UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

GUSTAVO HENRIQUE LOPES SPACHUK ZAFFANI

SISTEMA WEB PARA O GERENCIAMENTO DOS MATERIAIS DE LABORATÓRIO DE COMPUTAÇÃO

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO

2020

GUSTAVO HENRIQUE LOPES SPACHUK ZAFFANI

SISTEMA WEB PARA O GERENCIAMENTO DOS MATERIAIS DE LABORATÓRIO DE COMPUTAÇÃO

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Prof. Vinicius Pegorini

PATO BRANCO

2020

Ministério da Educação
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
UTFPR - CAMPUS PATO BRANCO
DIRETORIA GERAL - CÂMPUS PATO BRANCO
DIR. DE GRAD. E EDUCAÇÃO PROFISSIONAL-PB
DEP. ACADÊMICO DE INFORMÁTICA DAINF-PB

TERMO DE APROVAÇÃO

TRABALHO DE CONCLUSÃO DE CURSO

Sistema Web para o Gerenciamento dos Materiais de Laboratório de Computação

POR

Gustavo Henrique Lopes Spachuk Zaffani

Este trabalho de conclusão de curso foi apresentado em 07 de julho de 2020, como requisito parcial para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, pela Universidade Tecnológica Federal do Paraná. O acadêmico foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Banca examinadora:

Prof. MSc Vinicius Pegorini
Professor orientador

Prof. Dr. Fabio Favarim
Professor convidado

Profa. MSc Mainara Cristina Lorencena
Professora convidada

Prof. Dr. Edilson Pontarolo
Coordenador do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas

Profa. Dra. Mariza Miola Dosciatti
Responsável pela Atividade de Trabalho de Conclusão de Curso

Documento assinado eletronicamente por **MARIZA MIOLA DOSCIATTI**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 08/07/2020, às 10:12, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **FABIO FAVARIM**, **CHEFE DE DEPARTAMENTO ACADÊMICO**, em 08/07/2020, às 10:14, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **MAINARA CRISTINA LORENCENA**, **PROFESSOR MAGISTERIO SUPERIOR-SUBSTITUTO**, em 08/07/2020, às 10:17, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **VINICIUS PEGORINI**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 08/07/2020, às 10:21, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **EDILSON PONTAROLO**, **COORDENADOR(A) DE CURSO/PROGRAMA**, em 08/07/2020, às 10:56, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://sei.utfpr.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1512999** e o código CRC **8AD6B42D**.

Só sei que nada sei, e o fato de saber isso, me coloca em vantagem sobre aqueles que acham que sabem alguma coisa. (Leonardo da Vinci)

AGRADECIMENTOS

Primeiramente agradeço a Deus por ter me dado saúde e força durante todo o curso. Agradeço a todos os professores que incentivaram, ensinaram e me guiaram em todos os semestres. A todos os meus colegas que tiveram comigo durante este período. Aos meus pais que sempre acreditaram em mim, e sempre me deram apoio para eu chegar até onde estou hoje. Ao professor Fábio Favarim, por auxiliar no levantamento dos requisitos para o desenvolvimento do sistema. E por último ao professor Vinícius Pegorini por todo apoio durante o desenvolvimento deste trabalho e pelos conhecimentos passados durante o curso.

RESUMO

Os laboratórios das universidades são, na maioria das vezes, responsáveis por disponibilizar todos os equipamentos e materiais utilizados para o desenvolvimento de projetos acadêmicos. Os alunos e/ou professores, por sua vez, realizam empréstimos desses materiais para o desenvolvimento de seus projetos. Há uma grande necessidade de controlar adequadamente todo o fluxo de materiais e equipamentos que são utilizados nos projetos, os quais, na maioria das vezes, são gerenciados manualmente. No laboratório do Departamento Acadêmico de Informática (DAINF) da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, existem diversos materiais que são emprestados diariamente para todo o corpo de docentes e discentes da instituição, em prol de seus respectivos projetos. Um sistema web, como resultado da realização deste trabalho, foi desenvolvido para o gerenciamento dos materiais do laboratório do DAINF, visando melhorar o seu gerenciamento. O sistema auxiliará no controle de empréstimos e reservas de materiais e equipamentos disponibilizados no ambiente, em prol de melhorias no controle de estoque dos mesmos. As principais funcionalidades desenvolvidas foram o cadastro dos materiais, do empréstimo, da compra e da saída, no qual possibilitam todo o controle necessário de entrada e saída de materiais no laboratório.

Palavras-chave: Experiência do Usuário. Gerenciamento de Materiais e Equipamentos. Sistema Web.

ABSTRACT

Usually universities have laboratories to providing equipment and materials to be used for the development of academic projects. Students and/or teachers are able to loan these materials for the development of their projects. It's important to adequately control the entire flow of materials and equipment that is used in projects, which are often mismanaged. In the laboratory of the Academic Department of Informatics (DAINF) of the Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, there are several materials that are loaned daily to the entire faculty and students of the institution, in order to supply their respective projects. This work proposes the development of a web system for the management of DAINF laboratory. The system will provide the control of loans and reserves of materials and equipment available, in order to improve the inventory control of the laboratory. The main functionalities developed were the registration of materials, the loan, the purchase and the exit, in which they allow all the necessary control of entry and exit of materials in the laboratory.

Keywords: User Experience. Materials and Equipment Management. Web System.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTAGEM DOS CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface* (Interface de programação de aplicações)
CRUD	*Create, Read, Update and Delete*
CSS	*Cascading Style Sheets*
DAINF	Departamento Acadêmico de Informática
IP	*Internet Protocol* (Protocolo da Internet)
JPA	*Java Persistence API*
UTFPR	Universidade Tecnológica Federal do Paraná
UX	*User Experience* (Experiência do Usuário)
JDK	*Java Development Kit* (Kit de Desenvolvimento Java)
IDE	*Integrated Development Environment* (Ambiente de Desenvolvimento Integrado)

SUMÁRIO

1 INTRODUÇÃO

Este capítulo apresenta a introdução que é composta pelas considerações iniciais com o escopo e o contexto do trabalho, os objetivos e a justificativa. O texto é finalizado com a apresentação dos capítulos subsequentes.

1.1 CONSIDERAÇÕES INICIAIS

Nos últimos anos, a utilização de Sistemas de Informações (SI) nas empresas tornou-se algo imprescindível, pelo fato de permitir a obtenção e o processamento inúmeras informações para tomadas de decisão dentro das empresas em pouco tempo. Um dos SI utilizados e que apresentam uma proposta de ligação dos diversos processos que regem uma empresa, são denominados de Sistemas Integrados de Gestão Empresarial (SIGE). Os SIGE são "sistemas de informações gerenciais que têm como objetivo fundamental a integração, consolidação e aglutinação de todas as informações necessárias para a gestão do sistema empresa." (PADOZEVE, 2004, p. 68). Esses sistemas gerenciais vêm tomando conta cada dia mais da vida de diversas pessoas, tendo em vista a sua praticidade no controle de diferentes casos e aspectos. Ele proporciona uma solidez tanto para empresas, bancos e instituições em geral.

Diversas empresas buscam no SIGE, maneiras de controlar e administrar o estoque de seus produtos, para realizar o controle adequado da entrada e saída dos itens disponibilizados pela mesma. Para Daft (2007, p. 530) os itens de estoque são "os produtos que a organização mantém em mãos para uso no processo de produção até o ponto de venda dos produtos finais para os clientes", ou seja, o seu controle visa diretamente os principais objetivos dentro de uma empresa. Para O'Brien (2006, p. 231), "os sistemas de controle de estoque processam dados refletindo mudanças nos artigos em estoque". Portanto, o gerenciamento torna-se a peça chave para a evolução das empresas.

Estes sistemas são disponibilizados em diversas plataformas, como web, *desktop* e *mobile*. Entretanto, os sistemas web vêm ganhando muita força graças a sua usabilidade, pois é possível utilizá-los em qualquer lugar do mundo, necessitando apenas de acesso à Internet e um *web browser*. Além do fato desses sistemas abrangerem novas técnicas e estruturas focadas na

usabilidade do mesmo. Dito isso, vale ressaltar a importância da experiência do usuário (*User Experience* - UX) no desenvolvimento dos sistemas de gerenciamento, visando sempre nas melhores práticas de design e usabilidade para o usuário final.

Diante do exposto, neste é apresentado o desenvolvimento de um sistema web que possa ser utilizado para controlar os materiais de consumo (componentes eletrônicos) utilizado nas aulas práticas e equipamentos disponíveis na sala de apoio do Departamento Acadêmico de Informática (DAINF) da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco (UTFPR-PB), para utilização em aulas e projetos. A plataforma web foi escolhida devido a sua facilidade na utilização por diversos usuários, sejam eles alunos ou professores. Com isso, não é necessário possuir um programa instalado no computador ou dispositivo móvel, basta apenas um navegador web instalado, para poder verificar os materiais disponíveis e fazer os devidos controles de estoque e empréstimo.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Desenvolver um sistema web para controle de entradas e saídas de componentes eletrônicos e equipamentos disponíveis na sala de apoio do DAINF.

1.2.2 Objetivos Específicos

- Proporcionar o gerenciamento dos componentes eletrônicos e equipamentos da sala de apoio.
- Permitir o controle de estoque de todos os itens disponíveis na sala de apoio.
- Proporcionar o controle de empréstimos dos materiais disponíveis, para fins de desenvolvimento de projetos por docentes e discentes.

1.3 JUSTIFICATIVA

Diante da necessidade de um gerenciamento de componentes eletrônicos e equipamentos disponíveis na sala de apoio aos laboratórios do DAINF, o qual atualmente não dispõe de um sistema informatizado capaz de controlar o estoque desses itens, assim como os empréstimos, a automatização destes processos proporcionará à instituição um melhor controle e eficácia em sua distribuição.

Considerando esse contexto, o presente trabalho propõe o desenvolvimento de um sistema web que visa o controle de todos os materiais disponíveis na sala de apoio do DAINF, possibilitando, portanto, o controle de entrada e saída dos materiais disponíveis, para um melhor gerenciamento dos mesmos.

O sistema permitirá definir as quantidades mínimas dos materiais para que a compra dos mesmos possa ser planejada e executada de forma a não causar desabastecimento.

1.4 ESTRUTURA DO TRABALHO

Este trabalho está organizado em capítulos. Este é o primeiro capítulo, o qual apresenta as considerações iniciais com o contexto do trabalho a ser desenvolvido, seguido de seus objetivos e justificativas. O Capítulo 2 apresenta o referencial teórico voltado a Experiência de Usuário. No Capítulo 3 estão as tecnologias utilizadas para a modelagem do sistema e as que serão, posteriormente, utilizadas no desenvolvimento do mesmo, nesse capítulo também será apresentado a metodologia de desenvolvimento do trabalho. No Capítulo 4 é apresentado o resultado da realização do trabalho, que é a modelagem, apresentação e implementação do sistema. No Capítulo 5 é apresentado a conclusão deste trabalho. Por fim estão as referências utilizadas no texto.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta a fundamentação teórica deste trabalho, cujo conteúdo explana sobre o conceito e características da experiência de usuário e do design centrado no usuário.

2.1 Experiência do Usuário

O conceito de usabilidade tornou-se um dos principais pontos no desenvolvimento de um software. A interação do usuário passou a exigir detalhes maiores, de diversos profissionais da área, fazendo-os com que desenvolvessem certas peculiaridades capazes de facilitar a vida do usuário no manuseio destes softwares.

Diversos especialistas discutem e reconhecem a importância da usabilidade e experiência do usuário (*User Experience* - UX). Para Schmitt (2000, p. 74), a experiência corresponde a “acontecimentos individuais que ocorrem como resposta a algum estímulo”. Essas experiências apresentam um referencial em uma intencionalidade, ou seja, iniciam de algum lugar com o objetivo de alcançar algo. Isso se deve ao fato delas não se corresponderem de fenômenos espontâneos, e sim induzidos.

Hassenzahl (2013) e Reiss (2013) declaram que a UX pode ser conceituada como a experiência de uso de um indivíduo sobre determinado produto ou serviço, e consequentemente ela poderá ser boa ou ruim. Logo, após diversas interações, o usuário cria a sua percepção sobre o mesmo. Hassenzahl (2013) apresenta três níveis distintos de experiência por meio da interação com um produto: O que, o Como e o Porquê. “O que” corresponde as coisas que as pessoas podem realizar mediante de um produto interativo, ou seja, reflete pontualmente na funcionalidade de um produto. “O Como” retrata a ação por meio de um produto em um nível operacional, logo relaciona-se a parte de interação do usuário com o produto. Já “o Porquê” aborda o motivo real pelo qual os usuários se encorajaram a utilizar os produtos.

Para Buccini (2008), a experiência trata-se de um fenômeno individual que transcorre na mente de um indivíduo, fruto do processamento de um complexo conjunto de estímulos, tanto externo quanto interno, e dependente das interpretações subjetivas pertencentes de cada

pessoa. Portanto, cada indivíduo terá a sua opinião relacionado a experiência, assimilando de diferentes maneiras a usabilidade de determinado produto ou serviço.

Na UX é importante entender como ocorre a interação entre os usuários com os artefatos ou sistemas e, quais os objetivos os usuários almejam atingir no processo de interação. Nesse contexto, as interfaces devem ser pensadas focando no perfil do usuário final do sistema.

2.2 Conceitos do Design Centrado no Usuário

Diversos sistemas tem voltado cada vez mais a atenção para os usuários finais. Ou seja, formas de melhorar a usabilidade e tornar a utilização do software cada vez melhor. Para Rubin e Chisnell (2008, p.5), os profissionais que atuam em ambientes de desenvolvimento web devem relacionar-se com as melhores práticas de acessibilidade para serem implementadas no processo de design centrado ao usuário, acompanhados de outros métodos que são integrados nesse processo.

Segundo Abras, Maloney e Preece (2004), uma maneira de se evitar o insucesso de um projeto de software e mesmo a não aprovação do usuário final é utilizar-se de técnicas de design centrado no usuário. Os autores indicam algumas vantagens e desvantagens pensadas do design centrado no usuário, conforme é mostrado no Quadro 1.

Quadro 1 - Vantagens e Desvantagens do Design Centrado no Usuário

Vantagens	Desvantagens
Os produtos são mais eficientes, eficazes e seguros.	É mais caro.
Auxilia no gerenciamento das expectativas e nos níveis de contentamento dos usuários com o produto.	Necessário de mais tempo para desenvolvimento.
Os usuários desenvolvem um sentido de propriedade do produto.	Podem exigir a participação de membros da equipe de design adicional e uma vasta gana de outros intervenientes.
Produtos exigem menos re – design e integram o ambiente mais rapidamente.	Pode ser difícil de traduzir alguns tipos de dados no projeto.

O processo colaborativo concebe soluções de design mais inovadores para os problemas.	O produto pode ser muito específico para uso geral, portanto, não é facilmente adaptado para outros clientes, tornando-o mais caro.

Fonte: Abras, Maloney e Preece (2004).

Logo, é possível visualizar, seguindo a visão de Abras, Maloney e Preece (2004), a relevância de pensar no usuário final no passar do desenvolvimento de um sistema. Sabendo sempre levar em consideração os riscos de fracasso no projeto, caso não seja tomado as devidas medidas em seu desenvolvimento.

2.3 Arquitetura da Informação em Sistemas Web

A maneira de estruturar as informações que serão disponibilizadas em um sistema web é um fator determinante no sucesso do projeto. Para Silva e Dias (2008), o responsável por definir essa estrutura é a arquitetura da informação. Ainda segundo ele, essa arquitetura organizará a informação e será base para outras partes do site.

Para McGee e Prusak (1994, p. 129), o verdadeiro objetivo de uma arquitetura da informação é conceber um ‘mapa’ amplo dos dados organizacionais e em seguida desenvolver um sistema baseado nesse mapa.

Santos (2001, p.3) declara que a arquitetura da informação deve ser compreendida como uma das formas de se aperfeiçoar a usabilidade de um sistema por intermédio do desenvolvimento de uma estrutura de informação, no qual tornará possível o usuário alcançar suas metas de interação no decorrer do processo de busca de informação. Isso, ainda segundo Santos, é empregado a diferentes tipos de sistemas na web, podendo ser um website de compras, de entretenimento, de informação e outros.

Para Toms e Blades (1999, p.247), a arquitetura da informação compõe-se na maneira pela qual a informação é categorizada, classificada, armazenada, acessada e exibida, acarretando assim, as formas como o usuário poderá se deparar com a informação que necessita.

No livro *Information Architecture: for the World Wide Web*, Morville e Rosenfeld (2006) alegam que a arquitetura da informação não se pode ter uma declaração curta e simples,

visto que inclui desafios particulares da linguagem e representação, nos quais desintegram o conceito nos seguintes pontos:

a) o design estrutural do espaço informacional para simplificar a integridade das tarefas e o acesso intuitivo ao conteúdo;
b) o arranjo dos esquemas de organização, rotulação e navegação dentro de um sistema de informação;
c) a arte e a ciência de formar e classificar websites e intranets para ajudar as pessoas a encontrar e administrar a informação;
d) uma disciplina emergente e uma comunidade de prática propícia em conduzir princípios do design e da arquitetura ao espaço digital.

Ainda, segundo Morville e Rosenfeld (2006), são apresentados quatro sistemas que compõem a arquitetura da informação para diversos sistemas, como websites, intranet, entre outros. São eles:

a) sistemas de organização: responsável por qualificar a informação e agrupá-la;
b) sistemas de navegação: especifica as maneiras de percorrer pelo espaço informacional;
c) sistemas de rotulação: as formas de representação e apresentação da informação, definindo signos para cada elemento informativo, por exemplo uma terminologia científica;
d) sistemas de busca: responsável por definir as perguntas que o usuário poderá fazer e o total de respostas que poderá obter.

Conforme mencionado anteriormente a visão de diferentes autores, cabe ressaltar a importância da arquitetura de informação de um determinado sistema. A forma com que o mesmo é estruturado poderá acarretar tanto pontos positivos, quanto negativos. E, quando estruturado de maneira correta, ocasionará na melhor usabilidade para o usuário final.

Neste trabalho não serão realizados questionários com usuários com o objetivo de medir os seus índices de satisfação. Entretanto, durante o desenvolvimento serão respeitadas características importantes de usabilidade, como as sugeridas por Nielsen (2001). As telas e componentes utilizados serão responsivos visando atender um número maior de dispositivos compatíveis.

3 MATERIAIS E MÉTODO

Este capítulo apresenta as ferramentas e as tecnologias utilizadas na modelagem e na implementação do sistema. Também é apresentada a sequência das atividades desenvolvidas para a realização do trabalho.

