincronização de relógio entre servidores e clientes, o que representa uma melhoria em relação a protocolos como Kerberos que dependem dessa sincronização.

Em relação a ataques de senha, Jian (2009) propõem um esquema melhorado de protocolo SSO que adota sistema de criptografia de chave pública e chave criptográfica *Universal Serial Bus* (USB), melhorando a eficiência de trabalho do cliente e reduzindo sua gravidade de segurança. Adicionalmente, ainda conforme Jian (2009), são adicionados bancos de dados de clientes autenticados para validação de autenticação e bancos de dados de clientes autorizados para validação de autorização, o que aumenta a capacidade de prevenir ataques de repetição do sistema.

Em relação ao desempenho e escalabilidade, Chitpinityon e Tossa (2021) propõem uma abordagem de melhoria utilizando a distribuição de carga e armazenamento de dados em memória. Segundo Chitpinityon e Tossa (2021), em geral, sistemas SSO podem suportar apenas centenas de novas autenticações por segundo e milhares de acessos simultâneos, sendo que esses valores são fatores-chave para determinar a eficiência do sistema. Por conseguinte, Chitpinityon e Tossa (2021) demonstram que o armazenamento de dados em memória e métodos de distribuição de carga podem aumentar o suporte a novas autenticações e acessos simultâneos até escalas de dezenas de milhares e centenas de milhares, respectivamente.

2.2.6.7 Comunidade Acadêmica Federada (CAFe)

No contexto de sistemas distribuídos e integração de recursos entre múltiplas organizações, a Comunidade Acadêmica Federada (CAFe) representa uma implementação prática de federação voltada para a gestão de identidade e acesso (Rede Nacional de Ensino e Pesquisa, 2015). Conforme Rodrigues, Rocha e Ribeiro (2019), essa infraestrutura permite que instituições de ensino e pesquisa brasileiras estabeleçam uma relação de confiança mútua, viabilizando o compartilhamento seguro de serviços e recursos digitais. A arquitetura da CAFe fundamenta-se na cooperação entre dois atores principais: os Provedores de Identidade (IdP) e os Provedores de Serviço (SP) (WANGHAM et al., 2012). Segundo Instituto Federal de São Paulo (2024), os IdPs são mantidos pelas instituições de origem dos usuários e são responsáveis pela gestão e autenticação das credenciais, garantindo que senhas e dados sensíveis não sejam compartilhados diretamente com os serviços externos. Em contrapartida, ainda conforme Wangham et al. (2012), os SPs são as entidades que disponibilizam os recursos, como portais de periódicos ou laboratórios virtuais, confiando na validação de identidade realizada pelos IdPs participantes da federação.

A interoperabilidade técnica entre esses sistemas heterogêneos é assegurada pela adoção de padrões abertos, especificamente o protocolo SAML e o *framework* Shibboleth (CHAGAS; MELLO; WANGHAM, 2014). A Figura 10 ilustra a página de *login* para autenticação federada utilizando o *framework* Shibboleth no IFSC, disponível no endereço

Figura 10 – Página de *login* para autenticação federada utilizando o *framework* Shibboleth

Fonte: Elaborada pelo autor.

Neto (2014) explicam que o SAML padroniza a troca de dados de autenticação e au-

torização entre domínios de segurança distintos, permitindo a asserção de identidade através de mensagens XML assinadas digitalmente. Considerando esse aspecto, ainda conforme Rede Nacional de Ensino e Pesquisa (2015), a Rede Nacional de Ensino e Pesquisa (RNP) atua como o ente centralizador da federação, mantendo um repositório confiável de metadados que permite aos SPs e IdPs se reconhecerem e comunicarem de forma segura. Para integrar esse ecossistema, as instituições interessadas devem estar credenciadas e realizar procedimentos técnicos específicos, como a instalação dos servidores do serviço e a configuração da base de usuários local (Rede Nacional de Ensino e Pesquisa, 2024). Essa estrutura, uma vez estabelecida e homologada, viabiliza o mecanismo de SSO descrito na subseção 2.2.6, onde o usuário utiliza uma única credencial institucional para acessar múltiplos serviços distribuídos (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, 2020).

Além disso, ainda conforme Rodrigues, Rocha e Ribeiro (2019), a segurança e a governança da federação são reforçadas por estruturas de conformidade como o *Security Incident Response Trust Framework for Federated Identity* (SIRTFI). Esse *framework* estabelece requisitos operacionais para a resposta a incidentes de segurança, assegurando que as instituições participantes possuam maturidade suficiente para mitigar ameaças cibernéticas dentro da federação (WANGHAM et al., 2012). Por conseguinte, ainda conforme Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (2020), a CAFe transcende a simples autenticação, constituindo uma infraestrutura crítica para a colaboração acadêmica, permitindo, por exemplo, o acesso remoto ao conteúdo assinado do Portal de Periódicos da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) independentemente da localização geográfica do pesquisador. Ainda conforme Neto (2014), a integração de serviços em nuvem e a mobilidade acadêmica dependem intrinsecamente dessa capacidade de federar identidades mantendo a autonomia das instituições locais.

2.3 Controle de Acesso Baseado em Papéis (RBAC)

O desenvolvimento acelerado das tecnologias de rede e a crescente complexidade dos sistemas de informação distribuídos impuseram desafios significativos à gestão de segurança e ao controle de acesso a recursos corporativos (ZHOU; MEINEL, 2004). Nesse cenário, o Controle de Acesso Baseado em Papéis (*Role-Based Access Control* (RBAC)) emergiu na década de 1990 como uma tecnologia consolidada para o gerenciamento e a aplicação de políticas de segurança em sistemas de larga escala (HONG; HAI, 2010). Segundo Kuhn, Coyne e Weil (2010), o modelo RBAC oferece uma abordagem eficaz para reduzir a complexidade administrativa inerente à segurança da informação, facilitando a revisão de permissões atribuídas aos usuários e a determinação da exposição ao risco de uma organização. Em contrapartida aos modelos tradicionais, como o Controle de Acesso Discricionário (*Discretionary Access Control* (DAC)) e o Controle de Acesso Mandatório (*Mandatory Access Control* (MAC)), o RBAC introduz a noção de ”papel”como uma entidade intermediária entre usuários e permissões, promovendo uma

separação lógica que simplifica a gestão de direitos de acesso (HONG; HAI, 2010).

2.3.1 Conceito e Fundamentos do RBAC

A premissa fundamental do RBAC reside na associação de permissões a papéis específicos, em vez de atribuí-las diretamente a usuários individuais (CHEN, 2015). Conforme explicado por Zhou e Meinel (2004), o modelo geral do RBAC baseia-se em três conjuntos de entidades primárias: usuários (U), papéis (R) e permissões (P). Sob essa ótica, um usuário representa um indivíduo ou agente autônomo, um papel designa uma função de trabalho ou título que carrega autoridade e responsabilidade, e uma permissão constitui uma aprovação para executar uma operação específica em um ou mais objetos protegidos (KUHN; COYNE; WEIL, 2010). Hong e Hai (2010) complementam que essa estrutura permite que os usuários adquiram as permissões necessárias simplesmente ao serem associados aos papéis correspondentes, eliminando a necessidade de configurações manuais e repetitivas de listas de controle de acesso.

O funcionamento do RBAC pressupõe que, na maioria das aplicações, as permissões mudam lentamente ao longo do tempo, enquanto os usuários podem entrar, sair ou mudar de funções com maior frequência (KUHN; COYNE; WEIL, 2010). Dessa forma, a estabilidade relativa da estrutura de papéis em comparação com a volatilidade da população de usuários confere ao RBAC uma vantagem administrativa significativa (ZHOU; MEINEL, 2004). Cabe destacar que o modelo suporta o princípio do menor privilégio, assegurando que os usuários tenham apenas as permissões estritamente necessárias para realizar suas funções, e facilita a implementação da separação de deveres, impedindo que um único indivíduo detenha todas as permissões críticas para operações sensíveis (KUHN; COYNE; WEIL, 2010).

2.3.2 Mecanismos de Funcionamento e Implementação

A implementação prática do RBAC pode variar conforme o ambiente e os requisitos específicos do sistema. Em sistemas baseados na arquitetura Navegador/Servidor (B/S), Hong e Hai (2010) propõem um modelo estendido que introduz a entidade “Grupo de Usuários” para lidar com situações onde muitos usuários normais compartilham os mesmos direitos. Nessa abordagem, a relação entre usuários e grupos e entre grupos e papéis simplifica a administração, pois as alterações de permissões podem ser aplicadas a grupos inteiros, reduzindo a carga de trabalho do administrador do sistema (HONG; HAI, 2010). O processo de acesso nesses sistemas envolve a validação do usuário pelo servidor, a recuperação de seus papéis a partir de um banco de dados e a determinação das permissões de acesso às páginas e operações funcionais, mantendo essas informações em sessão para requisições subsequentes (HONG; HAI, 2010).

Em ambientes distribuídos e de serviços Web, a integração do RBAC com infraestruturas de segurança como a Infraestrutura de Chave Pública (*Public Key Infrastructure* (PKI)) e

a Infraestrutura de Gerenciamento de Privilégios (*Privilege Management Infrastructure* (PMI)) torna-se essencial (FUGKEAW; MANPANPANICH; JUNTAPREMJITT, 2007). Fugkeaw, Manpanpanich e Juntapremjitt (2007) descrevem uma implementação onde a autenticação é realizada via PKI utilizando certificados de chave pública X.509, enquanto a autorização é gerenciada pela PMI através de certificados de atributos (ACs). Nesse modelo, os papéis dos usuários são armazenados em certificados de atributos assinados digitalmente por uma autoridade, garantindo a integridade e a autenticidade das atribuições de papéis (FUGKEAW; MANPANPANICH; JUNTAPREMJITT, 2007). As decisões de controle de acesso são então tomadas por um motor de controle de acesso (*Access Control Engine*) que avalia as políticas de autorização, escritas em XML, e também armazenadas em certificados de atributos, contra os papéis apresentados pelo usuário (FUGKEAW; MANPANPANICH; JUNTAPREMJITT, 2007).

Para sistemas que requerem autenticação única (SSO) em múltiplos domínios, Fugkeaw, Manpanpanich e Juntapremjitt (2007) apresentam um modelo que utiliza a SAML para trocar informações de autenticação e autorização. O uso de SAML permite que asserções sobre a identidade e os atributos do usuário sejam transferidas de forma segura entre provedores de identidade e provedores de serviço, facilitando a aplicação de políticas RBAC dinâmicas em ambientes federados. O sistema proposto por Fugkeaw, Manpanpanich e Juntapremjitt (2007) emprega um Sistema Multiagente (*Multi Agent System* (MAS)) para mediar o acesso, onde agentes de usuário e de aplicação validam certificados e asserções SAML, mapeando identidades de usuários (PKC) para papéis (Role AC) através de certificados de ponte (*Bridge AC*).

2.3.3 Extensões: RBAC Baseado em Atributos e Negociação

Apesar de sua robustez, o RBAC tradicional enfrenta limitações em ambientes que exigem decisões de acesso baseadas em contexto dinâmico ou em atributos detalhados dos sujeitos (KUHN; COYNE; WEIL, 2010). Para endereçar essas questões, Kuhn, Coyne e Weil (2010) discutem a integração do RBAC com o Controle de Acesso Baseado em Atributos (ABAC), resultando em modelos híbridos (RBAC-A). O ABAC permite que o acesso seja determinado por regras baseadas em atributos do sujeito, do recurso e do ambiente, oferecendo maior flexibilidade para políticas complexas (LIU; GUO; SU, 2005). No entanto, o ABAC pode sofrer com complexidade de auditoria, enquanto o RBAC pode levar a uma ”explosão de papéis”ao tentar modelar contextos dinâmicos apenas com papéis estáticos (KUHN; COYNE; WEIL, 2010). A combinação proposta por Kuhn, Coyne e Weil (2010) sugere o uso de papéis para atributos estáticos (como cargo e departamento) e regras de atributos para restrições dinâmicas (como horário e localização), mantendo a facilidade de administração do RBAC e a flexibilidade do ABAC.

No contexto de serviços Web, Liu, Guo e Su (2005) propõem o modelo *Attribute and Role-Based Access Control* (ARBAC) (*Attribute and Role-Based Access Control*), que unifica o controle de acesso para métodos de serviços e recursos de dados. Esse modelo gera automa-

ticamente conjuntos de papéis com base nas restrições de atributos dos usuários e realiza o mapeamento dinâmico entre usuários, permissões e papéis. A abordagem de Liu, Guo e Su (2005) suporta políticas ricas, incluindo separação de deveres estática e dinâmica, e permite que provedores de serviço especifiquem políticas de acesso autonomamente, lidando com um grande número de usuários desconhecidos através de asserções de atributos confiáveis.

Para cenários de *Big Data* e redes heterogêneas, onde o volume de usuários é massivo, Chen (2015) introduzem um esquema de atribuição de papéis virtuais hierárquicos baseado em negociação. Este modelo permite que múltiplos agentes ou servidores de recursos negociem e cooperem na atribuição de acessos, estabelecendo uma estrutura de hierarquia de usuários eficiente. Segundo Chen (2015), essa abordagem resolve o desafio de controlar o acesso a recursos para grandes populações de usuários, permitindo que a atribuição de papéis seja extensível e flexível através de processos de negociação entre domínios de segurança distintos.

2.3.4 Aplicações em Sistemas Legados e Zero Trust

A aplicação do RBAC também se estende à modernização de segurança em sistemas legados, conforme explorado por Bello, Diyan e Asghar (2025). A implementação de uma arquitetura *Zero Trust* nesses ambientes utiliza microssegmentação dinâmica combinada com RBAC e ABAC para mitigar vulnerabilidades inerentes a infraestruturas obsoletas. Nesse contexto, o RBAC desempenha um papel crucial ao definir privilégios específicos para funções de usuário, prevenindo o acesso não autorizado e limitando a movimentação lateral de ameaças dentro da rede (BELLO; DIYAN; ASGHAR, 2025). Bello, Diyan e Asghar (2025) argumentam que a integração desses mecanismos permite uma postura de segurança resiliente que se adapta em tempo real a mudanças na rede e ao comportamento do usuário, superando as deficiências das defesas de perímetro tradicionais.

2.3.5 Limitações do RBAC

Embora o RBAC ofereça benefícios substanciais em termos de administração e segurança, ele não está isento de limitações. Kuhn, Coyne e Weil (2010) apontam que o RBAC é frequentemente criticado pela dificuldade e tempo necessários para estabelecer uma estrutura inicial de papéis, um processo conhecido como engenharia de papéis. Em acréscimo, o modelo puro pode apresentar inflexibilidade em domínios que mudam rapidamente e suporte inadequado para atributos dinâmicos, o que pode levar à criação de milhares de papéis distintos para cobrir diferentes combinações de permissões, fenômeno mencionado anteriormente como ”explosão de papéis”(KUHN; COYNE; WEIL, 2010). Hong e Hai (2010) também observam que, em sistemas onde a base de usuários muda mais frequentemente que a estrutura de papéis, a atribuição individual de papéis pode se tornar onerosa, justificando o uso de grupos de usuários como intermediários.

