# Documento de Arquitetura de Software

Alunos: Arthur Augusto Domingos Silva Júlio César Gonzaga Ferreira Silva Caio César Pereira Vinicius Figueiredo Ferreira

## Introdução

O sistema de Biblioteca Digital tem como objetivo principal fornecer uma plataforma eficiente para o gerenciamento de livros e empréstimos, permitindo que usuários pesquisem, reservem e realizem empréstimos de obras de forma prática e organizada. Além disso, oferece aos administradores funcionalidades para gerenciar o acervo, categorias, relatórios e notificações, garantindo o controle completo sobre as operações da biblioteca. O escopo do sistema abrange o cadastro e autenticação de usuários, consulta avançada de livros por título, autor, categoria e disponibilidade, reserva e empréstimo de livros, renovação e cancelamento de empréstimos, registro de histórico, avaliações e comentários. O sistema também inclui geração de relatórios sobre empréstimos, reservas e livros mais populares, com foco em alta disponibilidade, segurança, desempenho e interface intuitiva, funcionando de forma responsiva em dispositivos desktop e mobile.

## Arquitetura do Sistema

O sistema adota uma arquitetura em camadas (Layered Architecture) que separa claramente a interface do usuário, a lógica de negócio e a persistência de dados, garantindo modularidade, manutenção mais fácil e escalabilidade. A **camada de apresentação** é responsável pela interação com os usuários, oferecendo uma interface web responsiva e intuitiva, tanto para desktop quanto para dispositivos móveis. A **camada de aplicação**, ou de serviços, concentra toda a lógica de negócio, incluindo gerenciamento de usuários, livros, empréstimos, reservas, notificações e geração de relatórios. Por fim, a **camada de persistência** gerencia o armazenamento e recuperação de dados no banco de dados, garantindo integridade, segurança e suporte a backups automáticos. Essa estrutura permite que cada camada evolua independentemente, facilita testes unitários e integração, e assegura que requisitos de desempenho e disponibilidade sejam atendidos de forma eficiente.

## Componentes Principais

### 1. Módulo de Usuário

- **Serviços:** Autenticação, registro, atualização de perfil, recuperação de senha.

- **Controladores:** LoginController, RegistroController, PerfilController.
- **Descrição:** Gerencia todas as funcionalidades relacionadas aos usuários, incluindo cadastro, login, edição de perfil, histórico de empréstimos e avaliações de livros.

**2. Módulo de Livro**

- **Serviços:** Cadastro, edição, remoção, pesquisa e controle de disponibilidade.
- **Controladores:** LivroController, PesquisaController.
- **Descrição:** Responsável pelo gerenciamento do acervo, permitindo que administradores adicionem, editem ou removam livros e que usuários pesquisem obras por título, autor, categoria e filtros avançados.

**3. Módulo de Empréstimo e Reserva**

- **Serviços:** Registro de empréstimos, reservas, renovações e cancelamentos.
- **Controladores:** EmprestimoController, ReservaController.
- **Descrição:** Controla todo o fluxo de empréstimos e reservas, assegurando que a disponibilidade dos livros seja atualizada em tempo real e que usuários possam renovar ou cancelar empréstimos.

**4. Módulo de Relatórios**

- **Serviços:** Geração de relatórios de empréstimos, reservas e livros mais populares.
- **Controladores:** RelatorioController.
- **Descrição:** Fornece análises e relatórios detalhados para administradores, permitindo decisões baseadas em dados sobre o acervo e uso da biblioteca.