3.1 MATERIAIS

Para o desenvolvimento do projeto, foi utilizado a linguagem de programação Java, para estruturar o lado do servidor (*back-end*), sendo utilizado os *frameworks* Spring Boot, Spring MVC, Spring Security e Hibernate.

No lado do cliente (*front-end*), foi utilizado o *framework* Angular para estruturar e apresentar as páginas do sistema.

O Quadro 2 apresenta as ferramentas que foram utilizadas para o desenvolvimento do sistema proposto neste trabalho.

Quadro 2 – Lista de ferramentas e tecnologias

Ferramenta / Tecnologia	**Versão**	**Finalidade**	**Referência**
Java	11	Linguagem de Programação	https://docs.oracle.com/en/java/javase/11/
Spring Boot	2.2.4	Framework *back-end*	https://spring.io/guides
Spring Framework	2.2.4	Conjunto de ferramentas para *back-end*	https://spring.io/guides
Maven	3.6.1	Ferramenta de Automação back-end	https://maven.apache.org/guides/
Angular	9	Framework *front-end*	https://angular.io/cli

TypeScript	3.7.5	Linguagem de programação	
PrimeNg	9	Biblioteca *front-end*	https://www.primefaces.org/primeng/#/setup
HTML	5	Linguagem de Estruturação e Marcação de Conteúdo	
CSS	3	Linguagem de Estilização	
Bootstrap	4.4.1	Biblioteca *front-end*	https://getbootstrap.com/docs/4.3/getting-started/introduction/
IntelliJ Idea Ultimate	2020.1	IDE de Desenvolvimento	https://www.jetbrains.com/idea/documentation/index.html
PostgreSql	10	Banco de Dados Relacional	https://www.postgresql.org/docs/10/index.html
Visual Paradigm	16.1	Modelagem do sistema	https://www.visual-paradigm.com/tutorials/

Fonte: Autoria própria.

3.2 MÉTODO

O desenvolvimento deste projeto foi dividido em três fases. Na primeira fase do desenvolvimento do projeto foi realizado o levantamento de todos os requisitos necessários para a modelagem do sistema vigente, no qual teve como auxílio o orientador do projeto e um responsável pelo laboratório de informática da instituição. Nesta fase, foram validados todos os pontos importantes que deveriam ser desenvolvidos no sistema, de forma que tornasse viável a utilização do mesmo.

Um dos pontos abordados durante a fase do levantamento de requisitos foi a forma de disponibilizar as informações para os usuários do sistema, de forma que o fizesse intuitivo e melhorasse a experiência do usuário durante a navegação.

Na segunda etapa foi realizada uma análise mais objetiva dos requisitos do projeto e desenvolvido o diagrama de casos de uso e diagrama de entidades do banco de dados. Os requisitos e as tabelas do banco foram devidamente documentados.

Na terceira fase foi realizado o desenvolvimento do sistema e os testes. Iniciou-se pelo desenvolvimento do lado servidor da aplicação e na sequência foram desenvolvidas as funcionalidades do lado cliente da aplicação. Durante o desenvolvimento realizaram-se vários testes, em que todas as funcionalidades desenvolvidas foram testadas pelo desenvolvedor do sistema e validadas pelo responsável pelo laboratório de informática do DAINF da UTFPR-PB. Durante os testes, nos casos em que houve necessidade as funcionalidades do sistema foram ajustadas.

4 RESULTADOS

Este capítulo apresenta o resultado da realização deste trabalho que é a modelagem de um sistema web para o gerenciamento de materiais e equipamentos de laboratório em universidades. Na Seção 4.1 é apresentado o escopo geral do sistema, salientando suas funcionalidades. Essas funcionalidades são descritas de maneira mais detalha nos requisitos apresentado na Seção 4.2. Nas seções 4.3 e 4.4 são apresentadas a interface e a implementação do sistema, respectivamente.

4.1 ESCOPO DO SISTEMA

O sistema web de gerenciamento de materiais da sala de apoio visa possibilitar um maior controle dos itens disponíveis para o desenvolvimento das aulas e projetos de pesquisa e extensão, realizados tanto por discentes quanto docentes. O sistema terá quatro perfis de usuário, o administrador (responsável pelo laboratório), o laboratorista, o docente e o discente. Os usuários que tiverem cadastrados com perfil de “Administrador” ou “Laboratorista”, poderão realizar o cadastro dos materiais, e retirar relatórios de empréstimos e itens faltantes, sendo que, o usuário “Administrador” terá acesso a uma rotina exclusiva, que será o cadastro dos usuários no sistema. Já os que tiverem cadastrados como "Docente", poderão realizar empréstimos, consultar os materiais cadastrados e visualizar alguns relatórios. E por último, os que tiverem cadastrados como "Discente" poderão realizar empréstimos, consultar os materiais e solicitar o cadastro de itens.

Os equipamentos e componentes eletrônicos serão cadastrados no sistema, para possibilitar o seu gerenciamento de maneira mais concisa. Esses equipamentos são definidos como materiais de consumo ou permanente, sendo que somente os de consumo são retornados para o estoque. Todos esses materiais são cadastrados por tipo e armazenam a sua respectiva quantidade. Sendo que os permanentes possuem sempre apenas uma unidade. Cada material permanente possui um número de patrimônio que o individualiza. Esse número também é representado por um código de barras específico. Eles possuem também um controle de manutenções realizadas e quando baixados do estoque, deve ser explicitado o motivo da baixa.

A baixa ocorre geralmente quando o equipamento deixa de ser utilizado e isso pode ocorrer por defeito ou substituição por equipamento mais novo, por exemplo.

O controle de estoque realizado dentro do sistema inclui o acréscimo e a redução da quantidade de itens no estoque do laboratório. As alterações na quantidade dos materiais em estoque são realizadas geralmente por um laboratorista. Essa atualização é realizada aumentado a quantidade em estoque quando uma nova quantidade do produto é adquirida ou diminuindo a quantidade quando os materiais são utilizados em aulas práticas, cedidos para projetos, danificados ou quando equipamentos são baixados do estoque, por exemplo. Há também a possibilidade de gerenciar a quantidade reservada de equipamentos e materiais a serem utilizados em aulas ou projetos. Sendo que essa quantidade será apenas reservada do estoque e não subtraída. Todos os usuários podem consultar as quantidades dos materiais em estoque. A apresentação das quantidades é dada da seguinte forma: apresentada a quantidade real em estoque, a quantidade emprestada para a data atual e a quantidade disponível para a data atual.

Os materiais disponíveis no laboratório poderão ser emprestados para alunos e professores para a utilização em projetos. O empréstimo ocorre por um determinado período e no final desse período os mesmos devem ser devolvidos ou devem ser justificadas a sua não devolução. E nesse caso são baixados do estoque. Alguns materiais podem ser utilizados definitivamente no projeto (componentes soldados em placa) ou inutilizados (conectores danificados), por exemplo, e com isso não serão devolvidos. Nesse tipo de caso, a respectiva quantidade será baixada do estoque e uma observação deverá ser incluída com o motivo da baixa. O empréstimo de equipamentos é realizado por unidade. Exemplo: empréstimo do Arduino XYZ123.

Além do gerenciamento de estoque realizado pelo sistema, será efetuado também o envio de *e-mail* com as devidas pendências ou lista de compras de materiais para os seus respectivos usuários. Logo, os usuários que realizaram algum empréstimo no laboratório e não devolveram dentro do prazo vigente, receberão um *e-mail* avisando sobre o seu débito. E os usuários que administram o laboratório receberão *e-mail* quando um determinado material atingir o seu estoque mínimo ou tiver fora de estoque.

4.2 MODELAGEM DO SISTEMA

Os quadros numerados de 3 a 12 apresentam a listagem e descrição dos requisitos funcionais e não-funcionais do sistema, representados pelas siglas RF e RFN, respectivamente.

Quadro 3 – Requisitos para acesso ao sistema

RF 1 – Acesso ao sistema		
Descrição: O sistema deve permitir que o usuário realize o acesso ao sistema por meio de seu usuário e senha cadastrados.		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 1.1	Validação	Os campos usuário e senha deverão ser devidamente validados de forma com que seja realizada uma autenticação segura. Deverá ser verificado se contém algum usuário com o usuário e senha (criptografada) condizente ao informado no login.
RNF 1.2	Redirecionamento para a tela principal	Ao realizar a autenticação, o usuário deverá ser redirecionado à tela inicial do sistema.

Fonte: Autoria própria.

Quadro 4 – Requisitos para o cadastro de usuários

RF 2 – Cadastro de usuário		
Descrição: O sistema deve permitir o cadastro de novos usuários no sistema.		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 2.1	Validação	Todos os campos do cadastro serão validados no formulário, de forma com que o usuário preencha-os adequamente. Caso houver algum campo que desrespeite a validação, será alertado o usuário visualmente com uma mensagem de erro. Os dados só poderão ser salvos no banco após a validação.

Fonte: Autoria própria.

Quadro 5 – Requisitos para o cadastro de itens

RF 3 – Cadastro de itens		
Descrição: O sistema deve permitir o cadastro de novos itens, para a realização de futuros empréstimos.		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição

RNF 3.1	Validação	Todos os campos do cadastro serão validados no formulários, de forma com que o usuário preencha-os adequamente. Caso houver algum campo que desrespeite a validação, será alertado o usuário visualmente com uma mensagem de erro. Os dados só poderão ser salvos no banco após a validação.

Fonte: Autoria própria.

Quadro 6 – Requisitos para o cadastro de empréstimos

RF 4 – Cadastro de empréstimos		
Descrição: O sistema deve permitir o cadastro de novos empréstimos de materiais e equipamentos do laboratório.		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 4.1	Validação	Todos os campos do cadastro serão validados nos formulários, de forma com que o usuário preencha-os adequamente. Caso houver algum campo que desrespeite a validação, será alertado o usuário visualmente com uma mensagem de erro. Os dados só poderão ser salvos no banco após a validação.
RNF 4.2	Aviso para empréstimos com data de devolução atrasadas.	O sistema deverá enviar um *e-mail* para os usuários que realizaram um empréstimo e não devolveram o material até a data prevista.

Fonte: Autoria própria.

Quadro 7 – Requisitos para o controle de estoque

RF 5 – Controle de estoque		
Descrição: O sistema deve permitir o gerenciamento de estoque dos materiais e equipamentos do laboratório.		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 5.1	Validação	Não poderá ser alterado o estoque de um item caso o seu saldo final ficar negativo. Caso isso aconteça, será disparado um alerta para o usuário.

Fonte: Autoria própria.

Quadro 8 – Requisitos para a solicitação de compra de materiais e equipamentos

RF 6 – Solicitação de compra de materiais e equipamentos

Descrição: O sistema deve permitir que o usuário solicite a compra de materiais e equipamentos do laboratório.		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 6.1	Validação	Todos os campos do cadastro serão validados nos formulários, de forma com que o usuário preencha-os adequamente. Caso houver algum campo que desrespeite a validação, será alertado o usuário visualmente com uma mensagem de erro. Os dados só poderão ser salvos no banco após a validação.
RNF 6.2	Permitir solicitar a compra de materiais que não possuem estoque	O sistema deverá permitir a solicitação de compra apenas aos materiais que não constam no estoque.

Fonte: Autoria própria.

Quadro 9 – Requisitos para a reserva de materiais para empréstimo

RF 7 – Reservar materiais para empréstimo		
Descrição: O sistema deve permitir que o usuário realize a reserva de materiais que o mesmo pretende emprestar do laboratório		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 7.1	Disponibilizar apenas materiais que possuem estoque	Só poderão ser reservados os materiais disponíveis em estoque. Caso contrário, esses materiais só estarão disponíveis para a solicitação de compra.

Fonte: Autoria própria.

Quadro 10 – Requisitos para controlar a entrada e saída de materiais e equipamentos do laboratório

RF 8 – Controlar a entrada e saída de materiais e equipamentos do laboratório		
Descrição: O sistema deve permitir que o usuário realize o controle de entrada e saída dos materiais e equipamentos do laboratório.		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 8.1	Preenchimento adequado das informações.	Só poderá registrar uma entrada ou saída do material quando o mesmo não for proveniente de uma compra ou um defeito. Ou seja, as compras e os empréstimos deverão ser registrados em suas devidas telas.
RNF 8.2	Não permitir a saída de um material sem um	O sistema deverá obrigar o usuário informar o motivo pelo qual um material está deixando o estoque, de forma que

	motivo definido.	possa ter uma maior controle sobre eles.
RNF 8.3	Atualizar o preço do material adquirido	O sistema deverá atualizar o preço do material em seu cadastro, respeitando o preço pago na entrada vigente.

Fonte: Autoria própria.

Quadro 11 – Requisitos para a geração de relatórios

RF 9 – Gerar relatórios		
Descrição: O sistema deve permitir que o usuário gere relatórios que o auxiliem no gerenciamento dos materiais e equipamentos do laboratório.		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 9.1	Permitir informar alguns filtros na geração do relatório	O sistema deverá permitir que o usuário filtre o que deseja que sejá gerado no relatório. Como exemplo, filtrar os materiais que foram emprestados em um determinado período.

Fonte: Autoria própria.

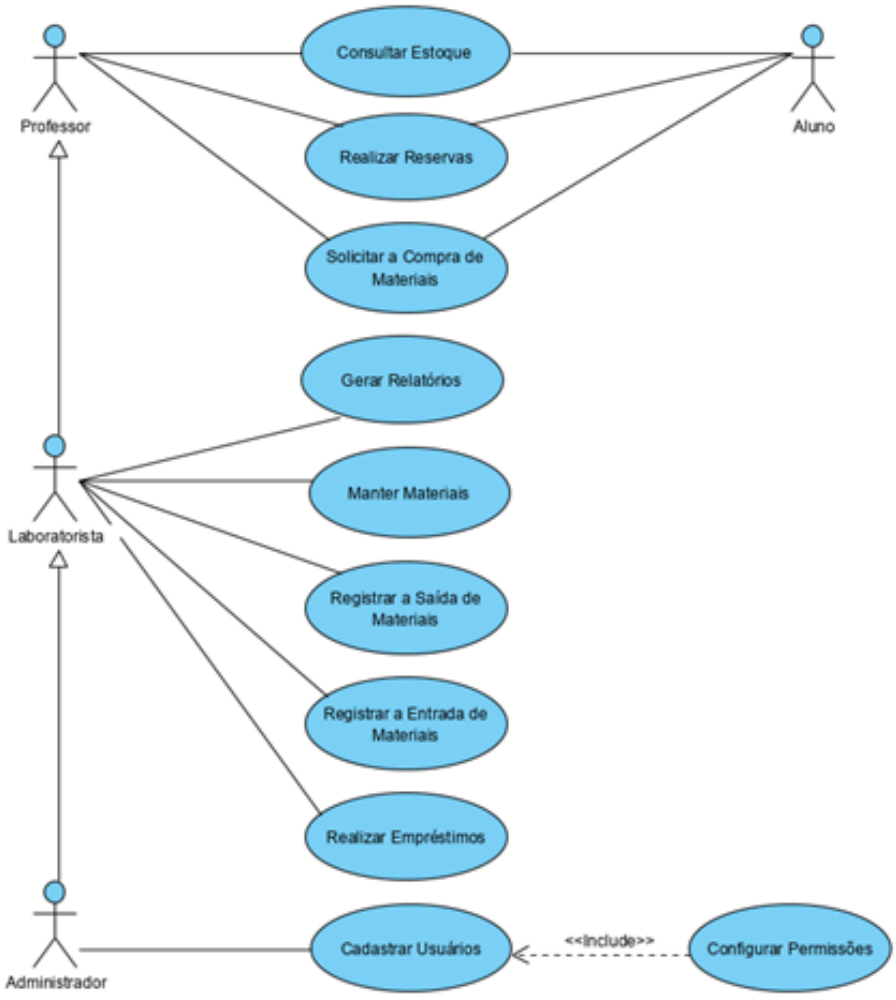
Quadro 12 – Requisitos para a compra de materiais e equipamentos

RF 10 – Realizar a compra de materiais e equipamentos		
Descrição: O sistema deve permitir que o usuário realize a compra de ma		
Requisitos Não-Funcionais		
Identificação	Nome	Restrição
RNF 10.1	Validação	Além dos campos obrigatórios, deverá ser informado ao menos um item na compra. Caso houver alguma informação que desrespeite a validação, será alertado o usuário visualmente com uma mensagem de erro. Os dados só poderão ser salvos no banco após a validação.
RNF 10.2	Atualizar o preço do material adquirido	O sistema deverá atualizar o preço do material em seu cadastro, respeitando o preço pago na compra realizada.

Fonte: Autoria própria.

A Figura 1 apresenta o diagrama de casos de uso do sistema. Esse diagrama contém as funcionalidades primordiais do sistema, nas quais são realizadas pelos atores Administrador, Laboratorista, Professor e Aluno. O administrador será responsável por manter usuários. Além disso, ele terá acesso total no sistema, podendo realizar todas as demais rotinas do mesmo. O laboratorista será responsável em manter materiais, realizar os empréstimos, registrar a entrada e saída de materiais e gerar relatórios. O professor e o aluno poderão solicitar a compra de materiais, realizar a reserva de materiais e consultar o estoque.

