Sistema para Gerenciamento de Empréstimos de Objetos

Camila Marques Vasconcelos Loureiro
Instituto Federal da Bahia
Rua Emídio dos Santos, S/N, Barbalho
Salvador, Brasil
E-mail:marquescami@gmail.com

Renato Novais
Departamento de Computação
Instituto Federal da Bahia
Salvador, Brasil
E-mail: renato@ifba.edu.br

***Resumo*—Muitas informações circulam a todo momento, sendo imprescindível o gerenciamento do fluxo de dados e processos que envolvem várias áreas. Nas instituições de ensino, a exemplo do Instituto Federal da Bahia (IFBA) ocorre o empréstimo de objetos, e comumente, não há um sistema informatizado para gerenciar essa atividade. Como solução, foi desenvolvido um sistema de gerenciamento de empréstimos de objetos (GEIFBA) para permitir o cadastro de ativos, a gestão dos empréstimos por parte dos administradores e dos usuários que realizarão os empréstimos. O objetivo da solução é a maior agilidade, transparência e controle sobre o processo, melhorando a experiência dos usuários e otimizando o uso dos recursos da universidade. Para mapeamento do uso da ferramenta, foi realizado um estudo preliminar de validação com alguns possíveis usuários para se ter uma percepção inicial sobre a visão da utilização do sistema.**

Keywords*—*aplicação, sistema de gerenciamento, GEIFBA, objetos, sistema, API, empréstimos, usuários, requisitos, solução.

***Abstract*—A lot of information circulates at all times, making it essential to manage the flow of data and processes that involve several areas. In educational institutions, such as the Federal Institute of Bahia (IFBA), objects are borrowed, and commonly, there is no computerized system to manage this activity. As a solution, an object loan management system (GEIFBA) was developed to allow the registration of assets, the management of loans by administrators and users who will carry out the loans. The objective of the solution is greater agility, transparency and control over the process, improving the user experience and optimizing the use of university resources. To map the use of the tool, a preliminary validation study was carried out with some potential users to have an initial perception of the vision of using the system.**

Keywords*—*application, management system, GEIFBA, objects, system, API, loans, users, requirements, solution.

I. INTRODUÇÃO

Com o avanço da tecnologia e o grande tráfego de informações em várias áreas, tornou-se cada vez mais necessária a gestão do fluxo de dados e processos. No âmbito das instituições de ensino, não é diferente, são usados softwares como facilitadores das atividades do dia a dia.

Este processo de informatização gera um grande volume de informações. Para que se tenha benefícios e conhecimento sobre esse processo, é necessária uma sistematização com o uso de ferramenta de gestão de processos, sistemas, pessoas, equipamentos, enfim, todo recurso disponível e que participe de alguma maneira da aquisição da informação deve ser gerenciado [1].

No Instituto Federal da Bahia (IFBA) além de livros emprestados na biblioteca, ocorre o empréstimo de objetos a exemplo de chaves de setores, projetores, entre outros. Segundo Pressman [2] "O software distribui o produto mais importante de nossa era - a informação". Atualmente não é feito o gerenciamento informatizado dessa atividade, diante disso surgiu a necessidade do desenvolvimento de uma solução para gestão possibilitando o cadastro dos ativos do instituto e gestão dos empréstimos realizados.

O GEIFBA é um sistema para gerenciamento de empréstimos de objetos, criado para solucionar esse problema. Seu intuito é permitir a gestão desses dados, de forma a auxiliar o acompanhamento desse processo para todos os envolvidos. Para mapeamento do uso da ferramenta, foi realizado um estudo preliminar de validação com alguns possíveis usuários. O objetivo foi o de ter uma percepção inicial sobre a visão dos usuários utilizando o sistema.

Sendo assim, o presente artigo está estruturado da seguinte forma, além da introdução, o trabalho tem a fundamentação teórica, no qual foi realizado um estudo sobre o Instituto Federal da Bahia e a forma de gestão de empréstimos atual, também foram apresentados conceitos de alguns termos técnicos que serão resgatados no decorrer do texto.