**5. Módulo de Notificações**

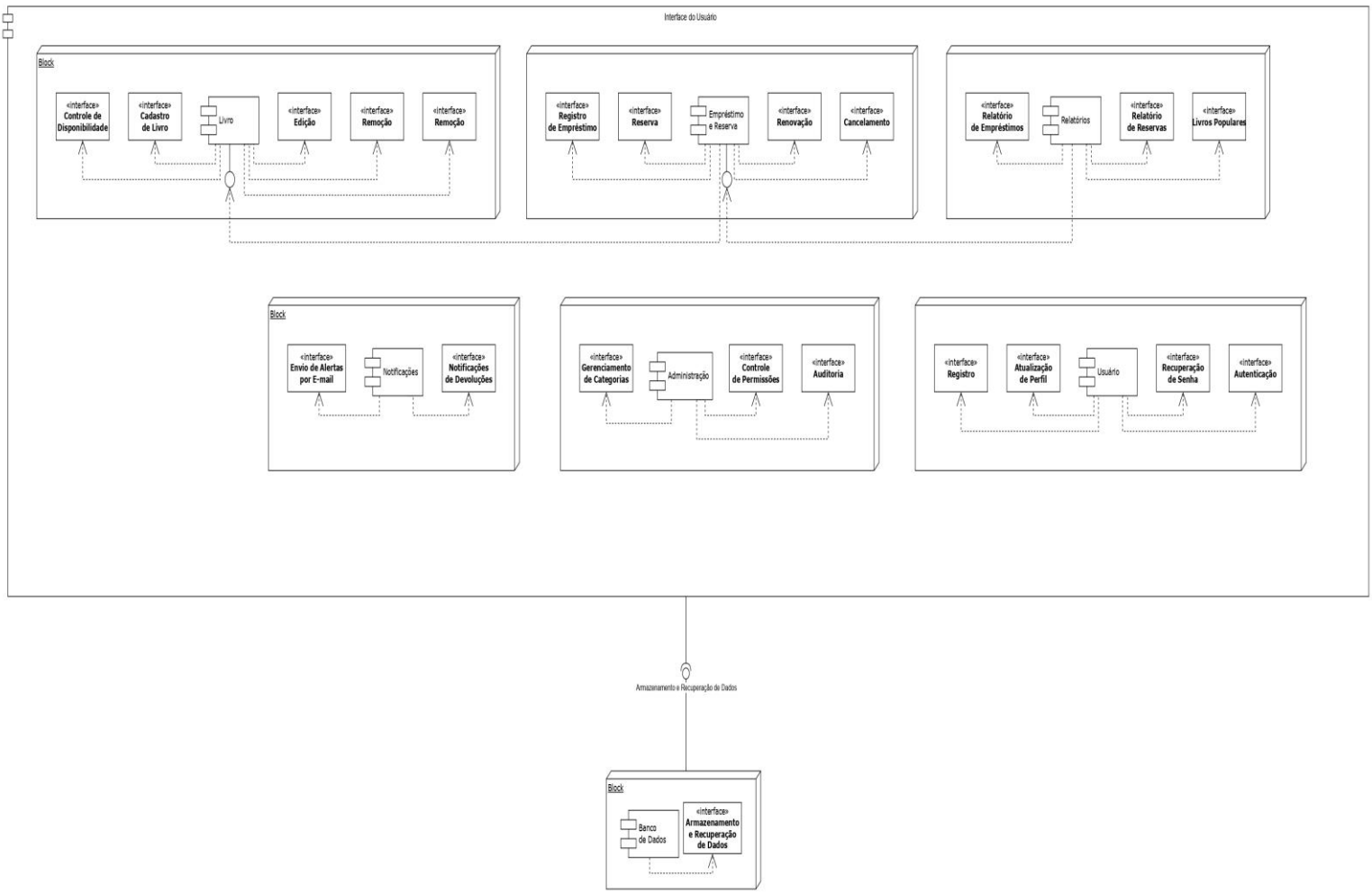
- **Serviços:** Envio de alertas por e-mail ou notificações sobre devoluções atrasadas.
- **Controladores:** NotificacaoController.
- **Descrição:** Mantém usuários informados sobre prazos de devolução e outras comunicações importantes, reduzindo atrasos e facilitando a gestão do acervo.