Figura 1 – Diagrama de casos de uso

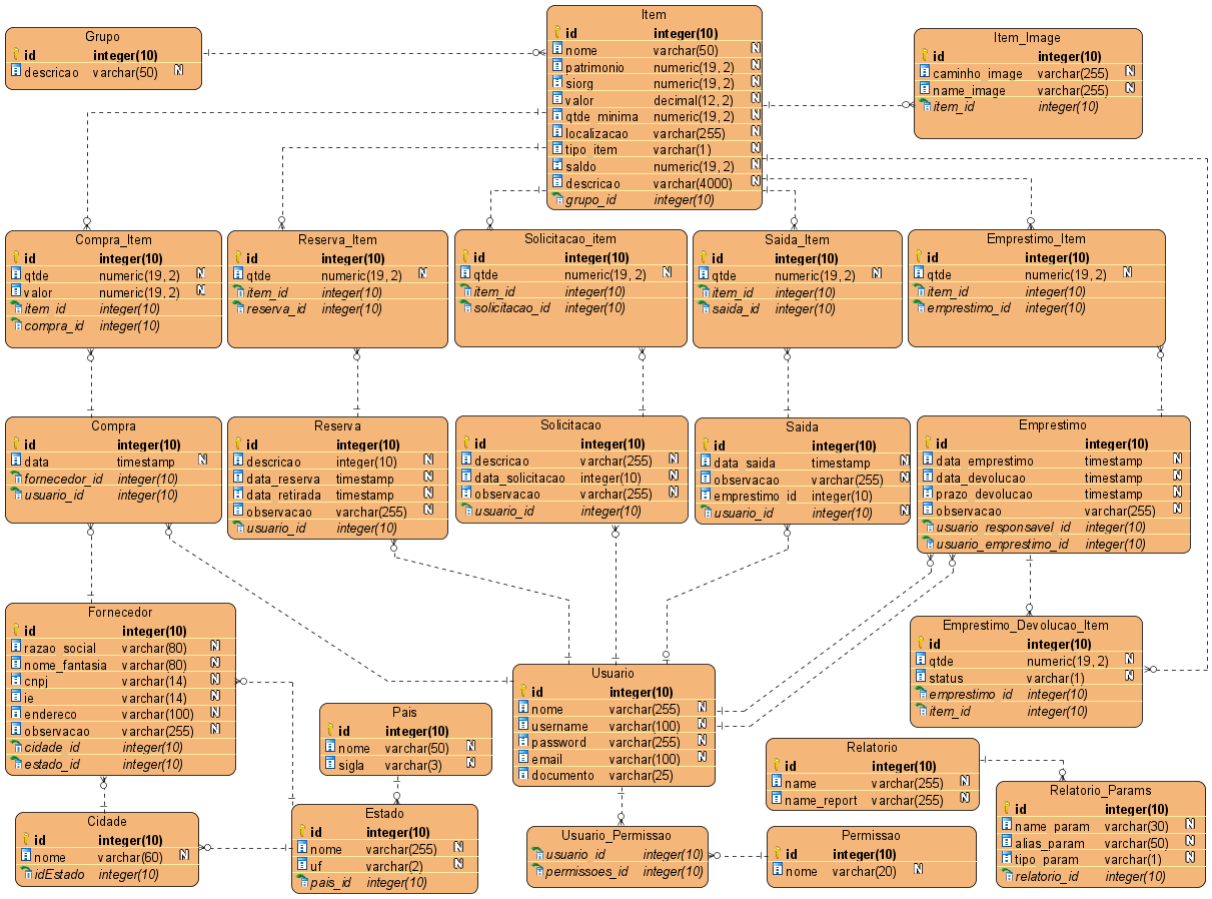


Fonte: Autoria própria.

A modelagem do sistema inclui os diagramas e as descrições textuais para representar o problema e a solução.

Na Figura 2 é apresentado o diagrama de entidades e relacionamentos do banco de dados do sistema. Esse diagrama representa a relação entre cada entidade, de forma com que possa ser definido e organizado devidamente a iteração entre os dados. As entidades que representam as funcionalidades principais do sistema são *Item*, *Emprestimo* e *Emprestimo_Item*. A entidade Item representa os materiais de consumo e permanentes que devem ser armazenados no sistema. Nessa entidade o campo patrimônio estará presente apenas nos materiais permanentes. A entidade Emprestimo contém as informações básicas de todos os empréstimos que serão realizados no sistema, tais como a data do empréstimo e data de devolução, o usuário que realizou o empréstimo e o usuário que emprestou determinado material no laboratório. Já a entidade Emprestimo_Item contém as informações de cada material que foi emprestado e as suas respectivas quantidades.

Figura 2 – Diagrama de entidade e relacionamentos



Fonte: Autoria própria.

Os quadros a seguir apresentam a descrição das tabelas que constam na Figura 3. Os campos da tabela de item são apresentados no Quadro 13.

Quadro 13 – Tabela Item

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
nome	Texto	Não	Não	Não	
patrimonio	Numérico	Não	Não	Não	
siorg	Numérico	Não	Não	Não	
valor	Numérico	Não	Não	Não	
qtde_minima	Numérico	Sim	Não	Não	
localizacao	Texto	Sim	Não	Não	
tipo_item	Enum	Não	Não	Não	
saldo	Numérico	Sim	Não	Não	
descricao	Texto	Sim	Não	Não	
grupo_id	Numérico	Sim	Não	Sim	Da tabela Grupo

Fonte: Autoria própria.

Os campos da tabela ItemImage são apresentados no Quadro 14.

Quadro 14 - Tabela ItemImage

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
caminho_image	Texto	Não	Não	Não	
name_image	Texto	Não	Não	Não	
item_id	Numérico	Não	Não	Sim	Da tabela Item

Fonte: Autoria própria.

Os campos da tabela de grupo são apresentados no Quadro 15.

Quadro 15 – Tabela Grupo

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
descricao	Texto	Não	Não	Não	

Fonte: Autoria própria.

Os campos da tabela de compra são apresentados no Quadro 15.

Quadro 16 – Tabela Compra

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
data_compra	Data	Não	Não	Não	
fornecedor_id	Numérico	Não	Não	Sim	Da tabela Fornecedor
usuario_id	Numérico	Não	Não	Sim	Usuário logado

Fonte: Autoria própria.

Os campos da tabela de itens da compra são apresentados no Quadro 16.

Quadro 17 – Tabela CompraItem

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
qtde	Texto	Não	Não	Não	
valor	Numérico	Não	Não	Não	
item_id	Numérico	Não	Não	Sim	Da tabela Item
compra_id	Numérico	Não	Não	Sim	Da tabela Compra

Fonte: Autoria própria.

Os campos da tabela de reserva são apresentados no Quadro 17.

Quadro 18 – Tabela Reserva

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
descricao	Texto	Não	Não	Não	
data_reserva	Data	Não	Não	Não	
observacao	Texto	Sim	Não	Sim	
usuario_id	Numérico	Não	Não	Sim	Usuário logado

Fonte: Autoria própria.

Os campos da tabela dos itens da reserva são apresentados no Quadro 18.

Quadro 19 – Tabela ReservaItem

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
qtde	Numérico	Não	Não	Não	
item_id	Numérico	Não	Não	Sim	Da tabela Item
reserva_id	Numérico	Não	Não	Sim	Da tabela Reserva

Fonte: Autoria própria.

Os campos da tabela de solicitação são apresentados no Quadro 19.

Quadro 20 – Tabela Solicitacao

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
descricao	Texto	Não	Não	Não	
data_solicitacao	Data	Não	Não	Sim	
usuario_id	Numérico	Não	Não	Sim	Usuário logado

observacao	Texto	Sim	Não	Não	

Fonte: Autoria própria.

Os campos da tabela de itens da solicitação são apresentados no Quadro 20.

Quadro 21 – Tabela SolicitacaoItem

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
qtde	Numérico	Não	Não	Não	
item_id	Numérico	Não	Não	Sim	Da tabela Item
solicitacao_id	Numérico	Não	Não	Sim	Da tabela Solicitação

Fonte: Autoria própria.

Os campos da tabela de empréstimos são apresentados no Quadro 21.

Quadro 22 – Tabela Emprestimo

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
data_emprestimo	Data	Não	Não	Não	
prazo_devolucao	Data	Não	Não	Não	
data_devolucao	Data	Sim	Não	Não	
usuario_responsavel_id	Numérico	Não	Não	Sim	Usuário logado
usuario_emprestimo_id	Numérico	Não	Não	Sim	Da tabela Usuário
observacao	Texto	Sim	Não	Não	

Fonte: Autoria própria.

Os campos da tabela EmprestimoItem são apresentados no Quadro 22.

Quadro 23 – Tabela EmprestimoItem

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
qtde	Numérico	Não	Não	Não	
item_id	Numérico	Não	Não	Sim	Da tabela Item
emprestimo_id	Numérico	Não	Não	Sim	Da tabela Empréstimo

Fonte: Autoria própria.

Os campos da tabela EmprestimoDevolucaoItem são apresentados no Quadro 23.

Quadro 24 - Tabela EmprestimoDevolucaoItem

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
qtde	Numérico	Não	Não	Não	
status	Enum	Não	Não	Não	
item_id	Numérico	Não	Não	Sim	Da tabela Item
emprestimo_id	Numérico	Não	Não	Sim	Da tabela Empréstimo

Os campos da tabela de saída são apresentados no Quadro 23.

Quadro 25 – Tabela Saida

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
data_saida	Data	Não	Não	Não	
observacao	Texto	Não	Não	Não	
emprestimo_id	Numérico	Sim	Não	Não	Id do Empréstimo que originar uma

					saída na devolução.
usuario_id	Numérico	Não	Não	Sim	Usuário logado

Fonte: Autoria própria.

Os campos da tabela de saída são apresentados no Quadro 26.

Quadro 26 - Tabela SaidaItem

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
qtde	Numérico	Não	Não	Não	
item_id	Numérico	Não	Não	Sim	Da tabela Item
saida_id	Numérico	Não	Não	Sim	Da tabela Saída

Fonte: Autoria própria.

Os campos da tabela de fornecedor são apresentados no Quadro 25.

Quadro 27 – Tabela Fornecedor

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
razao_social	Texto	Não	Não	Não	
nome_fantasia	Texto	Não	Não	Não	
telefone	Texto	Não	Não	Não	
email	Texto	Não	Não	Não	
ie	Texto	Não	Não	Não	
endereco	Texto	Não	Não	Não	
observacao	Texto	Sim	Não	Não	
cidade_id	Numérico	Sim	Não	Sim	Da tabela Cidade
estado_id	Numérico	Sim	Não	Sim	Da tabela Estado

Fonte: Autoria própria.

Os campos da tabela de cidade são apresentados no Quadro 26.

Quadro 28 – Tabela Cidade

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
nome	Texto	Não	Não	Não	
estado_id	Numérico	Não	Não	Sim	Da tabela Estado

Fonte: Autoria própria.

Os campos da tabela de estado são apresentados no Quadro 27.

Quadro 29 – Tabela Estado

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
nome	Texto	Não	Não	Não	
uf	Texto	Não	Não	Não	
pais_id	Numérico	Não	Não	Sim	Da tabela País

Fonte: Autoria própria.

Os campos da tabela de país são apresentados no Quadro 28.

Quadro 30 – Tabela País

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
nome	Texto	Não	Não	Não	
sigla	Texto	Não	Não	Não	

Fonte: Autoria própria.

Os campos da tabela de usuários são apresentados no Quadro 29.

Quadro 31 – Tabela Usuario

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
nome	Texto	Não	Não	Não	
username	Texto	Não	Não	Não	
password	Texto	Não	Não	Não	
documento	Texto	Sim	Não	Não	
e-mail	Texto	Sim	Não	Não	
telefone	Texto	Não	Não	Não	

Fonte: Autoria própria.

Os campos da tabela de permissões são apresentados no Quadro 30.

Quadro 32 – Tabela Permissao

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
nome	Texto	Não	Não	Não	

Fonte: Autoria própria.

Os campos da tabela de permissões de usuários são apresentados no Quadro 31.

Quadro 33 – Tabela UsuarioPermissao

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
usuario_id	Numérico	Não	Não	Sim	Da tabela Usuário
permissoes_id	Numérico	Não	Não	Sim	Da tabela Permissão

Fonte: Autoria própria.

Os campos da tabela de relatório são apresentados no Quadro 34.

Quadro 34 - Tabela Relatorio

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
nome	Texto	Não	Não	Não	
name_report	Texto	Não	Não	Não	

Fonte: Autoria própria.

Os campos da tabela RelatorioParams são apresentados no Quadro 35.

Quadro 35 - Tabela RelatorioParams

Campo	Tipo	Nulo	Chave Primária	Chave Estrangeira	Observações
id	Numérico	Não	Sim	Não	
name_param	Texto	Não	Não	Não	
alias_param	Texto	Não	Não	Não	
tipo_param	Enum	Não	Não	Não	
relatorio_id	Numérico	Não	Não	Sim	Da tabela Relatorio

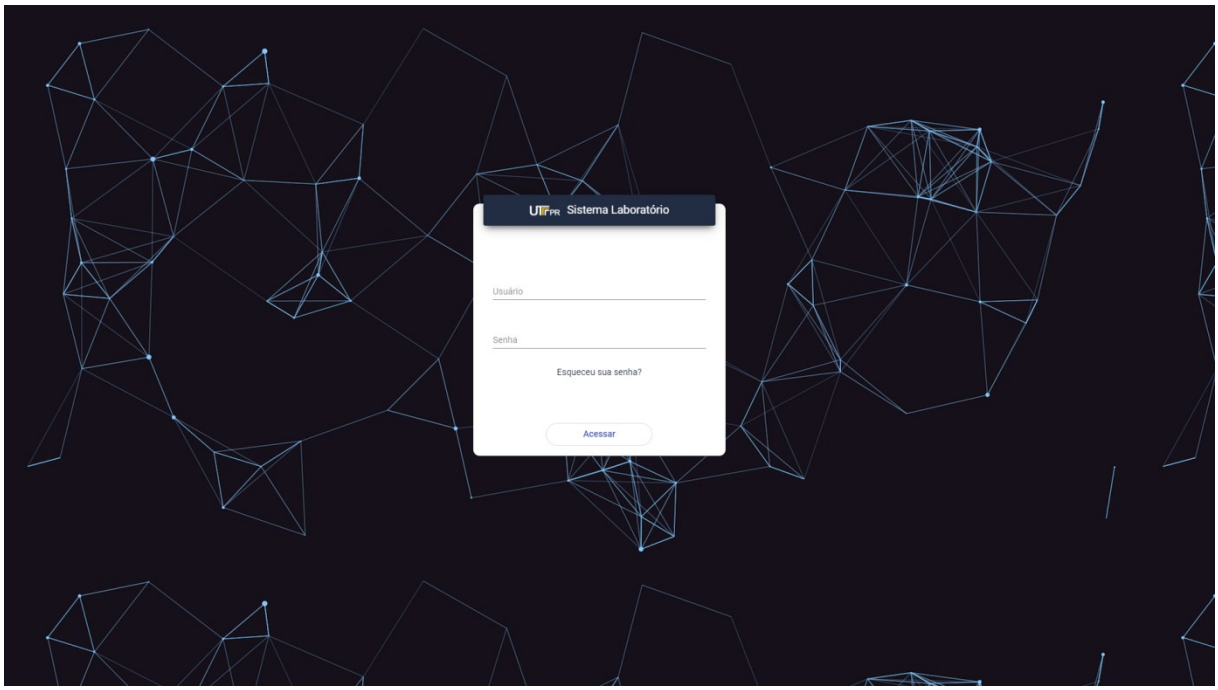
Fonte: Autoria própria.

4.3 APRESENTAÇÃO DO SISTEMA

Nesta seção serão apresentadas as principais funcionalidades desenvolvidas no sistema por meio de telas e descrições de suas funções. As páginas da aplicação foram criadas visando atender todos os dispositivos, levando em consideração o conceito da responsividade.

Na Figura 3 é apresenta a página de autenticação do sistema. Para autenticar-se é necessário informar o usuário e senha cadastrados.

Figura 3 - Tela de Autenticação

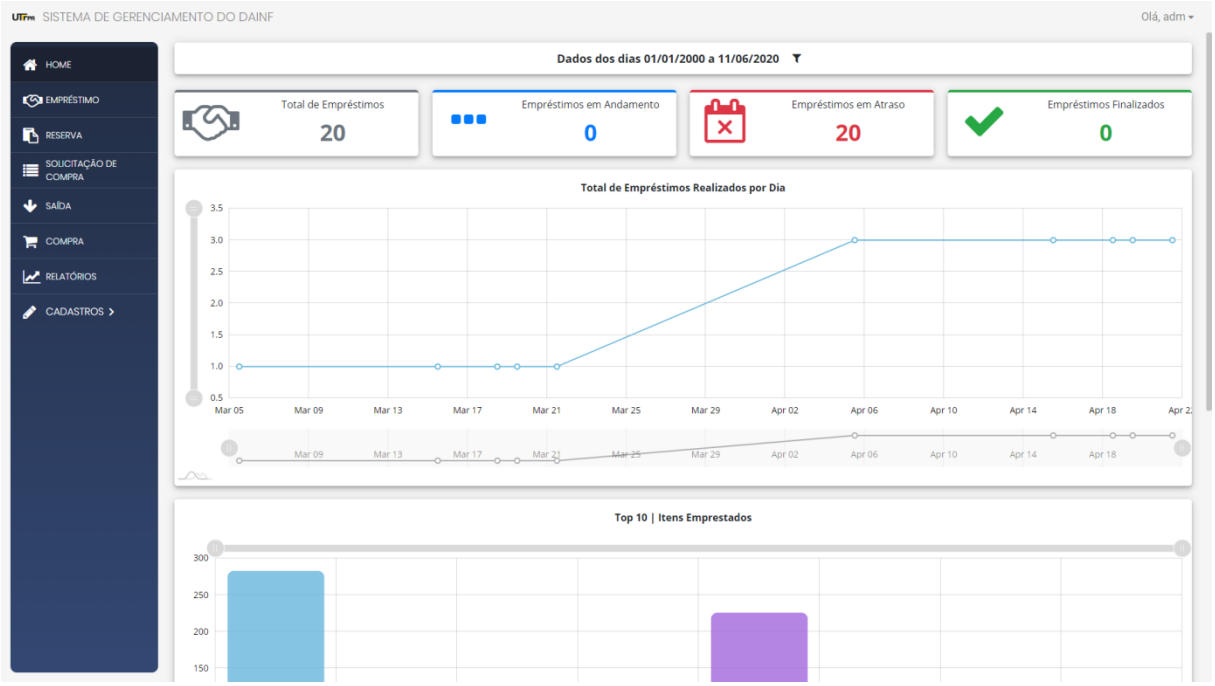


Fonte: Autoria própria.

Ao realizar a autenticação, o usuário é direcionado para a tela inicial do sistema, em que são apresentados alguns *dashboards*, como pode ser observado nas Figuras 4 e 5, com informações sobre as principais rotinas do sistema. Esses *dashboards* só são apresentados para os usuários que são responsáveis pelo laboratório.

