UNIVERSIDADE ESTADUAL DO PIAUÍ

CAMPUS DRA. JOSEFINA DEMES

CURSO DE GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

JAROD MATEUS DE SOUSA CAVALCANTE

SORA: SOFTWARE PARA ORGANIZAÇÃO DE RECURSOS ACADÊMICOS

FLORIANO

2024

JAROD MATEUS DE SOUSA CAVALCANTE

SORA: SOFTWARE PARA ORGANIZAÇÃO DE RECURSOS ACADÊMICOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciências da Computação da Universidade Estadual do Piauí, como requisito parcial à obtenção do grau de bacharel em Ciências da Computação.

Orientador: Danilo Borges da Silva.

FLORIANO

2024

SORA: Software para Organização de Recursos Acadêmicos

Jarod Mateus de Sousa Cavalcante[1], **Danilo Borges da Silva**[1]

[1]Ciência da Computação – Universidade Estadual do Piauí (UESPI)
Floriano – PI – Brazil

jarodcavalcante@aluno.uespi.br, danilo@prp.uespi.br

Abstract. *Resource management software is essential for optimizing processes in higher education institutions. The SORA software was developed to enhance resource allocation, such as classrooms and equipment, by eliminating reliance on manual methods like paper records. The system automates scheduling, improves control, and ensures traceability through a centralized and accessible platform. Built with ReactJS, Node.js, Prisma, and integrated with Telegram, SORA reduces scheduling conflicts, facilitates administrative decision-making, and significantly improves the efficiency of university resource management.*
Keywords*: Resource allocation. University management. Academic software. Resource tracking.*

Resumo. *Softwares de gestão de recursos são essenciais para otimizar processos em instituições de ensino superior. O software SORA foi desenvolvido para aprimorar a alocação de recursos, como salas de aula e equipamentos, eliminando a dependência de métodos manuais, como registros em papel. O sistema automatiza agendamentos, melhora o controle e garante a rastreabilidade, oferecendo uma plataforma centralizada e acessível. Desenvolvido com ReactJS, Node.js, Prisma e integrado ao Telegram, o SORA reduz conflitos, facilita decisões administrativas e melhora significativamente a eficiência da gestão universitária.*
Palavras-chave*: Alocação de recursos. Gestão universitária. Software acadêmico, Rastreamento de recursos.*

1. Introdução

A Tecnologia da Informação (TI) pode ser uma ferramenta poderosa para melhorar a eficiência e a eficácia da gestão de recursos em universidades. Os sistemas de TI podem automatizar tarefas manuais, fornecer *insights* em dados históricos e permitir que os administradores tomem decisões mais técnicas (ANDRADE et al., 2023). No entanto, muitas universidades ainda dependem de processos manuais e registros em papel para alocar recursos como: salas de aula, equipamentos audiovisuais, laboratórios, equipamentos em geral e outros ativos. Esse cenário apresenta inúmeros desafios que impactam tanto o corpo docente quanto os alunos, bem como a eficiência da instituição como um todo.

Uma das questões centrais que impulsionou a elaboração deste trabalho resultando no software proposto é a persistente dependência de registros em papel para a alocação de recursos nas instalações universitárias. Em muitos casos ainda se tem anotações manuais, não muito detalhadas, para a distribuição de recursos, como projetores multimídia, salas de aula e diversos outros ativos cruciais para o funcionamento acadêmico e administrativo.

Grande parte das instituições públicas carecem de um melhor controle das informações, uma vez que mesmo que usem sistemas digitais, os mesmos em geral são isolados e não possibilitam integração e nem expansão, diminuindo a perspectiva de auxiliar na tomada de decisões (CUNHA; JÚNIOR; ALMEIDA, 2005).

A continuidade do uso de registros em papel apresenta um leque de desafios substanciais, impactando significativamente a eficácia e a eficiência das operações universitárias. Uma das principais consequências dessa prática é a frequente perda de documentos físicos, resultando em dificuldades significativas para recuperar informações históricas e rastrear o uso anterior dos recursos. Isso pode ocasionar diversos problemas pois administrar informações envolve igualmente a consideração de dois aspectos essenciais: o custo associado e a durabilidade da informação (MOREIRA; NUNES, 2009). Essa lacuna na acessibilidade a registros antigos não apenas compromete a prestação de contas, mas também limita a capacidade de planejamento eficiente.

Além disso, a alocação manual de recursos frequentemente gera frustrações para professores, alunos e técnicos administrativos. Isso faz com que haja uma necessidade constante de verificar manualmente a disponibilidade de recursos, muitas vezes percorrendo um processo moroso e incerto para determinar se um projetor ou uma sala de aula específica está atualmente em uso ou disponível para agendamento. Essa abordagem, além de ser ineficiente, sobrecarrega a comunidade universitária e pode prejudicar o aproveitamento pleno dos recursos disponíveis.

Portanto, torna-se evidente a necessidade de uma solução alternativa que otimize o processo de alocação de recursos e aborde essas complexidades. A criação de um software específico tem como objetivo enfrentar esses desafios, oferecendo soluções mais eficientes, permitindo um controle preciso dos recursos e facilitando o processo para todos os envolvidos no mesmo. Com esse software espera-se otimizar a gestão e a utilização dos recursos nas universidades, impulsionando a inovação e aprimorando significativamente suas operações diárias.

O Software para Organização de Recursos Acadêmicos (SORA), desenvolvido neste trabalho, proporcionará uma plataforma centralizada que permite o agendamento eficiente e transparente de recursos, eliminando a dependência de registros em papel e a necessidade de busca manual. Isso trará benefícios substanciais, como a redução de conflitos de agendamento, o acesso a registros históricos, a capacidade de planejar com antecedência e a melhoria geral na experiência de professores, alunos e funcionários universitários.