**6. Módulo de Administração**

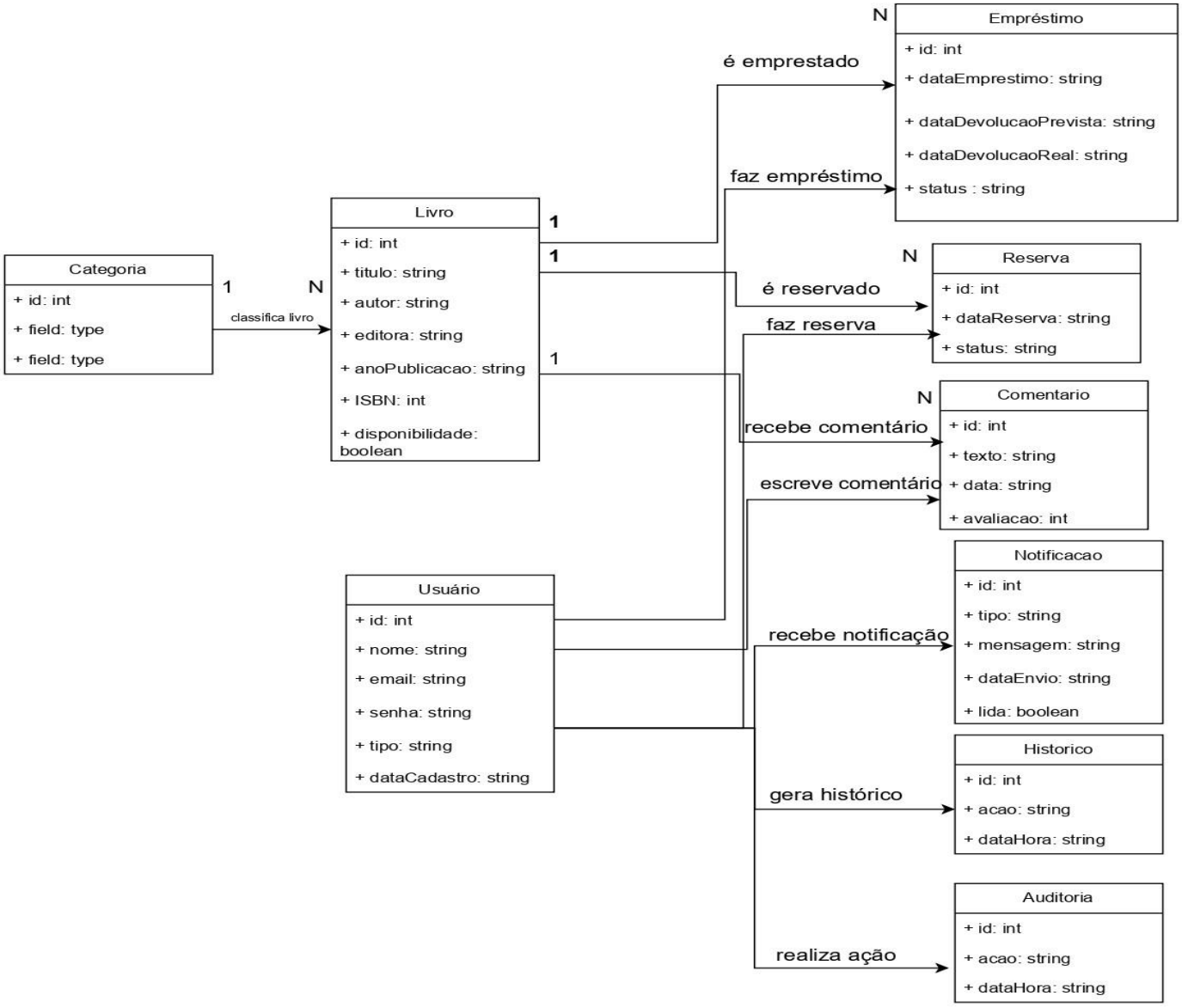
- **Serviços:** Gerenciamento de categorias de livros, controle de permissões, auditoria de ações importantes.
- **Controladores:** AdminController, CategoriaController, AuditoriaController.
- **Descrição:** Permite que administradores configurem categorias, gerenciem permissões de usuários e monitorem ações do sistema para garantir segurança e conformidade.

# Diagrama Arquitetural

## Diagrama de Componentes



## Diagrama de domínio




# Tecnologias Utilizadas

**FrontEnd**: React/Vue

**Backend**: NodeJS + Express

**Banco de Dados:** PostgreSQL

**Versionamento:** Git/GitHub

**Deploy:** Docker

## Padrões e Convenções

O sistema adota padrões e convenções de desenvolvimento que garantem legibilidade, consistência e manutenção do código. A arquitetura segue o padrão **MVC (Model-View-Controller)**, separando claramente a interface, a lógica de negócio e a persistência de dados. Para a transferência de informações entre camadas, são utilizados **DTOs (Data Transfer Objects)**, evitando acoplamento direto entre entidades e controladores. O código segue convenções de nomenclatura consistentes, padrões de indentação e organização de pastas, facilitando a colaboração entre desenvolvedores. Além disso, são aplicadas boas práticas de segurança, como **hashing e salting de senhas**, criptografia de dados sensíveis e validação de entrada do usuário, garantindo robustez e confiabilidade do sistema.

## Requisitos Não Funcionais (RNF)

- **RNF1:** Sistema web responsivo (desktop e mobile).
- **RNF2:** Disponibilidade mínima de 99%.
- **RNF3:** Dados de usuários e livros criptografados.