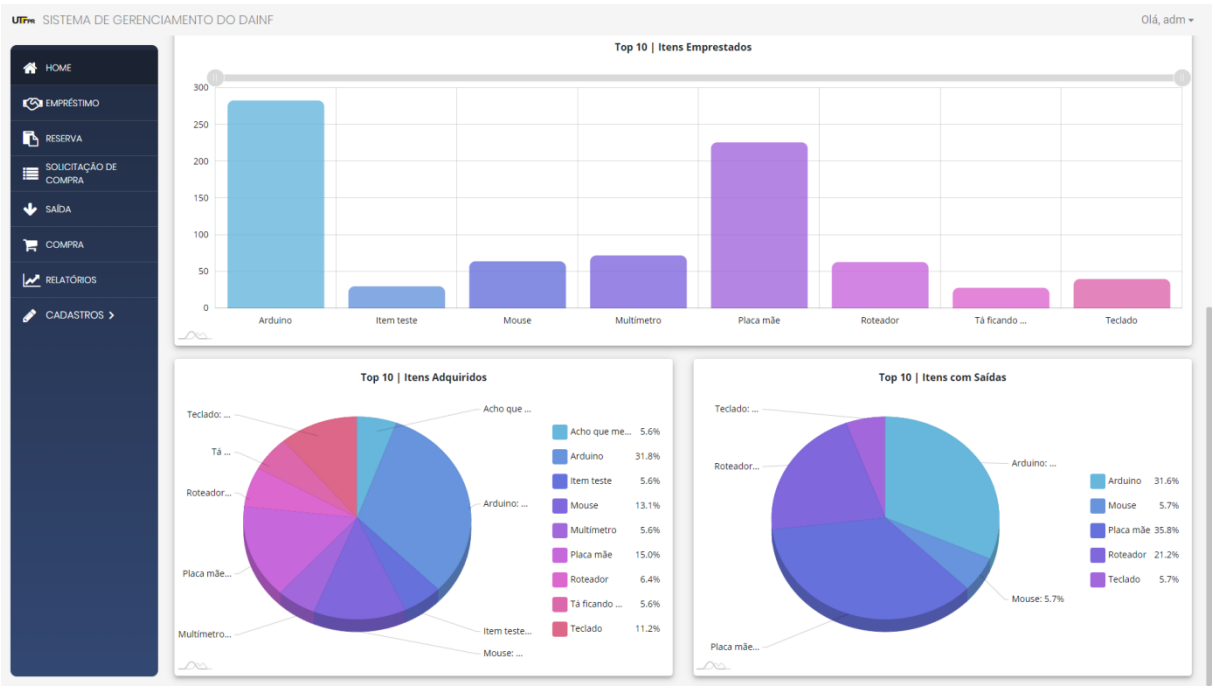
O menu com acesso às rotinas do sistema fica localizado no lado esquerdo da tela, nele estão os atalhos para todas as funcionalidades do sistema. Os usuários que tiverem permissão de laboratorista e administrador, terão acesso a todas opções do menu, com exceção que o laboratorista não poderá cadastrar novos usuários.

Figura 4 - Tela Inicial do Sistema (Parte 1)



Fonte: Autoria própria.

Figura 5 - Tela Inicial do Sistema (Parte 2)

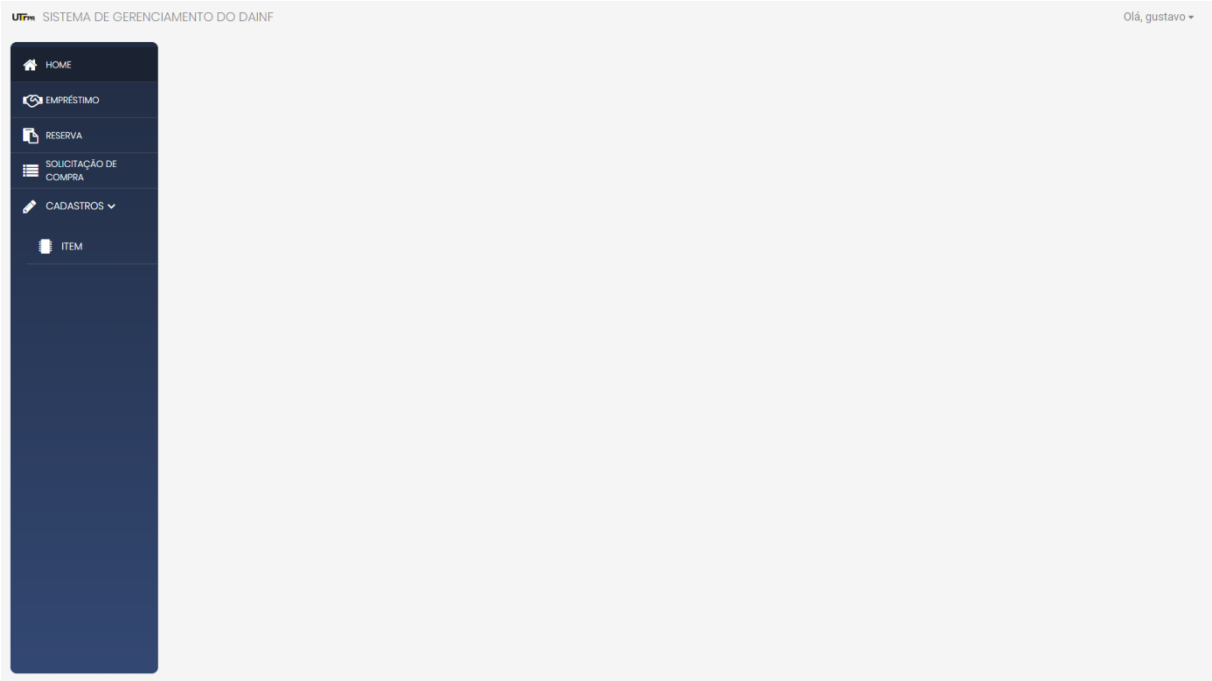


Fonte: Autoria própria.

Os usuários que tiverem a permissão de aluno ou professor poderão utilizar apenas algumas rotinas do sistema e, não poderão visualizar os *dashboards* da tela inicial. Estes

usuários poderão realizar o cadastro de solicitação de compra e reserva de materiais. Nas demais funcionalidades que eles possuem acesso só será possível visualizar as informações sem a opção alterá-las. Na Figura 6 é possível visualizar a tela inicial do sistema, quando um usuário com perfil de professor ou aluno está autenticado.

Figura 6 - Tela de Acesso dos Usuários com Permissão de Professor ou Aluno



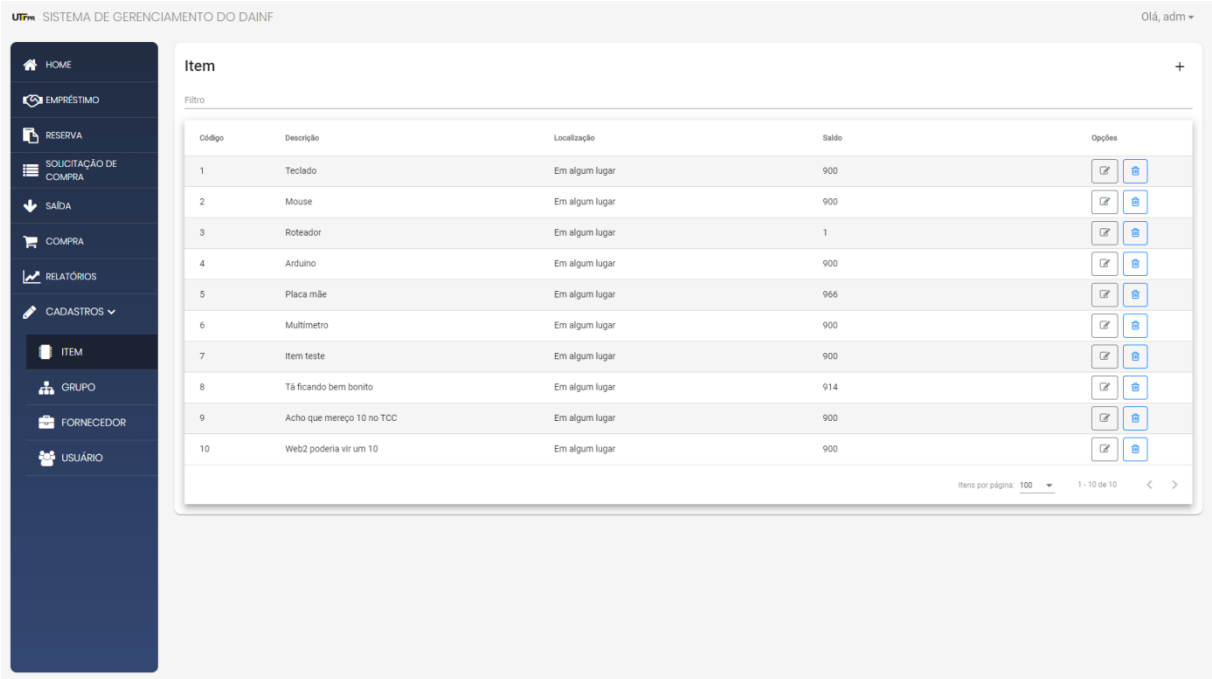
Fonte: Autoria própria.

A Figura 7 apresenta a lista de itens cadastrados dentro do sistema. Nela são apresentadas algumas informações básicas sobre o item, como: Código, Descrição, Localização e Saldo.

Para editar ou excluir algum item, basta clicar nos botões localizados no final de cada linha da tabela. Caso seja necessário criar um novo item, basta clicar no botão “Inserir Registro” (ícone de +).

Essa estrutura é apresentada em quase todas as tabelas do sistema, com exceção apenas da rotina de empréstimo e rotina de reserva de materiais.

Figura 7 - Tela de Itens



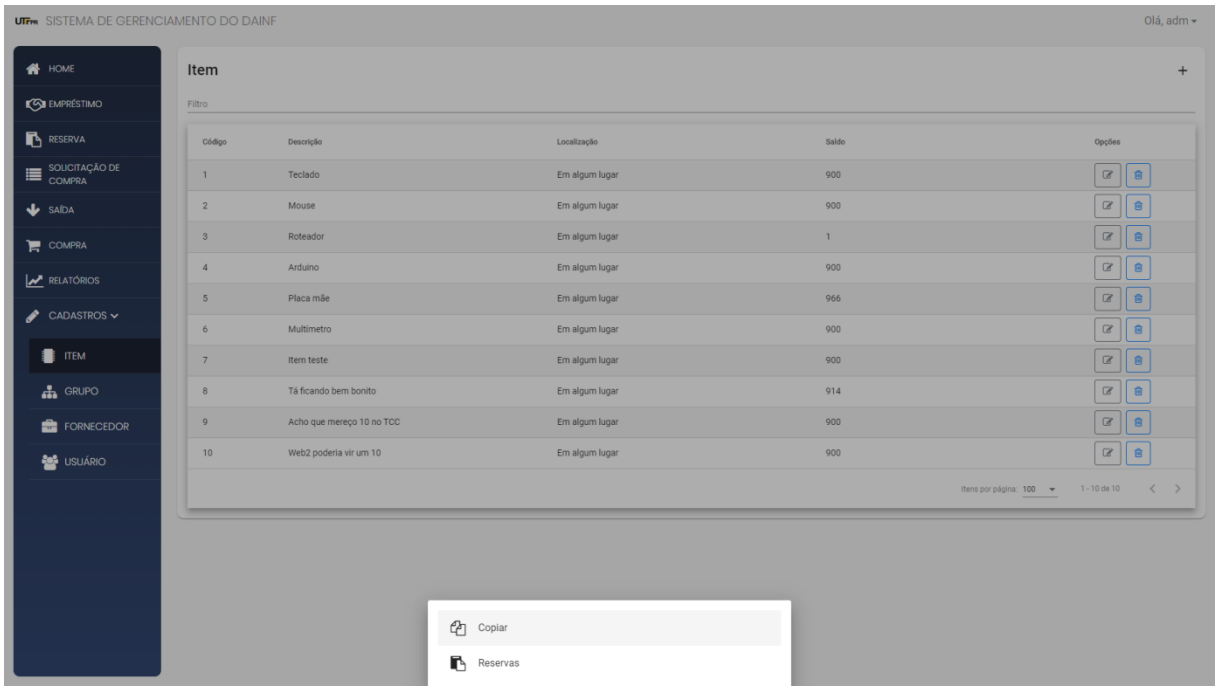
Fonte: Autoria própria.

Além das opções de editar e remover o registro, é possível também copiá-lo e ver as reservas vinculadas a ele. Para que essas opções fiquem visíveis, basta escolher o item e clicar sobre o mesmo. Feito isso elas irão aparecer, conforme a Figura 8.

A opção “Copiar” visa facilitar o cadastro de novos itens que possuem os dados semelhantes, sem a necessidade de preencher todo o formulário de cadastro novamente.

Já a opção “Reservas”, servirá para mostrar todas as reservas vinculadas ao item, informando a quantidade reservada e o dia previsto para a retirada do item reservado.

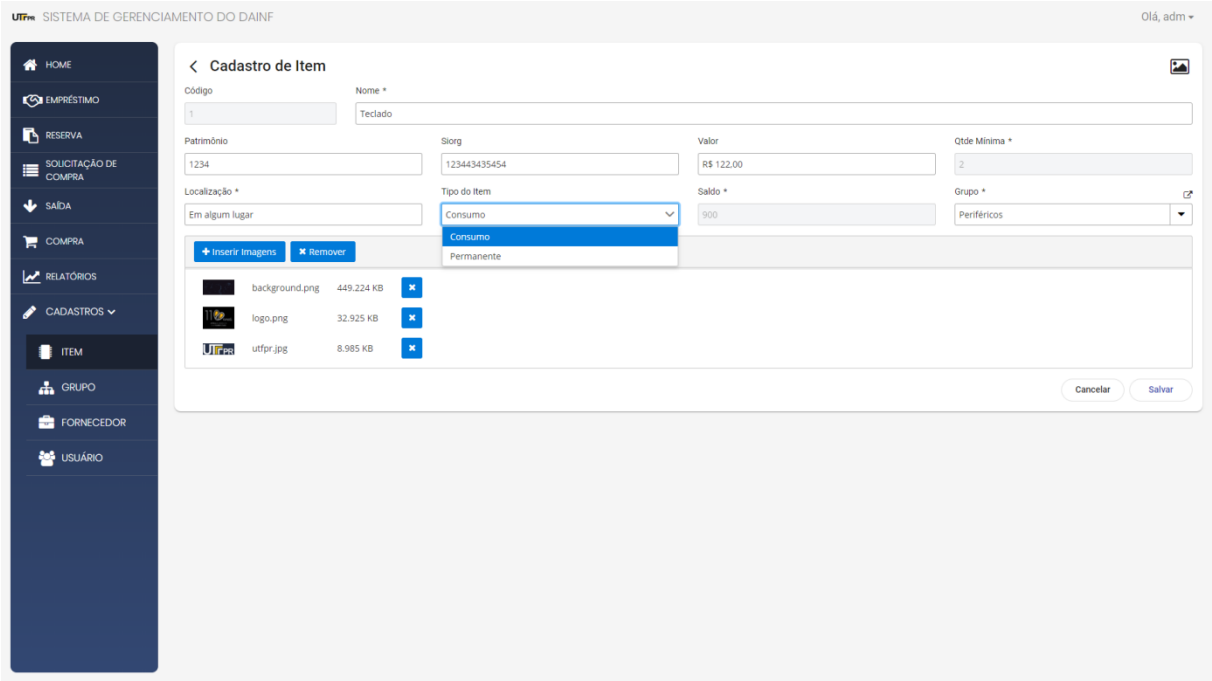
Figura 8 - Opções extras da tabela de itens



Fonte: Autoria própria.

Na Figura 9 é apresentado o formulário de cadastro do item, no qual, além de informar seus dados básicos, é possível anexar imagens relacionadas a ele. O campo “Tipo do Item” é o que define se o item deverá ser devolvido em um empréstimo ou não. Sendo que, caso for do tipo “Permanente”, o mesmo deverá retornar ao estoque, e caso for “Consumo”, não será necessário realizar a sua devolução.

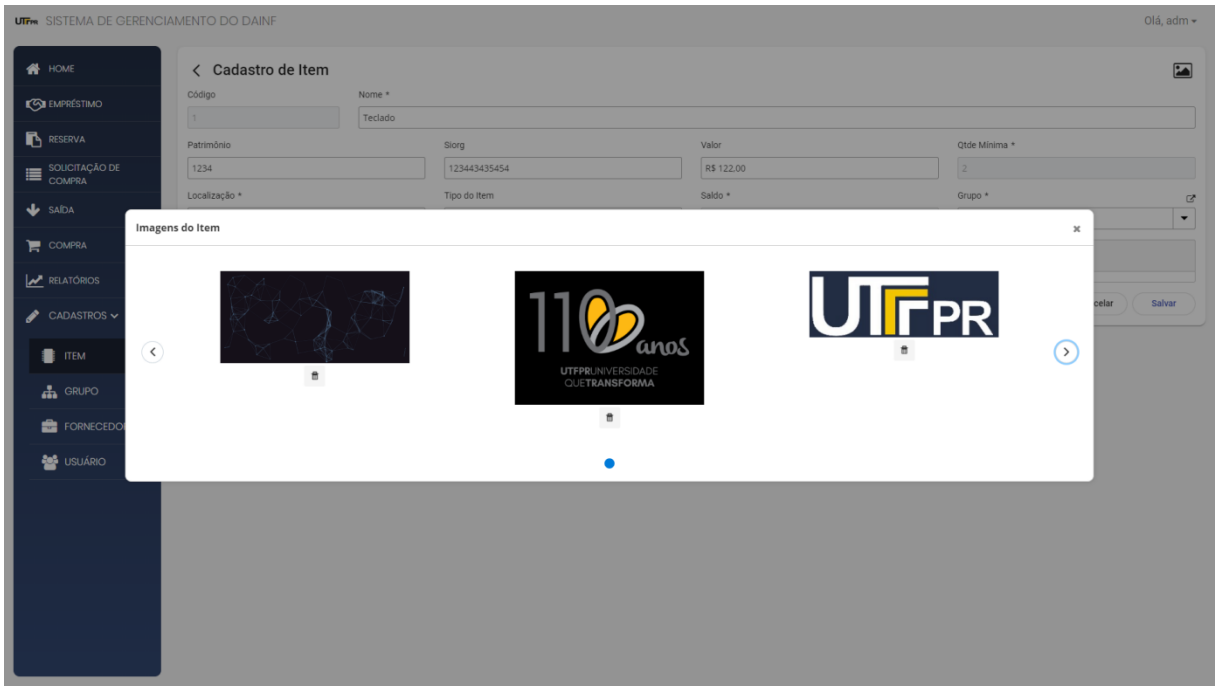
Figura 9 - Tela de Formulário do Item



Fonte: Autoria própria.

Na opção “Imagens” (ícone localizado na parte superior direita do formulário), é possível visualizar as imagens vinculadas ao registro, conforme pode ser visualizado na Figura 10. Abaixo de cada imagem está um botão com ícone de lixeira, utilizado para removê-la quando necessário.