Este trabalho está organizado em mais quatro seções. A Seção 2, apresenta estudos e soluções semelhantes que fundamentam e contextualizam o tema. A Seção 3, descreve o processo de criação do sistema, detalhando as tecnologias utilizadas e a modelagem do software. A Seção 4, expõe as funcionalidades implementadas e as contribuições práticas do SORA para o gerenciamento de alocação de recursos em ambientes acadêmicos. Por fim, a Seção 5 apresenta a conclusão do trabalho.

2. Trabalhos Relacionados

Nesta seção serão apresentados os principais trabalhos que possuem uma relação com o softwares similares ao desenvolvido neste trabalho.

A análise feita por Cunha et al. (2011) investiga os desafios na implementação de sistemas de informação em uma instituição pública de ensino, dando ênfase na influência de pressões ambientais, como restrições orçamentárias e a falta de controle no fluxo de informações. Esse estudo busca otimizar a alocação de recursos e processos institucionais, fornecendo uma solução que melhora a gestão e a tomada de decisões.

Diemer e Rehfeldt (2013) propõem um software para automatizar a organização de eventos acadêmicos, enfrentando a falta de sistematização e integração dos processos institucionais. O trabalho aborda a informatização de atividades administrativas, facilitando a gestão de recursos e eventos, assim como o controle e acompanhamento dos mesmos.

O trabalho de Campelo e Pinto (2010) debate a necessidade de informatizar os processos acadêmicos para poder melhorar a eficiência e consequentemente reduzir os custos. O mesmo identificou que o método atual de solicitação de aproveitamento de disciplinas é ineficaz e propenso a atrasos. O artigo propõe informatizar esse processo, permitindo que alunos façam solicitações e acompanhem o andamento de forma online, o que agilizaria a tramitação e reduziria o trabalho dos funcionários. Isso se relaciona ao SORA, que também visa informatizar e automatizar processos administrativos em instituições de ensino, promovendo uma gestão mais eficiente e transparente dos recursos educacionais.

Rezende (2019) propõe um módulo informatizado para gestão de vagas em cursos de graduação, buscando otimizar o controle e acompanhamento desde a criação da vaga até sua ocupação ou remanescência. A informatização visa aprimorar a eficiência da gestão acadêmica, fornecendo dados e relatórios que subsidiam a tomada de decisões estratégicas, como a oferta de novas vagas em processos seletivos.

Santos (2021) investigou soluções de otimização para a alocação de salas de aula, um problema que envolve diversas restrições e preferências. O autor comparou sistemas como o SALAS e o UniTime, destacando a importância da automatização para lidar com esse desafio e melhorar a eficiência da gestão acadêmica.

3. Desenvolvimento

Nesta seção, será descrito o processo de desenvolvimento do sistema com vistas a auxiliar de maneira eficaz o gerenciamento de alocação de recursos em uma universidade, que seguiu uma abordagem modular, com a separação clara entre *frontend* e *backend*. Durante o desenvolvimento, foram adotadas práticas de *Clean Code* (BALTA.IO, 2023), visando garantir a legibilidade, manutenibilidade e eficiência do código. Além disso, foram implementados testes unitários para cobrir as principais funcionalidades e assegurar a confiabilidade do sistema (FILHO et al., 2012). A seção também apresenta os diagramas de caso de uso e classes, essenciais para a modelagem e compreensão da arquitetura do sistema.

3.1. Status dos Agendamentos

No sistema, os agendamentos podem assumir diferentes tipos de status, dependendo do estágio em que se encontram no processo de uso dos recursos. Esses status são utilizados para controlar e acompanhar o ciclo de vida de um agendamento, e cada um reflete um ponto específico nesse ciclo. Esses status são:

- *Aguardando*: indica que o agendamento foi criado, mas o recurso ainda não foi utilizado.
- *Em Uso*: o recurso está sendo utilizado conforme o agendamento.
- *Finalizado*: o uso do recurso foi concluído com sucesso.
- *Cancelado*: o agendamento foi cancelado pelo usuário ou administrador.
- *Não Atendido*: o agendamento permaneceu aguardando, mas o recurso não foi utilizado no período.
- *Não Finalizado*: o recurso foi utilizado, mas não houve a finalização adequada.

Esses status são fundamentais para a gestão e controle dos recursos e agendamentos, pois permitem uma visão clara do estado de cada solicitação no sistema.

3.2. Tipos de Agendamento

O sistema suporta dois tipos de agendamentos: *único* e *recorrente*. Cada um deles apresenta características distintas, sendo eles apresentadas abaixo.

- *Agendamento único*: configura um agendamento isolado, que ocorre apenas uma única vez e para uma data e hora específica.
- *Agendamento recorrente*: permite que o usuário agende o uso de recursos de forma repetitiva em intervalos específicos. Existem três subtipos de agendamento recorrente:
    - *Diário*: o recurso é agendado todos os dias, respeitando a sua disponibilidade.
    - *Semanal*: o recurso é agendado semanalmente nos dias da semana selecionados pelo usuário (por exemplo, todas as terças e quintas-feiras).
    - *Mensal*: o recurso é agendado mensalmente em um dia específico do mês, como, por exemplo, na primeira segunda-feira de cada mês.

O sistema gerencia essas recorrências de forma automática, criando todas as instâncias de agendamentos necessárias de acordo com as opções selecionadas pelo usuário.

3.3. Requisitos do Sistema

Nesta seção, serão apresentados os requisitos do sistema, que descrevem as funcionalidades e características que o sistema deve atender para garantir que o produto final atenda às expectativas dos usuários e cumpra os objetivos propostos. Esses requisitos estão organizados em duas categorias principais: Requisitos Funcionais e Requisitos Não Funcionais.