A seguir, apresenta-se os trabalhos relacionados ao GEIFBA. No quarto tópico, descreve-se todas as funcionalidades do sistema para gerenciamento de empréstimos de objetos (GEIFBA), detalhando cada uma delas, bem como foram apresentados os requisitos funcionais e não funcionais do sistema, os diagramas, tecnologias utilizadas, implantação do software, estudo preliminar de validação do GEIFBA e por fim a conclusão e trabalhos futuros.

II. FUNDAMENTAÇÃO TEÓRICA

A. O Instituto Federal da Bahia

O Instituto Federal da Bahia (IFBA) foi criado no dia 29 de dezembro de 2008, é uma instituição que tem como compromisso de oferecer educação profissional pública, gratuita e de excelência a comunidade. A organização tem como órgão executivo a Reitoria, instalada em Salvador, capital baiana, e caracteriza-se como instituição *multicampi*, constituída por 22 (vinte e dois) *campi* (Salvador, Barreiras, Brumado, Camaçari,

Eunápolis, Euclides da Cunha, Feira de Santana, Ilhéus, Irecê, Jacobina, Jequié, Juazeiro, Lauro de Freitas, Paulo Afonso, Porto Seguro, Santo Amaro, Santo Antônio de Jesus, Seabra, Simões Filho, Ubaitaba, Valença e Vitória da Conquista); 01 (um) Núcleo Avançado (Salinas da Margarida); 02 (dois) *campi* em fase de implantação, localizados em Jaguaquara e Campo Formoso; 05 (cinco) Centros de referência, também em construção, localizados nas cidades de Itatim, Casa Nova, São Desidério, Camacã e Monte Santo; e 01 (um) Polo de Inovação Salvador, cuja unidade fica no Parque Tecnológico da Bahia (Paralela, em Salvador/Ba). O IFBA está presente em 113 cidades da Bahia, a instituição atualmente possui mais de 36 mil estudantes (presenciais e a distância, cerca de 1.700 professores e 1.000 técnicos administrativos [3].

O campus Salvador possui recursos físicos que são compartilhados para o desempenho das atividades na unidade, a administração desses recursos tem grande relevância para que se possa manter o controle e haja um fluxo contínuo nesse processo. Avaliando o fluxo de empréstimos atual desse *campi* foi identificado que ele possui deficiências e riscos, entre os quais o controle do empréstimo manual. Dito isto, a informatização dessas tarefas trará para o campus Salvador possibilidade de armazenamento dos dados, bem como a utilização dos recursos referentes aos empréstimos, maior segurança, facilidade, além da possibilidade de expansão da solução disponibilizada para outros campus.

B. Engenharia de Software

Engenharia de *Software* é uma área da ciência da computação que se concentra na aplicação de princípios científicos, técnicas e ferramentas para o desenvolvimento de *software* de alta qualidade, confiabilidade e segurança. É uma abordagem voltada para o ciclo de vida do *software*, que inclui a análise de requisitos, o projeto, a implementação, o teste, a manutenção e a evolução do *software*.

A Engenharia de *Software* envolve várias atividades, incluindo gerenciamento de projetos, gerenciamento de configuração, garantia de qualidade, engenharia de requisitos, engenharia de *software* orientada a objetos, desenvolvimento ágil, entre outros. O foco principal da engenharia de requisitos é a criação de um *software* que atenda às necessidades dos usuários, com qualidade, dentro do prazo e orçamento previstos.

Trata-se de um tema bastante complexo e abrangente que teve grande evolução ao longo dos ano e influência de vários autores, a exemplo dos citados abaixo:

- Roger Pressman e Bruce Maxim [2]: Definem a engenharia de *software* como uma tecnologia em camadas, conforme figura 1 sendo o processo de engenharia de *software* uma liga que mantêm as camadas coesas possibilitando o desenvolvimento de *software* de maneira racional e dentro do prazo. Ainda de acordo com os autores "A pedra fundamental que sustenta a engenharia de *software* é o foco na qualidade".