3 DESENVOLVIMENTO

Este capítulo apresenta o desenvolvimento do sistema de autenticação e autorização para a plataforma de laboratórios remotos eLab. Na seção 3.1, é apresentada a visão geral do sistema. Na seção 3.2, são definidos os requisitos funcionais do mesmo. A seção 3.3 apresenta a justificativa para a escolha do modelo de dados relacional. Na seção 3.4, detalha-se a modelagem do banco de dados. A seção 3.5 aborda o gerenciamento de sessões com persistência em banco de dados. As seções 3.6 a 3.9 tratam da autenticação federada e sua integração com o modelo RBAC. Por fim, a seção 3.11 descreve o sistema de auditoria.

3.1 Visão Geral do Sistema

A Figura 11 apresenta o diagrama geral da plataforma eLab, ilustrando os componentes principais e suas interações:

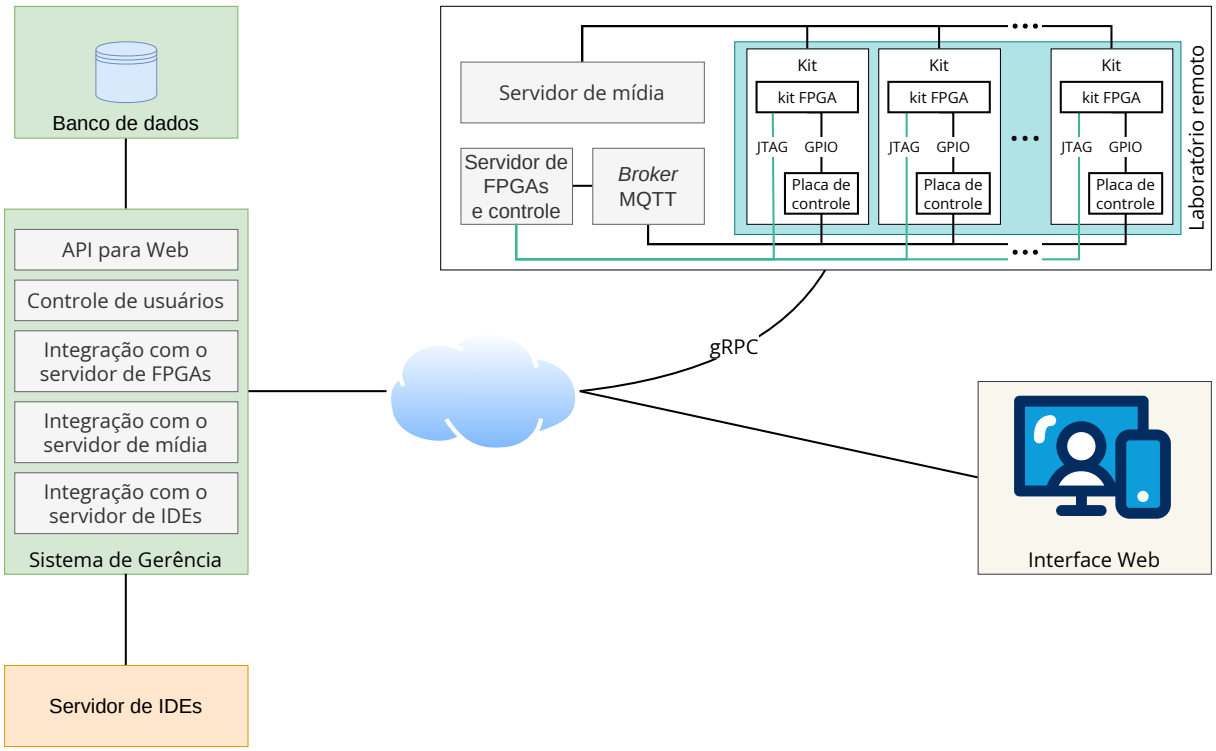


Figura 11 – Diagrama geral da plataforma eLab.

O eLab é um ambiente remoto de desenvolvimento para FPGA, concebido no âmbito do projeto "FPGA eLab: Plataforma de Desenvolvimento Avançado para Ensino e Pesquisa"[1], vinculado ao Edital nº 03/2023 do Câmpus São José do Instituto Federal de Santa Catarina (IFSC), vigente entre agosto e dezembro de 2023. O sistema oferece acesso a kits de desenvolvimento

[1] https://wiki.sj.ifsc.edu.br/index.php/Projeto_FPGA_eLab:_Plataforma_de_Desenvolvimento_Avan%C3%A7ado_para_Ensino_e_Pesquisa

FPGA e a ferramentas em nuvem, com o propósito de reduzir custos e centralizar o gerenciamento de recursos, permitir atividades práticas à distância por docentes e possibilitar que estudantes utilizem diversos kits sem investimento próprio. O protótipo contempla verificação de bitstreams, controle de acesso, feedback em tempo real e uma placa para acionamento remoto das entradas dos kits comerciais. Iniciativas análogas incluem o LabsLand para FPGA e o projeto VISIR de eletrônica analógica no IFSC .

3.2 Requisitos Funcionais

Esta seção apresenta os requisitos funcionais (RF) que orientaram as decisões de projeto do sistema eLab. A definição dos mesmos baseou-se nas características de sistemas de laboratórios remotos apresentadas por Aitor et al. (2022) e nas necessidades de integração com infraestruturas federadas acadêmicas descritas por Rodrigues, Rocha e Ribeiro (2019).

3.2.1 Requisitos de Autenticação

O sistema deve suportar múltiplos mecanismos de autenticação para atender diferentes perfis de usuários e instituições:

- **RF-01 – Suporte a Provedores de Identidade Externos**: O sistema deve permitir a integração com múltiplos IdPs externos, possibilitando que usuários utilizem credenciais de suas instituições de origem. Optou-se por este requisito porque laboratórios remotos frequentemente atendem usuários de múltiplas instituições, conforme destacado por Orduña et al. (2016);
- **RF-02 – Autenticação via Protocolo SAML 2.0**: Deve-se implementar autenticação federada através do protocolo SAML 2.0, utilizando o *framework* Shibboleth como SP. Escolheu-se o SAML 2.0 porque é o padrão adotado pela CAFe e amplamente utilizado em federações acadêmicas, conforme explicado por Sharma, Sharma e Dave (2015);
- **RF-03 – Integração com a Infraestrutura CAFe**: É necessário integrar-se à CAFe da RNP, permitindo que usuários de instituições brasileiras de ensino e pesquisa acessem a plataforma com suas credenciais institucionais. Optou-se pela CAFe porque ela representa a principal federação acadêmica brasileira, conforme descrito por Rodrigues, Rocha e Ribeiro (2019);
- **RF-04 – Autenticação Local**: O sistema deve oferecer autenticação local com credenciais armazenadas em banco de dados para usuários sem vínculo com instituições federadas. Incluiu-se este requisito para garantir acesso a usuários externos à federação, como administradores e desenvolvedores.

3.2.2 Requisitos de Autorização

O controle de acesso deve seguir um modelo baseado em papéis:

- **RF-05 – Controle de Acesso Baseado em Papéis (RBAC)**: É preciso implementar o modelo RBAC para gerenciamento de permissões. Essa escolha, conforme Kuhn, Coyne e Weil (2010), reduz a complexidade administrativa e facilita a revisão de permissões, sendo adequada para ambientes acadêmicos com diferentes perfis de usuários como estudantes, professores e administradores;
- **RF-06 – Mapeamento Automático de Atributos**: O sistema deve mapear automaticamente atributos SAML recebidos do IdP para papéis e grupos locais. Optou-se por este mapeamento automático para reduzir a carga administrativa e garantir consistência na atribuição de permissões.

3.2.3 Requisitos de Gerenciamento de Sessão

O gerenciamento de sessões deve garantir persistência e rastreabilidade:

- **RF-07 – Persistência de Sessão em Base NoSQL**: Deve-se armazenar informações de sessão em uma base de dados NoSQL orientada a documentos, como o MongoDB. Optou-se por este, em detrimento do modelo relacional, porque sessões são dados efêmeros que se beneficiam de: (i) esquema flexível para armazenar atributos variáveis; (ii) suporte nativo a índices *Time To Live* (TTL) para expiração automática; e (iii) escalabilidade horizontal nativa. Esta escolha segue as recomendações de Pokorny (2013) para dados com alta frequência de leitura e escrita;
- **RF-08 – Registro de Auditoria em Base NoSQL**: O sistema deve registrar todas as operações sensíveis em uma coleção de auditoria no MongoDB para fins de rastreabilidade e conformidade. Optou-se pelo armazenamento NoSQL porque registros de auditoria possuem estrutura semivariável (diferentes ações contêm diferentes detalhes) e o volume de registros pode crescer significativamente, beneficiando-se da escalabilidade horizontal. Conforme Chacón et al. (2025), o registro de operações é um requisito fundamental para sistemas de laboratórios remotos.

3.2.4 Escopo da Versão Inicial

Esta primeira versão do sistema implementa um modelo de confiança centralizado: a identidade dos usuários pode originar-se de provedores externos, sendo IdPs federados ou OAuth, porém a validação, o gerenciamento de sessões, a autorização e a auditoria são centralizadas no sistema eLab. Escolheu-se esta abordagem porque ela simplifica a implementação inicial e garante consistência dos dados.

3.3 Justificativa do Modelo de Dados

Esta seção apresenta a justificativa para a escolha de uma arquitetura híbrida de persistência de dados no sistema eLab, combinando um banco de dados relacional para entidades estruturadas com um banco de dados NoSQL orientado a documentos para sessões e auditoria.

Optou-se pelo modelo relacional, implementado através do SGBD MySQL, para armazenar as entidades estruturadas do sistema, como usuários, grupos, papéis, laboratórios, experimentos, equipamentos e agendamentos.

O domínio do sistema eLab apresenta relacionamentos complexos que justificam esta escolha. Um usuário pertence a exatamente um grupo, que por sua vez está associado a um papel com permissões específicas; um laboratório pode conter múltiplos experimentos, cada experimento pode utilizar múltiplos equipamentos, e um equipamento pode ser compartilhado entre experimentos distintos. Estas relações N:M (muitos-para-muitos) e hierarquias 1:N são representadas naturalmente através de tabelas intermediárias e chaves estrangeiras no modelo relacional (ELMASRI; NAVATHE, 2010).

A integridade referencial oferecida pelo MySQL previne inconsistências críticas para a operação do sistema. Por exemplo, não é possível excluir um laboratório que contenha experimentos ativos, nem remover um usuário que possua agendamentos futuros, sem tratamento explícito destas dependências através de restrições `ON DELETE` configuradas no esquema. Conforme Hassan (2021), as propriedades ACID garantem que operações compostas, como a criação de um agendamento que envolve verificação de disponibilidade do experimento, reserva do equipamento e registro do horário, sejam executadas de forma atômica.

Adicionalmente, o modelo relacional facilita consultas analíticas necessárias à gestão do sistema. Relatórios como “tempo médio de utilização por laboratório”, “experimentos mais acessados por instituição” ou “taxa de ocupação por período” requerem junções entre múltiplas tabelas e funções de agregação, operações para as quais a linguagem SQL oferece suporte consolidado.

3.3.1 Reconhecimento de Limitações

A escolha do modelo relacional apresenta limitações que foram consideradas no projeto:

1. **Atributos Dinâmicos**: Atributos do esquema `eduPerson`, recebidos via SAML, variam conforme o IdP de origem. Alguns provedores enviam apenas atributos obrigatórios, enquanto outros incluem atributos opcionais como `eduPersonEntitlement` ou `schacUserStatus`. A estrutura rígida de tabelas relacionais não acomoda naturalmente esta variabilidade. Conforme Ramzan et al. (2019), bases de dados NoSQL seriam mais adequadas para dados semiestruturados;

2. **Escalabilidade Horizontal**: Bancos de dados relacionais apresentam dificuldades de escalabilidade horizontal (*sharding*). Segundo Pokorny (2013), bancos de dados NoSQL são projetados para suportarem essa estruturação através da distribuição de dados em múltiplos servidores. A implementação de *sharding* em bancos relacionais requer configurações complexas que podem comprometer a integridade referencial.

3.3.2 Escolha do MongoDB para Sessões e Auditoria

Em um cenário com múltiplas instituições e usuários concorrentes, o volume de sessões ativas e de registros de auditoria tende a crescer de forma que a escalabilidade do armazenamento desses dados torna-se relevante. Para os dados de sessão e auditoria, optou-se pelo MongoDB, um banco de dados NoSQL orientado a documentos, pelos seguintes motivos:

1. **Esquema Flexível**: Documentos MongoDB não exigem esquema pré-definido, permitindo armazenar atributos variáveis sem alterações estruturais. Esta característica é vantajosa para registros de auditoria que contêm detalhes diferentes conforme o tipo de ação;

2. **Índices TTL Nativos**: Oferece suporte nativo a índices TTL, que expiram automaticamente documentos após um período configurado. Escolheu-se esta funcionalidade para sessões porque elimina a necessidade de processos externos para limpeza de sessões expiradas;

3. **Escalabilidade Horizontal**: Conforme Pokorny (2013), bancos de dados NoSQL são projetados para escalabilidade horizontal através de *sharding* nativo. Esta característica permite distribuir sessões e registros (*logs*) de auditoria em múltiplos servidores sem configurações complexas;

4. **Alto Desempenho de Escrita**: Registros de auditoria são gerados em alta frequência durante a operação do sistema. O MongoDB oferece operações de escrita otimizadas que suportam esta carga de trabalho.

3.3.3 Arquitetura Híbrida Adotada

A solução final adota uma arquitetura híbrida que combina as vantagens de ambos os modelos:

- **MySQL para Entidades Estruturadas**: Usuários, grupos, papéis, instituições, laboratórios, experimentos, equipamentos e agendamentos são armazenados em tabelas relacionais com integridade referencial. Atributos SAML variáveis na tabela `User` utilizam o tipo `JSON` nativo do MySQL 5.7+;

- **MongoDB para Sessões**: A coleção `sessions` armazena *tokens* de sessão com índice TTL para expiração automática. Cada documento contém o *token*, referência ao usuário (`userId`), endereço IP, *user-agent* e *timestamps*;
- **MongoDB para Auditoria**: A coleção `auditLogs` armazena registros de operações sensíveis com estrutura flexível para detalhes específicos de cada tipo de ação.

Esta arquitetura permite que entidades com relacionamentos complexos mantenham integridade referencial no MySQL, que atua como repositório principal de identidade e catálogo, com usuários, instituições, laboratórios, experimentos e equipamentos, ao passo que o MongoDB atua como repositório auxiliar para dados efêmeros e de alto volume. A aplicação coordena ambos. Por exemplo, cada documento de sessão no MongoDB referencia o `userId` armazenado no MySQL, garantindo consistência entre os dois sistemas.

3.4 Modelagem do Banco de Dados

Esta seção apresenta a modelagem do banco de dados relacional para o sistema eLab. O modelo incorpora os conceitos de RBAC apresentados na seção 2.3, permitindo controle de acesso baseado em papéis e grupos de usuários.

3.4.1 Estrutura relacional

A seguir, são apresentadas as tabelas do banco de dados organizadas em grupos lógicos. Após cada grupo, são explicados os relacionamentos e cardinalidades, além de exemplos de consultas SQL e como seriam os resultados esperados.