3.3.1. Requisitos Funcionais

Os Requisitos Funcionais (RF) especificam as funcionalidades principais do sistema, descrevendo como ele deve operar para resolver os problemas identificados. Esses requisitos devem satisfazer as necessidades dos usuários, garantindo que o software ofereça as capacidades necessárias para os mesmos, conforme discutido por Rocha e Magalhães (2005). A seguir são listados os RFs:

1. *Manutenção de usuários*: o sistema deve permitir que administradores criem, editem e excluam usuários, podendo ser do tipo "Membro da comunidade" ou "Administrador". E cada usuário, "Membro da comunidade", poderá editar suas próprias informações.
2. *Redefinição de senha*: o sistema deve permitir que administradores redefinam a senha de outros usuários. Após a redefinição, o usuário cuja senha foi redefinida deve receber um e-mail com uma nova senha temporária de acesso.
3. *Visualização de calendários de agendamentos*: o sistema deve permitir que qualquer usuário, incluindo visitantes não autenticados, visualize os recursos e seus respectivos agendamentos em uma visualização de calendário. Essa visualização deve oferecer opções de filtro por semana, recursos e usuários, facilitando a navegação e busca de informações específicas.
4. *Listagem de Agendamentos*: o sistema deve permitir que qualquer usuário, incluindo visitantes não autenticados, visualize os recursos e seus respectivos agendamentos em uma visualização de tabela. Essa visualização deve oferecer opções de filtro, facilitando a navegação e busca de informações específicas.
5. *Visualização de Agendamento*: o sistema deve permitir que qualquer usuário possa visualizar o agendamento e seus detalhes.
6. *Manutenção de agendamentos:* o sistema deve permitir que administradores criem e alterem agendamentos de recursos disponíveis, especificando detalhes como data, horário, recurso a ser utilizado e usuário.
7. *Manutenção de Recursos*: o sistema deve permitir que administradores criem, alterem e excluam recursos para agendamento, inserindo detalhes como nome, tipo, localização e disponibilidade.
8. *Geração de Relatórios*: o sistema deve permitir que administradores gerem relatórios sobre a utilização dos recursos, agendamentos realizados e dados de usuários, com a opção de exportar esses relatórios em diferentes formatos, como PDF e XLSX.
9. *Visualização de Auditoria*: o sistema deve permitir que administradores visualizem auditoria das ações de criar, editar e deletar realizadas no sistema.

3.3.2. Requisitos Não Funcionais

Os Requisitos Não Funcionais (RNF) tratam das qualidades e restrições que o sistema deve seguir, relacionadas a aspectos como desempenho, segurança, usabilidade, e confiabilidade. Essas características são fundamentais para garantir que o sistema funcione de forma eficiente e segura, conforme abordado por Rocha e Magalhães (2005). Seguem a seguir uma lista dos RNFs do sistema:

- *Desempenho*: o sistema deve ser capaz de suportar um grande número de usuários, recursos e agendamentos sem comprometer a usabilidade e disponibilidade do mesmo.
- *Segurança*: o sistema não deve armazenar senhas de usuários em texto simples no banco de dados. As senhas devem ser salvas na forma de *hashes* criptografados. Além disso, o sistema deve implementar um mecanismo de *refresh token*, com validade de 24 horas.

- *Escalabilidade*: o sistema deve ser capaz de escalar horizontalmente para acomodar o aumento do número de agendamentos, recursos e usuários, garantindo que a performance não seja degradada.
- *Usabilidade*: o sistema deve possuir uma interface intuitiva e objetiva, permitindo que os usuários realizem suas tarefas de forma eficiente e sem complicações.
- *Auditabilidade*: o sistema deve registrar todas as operações de criação, atualização e deleção, permitindo o rastreamento de todas as alterações realizadas no sistema.
- *Compatibilidade*: o sistema deve ser compatível com os principais navegadores (Chrome, Firefox e Safari), devendo oferecer uma experiência funcional em versões mobile.

3.3.3. Regras de Negócio

As Regras de Negócio (RN) são as regras que definem o negócio que deve ser seguido sobre determinado contexto. Estas regras são fundamentais para o funcionamento correto do sistema, como abordado em Souza e Figueredo (2014). Apesar de poderem se configurar como parte dos requisitos funcionais, destacam-se como elementos cruciais as seguintes RNs:

1. *Exclusão de usuários*: o sistema deve garantir que pelo menos um administrador permaneça ativo na plataforma. Um administrador não poderá deletar a si mesmo ou outro administrador se isso resultar na ausência de administradores no sistema.
2. *Automação de agendamentos:* o sistema deve automatizar a alteração de *status* dos agendamentos com base nas seguintes condições:
    - Se a data de início de um agendamento for ultrapassado e o recurso não for utilizado, o *status* deve ser alterado automaticamente para "não atendido", caso o mesmo seja um agendamento *recorrente*.
    - Se o uso de um recurso não for finalizado corretamente, o *status* deve ser alterado para "não finalizado", caso o mesmo seja um agendamento *recorrente*.
    - Após o término do uso de um recurso no qual o agendamento é do tipo *recorrente*, o *status* deve ser alterado para "finalizado".
3. *Exclusão de recursos com agendamentos ativos*: o sistema deve alertar o administrador ao tentar excluir um recurso com agendamentos futuros. Se o administrador confirmar a exclusão, todos os agendamentos associados ao recurso serão excluídos.
4. *Redefinição de Senha*: quando um administrador redefinir a senha de um usuário, a nova senha temporária enviada ao usuário deve ser única e gerada automaticamente pelo sistema.
5. *Alteração de Senha Temporária:* um usuário com senha temporária não poderá acessar outras funcionalidades da plataforma até que sua senha seja alterada para uma permanente no primeiro acesso do mesmo.
6. *Atualizações via Telegram*: o sistema deve enviar uma mensagem para o *telegram* do usuário sempre que um agendamento do mesmo for criado ou cancelado.