- Ian Sommerville [4]: Considera a engenharia de *software* uma abordagem sistemática para produção de *software*, levando em conta processos práticos de custo, prazo e confiança, assim como as necessidades dos clientes e produtores do *software*. A implementação dessa abordagem está suscetível a organização que está desenvolvendo o *software*, pessoas envolvidas nesse processo e o tipo do *software*. Sommerville destaca ainda que não há técnicas e métodos universais que se adequem a todos os sistemas e organizações, sendo um conjunto de métodos e ferramentas que estão sendo construídos aos longo dos anos. Para o autor "O foco está em todos os aspectos da produção de *software*, desde os estágios iniciais da especificação do sistema até sua manutenção, quando o sistema já está sendo usado."

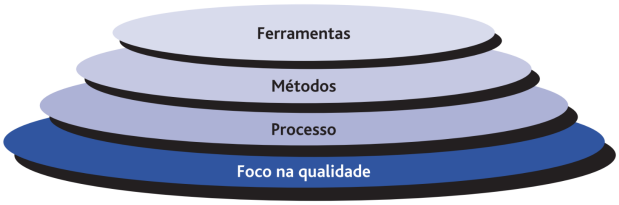

Figure 1: Camadas da engenharia de *software*. Autores: R. S. Pressman and B. R. Maxim

- Marco Valente [5] : Descreve a engenharia de *software* como um tema de grande relevância que busca investigar os desafios e propor soluções para o desenvolvimento de *software* de forma produtiva e com qualidade. O autor destaca a aplicação dos princípios da engenharia na construção do *software*, com abordagem no desenvolver, operar, manter e evoluir o *software*.

C. Engenharia de Requisitos

Vazquez [6] define a Engenharia de Requisitos como uma disciplina da Engenharia de *Software* que consiste no uso sistemático e repetitivo de técnicas para cobrir atividades de obtenção, documentação e manutenção de um conjunto de requisitos para *software* que atendam aos objetivos de negócio e sejam de qualidade. O conceito apresentado está diretamente ligado a aquisição e aplicação do conhecimento para a criação, o aperfeiçoamento e a implementação de sistemas de informação.

Para Pressman [2] "Do ponto de vista do processo de *software*, a engenharia de requisitos é uma ação de engenharia de *software* importante que se inicia durante a atividade de comunicação e continua na de modelagem. Ela deve ser adaptada às necessidades do processo, do projeto, do produto e das pessoas que estão realizando o trabalho". A engenharia de requisitos abrange sete tarefas distintas:

- Concepção: Permite entendimento do problema a ser resolvido e sua natureza;
- Levantamento: Investigação para definição do que é necessário para a construção da solução
- Elaboração: Ocorre a modificação e refinamento dos requisitos básicos;
- Negociação: São estipuladas prioridades, avaliados os custos e riscos e os requisitos são eliminados, combinados e/ou modificados;

- Especificação: É um artefato que apresenta uma descrição detalhada dos aspectos do *software* a ser construído e deve ser especificado antes do início do projeto;
- Validação: É examinada a especificação para garantir que os requisitos tenha sido declarados de forma não ambígua, para que estejam de acordo com os padrões estabelecidos nas etapas de processo, projeto e produto;
- Gestão: Na gestão de requisitos, o foco na identificação, controle e acompanhamento das mudanças necessárias no projeto.

Conforme destacado por Sommerville [4], os requisitos de um sistema consistem nas descrições das funcionalidades que o sistema deve possuir, os serviços que deve oferecer e as restrições que regem seu funcionamento. Esses requisitos são baseados nas necessidades dos clientes para um sistema específico, que tem como objetivo atender a uma finalidade determinada, como o controle de um dispositivo, a realização de pedidos ou a busca por informações. É por meio da definição e compreensão desses requisitos que podemos garantir o desenvolvimento de sistemas que atendam às expectativas e demandas dos usuários.