3.4.1.1 Tabelas de Identidade e Acesso

Quadro 1 – Estrutura da tabela `Institution`.

Coluna	Tipo de Dado	Uso
institutionId	INT (PK, AUTO_INCREMENT)	Identificador único da instituição
name	VARCHAR(100)	Nome da instituição
country	VARCHAR(100)	País da instituição
state	VARCHAR(100)	Estado da instituição
city	VARCHAR(100)	Cidade da instituição
address	VARCHAR(255)	Endereço da instituição
enabled	TINYINT(1)	Indica se a instituição está habilitada (1) ou desabilitada (0)
entityId	VARCHAR(255), UNIQUE, NULL	Identificador único da instituição em sistemas externos
metadataUrl	VARCHAR(500), NULL	URL dos metadados da instituição em sistemas externos
isFederated	TINYINT(1)	Indica se a instituição participa de federações externas (1) ou não (0)
createdAt	DATETIME	Data e hora de criação do registro
updatedAt	DATETIME	Data e hora da última atualização do registro

Fonte: Elaborada pelo autor.

Quadro 2 – Estrutura da tabela `Role`.

Coluna	Tipo de Dado	Uso
roleId	INT (PK, AUTO_INCREMENT)	Identificador único do papel
description	VARCHAR(100)	Descrição do papel (ex: Administrador, Professor, Estudante)
createdAt	DATETIME	Data e hora de criação do registro
updatedAt	DATETIME	Data e hora da última atualização do registro

Fonte: Elaborada pelo autor.

Quadro 3 – Estrutura da tabela `Group`.

Coluna	Tipo de Dado	Uso
groupId	INT (PK, AUTO_INCREMENT)	Identificador único do grupo
name	VARCHAR(45)	Nome do grupo de usuários
description	VARCHAR(100), NULL	Descrição do grupo
roleId	INT (FK)	Referência ao papel associado ao grupo
createdAt	DATETIME	Data e hora de criação do registro
updatedAt	DATETIME	Data e hora da última atualização do registro

Fonte: Elaborada pelo autor.

Quadro 4 – Estrutura da tabela `User`.

Coluna	Tipo de Dado	Uso
userId	INT (PK, AUTO_INCREMENT)	Identificador único do usuário
name	VARCHAR(100)	Nome completo do usuário
email	VARCHAR(100), UNIQUE	Endereço de e-mail do usuário (identificador único)
userType	ENUM('local', 'federated')	Tipo de usuário: local ou federado
federatedId	VARCHAR(255), UNIQUE, NULL	Identificador único recebido do IdP externo
passwordHash	VARCHAR(255), NULL	Hash da senha (apenas para usuários locais)
samlAttributes	JSON, NULL	Atributos SAML adicionais em formato JSON
groupId	INT (FK)	Referência ao grupo do usuário
institutionId	INT (FK)	Referência à instituição do usuário
lastLoginAt	DATETIME, NULL	Data e hora do último *login*
createdAt	DATETIME	Data e hora de criação do registro
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Os relacionamentos entre as tabelas deste grupo são descritos a seguir:

- **Group** $\rightarrow$ **Role** (N:1): Cada grupo está associado a exatamente um papel através da chave estrangeira `Group.roleId`. Múltiplos grupos podem compartilhar o mesmo papel, permitindo, por exemplo, que grupos "Professores de Eletrônica" e "Professores de Computação" possuam o papel "Professor" com as mesmas permissões;
- **User** $\rightarrow$ **Group** (N:1): Cada usuário pertence a exatamente um grupo através da chave estrangeira `User.groupId`. O grupo determina indiretamente as permissões do usuário, pois o papel do grupo é herdado por todos os seus membros;
- **User** $\rightarrow$ **Institution** (N:1): Cada usuário está vinculado a exatamente uma instituição através da chave estrangeira `User.institutionId`. Esta associação permite identificar a origem do usuário e possibilita relatórios segmentados por instituição.

Os exemplos a seguir demonstram consultas SQL que exploram estes relacionamentos:

Código 3.1 – Consulta para listar usuários com seus grupos e papéis.

```sql
SELECT
    u.name AS "Usuário",
    u.email AS "E-mail",
    g.name AS "Grupo",
    r.description AS "Papel"
FROM User u
INNER JOIN `Group` g ON u.groupId = g.groupId
INNER JOIN Role r ON g.roleId = r.roleId
ORDER BY r.description, g.name, u.name;
```

Código 3.2 – Exemplo de saída da consulta de usuários, grupos e papéis.

```
+----------------------+-------------------------+-------------+------------+
| Usuário              | E-mail                  | Grupo       | Papel      |
+----------------------+-------------------------+-------------+------------+
| Carlos Souza         | carlos@ifsc.edu.br      | Técnicos    | Técnico    |
| Ana Silva            | ana.silva@ufsc.br       | Professores | Professor  |
| João Oliveira        | joao@udesc.br           | Professores | Professor  |
| Maria Santos         | maria@ifsc.edu.br       | Estudantes  | Estudante  |
| Pedro Costa          | pedro@ufsc.br           | Estudantes  | Estudante  |
+----------------------+-------------------------+-------------+------------+
5 rows in set (0.02 sec)
```

Código 3.3 – Consulta para contar usuários por instituição federada.

```sql
SELECT
    i.name AS "Instituição",
    i.entityId AS "Identificador de Federação",
    COUNT(u.userId) AS "Total de Usuários"
FROM Institution i
LEFT JOIN User u ON i.institutionId = u.institutionId
WHERE i.isFederated = 1
GROUP BY i.institutionId
ORDER BY 3 DESC;
```

Código 3.4 – Exemplo de saída da consulta de usuários por instituição.

```
+---------------------------------------+-----------------------------------------------+-------------------+
| Instituição                           | Identificador de Federação                    | Total de Usuários |
+---------------------------------------+-----------------------------------------------+-------------------+
| Universidade Federal de Santa Catarina | https://idp.ufsc.br/idp/shibboleth           |                42 |
| Instituto Federal de Santa Catarina   | https://shibboleth.ifsc.edu.br/idp/shibboleth |                28 |
| Universidade do Estado de Santa Catarina | https://idp.udesc.br/idp/shibboleth        |                15 |
| Universidade Federal do Paraná        | https://idp.ufpr.br/idp/shibboleth           |                 8 |
+---------------------------------------+-----------------------------------------------+-------------------+
4 rows in set (0.01 sec)
```

3.4.1.2 Tabelas de Laboratórios e Experimentos

Quadro 5 – Estrutura da tabela `LaboratoryStatus`.

Coluna	Tipo de Dado	Uso
laboratoryStatusId	INT (PK, AUTO_INCREMENT)	Identificador único do status
description	VARCHAR(100)	Descrição do status (ex: Ativo, Manutenção)
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Quadro 6 – Estrutura da tabela `Laboratory`.

Coluna	Tipo de Dado	Uso
laboratoryId	INT (PK, AUTO_INCREMENT)	Identificador único do laboratório
name	VARCHAR(45)	Nome do laboratório
description	VARCHAR(1000)	Descrição detalhada
institutionId	INT (FK)	Referência à instituição proprietária
laboratoryStatusId	INT (FK)	Referência ao status atual
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Quadro 7 – Estrutura da tabela `ExperimentType`.

Coluna	Tipo de Dado	Uso
experimentTypeId	INT (PK, AUTO_INCREMENT)	Identificador único do tipo
description	VARCHAR(45)	Descrição do tipo (ex: FPGA, Microcontrolador)
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Quadro 8 – Estrutura da tabela `Experiment`.

Coluna	Tipo de Dado	Uso
experimentId	INT (PK, AUTO_INCREMENT)	Identificador único do experimento
name	VARCHAR(45)	Nome do experimento
description	VARCHAR(1000)	Descrição detalhada
experimentTypeId	INT (FK)	Referência ao tipo de experimento
laboratoryId	INT (FK)	Referência ao laboratório
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Os relacionamentos entre as tabelas deste grupo são descritos a seguir:

- **Laboratory** $\rightarrow$ **Institution** (N:1): Cada laboratório pertence a exatamente uma instituição através da chave estrangeira `Laboratory.institutionId`. Esta associação permite identificar qual instituição é proprietária e responsável pelo laboratório, possibilitando controle de acesso e relatórios segmentados por instituição;

- **Laboratory** → **LaboratoryStatus** (N:1): Cada laboratório possui exatamente um status operacional através da chave estrangeira `Laboratory.laboratoryStatusId`. Os status possíveis incluem “Ativo”, “Em Manutenção”,“Desativado”, “Totalmente Ocupado” ou “Em desativação”, permitindo controlar a disponibilidade do laboratório para agendamentos;
- **Experiment** → **Laboratory** (N:1): Cada experimento está vinculado a exatamente um laboratório através da chave estrangeira `Experiment.laboratoryId`. Um laboratório pode conter múltiplos experimentos, representando diferentes práticas ou atividades disponíveis naquele espaço;
- **Experiment** → **ExperimentType** (N:1): Cada experimento é classificado por exatamente um tipo através da chave estrangeira `Experiment.experimentTypeId`. Os tipos, como “FPGA” ou “Microcontrolador”, facilitam a organização e busca de experimentos por categoria.

Os exemplos a seguir demonstram consultas SQL que exploram estes relacionamentos:

Código 3.5 – Consulta para listar experimentos disponíveis por laboratório.

```sql
SELECT
    l.name AS "Laboratório",
    ls.description AS "Status do Laboratório",
    e.name AS   "Experimento",
    et.description AS "Tipo do Experimento"
FROM Laboratory l
INNER JOIN LaboratoryStatus ls ON l.laboratoryStatusId = ls.laboratoryStatusId
INNER JOIN Experiment e ON e.laboratoryId = l.laboratoryId
INNER JOIN ExperimentType et ON e.experimentTypeId = et.experimentTypeId
WHERE ls.description = 'Ativo'
ORDER BY l.name, e.name;
```

Código 3.6 – Exemplo de saída da consulta de experimentos por laboratório.

```
+--------------------------+---------------------+--------------------------+--------------------+
| Laboratório              | Status do Laboratório| Experimento              | Tipo do Experimento|
+--------------------------+---------------------+--------------------------+--------------------+
| Laboratório de Eletrônica | Ativo               | Contador Binário 4 bits  | FPGA               |
| Laboratório de Eletrônica | Ativo               | Multiplexador 4x1        | FPGA               |
| Laboratório de Eletrônica | Ativo               | Controle de LED com PWM  | Microcontrolador   |
| Laboratório de Sistemas   | Ativo               | Servidor Web Embarcado   | Microcontrolador   |
| Laboratório de Sistemas   | Ativo               | Comunicação Serial RS-232 | Microcontrolador  |
+--------------------------+---------------------+--------------------------+--------------------+
5 rows in set (0.03 sec)
```

Código 3.7 – Consulta para contar experimentos por tipo e instituição.

```sql
SELECT
    i.name AS "Instituição",
    et.description AS "Tipo do Experimento",
    COUNT(e.experimentId) AS "Total de Experimentos"
FROM Institution i
```

```
INNER JOIN Laboratory l ON l.institutionId = i.institutionId
INNER JOIN Experiment e ON e.laboratoryId = l.laboratoryId
INNER JOIN ExperimentType et ON e.experimentTypeId = et.experimentTypeId
GROUP BY i.institutionId, et.experimentTypeId
ORDER BY i.name, "Total de Experimentos" DESC;
```

Código 3.8 – Exemplo de saída da consulta de experimentos por tipo e instituição.

```
+----------------+--------------------+-----------------------+
| Instituição    | Tipo do Experimento | Total de Experimentos |
+----------------+--------------------+-----------------------+
| UFSC           | FPGA               |                     5 |
| UFSC           | Microcontrolador   |                     3 |
| IFSC           | Microcontrolador   |                     4 |
| IFSC           | FPGA               |                     2 |
| UESC           | FPGA               |                     2 |
+----------------+--------------------+-----------------------+
5 rows in set (0.02 sec)
```

3.4.1.3 Tabelas de Equipamentos e Agendamento

Quadro 9 – Estrutura da tabela `EquipmentStatus`.

Coluna	Tipo de Dado	Uso
equipmentStatusId	INT (PK, AUTO_INCREMENT)	Identificador único do status
description	VARCHAR(45)	Descrição do status (ex: Disponível, Em Uso)
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Quadro 10 – Estrutura da tabela `Equipment`.

Coluna	Tipo de Dado	Uso
equipmentId	INT (PK, AUTO_INCREMENT)	Identificador único do equipamento
name	VARCHAR(100)	Nome do equipamento
model	VARCHAR(45), NULL	Modelo do equipamento
manufacturer	VARCHAR(45), NULL	Fabricante
serialNumber	VARCHAR(45), NULL	Número de série
equipmentStatusId	INT (FK)	Referência ao status atual
experimentId	INT (FK), NULL	Referência ao experimento associado
description	VARCHAR(200), NULL	Descrição adicional
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Quadro 11 – Estrutura da tabela `ScheduleStatus`.

Coluna	Tipo de Dado	Uso
scheduleStatusId	INT (PK, AUTO_INCREMENT)	Identificador único do status
description	VARCHAR(100)	Descrição do status (ex: Agendado, Concluído)
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Quadro 12 – Estrutura da tabela `Schedule`.

Coluna	Tipo de Dado	Uso
scheduleId	INT (PK, AUTO_INCREMENT)	Identificador único do agendamento
experimentId	INT (FK)	Referência ao experimento agendado
userId	INT (FK)	Referência ao usuário que agendou
scheduleStatusId	INT (FK)	Referência ao status do agendamento
scheduledAt	DATETIME	Data e hora do agendamento
duration	INT	Duração em minutos
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

Os relacionamentos entre as tabelas deste grupo são descritos a seguir:

- **Equipment** → **EquipmentStatus** (N:1): Cada equipamento possui exatamente um status operacional através da chave estrangeira `Equipment.equipmentStatusId`. Os status possíveis incluem “Disponível”, “Em Uso”, “Em Manutenção” ou “Indisponível”, permitindo controlar a disponibilidade do equipamento para utilização em experimentos;
- **Equipment** → **Experiment** (N:1): Cada equipamento pode estar associado a no máximo um experimento através da chave estrangeira `Equipment.experimentId`. Esta relação é opcional (NULL), pois um equipamento pode não estar vinculado a nenhum experimento específico. Múltiplos equipamentos podem estar associados ao mesmo experimento, representando os recursos físicos necessários para sua execução;
- **Schedule** → **Experiment** (N:1): Cada agendamento refere-se a exatamente um experimento através da chave estrangeira `Schedule.experimentId`. Um experimento pode ter múltiplos agendamentos ao longo do tempo, representando as diferentes sessões de uso por usuários;
- **Schedule** → **User** (N:1): Cada agendamento está vinculado a exatamente um usuário através da chave estrangeira `Schedule.userId`. Esta associação identifica quem realizou a reserva e permite controle de acesso e rastreabilidade de uso;
- **Schedule** → **ScheduleStatus** (N:1): Cada agendamento possui exatamente um status através da chave estrangeira `Schedule.scheduleStatusId`. Os status possíveis incluem “Agendado”, “Em Andamento”, “Concluído” ou “Cancelado”, permitindo acompanhar o ciclo de vida de cada reserva.