3.4. Diagramas

Nas seções a seguir, serão apresentados os diagramas que representam a modelagem e a arquitetura do sistema, fornecendo uma visão detalhada de suas funcionalidades e

interações. Esses diagramas são essenciais para compreender a estrutura do sistema e como seus componentes se relacionam.

Para facilitar a compreensão dos componentes do sistema, foi utilizado um conjunto de ferramentas na elaboração dos diagramas com uso do software Lucidchart[1].

3.4.1. Diagrama de Casos de Uso

O diagrama de casos de uso ilustra como os diferentes atores interagem com o sistema, destacando as principais funcionalidades oferecidas. Ele é uma ferramenta fundamental para visualizar o comportamento do sistema em diversos cenários, de acordo com as necessidades dos usuários e as funcionalidades propostas (COCKBURN, 2005). O diagrama pode ser conferido na Figura 1.

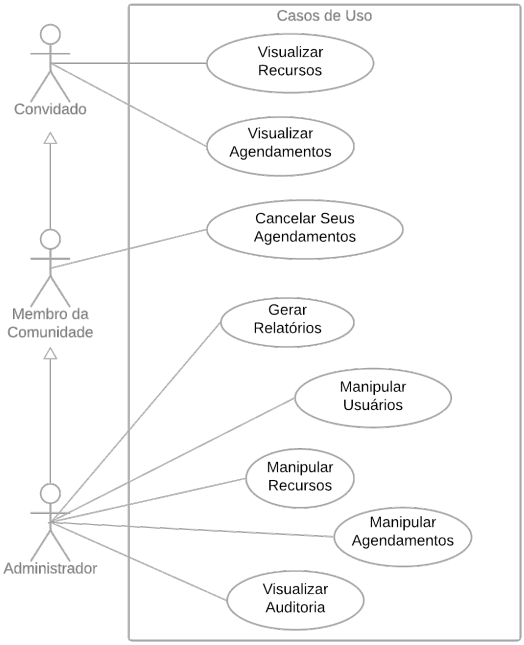

Figura 1. Diagrama de Casos de Uso do sistema.

3.4.2. Diagrama de Classe

Conforme discutido por Souza e Figueredo (2014), o diagrama de classe é utilizado para representar a estrutura estática de uma classe do sistema, trabalhando no paradigma orientado à objetos. No qual podem haver relacionamentos entre classes. Esses relacionamentos podem ser de: associação, especialização, dependências e em pacotes. No qual os mesmos devem ser mostrados no diagrama de classe em conjunto com seus atributos e operações. O diagrama pode ser conferido na Figura 2.