- **RNF4:** Interface intuitiva e rápida, sem atrasos significativos.
- **RNF5:** Backup diário automático do banco de dados.
- **RNF6:** Sistema deve suportar pelo menos 500 usuários simultâneos.
- **RNF7:** Tempo de resposta das pesquisas não deve exceder 2 segundos.
- **RNF8:** Sistema deve ter autenticação segura (senha + criptografia).
- **RNFG:** Layout deve seguir padrões de acessibilidade (contrast ratio, navegação por teclado).
- **RNF10:** Logs de ações importantes (empréstimos, reservas, cadastros) devem ser armazenados para auditoria.
- **RNF11:** Sistema deve permitir exportação de relatórios em PDF.
- **RNF12:** Backup e recuperação do banco de dados devem ser automatizados e diários.

## Riscos Arquiteturais

O sistema de Biblioteca Digital apresenta alguns riscos técnicos e limitações potenciais que precisam ser considerados durante o desenvolvimento e a operação:

1. **Sobrecarga em consultas simultâneas:** Com mais de 500 usuários simultâneos, consultas ao banco de dados podem gerar lentidão, principalmente em pesquisas avançadas ou geração de relatórios. Estratégias de cache e otimização de queries são essenciais para mitigar esse risco.

2. **Falhas de segurança:** Vulnerabilidades na autenticação, armazenamento de senhas ou criptografia de dados podem comprometer informações sensíveis de usuários e livros. É necessário implementar autenticação forte, validação de entradas e criptografia adequada para reduzir esse risco.
3. **Inconsistência de dados em reservas e empréstimos:** Se múltiplos usuários tentarem reservar ou emprestar o mesmo livro ao mesmo tempo, podem ocorrer conflitos e inconsistências. O uso de transações e bloqueios no banco de dados é crucial para manter a integridade.
4. **Dependência de serviços externos:** Notificações por e-mail ou serviços de deploy em nuvem podem falhar, impactando funcionalidades críticas como alertas de devolução ou disponibilidade do sistema. Estratégias de retry e monitoramento são recomendadas.
5. **Manutenção e escalabilidade:** A arquitetura precisa suportar futuras expansões, como integração com novas bibliotecas, módulos adicionais ou aumento de usuários. Caso o design não seja modular ou escalável, alterações futuras podem ser complexas e custosas.
6. **Performance do sistema:** Pesquisas complexas, geração de relatórios ou múltiplas operações simultâneas podem impactar o tempo de resposta, prejudicando a experiência do usuário. É necessário dimensionamento adequado do servidor, índices no banco e monitoramento de performance.

## Decisões Arquiteturais

Durante o desenvolvimento do sistema de Biblioteca Digital, foram tomadas decisões estratégicas para garantir modularidade, desempenho, segurança e facilidade de manutenção:

1. **Arquitetura em camadas:** Escolheu-se separar a interface, a lógica de negócio e a persistência de dados para facilitar manutenção, testes e escalabilidade, permitindo que cada camada evolua independentemente.
2. **Padrão MVC e uso de DTOs:** O padrão MVC foi adotado para organizar o código de forma clara, enquanto os DTOs garantem transferência de dados segura entre camadas, evitando acoplamento direto com entidades de banco.
3. **Banco de dados relacional:** Optou-se por MySQL/PostgreSQL para facilitar o controle de relacionamentos complexos entre usuários, livros, reservas e empréstimos, garantindo integridade e consistência dos dados.
4. **Frameworks modernos:** Frontend com Vue.js ou React e backend com Java (Spring Boot) ou Node.js (Express) foram escolhidos para acelerar o desenvolvimento, oferecer suporte a aplicações web responsivas e escaláveis, e garantir compatibilidade com boas práticas de desenvolvimento.
5. **Segurança e criptografia:** Implementação de autenticação segura (hash + sal de senhas) e criptografia de dados sensíveis para proteger informações de usuários e do acervo.

6. **Modularidade e manutenção:** Módulos separados para Usuário, Livro, Empréstimo/Reserva, Relatórios, Notificações e Administração, permitindo futuras expansões ou alterações sem impactar todo o sistema.
7. **Monitoramento e auditoria:** Inclusão de logs de ações importantes para rastreabilidade e auditoria, garantindo conformidade e suporte à manutenção preventiva.