Os exemplos a seguir demonstram consultas SQL que exploram estes relacionamentos:

Código 3.9 – Consulta para listar equipamentos com status e experimentos.

```sql
SELECT
    e.name AS "Equipamento",
    e.model AS "Modelo",
```

```
    e.manufacturer AS "Fabricante",
    es.description AS "Status",
    exp.name AS "Experimento"
FROM Equipment e
INNER JOIN EquipmentStatus es ON e.equipmentStatusId = es.equipmentStatusId
LEFT JOIN Experiment exp ON e.experimentId = exp.experimentId
ORDER BY exp.name, e.name;
```

Código 3.10 – Exemplo de saída da consulta de equipamentos.

```
+--------------------+--------------+-------------+------------+----------------------+
| Equipamento        | Modelo       | Fabricante  | Status     | Experimento          |
+--------------------+--------------+-------------+------------+----------------------+
| FPGA Artix-7       | XC7A35T      | Xilinx      | Disponível | Lógica Combinacional |
| Osciloscópio DSO   | DS1054Z      | Rigol       | Em Uso     | Lógica Combinacional |
| FPGA Spartan-6     | XC6SLX9      | Xilinx      | Disponível | Lógica Sequencial    |
| Analisador Lógico  | LA104        | Miniware    | Disponível | Lógica Sequencial    |
| Kit Arduino Mega   | ATmega2560   | Arduino     | Disponível | Sistemas Embarcados  |
| Multímetro Digital | UT61E        | UNI-T       | Em Uso     | Sistemas Embarcados  |
| Fonte DC Ajustável | DP832        | Rigol       | Manutenção | NULL                 |
+--------------------+--------------+-------------+------------+----------------------+
7 rows in set (0.01 sec)
```

Código 3.11 – Consulta para listar agendamentos futuros.

```
SELECT
    u.name AS "Usuário",
    exp.name AS "Experimento",
    l.name AS "Laboratório",
    ss.description AS "Status",
    s.scheduledAt AS "Data e Hora",
    s.duration AS "Duração em minutos"
FROM Schedule s
INNER JOIN User u ON s.userId = u.userId
INNER JOIN Experiment exp ON s.experimentId = exp.experimentId
INNER JOIN Laboratory l ON exp.laboratoryId = l.laboratoryId
INNER JOIN ScheduleStatus ss ON s.scheduleStatusId = ss.scheduleStatusId
WHERE s.scheduledAt >= CURRENT_DATE()
ORDER BY s.scheduledAt;
```

Código 3.12 – Exemplo de saída da consulta de agendamentos futuros.

```
+---------------+---------------------+-----------------+----------+---------------------+--------------------+
| Usuário       | Experimento         | Laboratório     | Status   | Data e Hora         | Duração em minutos |
+---------------+---------------------+-----------------+----------+---------------------+--------------------+
| Maria Santos  | Lógica Combinacional | Lab. de Eletrôn. | Agendado | 2025-12-20 14:00:00 |                 60 |
| Pedro Costa   | Sistemas Embarcados | Lab. de Sistemas | Agendado | 2025-12-20 15:30:00 |                 90 |
| Ana Silva     | Lógica Sequencial   | Lab. de Eletrôn. | Agendado | 2025-12-21 09:00:00 |                 60 |
| João Oliveira | Lógica Combinacional | Lab. de Eletrôn. | Agendado | 2025-12-21 10:30:00 |                 45 |
+---------------+---------------------+-----------------+----------+---------------------+--------------------+
4 rows in set (0.02 sec)
```

3.4.1.4 Tabela de Configuração do Sistema

A `SystemConfig` é uma tabela auxiliar que armazena pares chave-valor para configurações globais do sistema, como tempo de sessão, parâmetros de integração e outros aspectos administrativos. Ela não possui relacionamentos com outras tabelas do modelo.

Quadro 13 – Estrutura da tabela `SystemConfig`.

Coluna	Tipo de Dado	Uso
configId	INT (PK, AUTO_INCREMENT)	Identificador único da configuração
key	VARCHAR(100), UNIQUE	Chave da configuração
value	TEXT, NULL	Valor da configuração
description	VARCHAR(500), NULL	Descrição do propósito
createdAt	DATETIME	Data e hora de criação
updatedAt	DATETIME	Data e hora da última atualização

Fonte: Elaborada pelo autor.

3.4.2 Coleções MongoDB para Sessões e Auditoria

Conforme a arquitetura híbrida adotada, as entidades de sessão e auditoria são armazenadas no MongoDB. A seguir, são apresentadas as estruturas dos documentos em cada coleção.

Quadro 14 – Estrutura do documento na coleção `sessions`.

Campo	Tipo	Uso
_id	ObjectId	Identificador único gerado pelo MongoDB
sessionToken	String (índice único)	Token único da sessão gerado pelo sistema
userId	Number	Referência ao identificador do usuário no MySQL
federatedSessionId	String (opcional)	Identificador da sessão no IdP externo
ipAddress	String	Endereço IP de origem da sessão
userAgent	String	*User-Agent* do navegador
createdAt	ISODate	Data e hora de criação da sessão
expiresAt	ISODate (índice TTL)	Data e hora de expiração automática

Fonte: Elaborada pelo autor.

O campo `expiresAt` possui um índice TTL configurado, o que permite que o MongoDB remova automaticamente os documentos expirados sem necessidade de processos externos.

O Código 3.13 apresenta um exemplo de documento armazenado na coleção `sessions`:

Código 3.13 – Exemplo de documento na coleção `sessions`.

```
{
  "_id": { "$oid": "675a3b2f8c9d1e2a3b4c5d6e" },
  "sessionToken": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0",
  "userId": 42,
  "federatedSessionId": "_session_cafe_rnp_br_2025",
  "ipAddress": "192.168.1.100",
  "userAgent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36",
  "createdAt": { "$date": "2025-12-23T18:00:00.000Z" },
  "expiresAt": { "$date": "2025-12-23T23:59:59.999Z" }
}
```

Quadro 15 – Estrutura do documento na coleção `auditLogs`.

Campo	Tipo	Uso
_id	ObjectId	Identificador único gerado pelo MongoDB
userId	Number (opcional)	Referência ao usuário no MySQL (nulo se não autenticado)
action	String	Tipo de ação (ex: LOGIN, LOGOUT, USER_CREATED)
resource	String (opcional)	Recurso afetado pela ação
details	Object	Detalhes adicionais específicos da ação
ipAddress	String	Endereço IP de origem
userAgent	String	*User-Agent* do navegador
createdAt	ISODate	Data e hora do registro

Fonte: Elaborada pelo autor.

A coleção `auditLogs` utiliza o campo `details` como objeto flexível, permitindo armazenar informações específicas de cada tipo de ação sem necessidade de alterações no esquema.

O Código 3.14 apresenta um exemplo de documento armazenado na coleção `auditLogs`:

Código 3.14 – Exemplo de documento na coleção `auditLogs`.

```
{
  "_id": { "$oid": "675a3c4f9d8e2f3b4c5d6e7f" },
  "userId": 42,
  "action": "LOGIN",
  "resource": null,
  "details": {
    "authMethod": "federated",
    "idpEntityId": "https://shibboleth.ifsc.edu.br/idp/shibboleth",
    "sessionToken": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0"
  },
  "ipAddress": "192.168.1.100",
  "userAgent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36",
  "createdAt": { "$date": "2025-12-23T18:00:00.000Z" }
}
```

Os relacionamentos entre as coleções MongoDB e as tabelas MySQL são estabelecidos através do campo `userId`, que referencia o identificador primário da tabela `User` no MySQL. Esta referência é verificada pela aplicação, não havendo integridade referencial em nível de banco de dados entre os dois sistemas.

Os exemplos a seguir demonstram consultas MongoDB, escritas em JavaScript, que exploram as coleções `sessions` e `auditLogs`:

Código 3.15 – Consulta para listar sessões ativas de um usuário.

```
db.sessions.find(
    {
        userId: 42,
        expiresAt: { $gt: new Date() }
    },
    {
        sessionToken: 1,
        ipAddress: 1,
        userAgent: 1,
```

```
        createdAt: 1,
        expiresAt: 1
    }
).sort({ createdAt: -1 });
```

Código 3.16 – Exemplo de saída da consulta de sessões ativas.

```
[
  {
    "_id": ObjectId("675a3b2f8c9d1e2a3b4c5d6e"),
    "sessionToken": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0",
    "ipAddress": "192.168.1.100",
    "userAgent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36",
    "createdAt": ISODate("2025-12-23T18:00:00.000Z"),
    "expiresAt": ISODate("2025-12-23T23:59:59.999Z")
  },
  {
    "_id": ObjectId("675a2a1e7b8c0d1a2b3c4d5e"),
    "sessionToken": "z9y8x7w6v5u4t3s2r1q0p9o8n7m6l5k4j3i2h1g0",
    "ipAddress": "10.0.0.25",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)",
    "createdAt": ISODate("2025-12-23T14:30:00.000Z"),
    "expiresAt": ISODate("2025-12-23T22:30:00.000Z")
  }
]
```

Código 3.17 – Agregação para estatísticas de sessões por dia.

```
// Agregação para contar sessões por dia e calcular média de duração
db.sessions.aggregate([
    {
        $match: {
            createdAt: {
                $gte: ISODate("2025-12-01T00:00:00.000Z"),
                $lt: ISODate("2026-01-01T00:00:00.000Z")
            }
        }
    },
    {
        $group: {
            _id: { $dateToString: { format: "%Y-%m-%d", date: "$createdAt" } },
            totalSessions: { $sum: 1 },
            avgDuration: {
                $avg: { $subtract: ["$expiresAt", "$createdAt"] }
            }
        }
    },
    {
        $project: {
            _id: 0,
            date: "$_id",
            totalSessions: 1,
            avgDurationHours: { $divide: ["$avgDuration", 3600000] }
        }
    },
    { $sort: { date: -1 } }
```

```
]);
```

Código 3.18 – Exemplo de saída da agregação de estatísticas de sessões.

```
[
  { "date": "2025-12-23", "totalSessions": 47, "avgDurationHours": 5.2 },
  { "date": "2025-12-22", "totalSessions": 52, "avgDurationHours": 4.8 },
  { "date": "2025-12-21", "totalSessions": 38, "avgDurationHours": 6.1 },
  { "date": "2025-12-20", "totalSessions": 41, "avgDurationHours": 5.5 },
  { "date": "2025-12-19", "totalSessions": 55, "avgDurationHours": 4.3 }
]
```

Código 3.19 – Consulta para listar registros de auditoria por ação.

```
db.auditLogs.find(
    {
        action: "LOGIN",
        createdAt: {
            $gte: ISODate("2025-12-01T00:00:00.000Z"),
            $lt: ISODate("2026-01-01T00:00:00.000Z")
        }
    },
    {
        userId: 1,
        action: 1,
        "details.authMethod": 1,
        "details.idpEntityId": 1,
        ipAddress: 1,
        createdAt: 1
    }
).sort({ createdAt: -1 }).limit(5);
```

Código 3.20 – Exemplo de saída da consulta de auditoria por ação.

```
[
  {
    "_id": ObjectId("675a3c4f9d8e2f3b4c5d6e7f"),
    "userId": 42,
    "action": "LOGIN",
    "details": {
      "authMethod": "federated",
      "idpEntityId": "https://shibboleth.ifsc.edu.br/idp/shibboleth"
    },
    "ipAddress": "192.168.1.100",
    "createdAt": ISODate("2025-12-23T18:00:00.000Z")
  },
  {
    "_id": ObjectId("675a2b3c8d9e0f1a2b3c4d5e"),
    "userId": 15,
    "action": "LOGIN",
    "details": {
      "authMethod": "federated",
      "idpEntityId": "https://idp.ufsc.br/idp/shibboleth"
    },
    "ipAddress": "200.135.37.50",
```

```
    "createdAt": ISODate("2025-12-23T16:45:00.000Z")
  },
  {
    "_id": ObjectId("675a1a2b7c8d9e0a1b2c3d4e"),
    "userId": 8,
    "action": "LOGIN",
    "details": {
      "authMethod": "local",
      "idpEntityId": null
    },
    "ipAddress": "10.0.0.55",
    "createdAt": ISODate("2025-12-23T14:30:00.000Z")
  }
]
```

Código 3.21 – Agregação para contar ações de auditoria por tipo e método de autenticação.

```
db.auditLogs.aggregate([
    {
        $match: {
            createdAt: {
                $gte: ISODate("2025-12-01T00:00:00.000Z"),
                $lt: ISODate("2026-01-01T00:00:00.000Z")
            }
        }
    },
    {
        $group: {
            _id: {
                action: "$action",
                method: "$details.authMethod"
            },
            total: { $sum: 1 }
        }
    },
    {
        $project: {
            _id: 0,
            action: "$_id.action",
            authMethod: "$_id.method",
            total: 1
        }
    },
    { $sort: { action: 1, total: -1 } }
]);
```

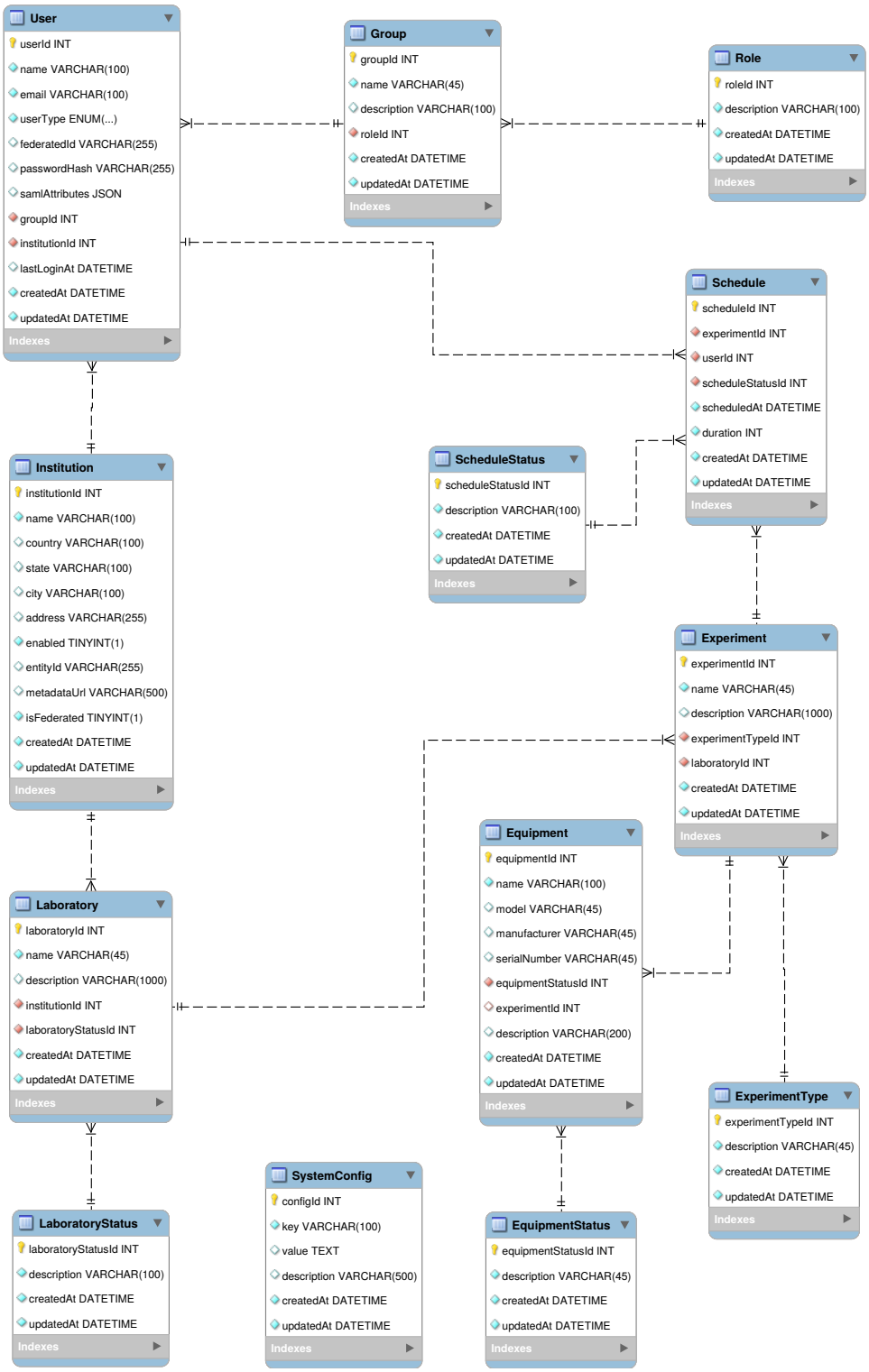
Código 3.22 – Exemplo de saída da agregação de ações de auditoria.

```
[
  { "action": "LOGIN", "authMethod": "federated", "total": 312 },
  { "action": "LOGIN", "authMethod": "local", "total": 87 },
  { "action": "LOGOUT", "authMethod": "federated", "total": 298 },
  { "action": "LOGOUT", "authMethod": "local", "total": 79 },
  { "action": "SCHEDULE_CREATED", "authMethod": null, "total": 156 },
  { "action": "SCHEDULE_CANCELLED", "authMethod": null, "total": 23 }
]
```

3.4.3 Diagrama Entidade-Relacionamento

A Figura 12 apresenta o diagrama entidade-relacionamento completo do banco de dados, ilustrando visualmente a estrutura e os relacionamentos entre as tabelas do sistema.