Figura 10 - Modal das imagens do item

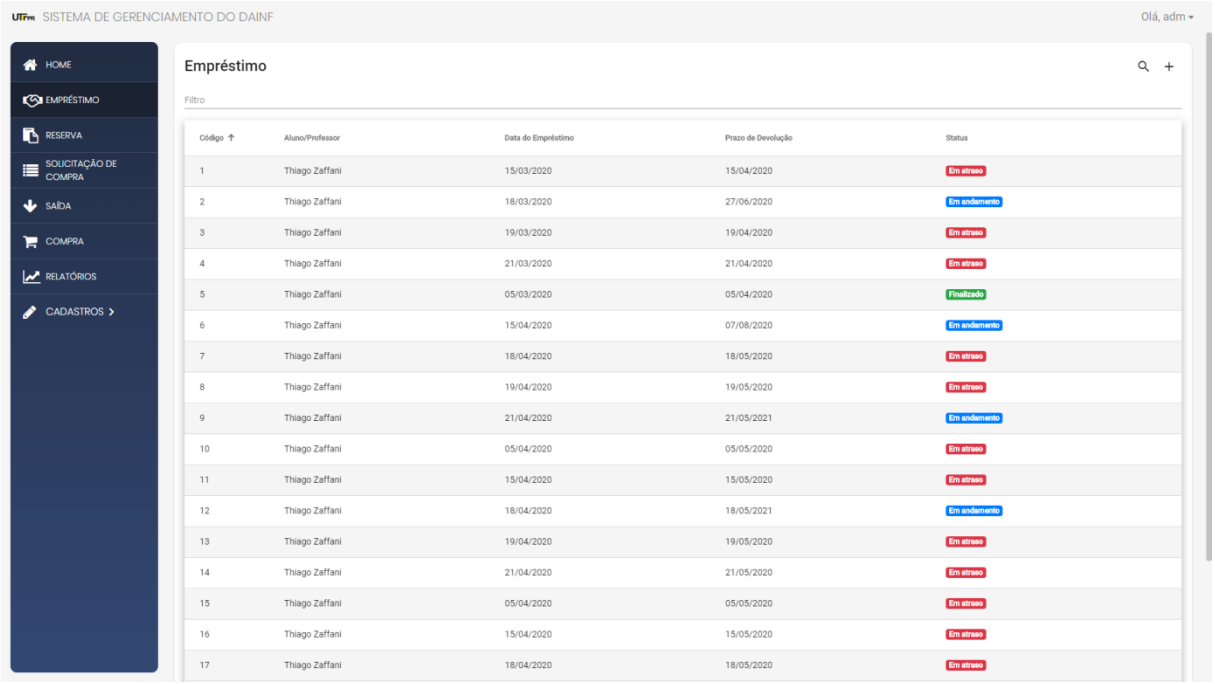


Fonte: Autoria própria.

A Figura 11 apresenta a lista de empréstimos realizados dentro do sistema, na qual é possível visualizar o usuário para quem foi realizado o empréstimo, a data de cadastro, o prazo de devolução e a situação do mesmo.

As situações são divididas em: Em atraso, Em andamento e Finalizado. Os que possuem a situação “Em andamento”, são aqueles que ainda não foram realizadas sua respectiva devolução e está dentro do prazo de devolução. Aqueles empréstimos que já passaram do prazo de devolução ficarão com a situação “Em atraso”, e aqueles que forem efetuados sua devida devolução, ficarão como “Finalizado”.

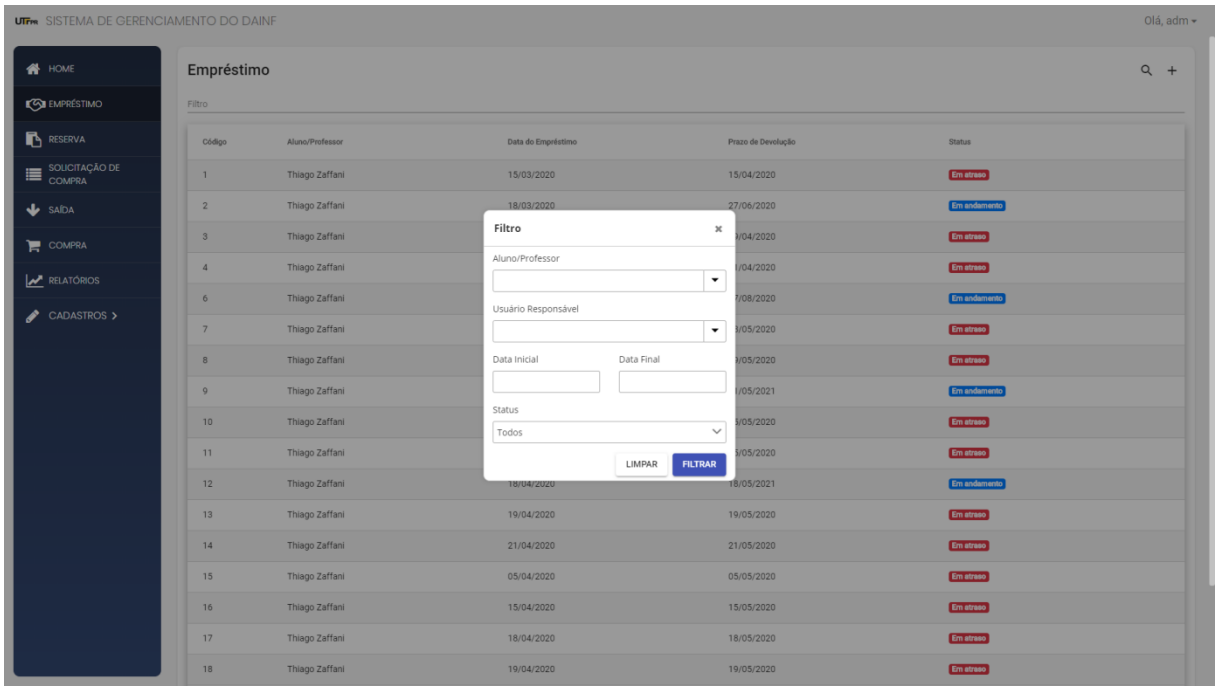
Figura 11 - Tela de Empréstimos



Fonte: Autoria própria.

Na parte superior direita da tela de empréstimos, há dois botões sendo, cada um deles, responsáveis por filtrar e criar novos registros. Ao clicar no botão de filtrar, será aberto uma janela *modal*, na qual o usuário terá a opção de filtrar por aluno/professor, usuário responsável, data de inicial e final do empréstimo, e também pela situação, como pode ser observado na Figura 12.

Figura 12 - Modal do filtro dos empréstimos



Fonte: Autoria própria.

Caso o usuário clique na opção “Novo Registro”, ele será redirecionado para o formulário do empréstimo, conforme apresentado na Figura 13.

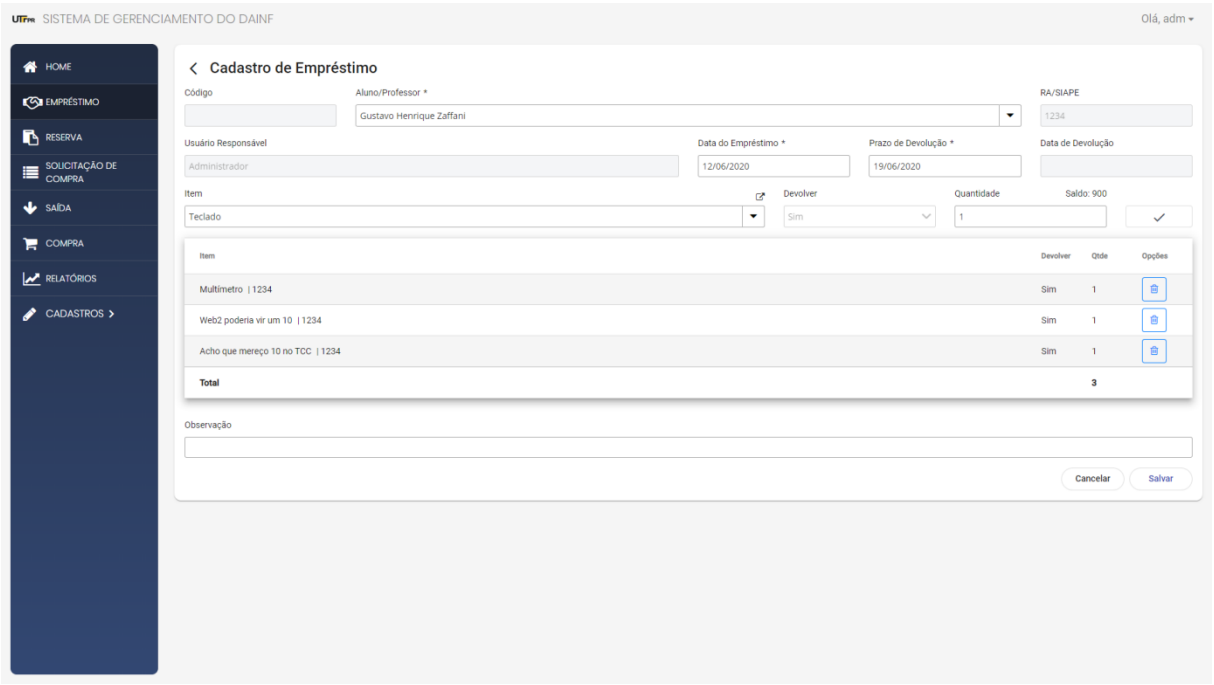
Ao escolher o usuário que está realizando o empréstimo, o campo “RA/SIAPE” será preenchido automaticamente, conforme estiver vinculado no cadastro do usuário. O campo “Data do Empréstimo” será iniciado por padrão com a data atual, porém pode ser alterada para datas anteriores. O campo “Prazo de Devolução” respeitará sempre a data do empréstimo, ou seja, não será permitido informar um prazo de devolução menor que a data do empréstimo. Ao buscar um item, por meio do campo “Item”, será informado automaticamente se o mesmo terá que ser devolvido ou não, respeitando o tipo do item (consumo ou permanente), essa informação ficará disponibilizada no campo “Devolver”. Além da informação da devolução, no campo de “Quantidade” será mostrado o saldo atual do item, deixando o processo mais simples e fazendo com que o usuário que está criando o empréstimo saiba de imediato o saldo desse item no estoque do laboratório.

Foram criados alguns atalhos para o teclado, que facilitam a inserção de novos itens no empréstimo. Caso pressionado as teclas CTRL + Enter, o item será adicionado no empréstimo, e o campo “Item” receberá o foco do cursor novamente, tornando assim o processo mais eficaz. Além desse comando, é possível navegar entre o campo “Item” e o campo

“Quantidade” de maneira mais prática. Logo, se o campo “Item” estiver com foco, basta pressionar CTRL + Seta Direita, que o campo “Quantidade” receberá o foco. E se tiver no campo “Quantidade”, basta fazer o processo ao contrário, ou seja, CTRL + Seta Esquerda.

Estes comandos também estão disponíveis em outras rotinas do sistema que possuem a inserção de itens, são elas: Reserva, Solicitação de Compra, Saída e Compra.

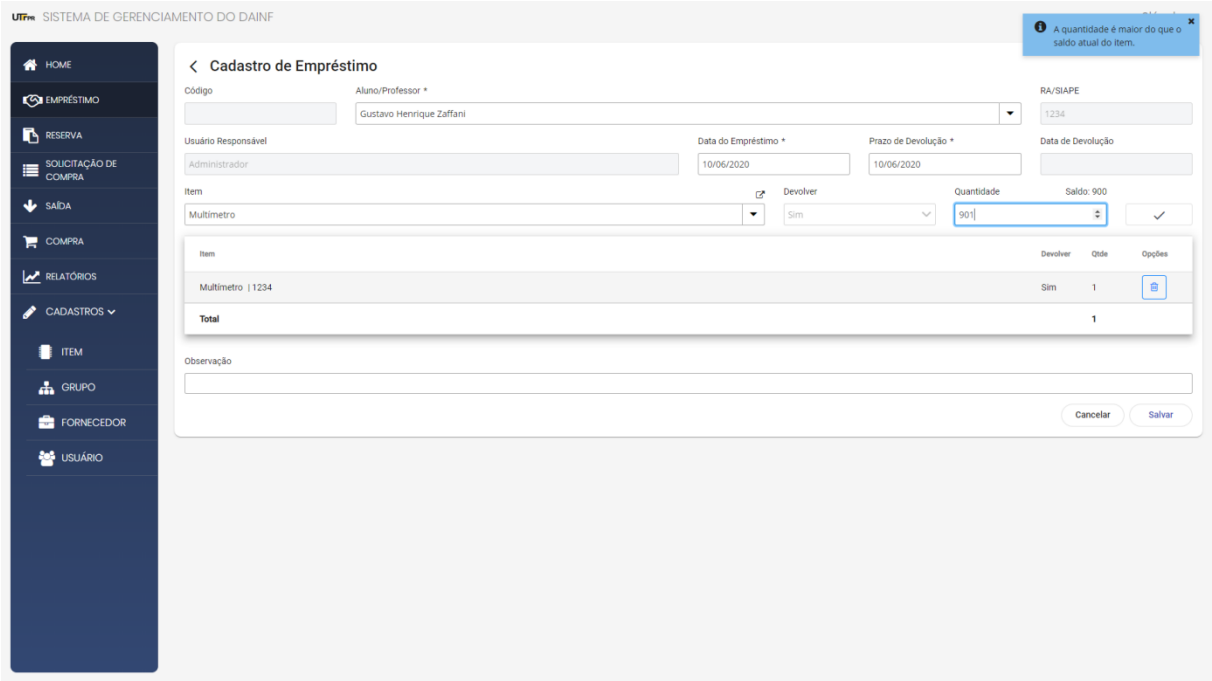
Figura 13 - Tela de formulário do empréstimo



Fonte: Autoria própria.

Caso o usuário tente inserir uma quantidade maior do que o saldo do item, será apresentado uma mensagem no canto superior direito da tela, informado que a quantidade informada é maior do que o saldo atual do item, conforme apresentado na Figura 14. E, consequentemente, não será possível adicionar o mesmo no empréstimo.

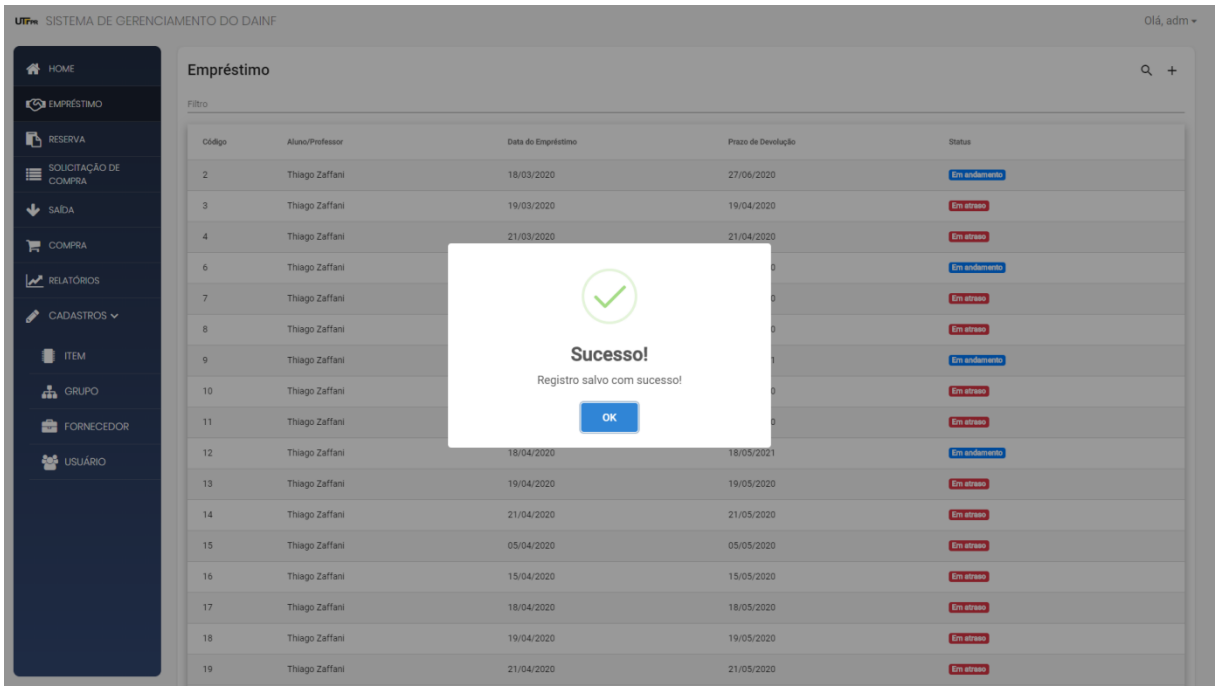
Figura 14 - Mensagem de erro ao tentar inserir um item com quantidade maior que o saldo



Fonte: Autoria própria.

Na Figura 15 é possível visualizar a mensagem de sucesso que é apresentada ao cadastrar um novo empréstimo. Essa mensagem de sucesso é padrão do sistema, logo todas as rotinas seguirão o mesmo padrão.