[1] www.lucidchart.com

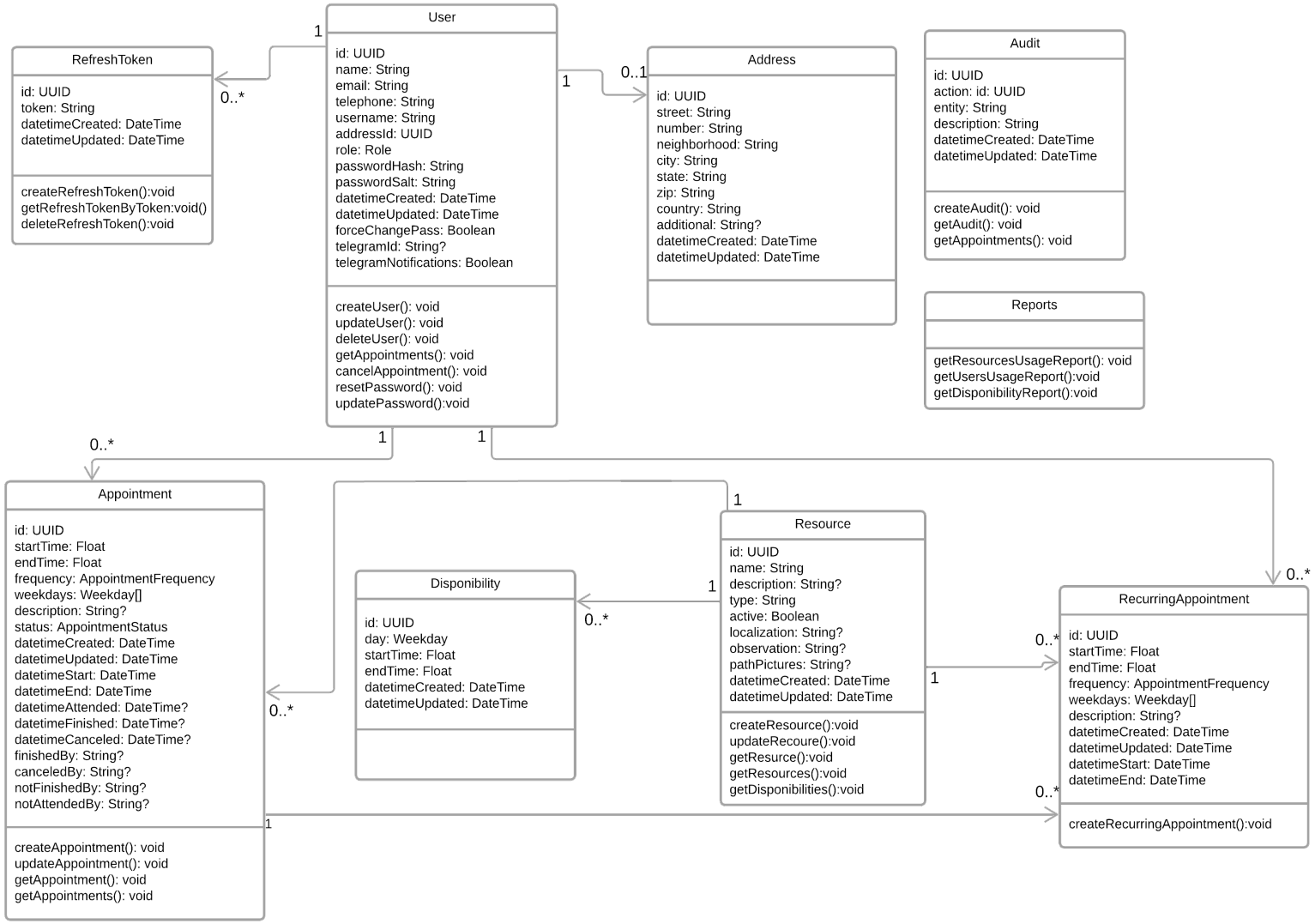

Figura 2. Diagrama de Classes.

3.5. Tecnologias

Nesta seção, serão apresentadas as tecnologias adotadas para o desenvolvimento do sistema, detalhando a infraestrutura dividida entre *frontend* e *backend*, além das bibliotecas e ferramentas utilizadas para atender aos requisitos funcionais e não funcionais do mesmo.

3.5.1. Frontend

O *frontend* da aplicação foi desenvolvido utilizando as seguintes tecnologias:

- React: é um *framework* desenvolvido na linguagem JavaScript, criado pela *Meta*. Ele foi utilizado para criar as interfaces de usuário (UI – *User Interface*, ou Interface de Usuário) em aplicações web.
- *Ant Design*: é uma biblioteca de componentes UI. O SORA utilizou seus componentes para criar uma interface moderna e com uma boa usabilidade.
- *Axios*: é uma biblioteca para realizar requisições HTTP. A mesma foi utilizada para fazer as requisições na nossa API REST do *backend*.
- *Vite*: é uma ferramenta de compilação que permite gerar projetos em *React*. É uma solução moderna, rápida e flexível, com tempos de compilação significativamente menores em comparação com outras ferramentas disponíveis no mercado.
- *Day.js*: é uma biblioteca para o tratamento de datas. A mesma foi escolhida por ser leve, fácil de utilizar e por oferecer uma gama de métodos que facilitam a implementação da mesma no sistema, bem como formatação e tratamento de datas.

3.5.2. Backend

O *backend* da aplicação foi desenvolvido utilizando as seguintes tecnologias:

- *Node.js com TypeScript*: o *Node.js* foi utilizado junto com o *TypeScript* pois oferece tipagem estática, garantindo maior segurança no desenvolvimento e manutenção do código.
- *Express*: *framework* para o desenvolvimento de *API RESTful*. O *Express* facilita a criação de rotas e o gerenciamento de *middleware*, facilitando integrações e também futuras manutenções.
- *Vitest*: biblioteca utilizada para o desenvolvimento dos testes unitários[2] e de integração[3], seguindo a metodologia de Desenvolvimento Orientado a Testes (TDD – *Test-Driven Development*)[4]. O Vitest permite rodar testes de maneira rápida e eficiente
- *Prisma*: ORM (*Object-Relational Mapping*, ou Mapeamento de Objeto Relacional) escolhido para a interação com o banco de dados PostgreSQL. O Prisma simplifica a modelagem do banco de dados e facilita as operações *CRUD*.

[2]São testes que verificam se uma parte específica do código, costumeiramente a nível de função, está funcionando corretamente (DEMOISELLE, 2013).

[3]Teste de integração é uma etapa do processo de desenvolvimento de software em que módulos ou componentes são combinados e testados em grupo (OBJECTIVE, 2023).

[4]TDD é uma metodologia de desenvolvimento de software que prioriza a escrita de testes unitários antes da implementação do código (DEMOISELLE, 2013).

- *PostgreSQL*: banco de dados relacional utilizado no sistema. Escolhido por sua robustez, confiabilidade e suporte a grandes volumes de dados, além de ser altamente compatível com o *Prisma*.
- *JWT (JSON Web Token)*: utilizado para a autenticação dos usuários. O JWT é uma solução leve e segura para autenticação em *APIs RESTful*, garantindo a integridade das informações do usuário.
- *Node-Telegram-Bot-API*: biblioteca utilizada para a integração do sistema com o *Telegram*, permitindo o envio de notificações automáticas para os usuários sobre agendamentos.

4. Resultados

Considerando as necessidades abordadas por este trabalho, o SORA se mostrou capaz de desempenhar um importante papel na organização e gerenciamento da alocação de recursos. A seguir, são apresentados os principais resultados obtidos com o desenvolvimento e testes do sistema, demonstrando como ele atende aos requisitos funcionais, regras de negócio e expectativas do projeto.

4.1. Funcionalidades Implementadas

O SORA foi projetado para otimizar o gerenciamento de alocação de recursos, com foco em acessibilidade, eficiência e uma experiência de usuário fluida. As principais funcionalidades desenvolvidas incluem:

- Visualização de calendário de agendamentos: permite que os usuários visualizem todos os agendamentos em um formato de calendário intuitivo, facilitando a identificação das reservas disponíveis e organizando o uso dos recursos (Figura 3).