Figura 12 – Diagrama Entidade-Relacionamento do Sistema eLab.



Fonte: Elaborada pelo autor.

O diagrama foi elaborado utilizando a ferramenta MySQL Workbench. Optou-se por

esta ferramenta porque ela permite a geração automática de *scripts* DDL e a criação do esquema em uma base de dados do servidor MySQL a partir do modelo visual.

3.5 Gerenciamento de Sessões

Esta seção descreve o mecanismo de gerenciamento de sessões do sistema eLab, com ênfase na persistência em MongoDB e na utilização de *cookies* conforme a RFC 6265 (BARTH, 2011).

3.5.1 Justificativa da Persistência em MongoDB

Sistemas web podem armazenar informações de sessão de diferentes formas: em memória da aplicação, em arquivos do sistema de arquivos, ou em banco de dados. Optou-se pelo armazenamento no MongoDB pelos seguintes motivos:

1. **Resiliência a Falhas**: Se a aplicação for reiniciada ou falhar, sessões armazenadas em memória são perdidas. Escolheu-se o MongoDB porque ele garante durabilidade dos dados mesmo em situações de falha;
2. **Escalabilidade Horizontal**: Em ambientes com múltiplas instâncias da aplicação, sessões em memória não são compartilhadas entre instâncias. Optou-se pelo MongoDB porque ele permite que qualquer instância da aplicação valide qualquer sessão, além de oferecer *sharding* nativo para distribuição de dados;
3. **Expiração Automática**: O MongoDB oferece índices TTL que removem automaticamente documentos expirados. Esta funcionalidade elimina a necessidade de processos externos (*cron jobs*) para limpeza de sessões.

3.5.2 Mecanismo de Cookies

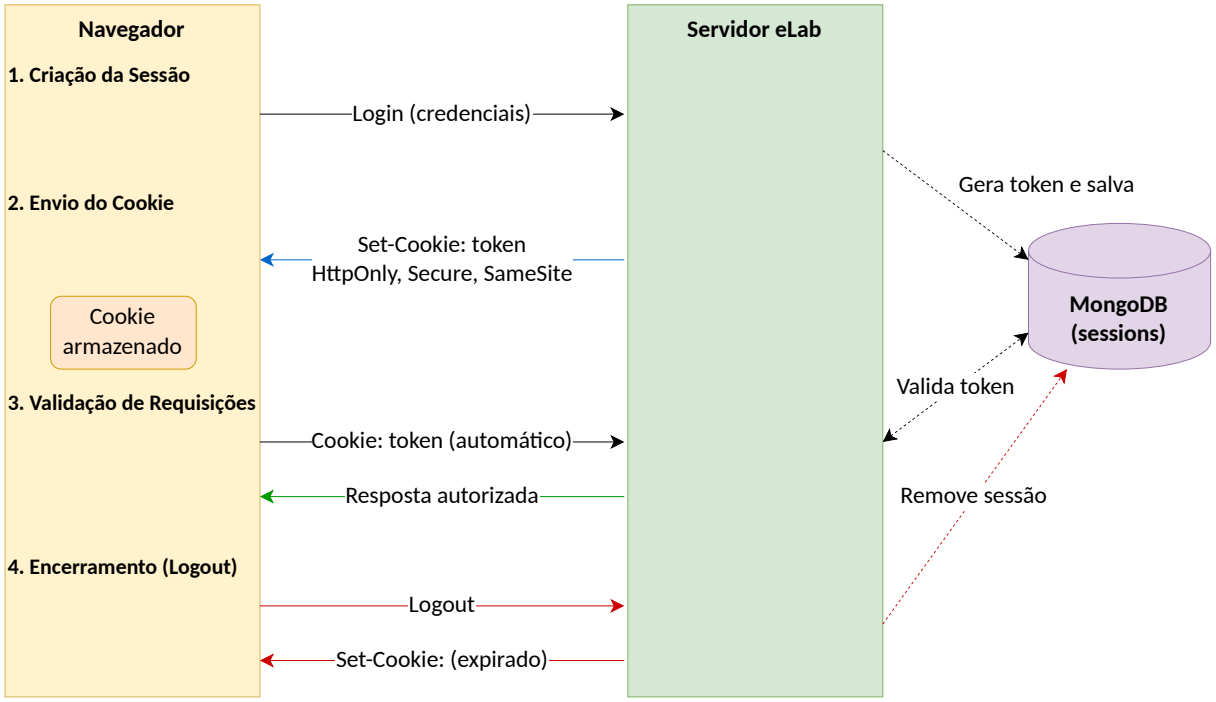
O gerenciamento de sessões utiliza *cookies* HTTP conforme especificado na RFC 6265 (BARTH, 2011). O mecanismo funciona da seguinte forma:

1. **Criação da Sessão**: Após autenticação bem-sucedida, o sistema gera um token de sessão único e o armazena na coleção `sessions` do MongoDB;
2. **Envio do Cookie**: O servidor envia o token ao navegador através de um *cookie* com os atributos `HttpOnly`, `Secure` e `SameSite`. Escolheram-se estes atributos porque eles mitigam ataques de roubo de sessão e *cross-site request forgery*;
3. **Validação de Requisições**: Em cada requisição subsequente, o navegador envia automaticamente o *cookie*. O servidor consulta a coleção `sessions` para validar o token e recuperar o identificador do usuário;

4. **Encerramento**: O *logout* consiste na remoção do documento da coleção `sessions` e na instrução ao navegador para excluir o *cookie*.

A Figura 13 ilustra o funcionamento do mecanismo de *cookies*:

Figura 13 – Mecanismo de *cookies* para gerenciamento de sessões.



Fonte: Elaborada pelo autor.

3.5.3 Ciclo de Vida da Sessão

O ciclo de vida de uma sessão segue as seguintes etapas:

1. **Criação**: Executada após autenticação bem-sucedida. O sistema insere um documento na coleção `sessions` com o token, identificador do usuário, endereço IP, *user-agent* e data de expiração;

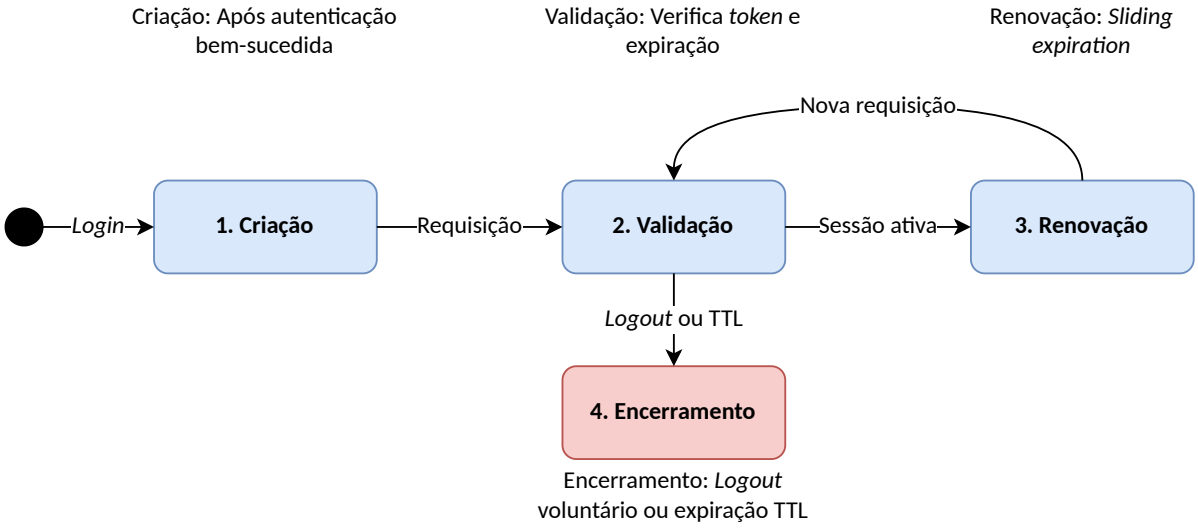
2. **Validação**: Em cada requisição, o sistema verifica se o token existe e se a sessão não expirou. Sessões expiradas são removidas automaticamente pelo índice TTL do MongoDB;

3. **Renovação**: Para sessões ativas, o sistema pode atualizar o campo `expiresAt` para estender a duração da sessão (*sliding expiration*);

4. **Encerramento**: O usuário pode encerrar a sessão voluntariamente (*logout*), ou a sessão pode expirar automaticamente através do índice TTL. No primeiro caso, o documento é removido explicitamente; no segundo, o MongoDB remove-o automaticamente.

A Figura 14 ilustra o ciclo de vida da sessão:

Figura 14 – Ciclo de vida da sessão no sistema eLab.



Fonte: Elaborada pelo autor.

3.5.4 Configuração de Expiração

O tempo de expiração das sessões é configurável através da tabela `SystemConfig` no MySQL ou via variáveis de ambiente. Optou-se por esta abordagem para permitir ajustes sem modificação do código-fonte. Os valores padrão são:

- Sessões federadas: 8 horas de inatividade;
- Sessões locais: 2 horas de inatividade.

3.6 Arquitetura de Autenticação Federada

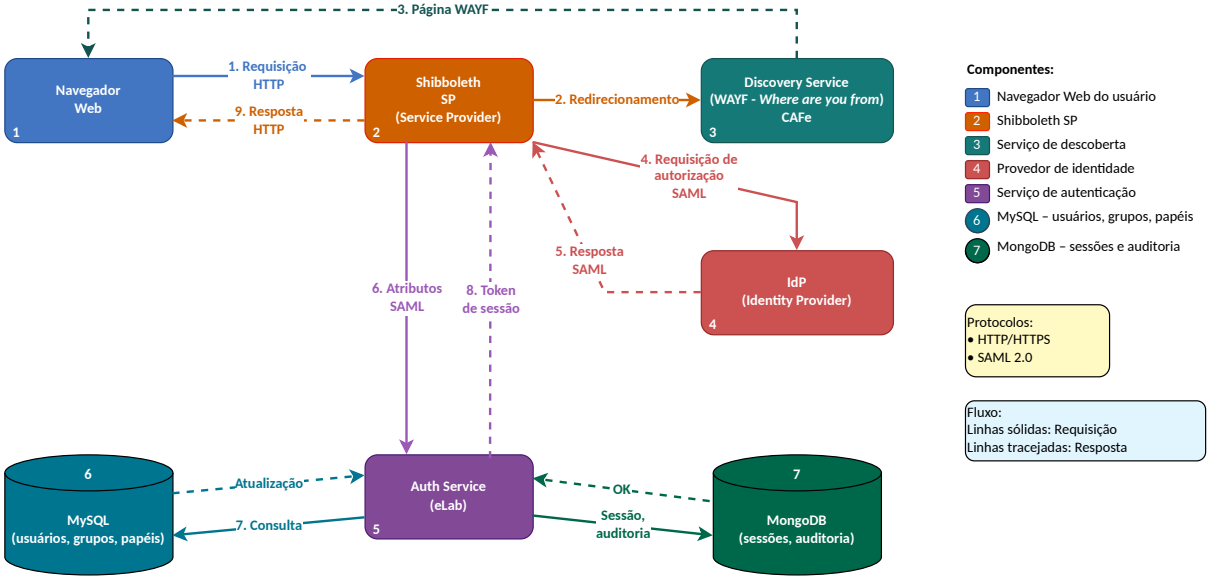
Esta seção apresenta a arquitetura do sistema de autenticação federada do eLab, baseada na integração com a infraestrutura CAFe através do *framework* Shibboleth e do protocolo SAML 2.0.

3.6.1 Componentes do Sistema

A arquitetura é composta pelos seguintes componentes:

1. **Navegador Web**: Interface do usuário que interage com o sistema através de requisições *Hypertext Transfer Protocol* (HTTP)/*Hypertext Transfer Protocol Secure* (HTTPS);

Figura 15 – Arquitetura do sistema de autenticação federada.



Fonte: Elaborada pelo autor.

2. **Shibboleth SP**: Componente que atua como SP, interceptando requisições protegidas e gerenciando o fluxo de autenticação federada. Optou-se pelo Shibboleth porque é o *framework* recomendado pela RNP para integração com a CAFe;

3. **Discovery Service (*Where Are You From* (WAYF))**: Serviço mantido pela CAFe que permite ao usuário selecionar sua instituição de origem;

4. **Identity Provider (IdP)**: Servidor de identidade mantido pela instituição do usuário, responsável pela autenticação e emissão de asserções SAML;

5. **Auth Service**: Serviço do eLab responsável por processar asserções SAML, gerenciar usuários e criar sessões;

6. **Database**: Banco de dados relacional MySQL que armazena usuários, sessões, grupos, papéis e registros de auditoria.

A Figura 15 ilustra a interação entre esses componentes.