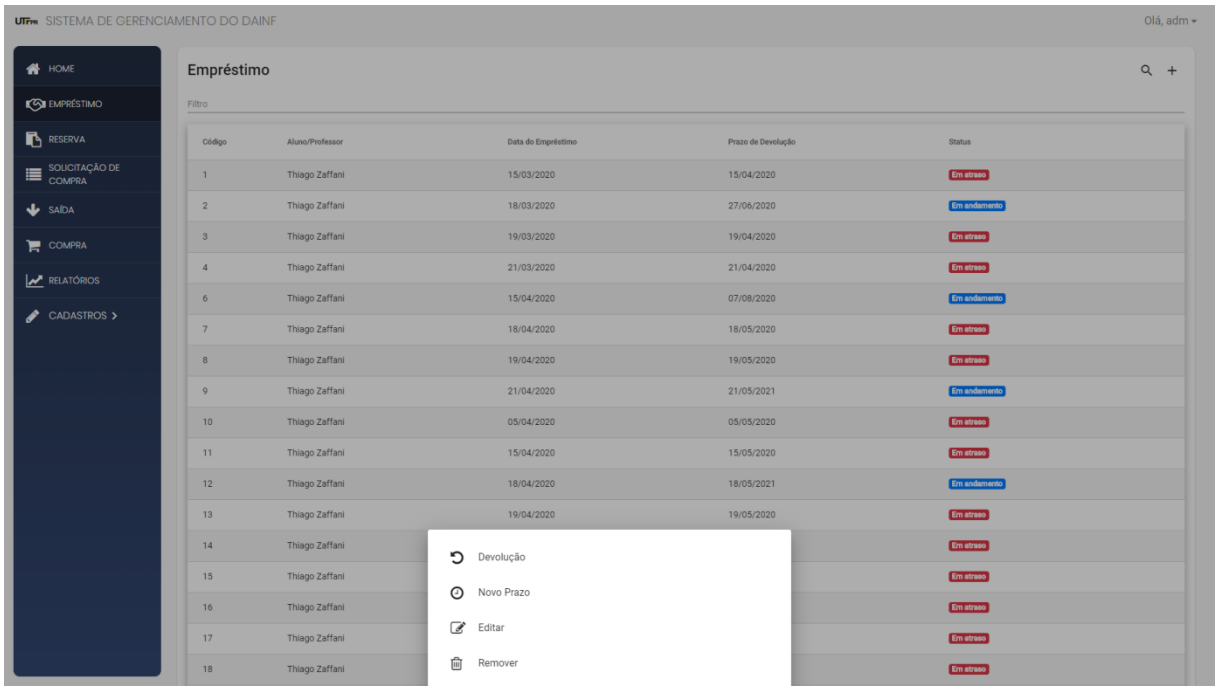
Figura 15 - Mensagem de sucesso ao finalizar o empréstimo (mensagem padrão do sistema)



Fonte: Autoria própria.

Ao clicar sobre um empréstimo, aparecerá as opções de manipulação do registro, sendo as opções de “Devolução”, “Novo Prazo”, “Editar” e “Remover”, como pode ser visualizado na Figura 16. Essas opções são apresentadas apenas para os usuários que tiverem o acesso para criar novos empréstimos, podendo ser um laboratorista ou um administrador. Os docentes e discentes poderão apenas visualizar as informações de seus próprios empréstimos, e não poderão alterá-los.

Figura 16 - Opções do empréstimo



Fonte: Autoria própria.

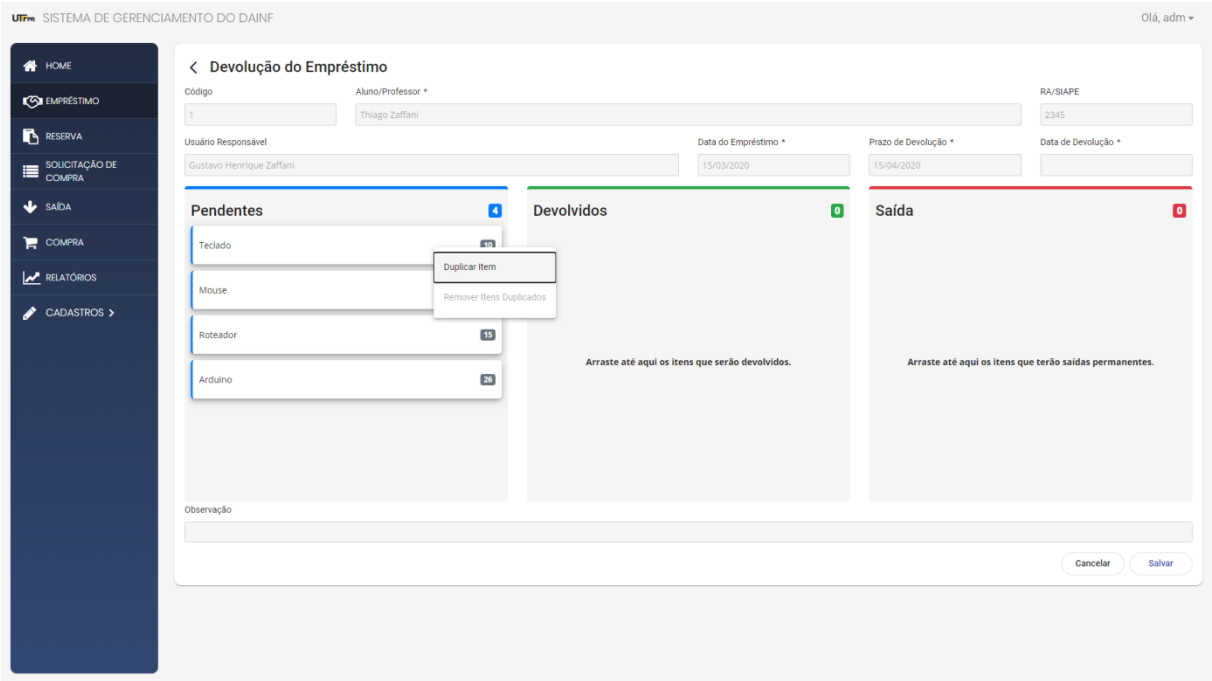
Ao clicar na opção “Devolução”, o usuário será redirecionado para uma nova tela, conforme apresentado na Figura 17, no qual ele poderá visualizar todos os dados do empréstimo e fazer a devolução de todos os itens. Nessa tela, há três quadros, sendo eles “Pendentes”, “Devolvidos” e “Saída”.

Ao iniciar uma devolução, todos os itens estarão no quadro de “Pendentes”, e para efetuar a devolução, basta arrastá-los para os demais quadros. Os itens que ficarem no quadro de “Devolvidos”, terão o retorno no estoque, já no quadro de “Saída”, os mesmos terão a baixa definitiva do estoque, e será realizado um registro na rotina de saída.

Se, por exemplo, foi realizado o empréstimo de dez arduinos, e um deles foi danificado, será possível dividir o registro no quadro de “Pendentes”, e dessa forma arrastar a quantia correta para os seus respectivos quadros. Ou seja, nove arduinos irão para o quadro de “Devolvidos” e um irá para o quadro de “Saída”. Para isso, basta clicar com o botão direito do mouse encima do item que deverá ser duplicado, e clicar na opção “Duplicar Item”, com isso será aberto um modal, no qual deverá ser informado a quantidade do novo registro que será duplicado. Será validado para que o novo registro no seja um valor válido, ou seja, não é possível informar a quantidade dez, sendo que o total já é 10. Na Figura 18 é possível visualizar a imagem da tela de duplicação do item.

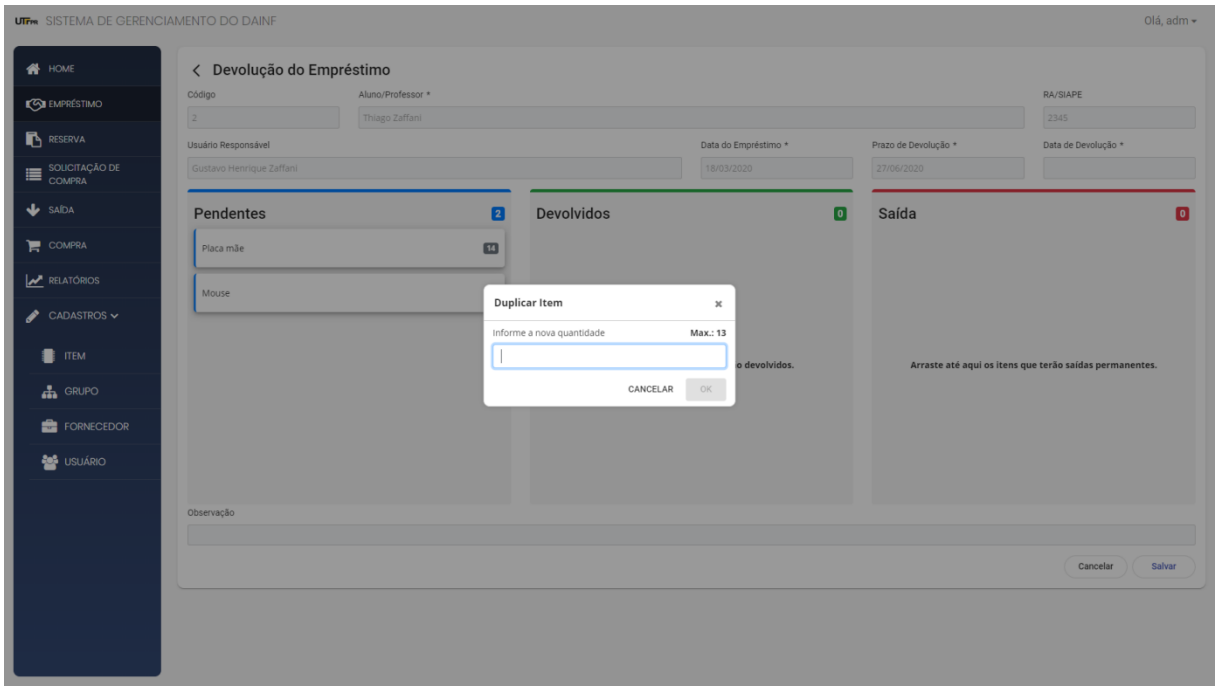
Além de dividir, é possível remover os itens que foram duplicados, para isso basta clicar na opção “Remover Itens Duplicados”. As opções serão habilitadas de acordo com o procedimento atual, portanto só pode ser itens duplicados quando houver.

Figura 17 - Tela de devolução de empréstimo



Fonte: Autoria própria.

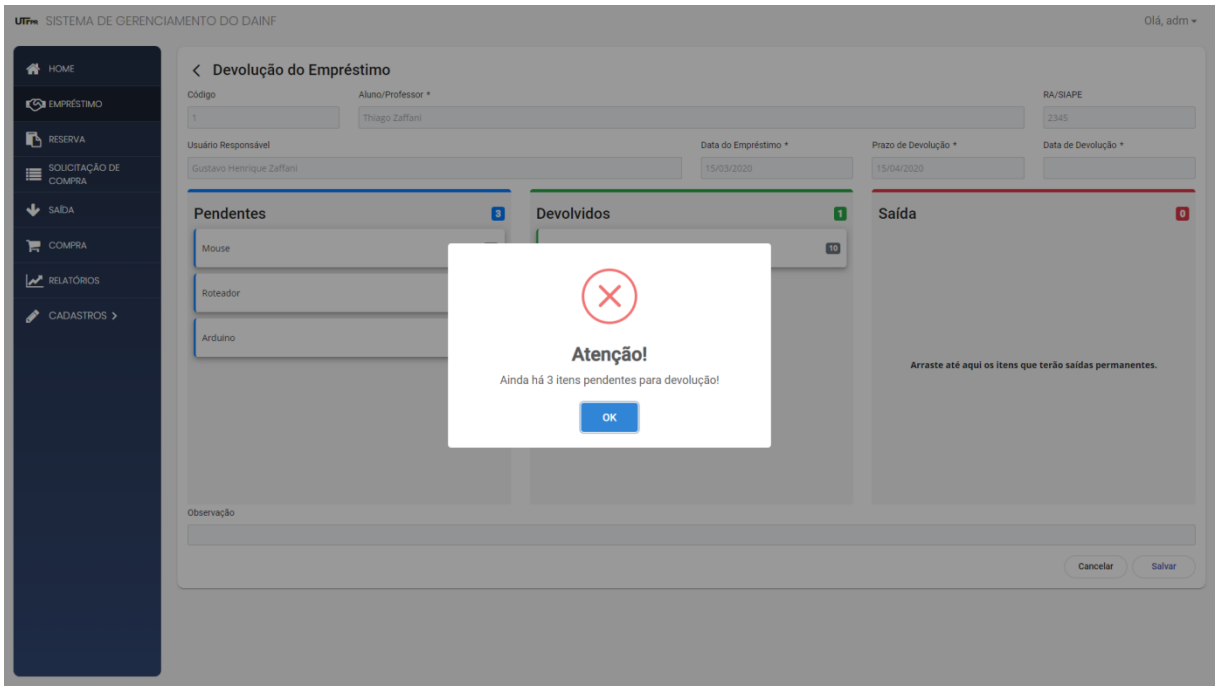
Figura 18 - Modal de duplicação do item da devolução



Fonte: Autoria própria.

Caso o usuário tente salvar a devolução sem ter realizado a devida devolução de todos os itens, será mostrado uma mensagem na tela, informado que ainda há itens pendentes para devolução, conforme mostra a Figura 19.

Figura 19 - Mensagem de erro ao tentar finalizar uma devolução com itens pendentes



Fonte: Autoria própria.

Ao efetuar de devolução de todos os itens pendentes e clicar em salvar, o empréstimo será finalizado e não será possível editá-lo novamente.

Como forma de confirmação de novos registros, é encaminhado um *e-mail* em algumas rotinas, sendo elas:

- Ao realizar o cadastro de um novo empréstimo, é encaminhado um *e-mail* de confirmação para o usuário que foi realizado o empréstimo. Nesse *e-mail* é possível verificar a data da realização do empréstimo, o prazo de devolução e os itens que foram emprestados.
- Ao realizar a devolução do empréstimo, informando todos os materiais que foram devolvidos.
- O usuário que realizar uma reserva de materiais receberá um *e-mail* de confirmação, e quando essa reserva é finalizada, ou seja, é transformada em empréstimo, também é encaminhado um *e-mail* de finalização da reserva.
- Três dias antes do vencimento do empréstimo, o usuário também receberá um *e-mail* de aviso de que a data da devolução do seu empréstimo está chegando.

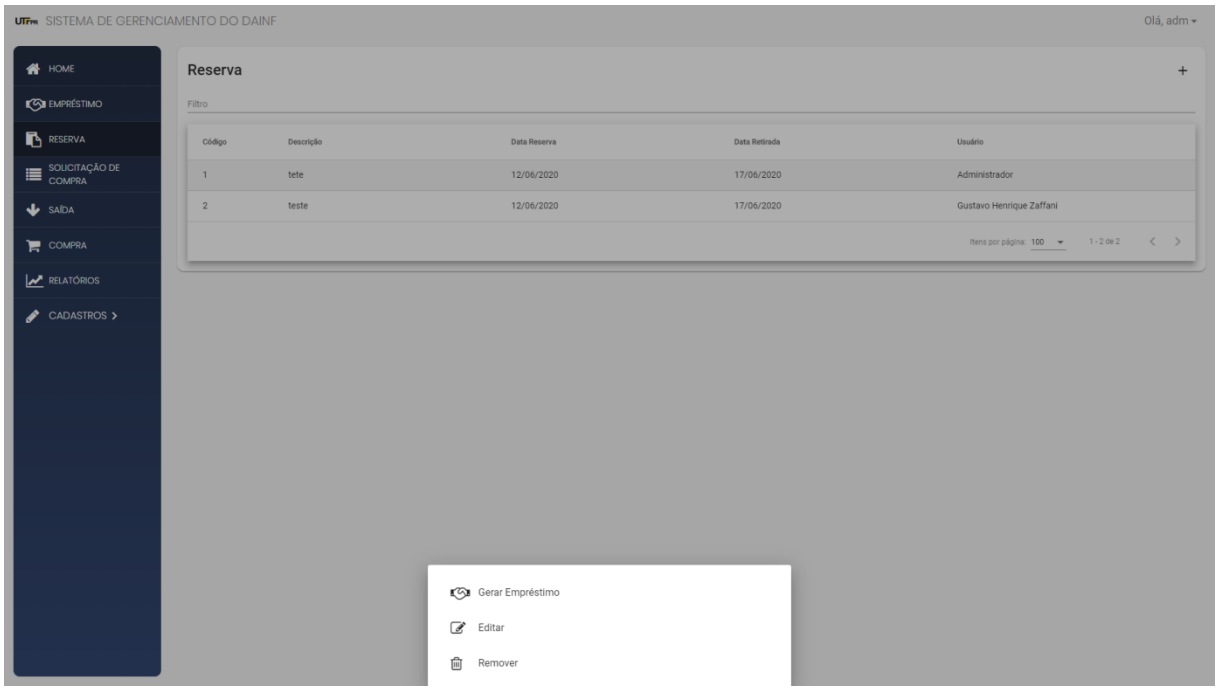
Todos estes *e-mails* encaminhados, seguem o padrão de leiaute do *e-mail* mostrado na Figura 20.

Figura 20 - Padrão de *e-mail* encaminhado pelo sistema em diferentes rotinas

Fonte: Autoria própria.

Na tela de reserva de materiais, o usuário laboratorista ou administrador, conseguirá visualizar a reserva de todos os usuários. E poderá transformar essa reserva em um empréstimo. Para isso, basta clicar sobre a reserva, e será mostrado as opções “Gerar Empréstimo”, “Editar” e “Remover”. As opções “Editar” e “Remover” são iguais as demais rotinas. Entretanto, a opção “Gerar Empréstimo”, irá criar um empréstimo com todos os dados da reserva. E caso o empréstimo for concluído com sucesso, essa reserva é removida do sistema, ficando assim apenas o registro empréstimo.

Figura 21 - Tela de reserva de materiais



Fonte: Autoria própria.

4.4 IMPLEMENTAÇÃO DO SISTEMA

Para iniciar o desenvolvimento do sistema, foi definida uma estrutura de pacotes para manter o projeto organizado e conciso. O sistema foi dividido em dois projetos, o primeiro é uma API REST que contém o *back-end* da aplicação e o segundo projeto contém a camada visual, ou seja, o *front-end* da aplicação.

4.4.1 Projeto *back-end*

Na Figura 22 é possível visualizar a estrutura utilizada durante o desenvolvimento do projeto, em que foi utilizado a linguagem de programação Java com o Spring Framework.