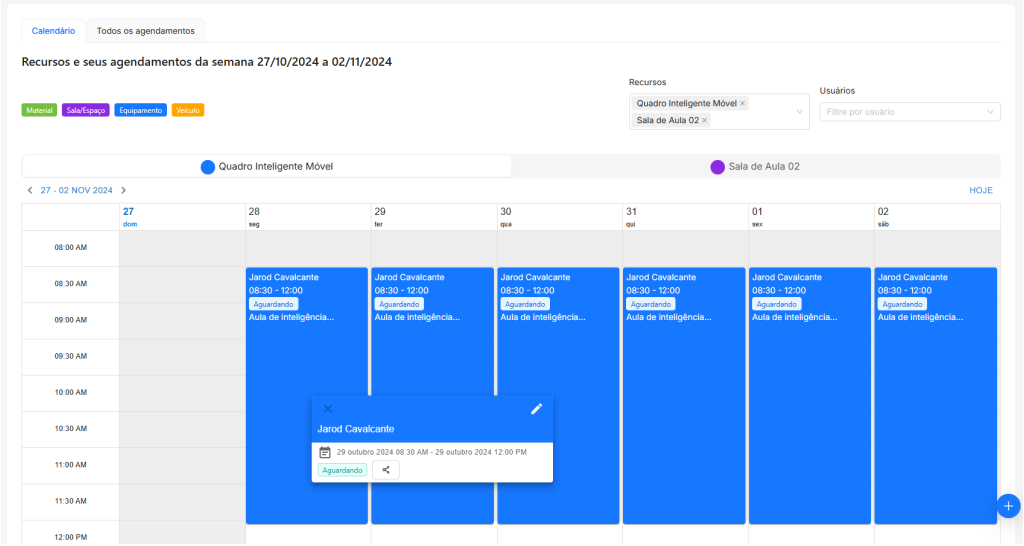

Figura 3. Visualização de Calendário de Agendamentos.

- Criação de agendamento único: funcionalidade que permite a criação de um agendamento único, proporcionando flexibilidade para alocar recursos em momentos específicos sem repetições (Figura 4).

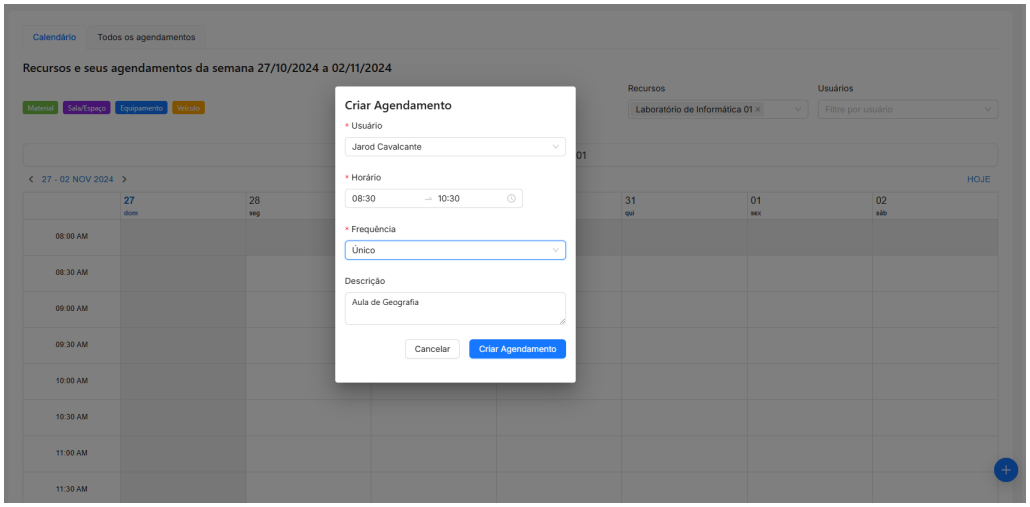

Figura 4. Criação de Agendamento Único.

- Criação de agendamento recorrente: Permite a criação de agendamentos recorrentes, como diários, semanais ou mensais, garantindo que recursos possam ser reservados de forma programada e contínua (Figura 5).

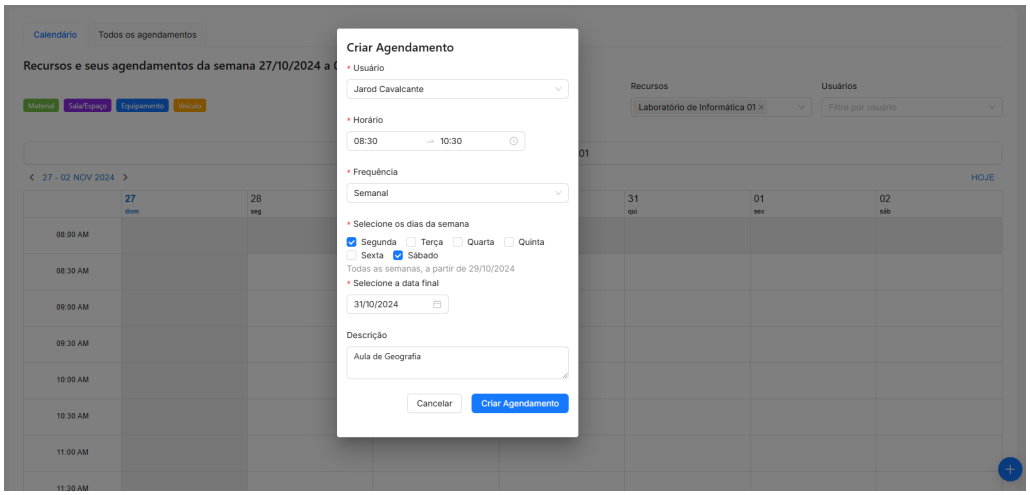

Figura 5. Criação de Agendamento Recorrente - Semanal.

- Geração de relatórios: oferece a funcionalidade de gerar relatórios detalhados sobre o uso dos recursos, permitindo que administradores acompanhem o histórico de agendamentos e tomem decisões baseadas em dados (Figura 6).
- Atualizações via *Telegram*: o sistema envia notificações automáticas via Telegram para informar os usuários sobre a criação, atualização ou cancelamento de agendamentos, mantendo-os sempre informados (Figura 7 e Figura 8).