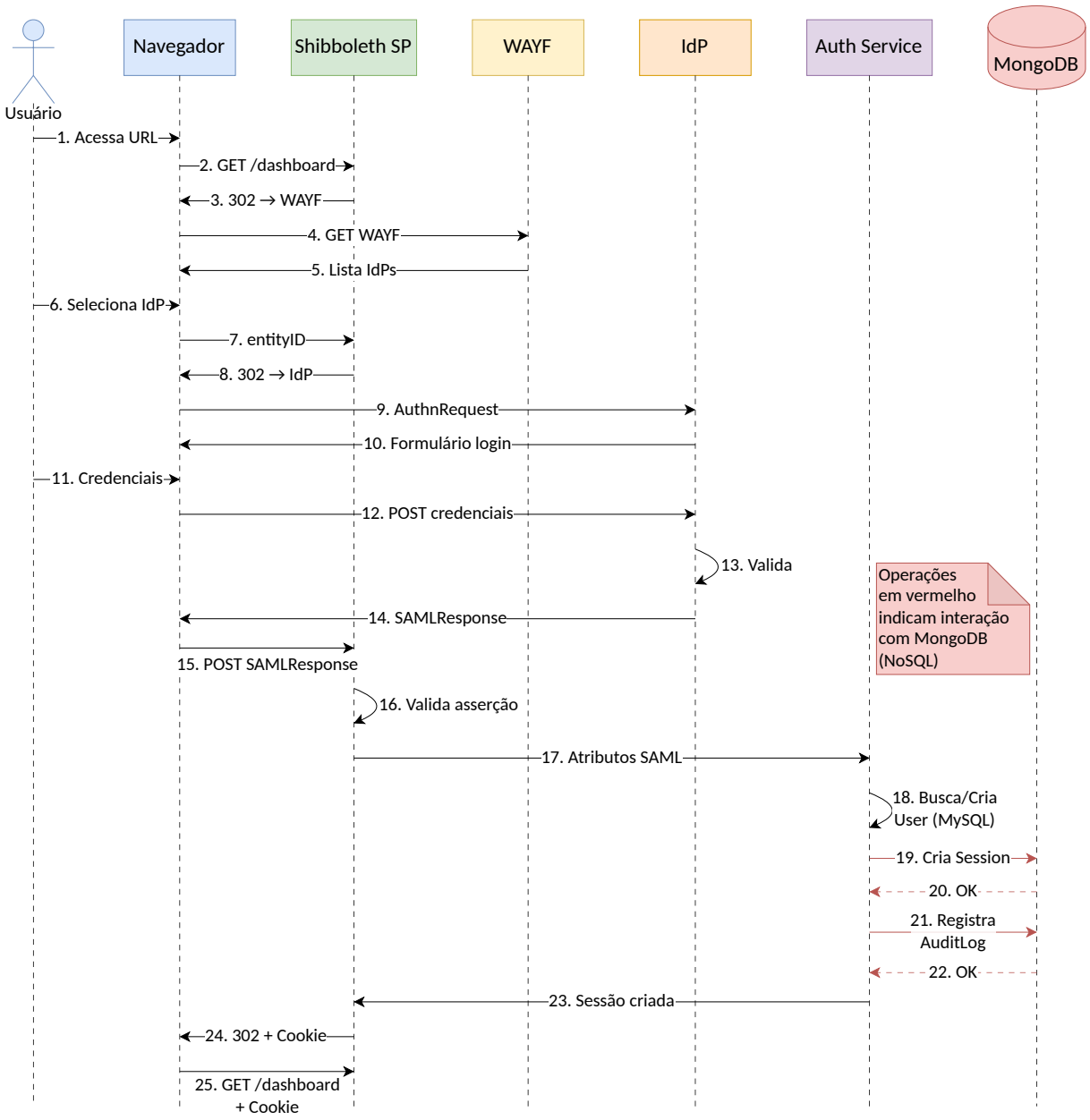
3.7 Fluxo de Autenticação Federada

Esta seção apresenta o fluxo detalhado de autenticação federada, desde o acesso inicial do usuário até o estabelecimento da sessão autenticada.

3.7.1 Visão Geral do Fluxo

Na Figura 16, são ilustradas as etapas do fluxo de autenticação federada.

Figura 16 – Diagrama de sequência da autenticação federada.



Fonte: Elaborada pelo autor.

3.7.2 Etapa 1: Acesso Inicial

O usuário acessa uma URL protegida do sistema eLab através do navegador web. O Shibboleth SP intercepta a requisição e verifica se existe uma sessão válida através da análise de *cookies*.

3.7.3 Etapa 2: Redirecionamento para Discovery Service

Se não existir sessão válida, o Shibboleth SP redireciona o navegador para o Discovery Service da CAFe através de uma resposta HTTP 302. O redirecionamento inclui parâmetros que

identificam o SP e a URL de retorno.

3.7.4 Etapa 3: Seleção do IdP

O Discovery Service apresenta uma lista de instituições participantes da CAFe. O usuário seleciona sua instituição de origem. O Discovery Service retorna o `entityID` do IdP selecionado ao Shibboleth SP.

3.7.5 Etapa 4: Autenticação no IdP

O Shibboleth SP redireciona o navegador para o IdP da instituição selecionada, enviando uma mensagem `AuthnRequest` SAML. O IdP apresenta uma página de *login* ao usuário. Após a inserção das credenciais, o IdP valida-as contra seu sistema de autenticação local (tipicamente LDAP).

3.7.6 Etapa 5: Resposta SAML

Após autenticação bem-sucedida, o IdP redireciona o navegador de volta ao Shibboleth SP através de HTTP POST, enviando uma mensagem `SAMLResponse` contendo a asserção de autenticação. A asserção inclui:

- **NameID**: Identificador único do usuário (`eduPersonPrincipalName`);
- **AttributeStatement**: Atributos do usuário (nome, e-mail, afiliação);
- **AuthnStatement**: Informações sobre o método de autenticação;
- **Signature**: Assinatura digital XML.

3.7.7 Etapa 6: Validação e Provisionamento

O Shibboleth SP valida a asserção SAML verificando a assinatura digital, a validade temporal e as restrições de audiência. Após validação, os atributos são encaminhados ao Auth Service, que executa as seguintes operações:

1. **Busca de Usuário**: Consulta a tabela `User` no MySQL pelo `federatedId`;
2. **Criação ou Atualização**: Se o usuário não existir, cria um novo registro. Se existir, atualiza os atributos;
3. **Mapeamento RBAC**: Determina o grupo apropriado com base nos atributos SAML;
4. **Criação de Sessão**: Insere um documento na coleção `sessions` do MongoDB;
5. **Registro de Auditoria**: Insere um documento na coleção `auditLogs` do MongoDB.

3.7.8 Etapa 7: Acesso Autenticado

O sistema redireciona o navegador para a URL originalmente solicitada, incluindo o *cookie* de sessão. Em requisições subsequentes, o navegador envia automaticamente o *cookie*, permitindo que o sistema valide a sessão e autorize o acesso.

3.8 Mensagens SAML

Esta seção apresenta detalhes técnicos sobre as mensagens SAML trocadas durante o processo de autenticação federada.

3.8.1 AuthnRequest

O `AuthnRequest` é a mensagem enviada pelo SP ao IdP para solicitar autenticação. Os elementos principais incluem:

- `ID`: Identificador único da requisição;
- `IssueInstant`: *Timestamp* da requisição;
- `Issuer`: `entityID` do SP;
- `NameIDPolicy`: Formato desejado para o identificador do usuário;
- `AssertionConsumerServiceURL`: URL de retorno para a resposta.

3.8.2 SAMLResponse

O `SAMLResponse` é a mensagem enviada pelo IdP ao SP contendo a asserção de autenticação. A asserção inclui declarações sobre o sujeito (*Subject*), condições de validade (*Conditions*), declaração de autenticação (*AuthnStatement*) e atributos do usuário (*AttributeStatement*).

3.8.3 Atributos SAML Utilizados

A Tabela 1 apresenta os principais atributos SAML utilizados pelo sistema eLab:

3.9 Integração RBAC com Autenticação Federada

O controle de acesso do eLab segue o modelo RBAC descrito na seção 2.3. Esta seção descreve como esse modelo integra-se com a autenticação federada para determinar automaticamente as permissões dos usuários.

Tabela 1 – Atributos SAML utilizados no sistema.

Atributo	Uso no Sistema
eduPersonPrincipalName	Identificador único persistente. Armazenado em `User.federatedId`
mail	Endereço de e-mail. Armazenado em `User.email`
displayName	Nome completo. Armazenado em `User.name`
eduPersonAffiliation	Afiliação institucional. Utilizado para mapeamento de grupos
schacHomeOrganization	Identificador da instituição. Utilizado para associação com `Institution`

Fonte: Elaborada pelo autor.

3.9.1 Mapeamento de Atributos para Papéis

O sistema implementa um mecanismo de mapeamento que analisa os atributos SAML recebidos e determina o grupo apropriado para cada usuário. O processo segue as etapas:

1. Extração do atributo `eduPersonAffiliation` da asserção;
2. Aplicação de regras de mapeamento configuráveis;
3. Determinação do grupo correspondente;
4. Recuperação do papel associado ao grupo através da relação `Group.roleId`;
5. Atualização do campo `User.groupId`.

3.9.2 Regras de Mapeamento

A Tabela 2 apresenta exemplos de regras de mapeamento:

Tabela 2 – Regras de mapeamento de atributos para grupos.

Condição	Grupo Atribuído	Papel
`eduPersonAffiliation = 'student'`	Estudantes	Estudante
`eduPersonAffiliation = 'faculty'`	Professores	Professor
`eduPersonAffiliation = 'staff'`	Técnicos	Técnico

Fonte: Elaborada pelo autor.

Optou-se por regras configuráveis porque diferentes instituições podem utilizar valores distintos para o atributo `eduPersonAffiliation`. Esta flexibilidade permite adaptação sem modificação do código-fonte.

3.10 Autenticação Local

O sistema oferece autenticação local como alternativa para usuários sem vínculo com instituições federadas.

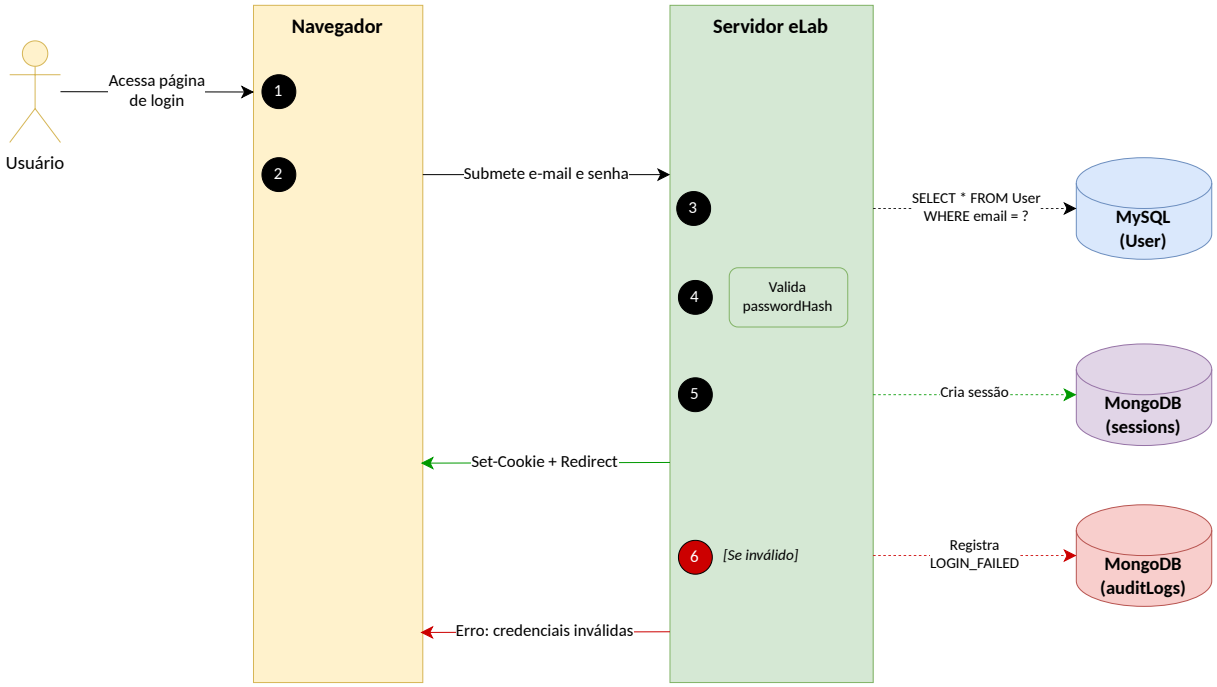
3.10.1 Fluxo de Autenticação Local

O fluxo de autenticação local segue as etapas:

1. Usuário acessa a página de *login* local;
2. Usuário submete e-mail e senha;
3. Sistema consulta a tabela `User` pelo e-mail;
4. Sistema valida o hash da senha fornecida contra o valor armazenado em `User.passwordHash`;
5. Se válido, sistema cria sessão e redireciona para a página solicitada;
6. Se inválido, sistema registra tentativa falha em `AuditLog` e exibe mensagem de erro.

A Figura 17 ilustra o fluxo de autenticação local:

Figura 17 – Fluxo de autenticação local.



Fonte: Elaborada pelo autor.

3.10.2 Armazenamento de Senhas

Para usuários locais, o campo `User.passwordHash` armazena um hash criptográfico da senha. Optou-se por algoritmos de hash com *salt* (como bcrypt ou Argon2) porque eles dificultam ataques de força bruta e de tabelas *rainbow*.

3.11 Auditoria

O sistema implementa um mecanismo de auditoria que registra operações sensíveis para fins de rastreabilidade e conformidade.

3.11.1 Eventos Registrados

A tabela `AuditLog` registra os seguintes tipos de eventos:

- **LOGIN**: Autenticação bem-sucedida (local ou federada);
- **LOGIN_FAILED**: Tentativa de autenticação falhada;
- **LOGOUT**: Encerramento voluntário de sessão;
- **SESSION_EXPIRED**: Expiração automática de sessão;
- **USER_CREATED**: Criação de novo usuário;
- **USER_UPDATED**: Atualização de dados de usuário;
- **EXPERIMENT_ACCESSED**: Acesso a experimento;
- **SCHEDULE_CREATED**: Criação de agendamento.

3.11.2 Informações Registradas

Cada registro de auditoria inclui:

- Identificador do usuário (quando autenticado);
- Tipo de ação (referência à tabela `AuditLogAction`);
- Recurso afetado;
- Detalhes adicionais em formato JSON;
- Endereço IP de origem;
- *User-Agent* do navegador;
- Data e hora do evento.

3.11.3 Consulta de Logs

O sistema permite consultas aos logs de auditoria para análise de atividades e investigação de incidentes. Administradores podem filtrar registros por usuário, tipo de ação, período ou endereço IP.