D. Modelagem de Software com UML

Booch [7] descreve a Linguagem Unificada de Modelagem (UML) como "linguagem gráfica para visualização, especificação, construção e documentação de artefatos de sistemas complexos de *software*. A UML proporciona uma forma padrão para a preparação de planos de arquitetura de projetos de sistemas, incluindo aspectos conceituais, tais como processos de negócios e funções do sistema, além de itens concretos como as classes escritas em determinada linguagem de programação, esquemas de bancos de dados e componentes de *software* reutilizáveis". Sendo que através dessa modelagem é possível simplificar o entendimento do sistema em desenvolvimento.

Os diagramas são a apresentação gráfica que permite a visualização por diferentes perspectivas. Segundo G. T. Guedes [8] "Cada diagrama da UML analisa o sistema, ou parte dele, sob uma determinada óptica. É como se o sistema fosse modelado em camadas. Alguns diagramas enfocam o sistema de forma mais geral, apresentando uma visão externa do sistema, enquanto outros oferecem uma visão de uma camada mais profunda do *software*, apresentando um enfoque mais técnico ou, ainda, visualizando apenas uma característica específica do sistema ou um determinado processo". Dito isto, os diagramas facilitam o entendimento do fluxo e os fatores envolvidos na aplicação trazendo uma visão mais clara e de melhor entendimento do contexto e fluxos em que o *software* faz parte.

Existe uma vasta quantidade de diagramas para as representações de um sistema, a seguir a relação de alguns destes diagramas:

- Diagrama de caso de uso: Esse tipo de diagrama organiza os comportamentos do sistema, mostrando um conjunto de casos de uso, seus atores e seus relacionamentos, ilustrando a visão estática do caso de uso de um sistema [7].
- Diagrama de classe: É um diagrama bastante usado na UML, seu foco é apresentar as classes de composição do sistemas, seus atributos e métodos, bem como seus relacionamentos. "Apresenta uma visão estática de como as classes estão organizadas, preocupando-se em como definir a estrutura lógica delas. O diagrama de classes serve ainda como apoio para a construção da maioria dos outros diagramas da linguagem UML" [8].

E. API REST

Application Programming Interface (API) Rest, também conhecida como API RESTful, é uma interface utilizada por dois sistemas de computador com a finalidade de trocar informações de maneira segura pela internet. Nas organizações, a comunicação com outras aplicações internas e de terceiros é uma realidade para a realização de diversas tarefas. Nesse contexto, a API REST oferece suporte à troca de informações, seguindo padrões de comunicação de software que são seguros, confiáveis e eficientes. Uma API define um conjunto de regras e protocolos que permitem que diferentes softwares se comuniquem entre si. Ela especifica como os componentes de software devem interagir, quais dados podem ser acessados e quais operações podem ser executadas. Através da API REST, os sistemas podem enviar solicitações e receber respostas no formato de dados estruturados, possibilitando a integração e a troca de informações de forma padronizada e eficiente.[9].

Conforme afirmado por Brito [10], uma API REST é uma aplicação cliente/servidor que realiza a troca de dados utilizando o protocolo HTTP e padrões de comunicação, como XML e JSON. Essa API pode ser implementada na linguagem de programação escolhida e o seu objetivo é permitir a interoperabilidade entre diferentes aplicações. Por exemplo, dois sistemas, um desktop e um mobile, podem consumir a mesma API para construir suas interfaces, o que significa que uma única API pode ser utilizada por sistemas distintos. Uma API é considerada RESTful quando é implementada seguindo os princípios e modelos de arquitetura REST. Esses princípios incluem o uso de recursos bem definidos, a utilização adequada dos métodos HTTP, a separação entre cliente e servidor, e a aplicação de cache, entre outros.