4.2. Automatização dos Status dos Agendamentos

Uma das principais implementações do SORA foi a automação do status dos agendamentos, que muda automaticamente conforme o uso do recurso, como “Aguardando”, “Atendido”, “Finalizado”, entre outros. Isso diminui o trabalho dos administradores durante esse processo, no qual mesmo com vários recursos para administrar, o mesmo não

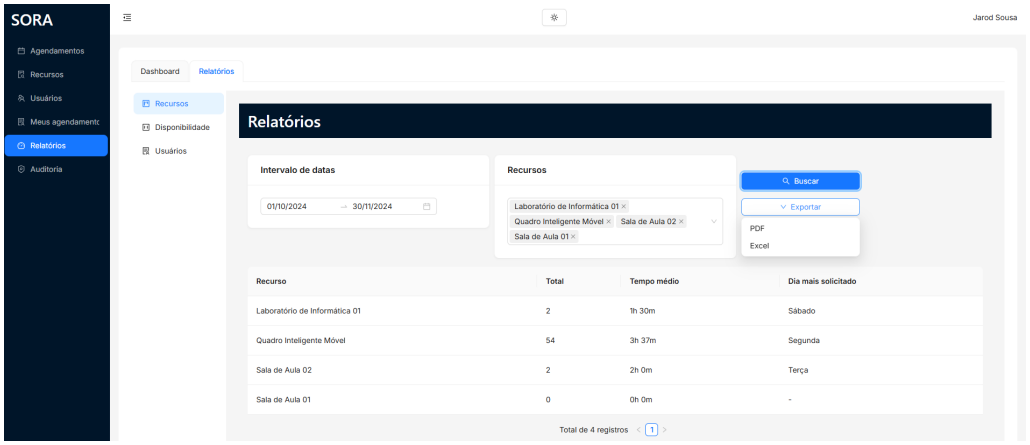

Figura 6. Geração de Relatórios.

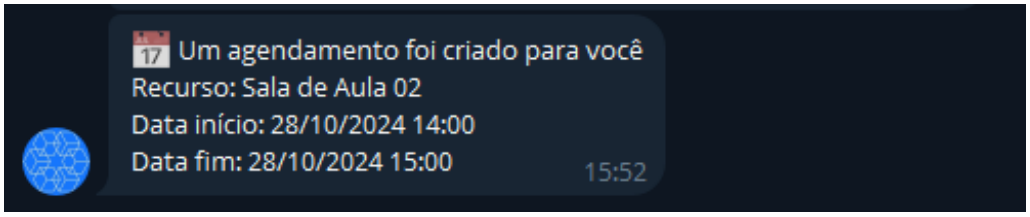

Figura 7. Mensagem de Agendamento Criado no Telegram.

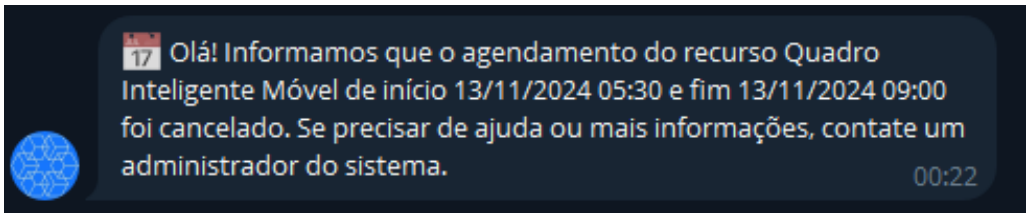

Figura 8. Mensagem de Agendamento Cancelado no Telegram.

precisará se preocupar tanto com a gerência de status dos mesmos, otimizando ainda mais todo o processo.

5. Conclusão e Trabalhos Futuros

O desenvolvimento do SORA atingiu os objetivos propostos de oferecer uma solução informatizada que otimiza o processo de alocação de recursos em instituições. Através da automação de agendamentos, o sistema abordou problemas como a dependência de registros manuais e a dificuldade no controle dos recursos.

Os resultados obtidos durante o desenvolvimento destacaram pontos positivos, como a redução de conflitos de agendamentos, o acesso fácil a históricos de uso e a geração de relatórios dos mesmos, cooperando com a melhoria na tomada de decisões administrativas. O sistema também automatizou o controle de status dos agendamentos reduzindo consideravelmente o trabalho manual dos administradores. Entretanto, limitações também foram identificadas, como o fato de o sistema ainda não ser *multi-tenancy*[5], o que restringe seu uso simultâneo por diferentes organizações com isolamento

[5]"Multi-tenancy é um estilo de arquitetura onde você tem uma aplicação centralizada que atende a vários

completo de dados.

Este trabalho contribui para a gestão acadêmica ao apresentar uma ferramenta prática e acessível para instituições de ensino superior. A centralização das operações de agendamento e a automação de tarefas antes manuais são diferenciais que melhoram a eficiência administrativa e promovem o uso racional e otimizado dos recursos. Além disso, a integração com o *Telegram* demonstra o potencial da informatização por meio de notificações em tempo real.

Embora o SORA tenha alcançado resultados significativos, algumas melhorias e extensões podem ser realizadas em trabalhos futuros. Como a integração com ferramentas externas, como o *Google Calendar* e outros calendários digitais, pode ampliar a usabilidade e tornar o sistema ainda mais alinhado às necessidades dos usuários. Outra melhoria relevante seria a inclusão de login com redes sociais, simplificando o acesso e incentivando a adoção do sistema.