À seguir, estão exemplos de consultas, escritas na linguagem JavaScript, que podem ser executadas no MongoDB Shell, e seus resultados esperados:

Código 3.23 – Consulta para listar eventos de login de um usuário.

```
// Consulta para listar eventos de login de um usuário específico
db.auditLogs.find({
    userId: 42,
    action: { $in: ["LOGIN", "LOGIN_FAILED", "LOGOUT"] },
    createdAt: {
        $gte: ISODate("2025-12-01T00:00:00.000Z"),
        $lt: ISODate("2026-01-01T00:00:00.000Z")
    }
}).sort({ createdAt: -1 }).limit(10);
```

Código 3.24 – Exemplo de saída da consulta de logs de usuário.

```
[
  {
    "_id": ObjectId("6766a1b2c3d4e5f6a7b8c9d0"),
    "userId": 42,
    "action": "LOGOUT",
    "details": { "authMethod": "federated" },
    "ipAddress": "200.135.37.65",
    "createdAt": ISODate("2025-12-20T16:45:30.000Z")
  },
  {
    "_id": ObjectId("6766a0a1b2c3d4e5f6a7b8c9"),
    "userId": 42,
    "action": "LOGIN",
    "details": {
      "authMethod": "federated",
      "idpEntityId": "https://shibboleth.ifsc.edu.br/idp/shibboleth"
    },
    "ipAddress": "200.135.37.65",
    "createdAt": ISODate("2025-12-20T08:30:15.000Z")
  },
  {
    "_id": ObjectId("67658f90a1b2c3d4e5f6a7b8"),
    "userId": 42,
    "action": "LOGIN_FAILED",
    "details": { "authMethod": "local", "reason": "invalid_password" },
    "ipAddress": "189.44.120.33",
    "createdAt": ISODate("2025-12-19T14:22:45.000Z")
  }
]
```

O resultado exemplificado no Código 3.24 apresenta três registros de auditoria do usuário com identificador 42: um evento de *logout*, um *login* bem-sucedido via autenticação federada (CAFe/IFSC) e uma tentativa de *login* local falha por senha inválida. Cada registro inclui o endereço IP de origem, auxiliando no rastreio do acesso.

Código 3.25 – Agregação para detectar tentativas de login falhas por IP.

```
// Consulta para listar tentativas de login falhas por IP
db.auditLogs.aggregate([
    {
        $match: {
            action: "LOGIN_FAILED",
            createdAt: { $gte: ISODate("2025-12-20T00:00:00.000Z") }
        }
    },
    {
        $group: {
            _id: "$ipAddress",
            totalAttempts: { $sum: 1 },
            lastAttempt: { $max: "$createdAt" }
        }
    },
    { $match: { totalAttempts: { $gte: 3 } } },
    { $sort: { totalAttempts: -1 } }
]);
```

Código 3.26 – Exemplo de saída da agregação de tentativas falhas.

```
[
  {
    "_id": "189.44.120.33",
    "totalAttempts": 15,
    "lastAttempt": ISODate("2025-12-20T14:35:22.000Z")
  },
  {
    "_id": "45.227.89.102",
    "totalAttempts": 8,
    "lastAttempt": ISODate("2025-12-20T11:18:45.000Z")
  },
  {
    "_id": "177.52.33.201",
    "totalAttempts": 5,
    "lastAttempt": ISODate("2025-12-20T09:42:10.000Z")
  }
]
```

Já o resultado exemplificado no Código 3.26 é útil para detectar possíveis ataques de força bruta. São agrupadas as tentativas falhas por endereço IP e filtradas apenas aquelas com três ou mais tentativas no período. No exemplo, o IP 189.44.120.33 apresenta 15 tentativas falhas, indicando comportamento suspeito que pode justificar bloqueio temporário.

4 CONCLUSÃO

Este trabalho teve como objetivo geral a modelagem de um sistema de autenticação e autorização para uma plataforma de laboratórios remotos distribuídos. A proposta desenvolvida atende a esse objetivo ao apresentar uma arquitetura híbrida de dados que suporta autenticação federada via protocolo SAML 2.0, integração com a infraestrutura CAFe e controle de acesso baseado no modelo RBAC.

No âmbito da autenticação federada, foram analisados e documentados os protocolos SAML 2.0 e OAuth 2.0, bem como a integração com a federação CAFe por meio do *framework* Shibboleth. Adicionalmente, o fluxo de autenticação foi detalhado desde o acesso inicial até o estabelecimento da sessão, incluindo o mapeamento de atributos recebidos do provedor de identidade para o sistema local.

Em relação à distribuição de laboratórios entre múltiplas instituições, a estrutura desenvolvida suporta a participação de diversas entidades de ensino e pesquisa. Cada instituição cadastrada pode registrar seus próprios laboratórios, experimentos e equipamentos, mantendo a autonomia sobre seus recursos. Sob essa ótica, a tabela `Institution` armazena não apenas o identificador e o nome da instituição, mas também seus dados de localização geográfica, incluindo país, estado, cidade e endereço. Por conseguinte, a associação entre as tabelas `Laboratory` e `Institution`, estabelecida através da chave estrangeira `institutionId`, permite identificar a localização física de cada laboratório remoto. Essa estrutura viabiliza consultas que relacionam experimentos disponíveis com suas respectivas localizações, possibilitando que usuários identifiquem a origem geográfica dos recursos que desejam acessar.

No que concerne ao armazenamento de dados, a arquitetura híbrida adotada combina dois sistemas de gerenciamento de banco de dados. As entidades estruturadas, como usuários, grupos, papéis, instituições, laboratórios, experimentos, equipamentos e agendamentos, são armazenadas no MySQL com integridade referencial. Paralelamente, dados de sessão e registros de auditoria são mantidos no MongoDB, que oferece esquema flexível para atributos variáveis e índices TTL para expiração automática.

Somado a isso, a arquitetura híbrida adotada apresenta vantagens específicas para o domínio de laboratórios remotos. O componente relacional garante consistência entre usuários, grupos, papéis, instituições, laboratórios e experimentos. O componente não relacional, por sua vez, oferece flexibilidade para armazenamento de atributos SAML variáveis e escalabilidade horizontal para registros de auditoria.

4.1 Limitações da Versão Inicial

A primeira versão do sistema apresenta limitações que devem ser consideradas. O modelo de confiança adotado é centralizado, ou seja, embora a identidade dos usuários possa originar-se de provedores externos, a validação, o gerenciamento de sessões e a autorização são executados exclusivamente no sistema eLab. Essa abordagem simplifica a implementação inicial, porém limita a autonomia das instituições participantes.

A integração com provedores de identidade está restrita ao protocolo SAML 2.0 e ao OAuth 2.0. Outros protocolos de autenticação federada, como OpenID Connect, não foram contemplados nesta versão. Essa limitação pode dificultar a integração com instituições que não participam da federação CAFe ou que não oferecem suporte aos protocolos implementados.

O mapeamento de atributos SAML para o modelo RBAC depende de regras configuradas manualmente no sistema. Não foi implementado um mecanismo automático de descoberta ou sincronização de grupos e papéis com base em políticas externas. A ausência desse mecanismo requer intervenção administrativa para ajustes na atribuição de permissões.

A escalabilidade horizontal do componente relacional não foi abordada nesta versão. A estrutura MySQL foi projetada para operação em servidor único, o que pode representar um gargalo em cenários com elevado número de usuários ou instituições simultâneas. A migração para configurações de replicação ou *sharding* demandaria alterações na arquitetura proposta.

4.2 Trabalhos Futuros

Como continuidade deste trabalho, identificam-se as seguintes possibilidades de desenvolvimento:

- Implementação de suporte ao protocolo OpenID Connect, ampliando as opções de integração com provedores de identidade;
- Avaliação de estratégias de escalabilidade horizontal para o componente relacional, incluindo replicação de leitura e particionamento de dados;
- Implementação completa de federação de autorização, permitindo que instituições participantes definam políticas de acesso próprias;
- Integração com sistemas de monitoramento e alertas para detecção de comportamentos anômalos nos registros de auditoria;
- Desenvolvimento de uma interface administrativa para gerenciamento de instituições, laboratórios e experimentos;

- Implementação de mecanismos de *Single Logout* (SLO) para encerramento sincronizado de sessões entre o sistema eLab e os provedores de identidade federados.
- Implementação da federação completa de auditoria, permitindo que instituições participantes armazenem seus registros e definam políticas de auditoria próprias.

A modelagem desenvolvida constitui a base para a implementação do sistema eLab. A arquitetura proposta oferece flexibilidade para incorporar as funcionalidades identificadas como trabalhos futuros, mantendo compatibilidade com a estrutura de dados definida neste trabalho.

REFERÊNCIAS

AGGOUNE, A.; NAMOUNE, M. S. A method for transforming object-relational to document-oriented databases. In: *2020 2nd International Conference on Mathematics and Information Technology (ICMIT)*. [S.l.: s.n.], 2020. p. 154–158. 25, 26

AITOR, V.-M. et al. Toward widespread remote laboratories: Evaluating the effectiveness of a replication-based architecture for real-world multiinstitutional usage. *IEEE Access*, v. 10, p. 86298–86317, 2022. 15, 19, 46

ALMEIDA, D. et al. Performance comparison of redis, memcached, mysql, and postgresql: A study on key-value and relational databases. In: *2023 Second International Conference On Smart Technologies For Smart Nation (SmartTechCon)*. [S.l.: s.n.], 2023. p. 902–907. 22

ALVES, G. R. et al. A roadmap for the visir remote lab. *European Journal of Engineering Education*, Taylor & Francis, v. 48, n. 5, p. 880–898, 2023. Disponível em: <https://doi.org/10.1080/03043797.2022.2054312>. 18

AMD. *Vivado Design Suite*. 2024. Online. Disponível em: <https://www.xilinx.com/products/design-tools/vivado.htmloverview>. Acesso em: 2 de abr. 2024. 15

ARIZA, J. Álvarez; GALVIS, C. N. Raspycontrol lab: A fully open-source and real-time remote laboratory for education in automatic control systems using raspberry pi and python. *HardwareX*, v. 13, p. e00396, 2023. ISSN 2468-0672. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2468067223000032>. 15, 18

BARTH, A. *HTTP State Management Mechanism*. RFC Editor, 2011. RFC 6265. (Request for Comments, 6265). Disponível em: <https://www.rfc-editor.org/info/rfc6265>. 64

BAČA, J.; HAULIŠ, M.; ŠUBA, S. Remote laboratory for digital systems. In: *2011 14th International Conference on Interactive Collaborative Learning*. [S.l.: s.n.], 2011. p. 620–625. 16

BEKASIEWICZ, A. et al. Application of open-hardware-based solutions for rapid transition from stationary to the remote teaching model during pandemic. *IEEE Transactions on Education*, v. 64, n. 3, p. 299–307, 2021. 15

BELLO, A. J.; DIYAN, M.; ASGHAR, I. Zero trust implementation for legacy systems using dynamic microsegmentation, role-based access control (rbac), and attribute-based access control (abac). In: *2025 4th International Conference on Computing and Information Technology (ICCIT)*. [S.l.: s.n.], 2025. p. 181–189. 44

BENMOHAMED, H.; LELEVE, A.; PREVOT, P. Remote laboratories: new technology and standard based architecture. In: *Proceedings. 2004 International Conference on Information and Communication Technologies: From Theory to Applications, 2004.* [S.l.: s.n.], 2004. p. 101–102. 19

CAPRIS, T. et al. Comparison of sql and nosql databases with different workloads: Mongodb vs mysql evaluation. In: *2022 International Conference on Data Analytics for Business and Industry (ICDABI)*. [S.l.: s.n.], 2022. p. 214–218. 22

CHACóN, J. et al. An integrated framework for the agile development and deployment of low cost remote laboratories. *Multimedia Tools and Applications*, v. 84, n. 25, p. 29207–29227, 2025. ISSN 1573-7721. 16, 47

CHAGAS, M.; MELLO, E. R. de; WANGHAM, M. S. Cafe expresso: Comunidade acadêmica federada para experimentação usando framework shibboleth. In: *Anais do XIV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)*. Belo Horizonte, MG, Brasil: SBC, 2014. p. 405–408. 40

CHEN, H.-C. A hierarchical virtual role assignment for negotiation-based rbac scheme. In: *2015 10th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*. [S.l.: s.n.], 2015. p. 538–543. 42, 44

CHEN, J.-K.; LEE, W.-Z. The transformation of relational database to wide column store database. In: *2020 International Symposium on Computer, Consumer and Control (IS3C)*. [S.l.: s.n.], 2020. p. 384–386. 28

CHICKERUR, S.; GOUDAR, A.; KINNERKAR, A. Comparison of relational database with document-oriented database (mongodb) for big data applications. In: *2015 8th International Conference on Advanced Software Engineering Its Applications (ASEA)*. [S.l.: s.n.], 2015. p. 41–47. 23, 24, 25

CHITPINITYON, S.; TOSSA, M. New approach for single sign-on improvement using load distribution method. In: *2021 Research, Invention, and Innovation Congress: Innovation Electricals and Electronics (RI2C)*. [S.l.: s.n.], 2021. p. 44–47. 39

Coordenação de Aperfeiçoamento de Pessoal de Nível Superior. *Acesso remoto ao Portal de Periódicos está disponível aos membros da CAFe*. CAPES, 2020. Acesso em: 30 nov. 2025. Disponível em: <https://www.gov.br/capes/pt-br/assuntos/noticias/acesso-remoto-ao-portal-de-periodicos-esta-disponivel-aos-membros-da-cafe>. 41

DHARMASIRI, H. M. L.; GOONETILLAKE, M. D. J. S. A federated approach on heterogeneous nosql data stores. In: *2013 International Conference on Advances in ICT for Emerging Regions (ICTer)*. [S.l.: s.n.], 2013. p. 234–239. 30, 31, 32, 33, 34

ELMASRI, R.; NAVATHE, S. B. *Sistemas de Banco de Dados*. 6th. ed. [S.l.]: Pearson, 2010. ISBN 8579360854. 21, 23, 48

FLORES, A. et al. Remote laboratory for teaching digital design using a vpn and embedded system. In: *2021 IEEE XXVIII International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*. [S.l.: s.n.], 2021. p. 1–4. 16

FUGKEAW, S.; MANPANPANICH, P.; JUNTAPREMJITT, S. A development of multi-sso authentication and rbac model in the distributed systems. In: *2007 2nd International Conference on Digital Information Management*. [S.l.: s.n.], 2007. v. 1, p. 297–302. 43

GAO, W.; WEN, Y.; ZHANG, H. An optimization method of federated database join query based on computational push-down. In: *2024 IEEE 2nd International Conference on Control, Electronics and Computer Technology (ICCECT)*. [S.l.: s.n.], 2024. p. 225–229. 33

GHDL. *GHDL*. 2017. Online. Disponível em: <http://ghdl.free.fr>. Acesso em: 2 de abr. 2024. 15

GUEYE, A. D. et al. Decentralized authentication method for accessing pedagogical resources in a cloud computing based virtual organization. In: *2014 International Conference on Interactive Collaborative Learning (ICL)*. [S.l.: s.n.], 2014. p. 656–662. 37, 38

HAMOUDA, S. Seamless transition: Migrating from relational databases to document-oriented databases. In: *2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*. [S.l.: s.n.], 2023. v. 1, p. 883–888. 26

HASSAN, M. A. Relational and nosql databases: The appropriate database model choice. In: *2021 22nd International Arab Conference on Information Technology (ACIT)*. [S.l.: s.n.], 2021. p. 1–6. 22, 23, 24, 25, 26, 29, 48

HEUSER, C. A. *Projeto de Banco de Dados*. 4th. ed. [S.l.]: Sagra Luzzatto, 2009. ISBN 8577803821. 21, 22