As principais pastas apresentadas são:

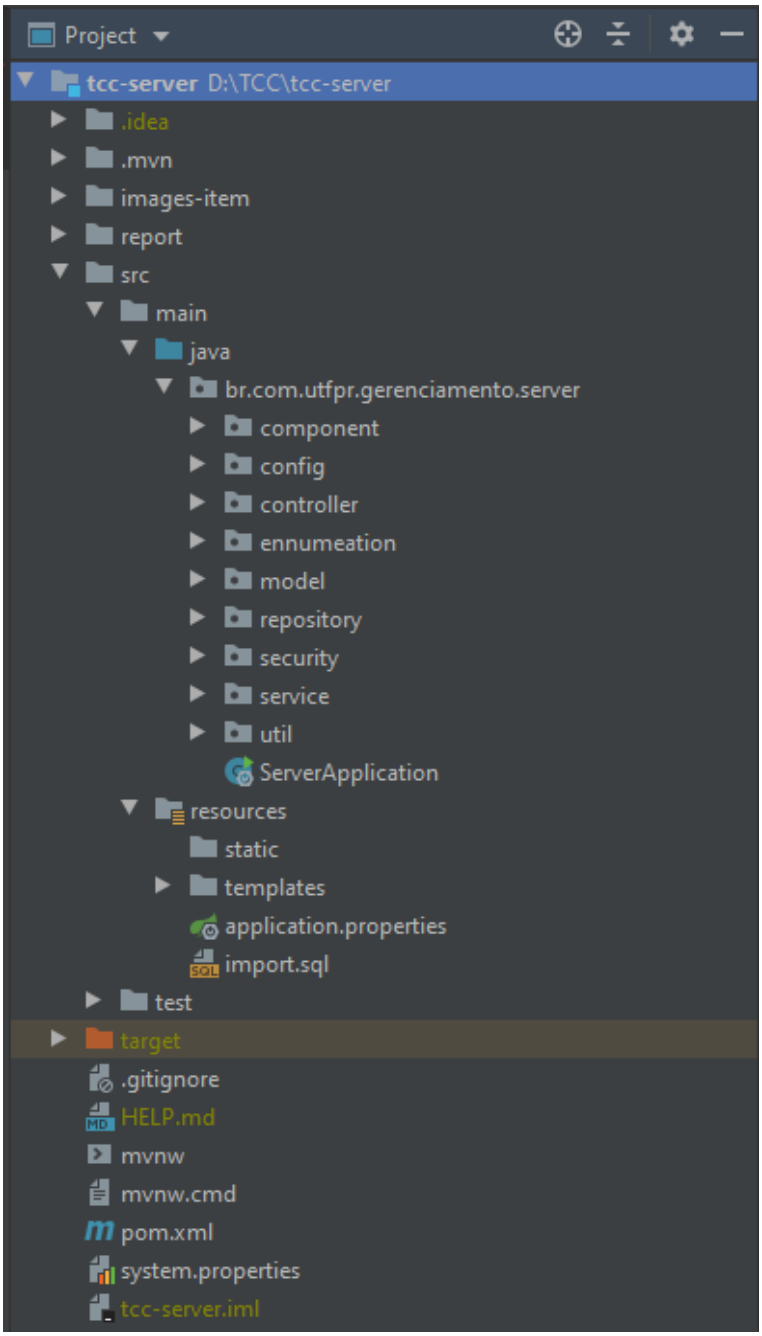
a) *Model*: contém todas as classes de persistência do sistema.

b) *Repository*: possui todas as classes que estendem do Spring JPA, para realizar a persistência de dados no banco.

c) *Controller*: possui o *controller* de cada classe do sistema, que é o responsável pela comunicação com o *front-end*.

d) *Service*: contém todas as classes de comunicação entre o *controller* e o *repository*, como por exemplo os métodos de salvar, editar, remover.

Figura 22 - Estrutura do projeto *back-end*



Fonte: Autoria própria.

Na Listagem 1 é apresentada a classe abstrata utilizada por todos os *controllers*. Nele contém os principais métodos utilizados para a manipulação de dados no sistema, são eles:

a) findAll(): responsável por buscar todos os registros de uma determinada tabela.

b) save(): é o responsável por inserir ou atualizar um registro.

c) findOne(): responsável por buscar um determinado registro do banco de dados

d) delete(): responsável por remover um registro.

A utilização dessa classe visa a reutilização de código e boas práticas de programação, visto que todos os formulários terão métodos semelhantes.

Listagem 1 - Classe abstrata utilizada pelos *controllers* do projeto

```java
package br.com.utfpr.gerenciamento.server.controller;

import br.com.utfpr.gerenciamento.server.service.CrudService;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Sort;
import org.springframework.web.bind.annotation.*;

import java.io.Serializable;
import java.util.List;

public abstract class CrudController<T, ID extends Serializable> {

    protected abstract CrudService<T, ID> getService();

    @GetMapping
    public List<T> findAll() { return getService().findAll(Sort.by("id")); }

    @PostMapping
    public T save(@RequestBody T object) {
        preSave(object);
        T toReturn = getService().save(object);
        postSave(object);
        return toReturn;
    }

    public void preSave(T object) {
    }

    public void postSave(T object) {
    }

    @GetMapping("{id}")
    public T findone(@PathVariable("id") ID id) {
        return getService().findOne(id);
    }

    @DeleteMapping("{id}")
    public void delete(@PathVariable("id") ID id) {
        T object = getService().findOne(id);
        getService().delete(id);
        postDelete(object);
    }

    public void postDelete(T object) {
    }

    @GetMapping("exists/{id}")
    public boolean exists(@PathVariable ID id) { return getService().exists(id); }

    @GetMapping("count")
    public long count() { return getService().count(); }
```

```
    @GetMapping("page")
    public Page<T> findAllPaged(@RequestParam("page") int page,
                                @RequestParam("size") int size,
                                @RequestParam(required = false) String order,
                                @RequestParam(required = false) Boolean asc) {
        PageRequest pageRequest = PageRequest.of(page, size);
        if (order != null && asc != null) {
            pageRequest = PageRequest.of(page, size, asc ? Sort.Direction.ASC : Sort.Direction.DESC, order);
        }
        return getService().findAll(pageRequest);
    }
}
```

Fonte: Autoria própria.

A Listagem 2 apresenta os métodos utilizados do projeto no *back-end*, em que é realizado o aumento ou diminuição do saldo do item. Esses métodos são chamados ao realizar um empréstimo, uma saída ou uma compra.

Ao cadastrar um novo empréstimo, por exemplo, os itens terão os seus saldos diminuídos. E, ao realizar a sua devolução, a quantidade emprestada será retornada ao saldo do item.

Listagem 2 - Métodos de diminuir e aumentar o saldo do item

```
    @Override
    public void diminuiSaldoItem(Long idItem, BigDecimal qtde, boolean needValidationSaldo) {
        Item itemToSave = itemRepository.findById(idItem).get();
        if (!needValidationSaldo || this.saldoItemIsValid(itemToSave.getSaldo(), qtde)) {
            itemToSave.setSaldo(itemToSave.getSaldo().subtract(qtde));
            itemRepository.save(itemToSave);
        }
    }

    @Override
    public void aumentaSaldoItem(Long idItem, BigDecimal qtde) {
        Item itemToSave = itemRepository.findById(idItem).get();
        itemToSave.setSaldo(itemToSave.getSaldo().add(qtde));
        itemRepository.save(itemToSave);
    }
```

Fonte: Autoria própria.

Na Listagem 3 é apresentado o método do servidor responsável pelo envio de *e-mail*. O envio de *e-mail* é utilizado em algumas rotinas do sistema, como, por exemplo, no cadastro de um novo empréstimo, ao realizar a devolução de um empréstimo, ao cadastrar uma nova reserva de materiais entre outras rotinas.

Para evitar o travamento no sistema, o envio de *e-mail* é realizado por meio de uma nova *Thread*. Dessa maneira, o usuário pode continuar utilizando o sistema, enquanto o servidor faz o encaminhamento do *e-mail*.

Listagem 3 - Método do servidor responsável pelo envio de *e-mail*

```
@Override
public void enviar(Email email) throws Exception {
    new Thread(() -> {
        try {
            MimeMessage message = javaMailSender.createMimeMessage();

            MimeMessageHelper helper = new MimeMessageHelper(message, multipart: true, encoding: "UTF-8");

            helper.setFrom(email.getDe(), personal: "Laboratório de Informática - UTFPR/PB");
            helper.setReplyTo(email.getDe());

            if (email.getPara() != null && !email.getPara().equals("")) {
                helper.setBcc(email.getPara());
            } else if (email.getParaList() != null && email.getParaList().size() > 0) {
                helper.setBcc(email.getParaList().toArray(new String[0]));
            } else {
                throw new Exception("Nenhum email encontrado para envio.");
            }

            helper.setSubject(email.getTitulo());
            helper.setText(email.getConteudo(), html: true);

            for (Map.Entry<String, byte[]> entry : email.getFileMap().entrySet()) {
                helper.addAttachment(entry.getKey(), new ByteArrayResource(entry.getValue()));
            }

            javaMailSender.setUsername("dainff.labs@gmail.com");
            javaMailSender.setPassword("password");
            javaMailSender.send(message);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }).start();
}
```

Fonte: Autoria própria.

Na Listagem 4 é apresentado o método de segurança responsável por definir as URLs para cada perfil de acesso. Nela, é possível observar que várias funcionalidades do sistema são acessadas apenas por usuários que possuem o perfil de acesso “Laboratorista” ou “Administrador”. As funcionalidades cujas URLs não estão elencadas na Listagem 4, não possuem um perfil de acesso específico, ou seja, todos os usuários terão acesso, desde que estejam autenticados. Por exemplo, as rotinas de reserva de estoque e solicitação de compra.

Listagem 4 - Método responsável pela definição do acesso das URLs

```
@Override
protected void configure(HttpSecurity http) throws Exception {
    http.csrf().disable().authorizeRequests() ExpressionUrlAuthorizationConfigurer<H>.ExpressionInterceptUrlRegistry
            .antMatchers( ...antPatterns: "/cidade/**",
                    "/estado/**",
                    "/pais/**",
                    "/relatorio/**",
                    "/fornecedor/**",
                    "/compra/**",
                    "/entrada/**",
                    "/grupo/**",
                    "/saida/**").hasAnyRole( ...roles: "LABORATORISTA", "ADMINISTRADOR")
            .antMatchers(HttpMethod.POST,  ...antPatterns: "/item/**").hasAnyRole( ...roles: "LABORATORISTA", "ADMINISTRADOR")
            .antMatchers(HttpMethod.DELETE,  ...antPatterns: "/item/**").hasAnyRole( ...roles: "LABORATORISTA", "ADMINISTRADOR")
            .antMatchers(HttpMethod.POST,  ...antPatterns: "/usuario/**").hasRole("ADMINISTRADOR")
            .antMatchers(HttpMethod.DELETE,  ...antPatterns: "/usuario/**").hasRole("ADMINISTRADOR")
            .antMatchers(HttpMethod.POST,  ...antPatterns: "/emprestimo/**").hasAnyRole( ...roles: "LABORATORISTA", "ADMINISTRADOR")
            .antMatchers(HttpMethod.DELETE,  ...antPatterns: "/emprestimo/**").hasAnyRole( ...roles: "LABORATORISTA", "ADMINISTRADOR")
            .antMatchers( ...antPatterns: "/usuario/user-info").permitAll()
            .anyRequest().authenticated()
            .and() HttpSecurity
            .addFilter(new JWTAuthenticationFilter(authenticationManager()))
            .addFilter(new JWTAuthorizationFilter(authenticationManager()))
            .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS);
}
```

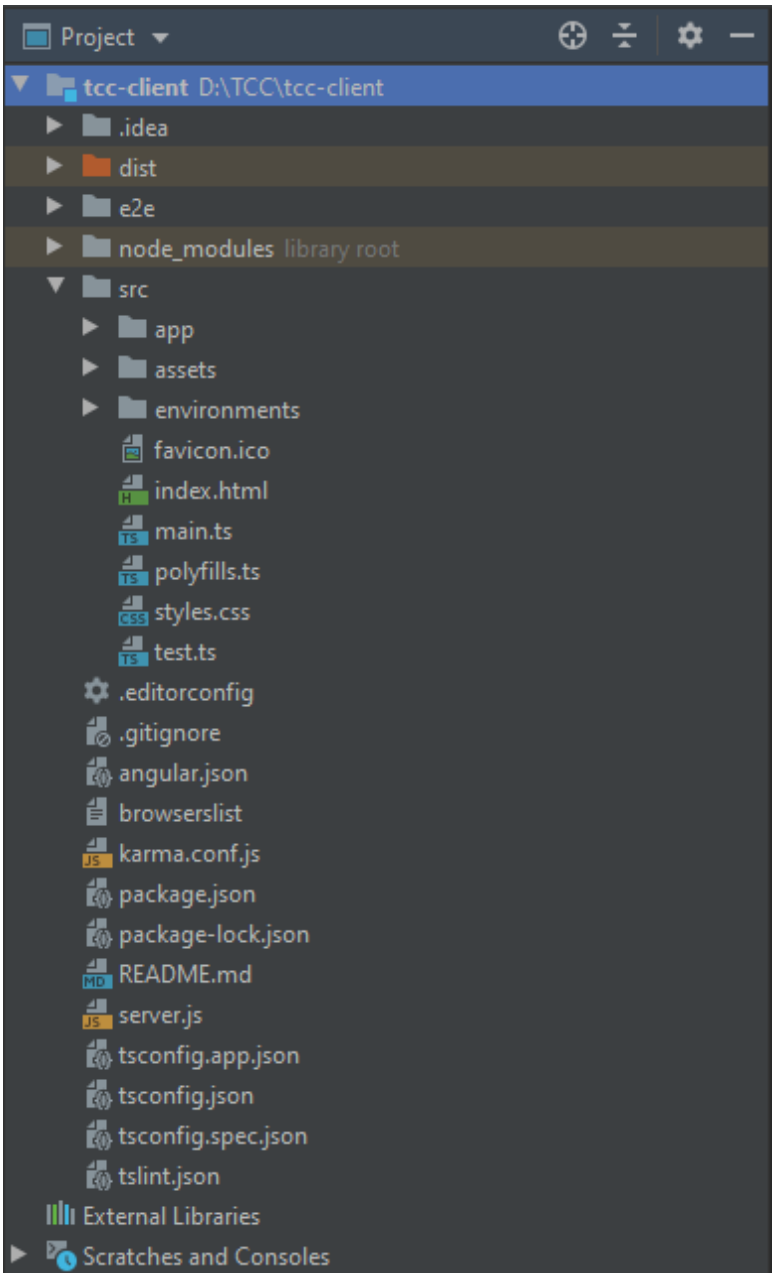
Fonte: Autoria própria.

4.4.2 Projeto *front-end*

Na Figura 23 é apresentado a estrutura que segue o padrão do Angular CLI, ao criar um projeto.

Na pasta *app* ficam todos os arquivos responsáveis pelas funcionalidades do sistema, sendo eles: *typescript*, *html* e *css*. Na pasta *assets* é ficam armazenadas todas as imagens utilizadas no sistema. E, na pasta *environments* ficam os arquivos de configurações da API, como por exemplo, o endereço IP e porta que está sendo executado o servidor. Os demais arquivos são os arquivos padrões do Angular, que são responsáveis por toda a configuração do projeto.

Figura 23 - Estrutura do projeto *front-end*



Fonte: Autoria própria.

Na Listagem 5 é apresentado a classe abstrata de comunicação com o servidor. Ela é utilizada por todas as classes de serviço do projeto. A mesma possui os métodos semelhantes entre todas as classes que fazem CRUD. A utilização dessa classe evita possíveis divergências na URL da requisição.

Listagem 5 - Classe abstrata utilizada pelos *services* no projeto *front-end*

```
import {HttpClient} from '@angular/common/http';
import {Observable} from 'rxjs';

export abstract class CrudService<T, ID> {

  constructor(protected url: string, protected http: HttpClient) {
  }

  protected getUrl(): string {
    return this.url;
  }

  findAll(): Observable<T[]> {
    return this.http.get<T[]>(this.getUrl());
  }

  findAllByUsername(username: string): Observable<T[]> {
    return this.http.get<T[]>( url: this.getUrl() + `find-all-by-username/${username}`);
  }

  findOne(id: ID): Observable<T> {
    return this.http.get<T>( url: this.getUrl() + id);
  }

  save(t: T): Observable<T> {
    return this.http.post<T>(this.getUrl(), t);
  }

  delete(id: ID): Observable<void> {
    return this.http.delete<void>( url: `${this.url + id}`);
  }

  complete(query: string): Observable<T[]> {
    return this.http.get<T[]>( url: `${this.url}complete?query=${query}`);
  }
}
```

Fonte: Autoria própria.

A Listagem 6 apresenta a classe abstrata utilizada pelos componentes de lista da aplicação. Todas as rotinas do sistema que possuem uma listagem de registros, estendem essa classe.

Ela possui diversas *functions* básicas importantes na criação da lista, como:

a) findAll(): responsável por buscar os registros do banco, e montar a tabela com as informações;

b) delete(): responsável por remover um determinado registro e atualizar a tabela;

c) edit(): direciona o usuário para o formulário de edição.

d) findAllByUsername(): realiza uma busca de todos os registros que estão vinculados a um usuário. Essa *function* é utilizada nas rotinas de empréstimo, solicitação de compra

e reserva de materiais, quando o usuário autenticado possui o perfil de acesso “Professor” ou “Aluno”, pois estes só poderão visualizar os próprios registros.

e) buildColumnsTable(): essa *function* utiliza um *HostListener* para capturar o tamanho da tela em tempo real, conforme a mesma é redimensionada. Com isso, caso a largura total da tela for menor ou igual a 1200 pixels, os botões de editar e excluir serão removidos da lista, e um novo componente será inicializado para suprir as funcionalidades de edição e exclusão.

Ainda na Listagem 6 é possível observar que no método construtor é realizada a injeção de dependência do *service* que será responsável pela comunicação do projeto *front-end* com o projeto do *back-end*. Além deste, é realizado outras injeções de dependências em *services* que são utilizados em todas as páginas, como por exemplo: o *messageService*, *loaderService*, *loginService* e outros.