Por fim, uma área promissora é a implementação de integrações com inteligências artificiais para gerar relatórios mais otimizados e com análises preditivas. Essa funcionalidade pode fornecer *insights* mais avançados sobre o uso dos recursos, apoiando ainda mais a tomada de decisões pelos administradores.

Referências

ANDRADE, M. M. de et al. Excelência na gestão do ensino superior: desafios e oportunidades. *Cuadernos de Educación y Desarrollo*, v. 15, n. 5, p. 4646–4663, jul. 2023. Disponível em: ⟨https://ojs.cuadernoseducacion.com/ojs/index.php/ced/article/view/1429⟩. 1

BALTA.IO. *Clean Code - Guia e Exemplos*. 2023. Acessado em 02 de novembro de 2023. Disponível em: ⟨https://balta.io/blog/clean-code⟩. 3

CAMPELO, J. d. S.; PINTO, R. S. Proposta de implantação de um sistema informatizado para o gerenciamento dos processos de solicitação de aproveitamento de disciplinas no departamento de registros acadêmicos da Universidade Federal de Pelotas. In: *X Colóquio Internacional sobre Gestão Universitária na América do Sul*. [S.l.]: INPEAU, 2010. 3

COCKBURN, A. *Escrevendo Casos de Usos Eficazes: Um guia prático para desenvolvedores de software*. Bookman, 2005. ISBN 9788577800193. Disponível em: ⟨https://books.google.com.br/books?id=gbBRo8CxmFUC⟩. 7

CUNHA, M. et al. Análise da implantação dos sistemas de informação em uma instituição federal de ensino de alagoas à luz da teoria institucional. *Revista de Administração, Contabilidade e Economia da Fundace*, v. 2, 02 2011. Disponível em: ⟨https://www.researchgate.net/publication/277933585_Analise_da_Implantacao_dos_Sistemas_de_Informacao_em_uma_Instituicao_Federal_de_Ensino_de_Alagoas_a_Luz_da_Teoria_Institucional⟩. 3

CUNHA, M. X. C. da; JÚNIOR, M. F. de S.; ALMEIDA, H. O. de. Dificuldades com integração e interoperabilidade de sistemas de informação nas instituições públicas de

clientes" (PEREIRA, 2019).

ensino: um estudo de caso no cefet-al. In: UNESP. *Anais do XII Simpósio de Engenharia de Produção (SIMPEP)*. Bauru, SP, Brasil, 2005. Evento realizado de 07 a 09 de novembro de 2005. 2

DEMOISELLE. *Orientações técnicas/Conceitos Gerais Testes Unitarios*. 2013. Acessado em 27 de outubro de 2024 às 14:00. Disponível em: ⟨https://www.frameworkdemoiselle.gov.br/wikibe16.html⟩. 9

DIEMER, M. H.; REHFELDT, M. J. H. Modelagem de um software para subsidiar a organização de eventos: O caso da univates. In: *XIII Coloquio de Gestión Universitaria en Américas*. [S.l.: s.n.], 2013. 3

FILHO, M. C. et al. Um estudo de caso sobre o aumento de qualidade de software em projetos de sistemas de informação que utilizam test driven development. In: SOCIEDADE BRASILEIRA DE COMPUTAçãO (SBC). *Anais do VIII Simpósio Brasileiro de Sistemas de Informação (SBSI 2012)*. São Paulo, SP, Brasil, 2012. p. 633–644. Trilhas Técnicas. 3

MOREIRA, J. V. T.; NUNES, M. G. Gestão da informação em uma instituição de ensino superior: registros acadêmicos em foco. *Gestão & Planejamento (G&P)*, v. 10, n. 2, 2009. ISSN 2178-8030. Disponível em: ⟨https://revistas.unifacs.br/index.php/rgb/article/view/765⟩. 2

OBJECTIVE. *Testes de integração: conheça os tipos e as vantagens em automatizar*. 2023. Acessado em 27 de outubro de 2024 às 14:18. Disponível em: ⟨https://www.objective.com.br/insights/teste-de-integracao⟩. 9

PEREIRA, E. *Arquitetura Multi-Tenancy*. 2019. Acessado em 17 de novembro de 2024 às 22:18. Disponível em: ⟨https://medium.com/@edytarcio/arquitetura-multi-tenancy-bb7b47d7ba⟩. 13

REZENDE, R. S. R. Gestão acadêmica no ensino superior: proposta de informatização do controle de vagas dos cursos de graduação da universidade federal do triângulo mineiro (uftm). *Universidade Federal do Triângulo Mineiro*, Universidade Federal do Triângulo Mineiro, 2019. Dissertação (Mestrado em Inovação Tecnológica) - Programa de Mestrado Profissional em Inovação Tecnológica. Disponível em: ⟨http://bdtd.uftm.edu.br/handle/tede/761⟩. 3

ROCHA, R. da S.; MAGALHÃES, T. M. de. Engenharia de requisitos. *Fundação Educacional São José, Revista Eletrônica*, v. 4, n. 1, 2005. ISSN 2178-3098. 4ª Edição. Disponível em: ⟨https://www.fsd.edu.br/wp-content/uploads/2019/12/artigo27-RAFAEL-e-TERESINHA.pdf⟩. 4, 5

SANTOS, H. de A. *Um estudo sobre soluções de otimização para elaboração de ofertas no âmbito acadêmico*. Trabalho de Conclusão de Curso (Graduação), Fortaleza, 2021. Orientador: Leonardo Oliveira Moreira. 3

SOUZA, J. P. d.; FIGUEREDO, R. *Sistema de gestão de imóveis*. 82 p. Monografia de Graduação — Centro Universitário de Brasília - UNICEUB, 2014. Orientador: Queiroz, Wander Jácome de. Disponível em: ⟨https://repositorio.uniceub.br/jspui/handle/235/6553⟩. 6, 7