HONG, H. lian; HAI, Q. lin. Application and implementation of role-based access control in b/s systems. In: *2010 5th International Conference on Computer Science Education*. [S.l.: s.n.], 2010. p. 1155–1157. 41, 42, 44

HOSSAIN, N. et al. Oauth-sso: A framework to secure the oauth-based sso service for packaged web applications. In: *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. [S.l.: s.n.], 2018. p. 1575–1578. 37

HU, H.; GUO, Z. The application of cross-domain single sign-on in municipal portal. In: *2013 IEEE International Conference of IEEE Region 10 (TENCON 2013)*. [S.l.: s.n.], 2013. p. 1–4. 38

HU, J.; SUN, Q.; CHEN, H. Application of single sign-on (sso) in digital campus. In: *2010 3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT)*. [S.l.: s.n.], 2010. p. 725–727. 37, 38

HUANG, T.; GUO, F. Research on single sign-on technology for educational administration information service platform. In: *2021 3rd International Conference on Computer Communication and the Internet (ICCCI)*. [S.l.: s.n.], 2021. p. 69–72. 38

IFSC, C. F. *Sobre*. 2024. Online. Disponível em: <https://visir.florianopolis.ifsc.edu.br/visir/index.php/pt?page=AboutPage>. Acesso em: 22 de jul. 2024. 20

Instituto Federal de São Paulo. *CAFe - Comunidade Acadêmica Federada*. 2024. Acesso em: 30 nov. 2025. Disponível em: <https://www.ifsp.edu.br/ceua/122-assuntos/desenvolvimento-institucional/tecnologia-da-informacao/diretoria-de-infraestrutura-e-redes/2222-cafe>. 40

Intel. *Intel® FPGA Academic Program Teaching Materials*. 2024. Online. Disponível em: <https://www.intel.com/content/www/us/en/developer/topic-technology/fpga-academic/tools.html>. Acesso em: 4 de abr. 2024. 15

JIAN, Y. An improved scheme of single sign-on protocol. In: *2009 Fifth International Conference on Information Assurance and Security*. [S.l.: s.n.], 2009. v. 1, p. 495–498. 36, 39

JUN, Y.; ZHISHU, L.; YANYAN, M. Json based decentralized sso security architecture in e-commerce. In: *2008 International Symposium on Electronic Commerce and Security*. [S.l.: s.n.], 2008. p. 471–475. 37

KALENDAR, M. et al. The ubilab framework for remote laboratories. In: *2023 30th International Conference on Systems, Signals and Image Processing (IWSSIP)*. [S.l.: s.n.], 2023. p. 1–5. 16

KAMRA, V.; SHEKHAWAT, A. Enhancing user authentication with single sign-on and passkey integration. In: *2025 International Conference on Engineering Innovations and Technologies (ICoEIT)*. [S.l.: s.n.], 2025. p. 495–499. 36

KANADE, A. S.; GOPAL, A. Choosing right database system: Row or column-store. In: *2013 International Conference on Information Communication and Embedded Systems (ICICES)*. [S.l.: s.n.], 2013. p. 16–20. 26, 28

KARNITIS, G.; ARNICANS, G. Migration of relational database to document-oriented database: Structure denormalization and data transformation. In: *2015 7th International Conference on Computational Intelligence, Communication Systems and Networks*. [S.l.: s.n.], 2015. p. 113–118. 26

KARUNANITHI, M. D.; KIRUTHIKA, B. Single sign-on and single log out in identity. In: *International Conference on Nanoscience, Engineering and Technology (ICONSET 2011)*. [S.l.: s.n.], 2011. p. 607–611. 34, 38

KAUR, G. K.; SINGLA, S.; KHAWAS, V. Database management system: A study of increasing impact of nosql databases. In: *2023 International Conference on Advanced Computing Communication Technologies (ICACCTech)*. [S.l.: s.n.], 2023. p. 370–374. 24, 25, 29

KRISHAN, K.; GUPTA, G.; BHATHAL, G. S. Striking the balance: Comprehensive insights into data consistency in nosql realms. In: *2024 11th International Conference on Computing for Sustainable Global Development (INDIACom)*. [S.l.: s.n.], 2024. p. 715–720. 29

KUHN, D. R.; COYNE, E. J.; WEIL, T. R. Adding attributes to role-based access control. *Computer*, v. 43, n. 6, p. 79–81, 2010. 41, 42, 43, 44, 47

LabsLand. *O que a LabsLand oferece?* 2024. Online. Disponível em: <https://labsland.com/pt$_BR/about$>. $Acessoem : 22dejul.2024.19$

LEE, K. K.-Y.; TANG, W.-C.; CHOI, K.-S. Alternatives to relational database: Comparison of nosql and xml approaches for clinical data storage. *Computer Methods and Programs in Biomedicine*, v. 110, n. 1, p. 99–109, 2013. ISSN 0169-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169260712002805>. 24

LIU, M.; GUO, H.-Q.; SU, J.-D. An attribute and role based access control model for web services. In: *2005 International Conference on Machine Learning and Cybernetics*. [S.l.: s.n.], 2005. v. 2, p. 1302–1306 Vol. 2. 43, 44

LOBO, J. A remote reconfigurable logic laboratory for basic digital design. *ISR - Institute of Systems and Robotics, DEEC, FCT, University of Coimbra*, 2011. 15, 18

MATHOLIA, H.; ADEDAYO, O. M. Investigating nosql injection attacks on mongodb web applications. In: *2025 13th International Symposium on Digital Forensics and Security (ISDFS)*. [S.l.: s.n.], 2025. p. 1–6. 30

MathWorks. *HDL Coder*. 2024. Online. Disponível em: <https://www.mathworks.com/products/hdl-coder.html>. Acesso em: 4 de abr. 2024. 15

MAXFIELD, C. *The Design Warrior's Guide to FPGAs*. 1st. ed. [S.l.]: Newnes, 2004. ISBN 0750676043. 15

MAYOZ, C. A. et al. Fpga remote laboratory: experience of a shared laboratory between upna and unifesp. In: *2020 XIV Technologies Applied to Electronics Teaching Conference (TAEE)*. [S.l.: s.n.], 2020. p. 1–8. 16

National Instruments. *What is NI LabVIEW*. 2024. Online. Disponível em: <https://www.ni.com/en/shop/labview.html>. Acesso em: 4 de abr. 2024. 15

NEDIC, Z.; MACHOTKA, J.; NAFALSKI, A. Remote laboratory netlab for effective interaction with real equipment over the internet. In: *2008 Conference on Human System Interactions*. [S.l.: s.n.], 2008. p. 846–851. 20

NENOV, T. R.; EVSTATIEV, B. I.; KADIROVA, S. Y. A remote lab for experimental study of dc motors. In: *2023 18th Conference on Electrical Machines, Drives and Power Systems (ELMA)*. [S.l.: s.n.], 2023. p. 1–4. 15, 18

NETO, L. A. B. *Um Mecanismo de Integração de Identidades Federadas entre Shibboleth e SimpleSAMLphp para aplicações de Nuvens*. Dissertação (Dissertação (Mestrado em Engenharia de Eletricidade)) — Universidade Federal do Maranhão (UFMA), São Luís, MA, Brasil, 2014. 40, 41

NISHIOKA, S.; OKABE, Y. Centralized control of account migration at single sign-on in shibboleth. In: *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*. [S.l.: s.n.], 2020. p. 1597–1603. 37

ORDUñA, P. et al. Labsland: A sharing economy platform to promote educational remote laboratories maintainability, sustainability and adoption. In: *2016 IEEE Frontiers in Education Conference (FIE)*. [S.l.: s.n.], 2016. p. 1–6. 15, 19, 46

PEDRONI, V. A. *Circuit Design and Simulation with VHDL, Second Edition*. 2nd. ed. [S.l.]: The MIT Press, 2010. ISBN 0262014335. 15

PINO, C.; RAVIDá, S.; SCIBILIA, S. Evaluation of federated database for distributed applications in e-government. In: *2013 7th International Conference on Application of Information and Communication Technologies*. [S.l.: s.n.], 2013. p. 1–5. 30, 31, 32, 33

POKORNY, J. Nosql databases: A step to database scalability in web environment. *International Journal of Web Information Systems*, v. 9, n. 1, p. 69 – 82, 2013. ISSN 17440084. CAP theorem;Computing;Data distribution;Horizontal scaling;Nosql database;Vertical scaling;Weak consistency;. Disponível em: <http://dx.doi.org/10.1108/17440081311316398>. 22, 24, 47, 49

PRAKOSO, R. H. P.; AZIZAH, F. N. The study of data modeling methodologies for column-oriented databases. In: *2023 IEEE International Conference on Data and Software Engineering (ICoDSE)*. [S.l.: s.n.], 2023. p. 238–243. 28

QIYUE, W. Research on column-store databases optimization techniques. In: *2015 International Conference on Logistics, Informatics and Service Sciences (LISS)*. [S.l.: s.n.], 2015. p. 1–7. 27

RAMAKRISHNAN, R.; GEHRKE, J. *Sistemas de Gerenciamento de Bancos de Dados*. 3rd. ed. [S.l.]: McGraw-Hill, 2008. ISBN 8577260275. 21, 22, 23

RAMZAN, S. et al. Intelligent data engineering for migration to nosql based secure environments. *IEEE Access*, v. 7, p. 69042–69057, 2019. 16, 24, 25, 48

Rede Nacional de Ensino e Pesquisa. *Catálogo de Serviços: Comunidade Acadêmica Federada (CAFe)*. RNP, 2015. Acesso em: 30 nov. 2025. Disponível em: <https://memoria.rnp.br/$_arquivo/servicos/catalogo2.pdf$>. 40, 41

Rede Nacional de Ensino e Pesquisa. *Adesão IdP: Processo de Adesão à CAFe*. RNP, 2024. Acesso em: 30 nov. 2025. Disponível em: <https://ajuda.rnp.br/cafe/processo-de-adesao/adesao-idp>. 41

RELLE. *Sobre*. 2024. Online. Disponível em: <http://relle.ufsc.br/about>. Acesso em: 22 de jul. 2024. 20

RODRIGUES, C. F.; ROCHA, L. F. da; RIBEIRO, R. de Q. A contribuição da rnp para a conformidade dos provedores de identidade da federação cafe ao sirtfi. In: *Anais do Workshop de Gestão de Identidades Digitais (WGID)*. Porto Alegre, RS, Brasil: SBC, 2019. Disponível em: <https://sol.sbc.org.br/index.php/wgid/article/view/14029>. 40, 41, 46

SAKIB, F. F. et al. Nosql database selection process: An integrated proof of concept and comparison for targeted document store databases. In: *2025 4th International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*. [S.l.: s.n.], 2025. p. 151–155. 30

SAMARTA, M. T.; GUNAWAN, A. A. S.; SYAHPUTRA, M. E. Systematic literature review and comparative performance analysis of sql and nosql databases in big data applications. In: *2024 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)*. [S.l.: s.n.], 2024. p. 218–222. 29, 30

SERGEY, K. The comparative analysis of technologies and software for single sign-on. In: *2023 19th International Asian School-Seminar on Optimization Problems of Complex Systems (OPCS)*. [S.l.: s.n.], 2023. p. 44–47. 35, 36, 37, 39

SERRANO, P. A. M.; OñATE, J. J. S. Integration of restful api to student information system for secured data sharing and single sign-on. In: *2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*. [S.l.: s.n.], 2021. p. 1–6. 38

SHAIKH, N.; KASAT, K.; JADHAV, S. Secured authentication by single sign on (sso): A big picture. In: *2022 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*. [S.l.: s.n.], 2022. p. 951–955. 34, 35, 36, 39

SHARMA, A.; KAUR, P. Implementing a multitenant system using a document-based nosql database. In: *2021 5th International Conference on Information Systems and Computer Networks (ISCON)*. [S.l.: s.n.], 2021. p. 1–5. 25

SHARMA, A.; SHARMA, S.; DAVE, M. Identity and access management- a comprehensive study. In: *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*. [S.l.: s.n.], 2015. p. 1481–1485. 34, 35, 36, 37, 46

SHARMA, P.; SIHAG, V. K. Hybrid single sign-on protocol for lightweight devices. In: *2016 IEEE 6th International Conference on Advanced Computing (IACC)*. [S.l.: s.n.], 2016. p. 679–684. 37

SHETH, A. P.; LARSON, J. A. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 22, n. 3, p. 183–236, set. 1990. ISSN 0360-0300. Disponível em: <https://doi-org.ez130.periodicos.capes.gov.br/10.1145/96602.96604>. 30, 31, 32, 33, 34

SHI, S.; YAN, W. Q.; LI, M. Z. A secure sso protocol without clock synchronization. In: *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*. [S.l.: s.n.], 2010. v. 3, p. V3–422–V3–424. 39

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. *Sistemas de Banco de Dados*. 5th. ed. [S.l.]: Elsevier, 2006. ISBN 8535211078. 21, 23

SIREGAR, R. A. S.; AZIZAH, F. N. Migration from relational database to column-oriented nosql database. In: *2023 IEEE International Conference on Data and Software Engineering (ICoDSE)*. [S.l.: s.n.], 2023. p. 174–179. 28

Stephen Williams. *ICARUS Verilog*. 2019. Online. Disponível em: <https://steveicarus.github.io/iverilog/>. Acesso em: 3 de abr. 2024. 15

TRINQUET, V. et al. Optical materials discovery and design with federated databases and machine learning. *Faraday Discussions*, Royal Society of Chemistry (RSC), v. 256, p. 459–482, 2025. ISSN 1364-5498. Disponível em: <http://dx.doi.org/10.1039/D4FD00092G>. 30, 33

UFSC, C. A. *Sobre*. 2024. Online. Disponível em: <https://rexlab.ufsc.br/about/>. Acesso em: 22 de jul. 2024. 20

VERA, R. G. et al. A flexible framework for the deployment of stem real remote laboratories in digital electronics and control systems. *IEEE Access*, v. 12, p. 14563–14579, 2024. 16

WANG, Y.; WEN, Q.; ZHANG, H. A single sign-on scheme for cross domain web applications using identity-based cryptography. In: *2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing*. [S.l.: s.n.], 2010. v. 1, p. 483–485. 38

WANGHAM, M. S. et al. Geração de certificados digitais a partir da autenticação federada shibboleth. In: *Anais do Workshop de Gestão de Identidade (WGID)*. Curitiba, PR, Brasil: SBC, 2012. p. 1–12. 40, 41

ZHANG, G.; CHEN, M.; SHEN, M. Authorization model of sso for a distributed environment based on the attributes. In: *2012 International Conference for Internet Technology and Secured Transactions*. [S.l.: s.n.], 2012. p. 784–789. 38

ZHAO, D. Frag: Toward federated vector database management for collaborative and secure retrieval-augmented generation. 2024. Disponível em: <https://arxiv.org/abs/2410.13272>. 33

ZHAO, G.; ZHENG, D.; CHEN, K. Design of single sign-on. In: *IEEE International Conference on E-Commerce Technology for Dynamic E-Business*. [S.l.: s.n.], 2004. p. 253–256. 34

ZHOU, W.; MEINEL, C. Implement role based access control with attribute certificates. In: *The 6th International Conference on Advanced Communication Technology, 2004.* [S.l.: s.n.], 2004. v. 1, p. 536–540. 41, 42