Listagem 6 - Classe abstrata responsável pelo *component* de lista das rotinas do sistema

```
import {HostListener, Injector, OnInit, ViewChild} from '@angular/core';
import {Router} from '@angular/router';
import {CrudService} from '../service/crud.service';
import {ConfirmationService, MessageService} from 'primeng';
import {MatTableDataSource} from '@angular/material/table';
import {MatPaginator} from '@angular/material/paginator';
import {MatSort} from '@angular/material/sort';
import {BottomSheetComponent} from '../../geral/bottomScheet/bottomSheet.component';
import {MatBottomSheet} from '@angular/material/bottom-sheet';
import Swal from 'sweetalert2';
import {Exception} from '../../exception/exception';
import {LoaderService} from '../loader/loader.service';
import {LoginService} from '../../login/login.service';

export abstract class CrudListComponent<T, ID> implements OnInit {

  protected router: Router;
  protected messageService: MessageService;
  protected confirmationService: ConfirmationService;
  protected bottom: MatBottomSheet;
  protected loaderService: LoaderService;
  protected loginService: LoginService;
  public displayedColumns: string[] = this.columnsTable;
  public dataSource: MatTableDataSource<T>;
  public bottomSheetEnabled = true;
  public hostListenerColumnEnable = true;
  public isAlunoOrProfessor = false;
  @ViewChild(MatPaginator, {static: true}) paginator: MatPaginator;
  @ViewChild(MatSort, {static: true}) sort: MatSort;
  objects: T[];

  ngOnInit(): void {
    this.loginService.userLoggedIsAlunoOrProfessor()
      .then(value => this.isAlunoOrProfessor = value);
    this.findAll();
  }

  constructor(protected service: CrudService<T, ID>,
              protected injector: Injector,
              protected columnsTable: string[],
              protected urlForm: string) {
    this.router = this.injector.get(Router);
    this.messageService = this.injector.get(MessageService);
    this.confirmationService = this.injector.get(ConfirmationService);
    this.bottom = injector.get(MatBottomSheet);
    this.loaderService = injector.get(LoaderService);
    this.loginService = injector.get(LoginService);
  }
```

```
applyFilter(filterValue: string) {
  this.dataSource.filter = filterValue.trim().toLowerCase();
  if (this.dataSource.paginator) {
    this.dataSource.paginator.firstPage();
  }
}

findAllCustom(): void {
}

findAll() {
  this.loaderService.display( display: true);
  this.service.findAll()
    .subscribe( next: e => {
      this.objects = e;
      this.buildList();
      this.loaderService.display( display: false);
      this.postFindAll();
    }, error: error => {
      this.loaderService.display( display: false);
    });
  this.buildColumnsTable();
}

findAllByUsername() {
  this.loaderService.display( display: true);
  const u = localStorage.getItem( key: 'username');
  this.service.findAllByUsername(u)
    .subscribe( next: e => {
      this.objects = e;
      this.buildList();
      this.loaderService.display( display: false);
      this.postFindAll();
    }, error: error => {
      this.loaderService.display( display: false);
    });
}

buildList() {
  if (this.objects != null) {
    this.dataSource = new MatTableDataSource(this.objects);
    this.dataSource.paginator = this.paginator;
    this.dataSource.sort = this.sort;
  }
}

edit(id: number) {
  this.router.navigate( commands: [this.urlForm, id]);
}
```

```
  postFindAll(): void {
  }

  delete(id: any) {
    Swal.fire( options: {
      title: `Tem certeza que deseja remover o registro?`,
      text: 'A ação não poderá ser desfeita.',
      icon: 'warning',
      showCancelButton: true,
      confirmButtonColor: '#3085d6',
      cancelButtonColor: '#d33',
      confirmButtonText: 'Sim',
      cancelButtonText: 'Não'
    }).then((result : SweetAlertResult ) => {
      if (result.value) {
        this.loaderService.display( display: true);
        this.service.delete(id)
          .subscribe( next: e => {
            Swal.fire( title: 'Sucesso!',  html: 'Registro excluído com sucesso!',  icon: 'success');
            this.findAll();
            this.loaderService.display( display: false);
          },  error: error => {
            this.loaderService.display( display: false);
            this.showError(error);
          });
      }
    });
  }

  openBottomSheet(id): void {
    if (window.innerWidth <= 1200 && this.bottomSheetEnabled) {
      const sheet = this.bottom.open(BottomSheetComponent);
      sheet.afterDismissed().subscribe( next: action => {
        if (action === 'E') {
          this.edit(id);
        } else if (action === 'R') {
          this.delete(id);
        }
      });
    }
  }

  openForm() {
    this.router.navigate( commands: [this.urlForm]);
  }
```

```
  @HostListener('window:resize', ['$event'])
  buildColumnsTable() {
    if (this.hostListenerColumnEnable) {
      if (window.innerWidth <= 1200) {
        this.columnsTable.forEach((value : string , index : number ) => {
          if (value === 'actions') {
            this.columnsTable.splice(index,  deleteCount: 1);
          }
        });
      } else if (this.columnsTable.filter(value => value === 'actions').length === 0) {
        this.columnsTable.push('actions');
      }
    }
  }

  showError(error: any): void {
    Exception.addMessage(error);
  }
}
```

Fonte: Autoria própria.

Para efetuar as operações de CRUD de maneira genérica e visando o reaproveitamento de código, foi criado a classe abstrata CrudFormComponent, conforme mostrado na Listagem 7.

Ela é responsável por várias *functions* utilizadas nos formulários do sistema, como por exemplo o edit() e o save(). Ao iniciar um formulário que estende esta classe, será verificado se na URL contém um ID como parâmetro. Caso contenha o ID, o método findOne() será chamado, para que assim seja buscado todos os dados do devido registro, caso contrário será iniciado um formulário sem informações.

Listagem 7 - Classe abstrata responsável pelos componentes de formulário do projeto

```
import {ActivatedRoute, Router} from '@angular/router';
import {CrudService} from '../service/crud.service';
import {Injector, OnInit} from '@angular/core';
import {MessageService} from 'primeng';
import {BaseFormComponent} from './base.form.component';
import Swal from 'sweetalert2';
import {LoaderService} from '../loader/loader.service';
import {LoginService} from '../../login/login.service';

export abstract class CrudFormComponent<T, ID> extends BaseFormComponent implements OnInit {

  protected router: Router;
  protected messageService: MessageService;
  protected route: ActivatedRoute;
  protected loaderService: LoaderService;
  protected loginService: LoginService;
  // utilizado para validações extras
  public validExtra = true;
  public editando = false;
  public isAlunosOrProfessor = false;
  public object: T;

  constructor(protected service: CrudService<T, ID>,
              protected injector: Injector,
              protected urlList: string,
              private type?: new () => T) {
    super();
    this.router = this.injector.get(Router);
    this.route = this.injector.get(ActivatedRoute);
    this.messageService = this.injector.get(MessageService);
    this.loaderService = this.injector.get(LoaderService);
    this.loginService = this.injector.get(LoginService);
  }

  ngOnInit(): void {
    this.loginService.userLoggedIsAlunoOrProfessor()
      .then(value => this.isAlunosOrProfessor = value);
    this.newInstance();
    this.preOnInit();
    this.route.params.subscribe( next: params => {
      if (params.id) {
        if (isNaN(params.id)) {
          this.initializeValues();
        } else {
          this.edit(params.id);
        }
      } else {
        this.initializeValues();
      }
    });
  }
```

```
save() {
  this.loaderService.display( display: true);
  if (this.isValid() && this.validExtra) {
    this.service.save(this.object)
      .subscribe( next: e => {
        this.object = e;
        this.postSave( callback: value => {
          this.loaderService.display( display: false);
          Swal.fire( title: 'Sucesso!',  html: 'Registro salvo com sucesso!',  icon: 'success');
          this.back();
        });
      },  error: error => {
        this.loaderService.display( display: false);
        Swal.fire( title: 'Atenção!',  html: 'Ocorreu um erro ao salvar o registro!',  icon: 'error');
        console.log(error);
      });
  } else {
    this.loaderService.display( display: false);
    this.messageService.add({severity: 'info', summary: 'Atenção', detail: 'Necessário preencher todos os campos corretamente!'});
    this.validarFormulario();
  }
}

postSave(callback: Function): void {
  callback();
}

initializeValues(): void {
}

preOnInit(): void {
}

postEdit(): void {
}

edit(id: ID) {
  this.loaderService.display( display: true);
  this.service.findOne(id)
    .subscribe( next: e => {
      this.object = e;
      this.editando = true;
      this.postEdit();
      this.loaderService.display( display: false);
    },  error: error => {
      this.loaderService.display( display: false);
      Swal.fire( title: 'Atenção!',  html: 'Ocorreu um erro ao buscar o registro!',  icon: 'error');
    });
}
```

```
 back() {
   this.router.navigate( commands: [this.urlList]);
 }

 protected newInstance(): void {
   if (this.type) {
     this.object = new this.type();
   } else {
     this.object = {} as T;
   }
 }
}
```

Fonte: Autoria própria.

A Listagem 8 apresenta o método responsável por transformar uma reserva de materiais em empréstimo. Quando um usuário realiza uma reserva de materiais no sistema, essa reserva fica pendente até que um responsável possa transformá-la em empréstimo. Quando o usuário clica na opção “Gerar Empréstimo”, as informações da reserva são armazenadas no *localStorage*, e o usuário é redirecionado para o formulário de empréstimo. Ao abrir o formulário, o sistema verificará que o usuário quer gerar um empréstimo utilizando uma reserva, e chamará a *function* generateEmprestimoByReserva().

Listagem 8 - Método responsável por gerar empréstimo por meio da reserva de materiais

```
generateEmprestimoByReserva() {
  let reserva = JSON.parse(localStorage.getItem( key: 'reserva-to-emprestimo'));
  this.idReserva = reserva.id;
  this.object.usuarioEmprestimo = reserva.usuario;
  this.object.observacao = reserva.observacao;
  this.documentoUsuario = reserva.usuario.documento;
  this.object.emprestimoItem = new Array();
  reserva.reservaItem.forEach(reserva => {
    let emprestimoItem = new EmprestimoItem();
    emprestimoItem.item = reserva.item;
    emprestimoItem.qtde = reserva.qtde;
    this.object.emprestimoItem.push(emprestimoItem);
  });
  localStorage.removeItem( key: 'reserva-to-emprestimo');
}
```

Fonte: Autoria própria.

Quando o usuário adiciona um novo item no empréstimo, é chamado o método insertItem(), conforme apresentado na Listagem 9. Esse método irá verificar se foi informado corretamente o item e a quantidade que será adicionada. Ao fazer essa verificação, será validado se a quantidade informada é válida para o saldo atual do item, utilizando um outro método, conforme mostrado na Listagem 10. Após realizar ambas verificações, o item será adicionado no empréstimo.

Nas rotinas de compra, saída, reserva e solicitação de compra, é utilizado um método parecido a esse para adicionar os itens, diferenciando apenas algumas regras de negócio vigentes da rotina.

Listagem 9 - Método responsável pela inserção de itens no empréstimo

```
insertItem() {
  if (this.emprestimoItem.item && this.emprestimoItem.qtde && typeof this.emprestimoItem.item === 'object') {
    if (this.saldoItemIsValid(this.emprestimoItem.qtde)) {
      if (!this.object.emprestimoItem) {
        this.object.emprestimoItem = new Array();
      }
      const upQtde = this.object.emprestimoItem.some(value => value.item.id === this.emprestimoItem.item.id);
      if (upQtde) {
        this.object.emprestimoItem.forEach(empItem => {
          if (empItem.item.id === this.emprestimoItem.item.id) {
            const novaQtde = Number(empItem.qtde) + Number(this.emprestimoItem.qtde);
            if (this.saldoItemIsValid(novaQtde)) {
              empItem.qtde = novaQtde;
            }
          }
        });
      } else {
        this.object.emprestimoItem.push(this.emprestimoItem);
      }
      this.postInsertItemList();
    }
  } else {
    this.messageService.add({severity: 'info', detail: 'Necessário informar o item e a quantidade.'});
  }
}
```

Fonte: Autoria própria.

Listagem 10 - Método responsável por validar o saldo atual do item

```
saldoItemIsValid(qtdeInserir) {
  const isValid = this.emprestimoItem.item.saldo > 0 && qtdeInserir <= this.emprestimoItem.item.saldo;
  if (!isValid) {
    this.messageService.add({severity: 'info', detail: 'A quantidade é maior do que o saldo atual do item.'});
    return false;
  }
  return true;
}
```

Fonte: Autoria própria.

Quando um usuário realiza a devolução de empréstimo e clica no botão “Salvar”, será chamado o método saveDevolucao(), conforme mostrado na Listagem 11. Esse método irá verificar se todos os itens pendentes foram arrastados para os seus devidos quadros, e, caso não ainda exista algum item pendente, será disparado uma mensagem de alerta para o usuário. Após fazer essa validação, será atualizado o campo statusDevolucao de cada item do empréstimo, para vincular se o item foi devolvido ou será vinculado à uma saída.

Listagem 11 - Function responsável por salvar a devolução de empréstimo

```
saveDevolucao() {
  if (this.itensPendentes.length === 0) {
    this.loaderService.display( display: true);
    this.object.emprestimoDevolucaoItem.forEach(empDevItem => {
      this.itensPendentes.forEach(pendente => {
        if (empDevItem.id === pendente.id) {
          empDevItem.statusDevolucao = StatusDevolucao.P;
        }
      });
      this.itensDevolvidos.forEach(devolvido => {
        if (empDevItem.id === devolvido.id) {
          empDevItem.statusDevolucao = StatusDevolucao.D;
        }
      });
      this.itensSaida.forEach(saida => {
        if (empDevItem.id === saida.id) {
          empDevItem.statusDevolucao = StatusDevolucao.S;
        }
      });
    });
    this.emprestimoService.saveDevolucao(this.object)
      .subscribe( next: e => {
        this.loaderService.display( display: false);
        Swal.fire( title: 'Sucesso!',  html: 'Devolução efetuada com sucesso!',  icon: 'success');
        this.back();
      },  error: error => {
        this.loaderService.display( display: false);
        Swal.fire( title: 'Atenção!',  html: 'Ocorreu um erro ao salvar a devolução!',  icon: 'error');
      });
  } else {
    Swal.fire( title: 'Atenção!',  html: 'Ainda há ' + this.itensPendentes.length + ' itens pendentes para devolução!',  icon: 'error');
  }
}
```

Fonte: Autoria própria.

4.5 IMPLANTAÇÃO DO SISTEMA

Os projetos do sistema estão disponíveis publicamente no github, por meio dos links: https://github.com/GustavoZaffani/tcc-server e https://github.com/GustavoZaffani/tcc-client.

O projeto “tcc-server” possui o código fonte do servidor do sistema, onde contém todas as classes de persistência do banco e as regras de negócio. Para implantar o *back-end*, pode-se utilizar um servidor web, como por exemplo: Apache Tomcat ou Wildfly. A máquina que será responsável por executar o servidor, precisará ter instalado o Kit de Desenvolvimento Java (JDK) na versão 11 e o banco de dados PostgreSQL na versão 11. Será necessário configurar o usuário e senha do banco de dados no arquivo application.properties, que está localizado na pasta /resources/ da aplicação. Após esse processo basta utilizar o Ambiente de Desenvolvimento Integrado (IDE) de desenvolvimento para gerar o arquivo .jar ou .war para implantar a aplicação no servidor.

Já o projeto “tcc-client” contém o código fonte de todas as telas do sistema, ou seja, o lado em que o usuário irá utilizar. Para implantar uma aplicação Angular é necessário executar o comando *ng build*, então os arquivos Typescript serão transpilados em javascrip em uma pasta chamada /dist/. O conteúdo dessa pasta pode ser copiado para implantar a aplicação em um servidor Nginx ou Apache.

5 CONCLUSÃO

Neste trabalho foi desenvolvido um sistema web para facilitar o gerenciamento dos materiais disponibilizados pelo laboratório de informática da UTFPR. O mesmo foi implementado utilizando as tecnologias Java para o *back-end* e Angular para o *front-end*.

Todos os objetivos propostos para o trabalho foram alcançados. Algumas melhorias e ideias surgiram no decorrer do desenvolvimento, viabilizando sempre a praticidade de gerenciamento dos materiais, tendo como base o *User Experience* (UX).

Pelo fato de já trabalhar com o Spring Boot, não houve grandes dificuldades no desenvolvimento do projeto no *back-end*. O Angular também se mostrou um *framework* produtivo para desenvolvimento do projeto no *front-end*. Durante o desenvolvimento das rotinas do sistema surgiram dúvidas relacionadas ao uso de alguns componentes, entretanto essas dúvidas foram sanadas por meio de pesquisas na documentação das bibliotecas e fóruns de discussão.

No *front-end* foram utilizadas várias bibliotecas para obter-se os resultados desejados, pois somente o *framework* Angular e os componentes do PrimeNG não conseguiram suprir todas as demandas necessárias. Além de priorizar interfaces de fácil utilização, o sistema foi desenvolvido telas com responsividade para diferentes dispositivos.

Como trabalhos futuros poderão ser desenvolvidas novas rotinas, relatórios e gráficos no sistema, de forma que possa agregar ainda mais o gerenciamento dos materiais do laboratório.

REFERÊNCIAS

ABRAS, Chadia; MALONEY-KRICHMAR, Diane; PREECE, Jenny. **User-Centered Design.** W. Encyclopedia Of Human-computer Interaction, 2004. Disponível em:

<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.94.381&rep=rep1&type=pdf> Acesso em: 17 nov. 2019.

BUCCINI, M. **Introdução ao design experiencial.** Recife: Edição do autor, 2008.

DAFT, R. L. **Administração**. 6. ed. São Paulo: Thomson, 2007.

HASSENZAHL, M. Encyclopedia entry on User Experience e Design Experience. In: **The interaction-design.org foundation.** 2013. Disponível em: <http://www.interactiondesign.org/printerfriendly/encyclopedia/user_experience_and_experience_design.html>. Acesso em: 17 nov. 2019.

MORVILLE, Peter; ROSENFELD, Louis. **Information Architecture: for the World Wide Web**. 3. ed. United States Of America: O'reilly Media, 2006. 528 p.

NIELSEN, Jakob. **Design Guidelines for Homepage Usability.** 2001**.** Disponível em: <https://www.nngroup.com/articles/113-design-guidelines-homepage-usability/>. Acesso em: 20 nov. 2019.

O'BRIEN, J. A. **Sistemas de informações e as decisões gerenciais na era da Internet**. 9. ed. São Paulo: Saraiva, 2006.

PADOVEZE, C. L. **Sistemas de informações contábeis**. São Paulo: Atlas, 2004.

RUBIN, Jeff; CHISNELL, Dana. **Handbook of Usability Testing**, Second Edition: How to Plan, Design, and Conduct Effective Tests. 2. ed. Indianapolis, Indiana: Wiley Publishing, Inc., 2008. 386 p.

SANTOS, Robson. **Usabilidade de interfaces e arquitetura de informação: alguns aspectos da organização de conteúdo para o meio digital.** In: Congresso Da Brasileiro de Ergonomia, XI; Congresso Latino Americano de Ergonomia, VI; Encontro África-Brasil de Ergonomia, III; Fórum Sul Brasileiro de Ergonomia. Gramado, 2001. 6p.

SCHMITT, Bernd. **Marketing experimental.** São Paulo: Nobel, 2000.

SILVA, Patrícia Maria da; DIAS, Guilherme Ataíde. **A arquitetura da informação centrada no usuário: estudo do website da biblioteca virtual em saúde (bvs).** Revista Eletronica Bibliotecon, Florianopolis, v. 2, n. 26, p.1-12, 2008. Disponível em: <https://periodicos.ufsc.br/index.php/eb/article/view/1518-2924.2008v13n26p119>. Acesso em: 18 nov. 2019.

TOMS, E.G; BLADES, R.L. **Information Architecture and web site design**. Feliciter, v.45, n.4, 1999.