Geralmente as APIs *Restful* são implementadas, utilizando o método Hypertext Transfer Protocol (HTTP), em português Protocolo de Transferência de Hipertexto, que informa o que o servidor necessita fazer com o recurso de cada endpoint [9]. O protocolo HTTP define oito métodos *GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS* e *CONNECT*, mas os mais comuns são:

- *GET*: Esse método tem como objetivo requisitar algum dado, é comumente usado quando há necessidade de capturar alguma informação, como por exemplo a lista de empréstimos existentes. É importante destacar que nesse caso nenhuma alteração é feita, apenas os dados são retornados nesse solicitação;
- *POST*: Usado no envio de dados ao servidor, como por exemplo o cadastro de objeto, cadastro de empréstimo

onde nesses casos é necessário o armazenamento dos dados enviados;

- *PUT*: É utilizado para a edição de informações já existentes. Nesse caso, é necessário ter a informação previamente armazenada e realizar ajustes nela. Por exemplo, é possível utilizar o método *PUT* para modificar a descrição de um objeto.
- *DELETE*: Como indica o próprio nome esse método deleta arquivos ou informações no banco de dados.

Na figura 2 é possível visualizar o funcionamento de uma API, o cliente requisita algo a API REST e conforme o respectivo método é realizada a operação no banco de dados, seja de consulta, inserção, edição ou deleção de dados. A resposta a solicitação segue o padrão JSON (JavaScript Object Notation) conforme exemplificado na figura 3 ou XML (Extensible Markup Language).

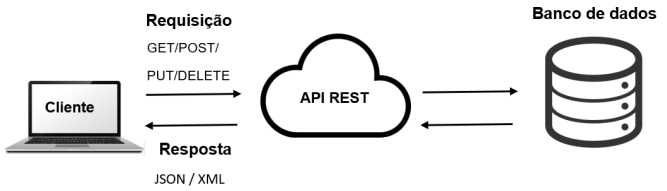

Figure 2: Representação comunicação API. Fonte: próprio autor.

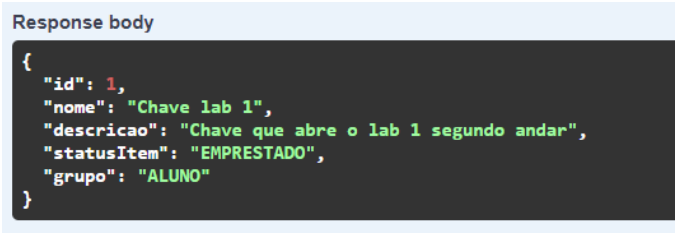

Figure 3: Estrutura de uma resposta JSON. Fonte: próprio autor.

Para haver um melhor entendimento do conteúdo, se faz necessário entender os conceitos que estão atrelados ao que já foi descrito acima, sendo os principais os conceitos:

- *Model-View-Controller*: Conhecido como padrão MVC modelo-visão-controlador. Sommerville[4] descreve como um padrão que serve como base do gerenciamento de interação em muitos sistemas baseados em *Web*. Como pode ser visualizado na figura 4, o modelo separa a representação e a interação dos dados do sistema, estruturando em três componentes lógicos que interagem entre si. Sendo o *Model* o que gerencia o sistema de dados e operações associados a esses dados, a *View* define e gerencia os dados que são apresentados para os usuários e o *Controller* é um intermediador desse processo, gerenciando a interação do usuário e passando as informações para a visão e o modelo. Pressman [2] exemplifica "Na arquitetura MVC, o comando do usuário é enviado da janela do navegador para um processador de comandos (controlador), o qual gerencia o acesso ao conteúdo (modelo) e instrui o modelo de renderização de informações (visão) a transformá-lo para exibição pelo *software* do navegador".

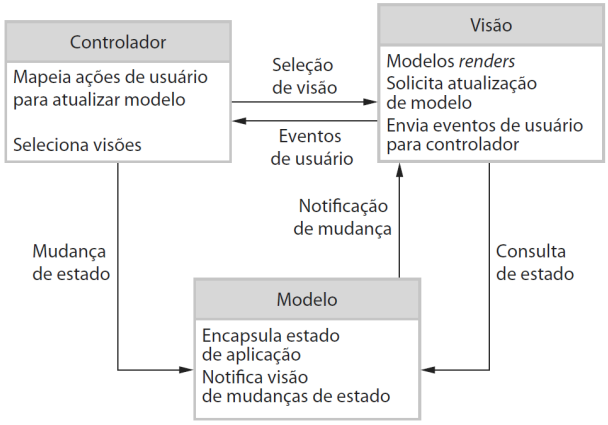

Figure 4: A organização do MVC. Autor: Ian Sommerville

- API: *Application Programming Interface*, em português Interface de Programação de Aplicação, é um conjunto de definições e protocolos que são comumente usados no desenvolvimento e na integração de aplicações. As APIs também podem ser descritas como um contrato entre um provedor e um usuário de informações, estabelecendo a chamada, conteúdo solicitado pelo consumidor e a resposta, conteúdo exigido pelo produtor. [11]. É importante destacar que uma API pode ser criada nas mais diversas linguagens de programação, sendo de suma importância que seja bem desenvolvida e que se tenha documentação com informações claras para facilitar a implementação e estruturar solicitações e respostas. A interface da arquitetura pode ser descrita como um contrato de serviço entre duas aplicações, esse contrato define como as duas se comunicam usando solicitações e respostas. A arquitetura da API é comumente explicada como estrutura cliente e servidor, com o fluxo de envio de chamada do cliente e a aplicação com o envio da chamada feita pelo servidor,. Existem quatro maneiras que as APIs podem se comunicar APIs *SOAP Simple Object Access Protocol* (Protocolo de Acesso a Objetos Simples), APIs RPC *Remote Procedure Calls* (Chamadas de Procedimento Remoto), APIs *WebSocket* e a já descrita acima e utilizada na implementação realizada APIs REST, sendo uma das mais populares e flexíveis encontradas atualmente [9].
- REST: *Representational State Fransfer* (REST – transferência de estado representacional) é uma arquitetura que impõe como uma arquitetura de *software* deve funcionar. Seu uso possibilita uma comunicação confiável e de alta performance em escala, podendo ser implementada e modificada facilmente o que traz o

benefício de visibilidade e portabilidade entre plataformas para qualquer sistema de API [9]. Esse modelo organiza recursos de um sistema *web*, permitindo uma comunicação padronizadas, podendo funcionar de forma independente com seus recursos, componentes e evoluir algumas partes sem comprometer o funcionamento do todo [12].

F. Computação em nuvem

A Computação em Nuvem ou *Cloud Computing* é uma tendência no mercado de Tecnologia da Informação (TI) e tem sido amplamente utilizada em organizações de todos os tipos, portes e setores, com uma grande variedade de casos de uso, como *backup* de dados, recuperação de desastres, e-mail, desktops virtuais, desenvolvimento e teste de *software*, análises de *big data* e aplicativos *web* voltados ao cliente [13].

Segundo Souza, "As infraestruturas existentes permitem entregar tais serviços em qualquer lugar e a qualquer hora, de forma que possamos simplesmente acender a luz, abrir a torneira ou usar o fogão. O uso desses serviços é, então, cobrado de acordo com as diferentes políticas de tarifação para o usuário final. Recentemente, a mesma ideia de utilidade tem sido aplicada no contexto da informática e uma mudança consistente neste sentido tem sido feita com a disseminação de *Cloud Computing* ou Computação em Nuvem [14]". Trata-se de um grande avanço em termos de custos, desempenho e segurança, pois os recursos não necessitam ser provisionados antecipadamente como é feito com os servidores locais (*on-premise*) que necessitavam de uma estimativa de uso para a compra de servidores. Estimativa essa que é bastante perigosa, já que na prática é complicado ter a real noção de quanto será necessário para uso, além disso a necessidade de recursos muitas vezes pode ser sazonal, como por exemplo uma loja que tem sua quantidade de acessos aumentada no período de *black friday* ou alguma promoção.

Cloud Computing permite desacoplar os processos de negócio da TI necessários para rodá-los, introduzindo a ideia de elasticidade na utilização de infraestrutura. Recursos podem ser utilizados em períodos de alta demanda e devolvidos em períodos de baixa demanda, oferecendo as organizações maior escalabilidade, atendendo de forma mais efetiva as necessidades comerciais. Também, permite flexibilizar a alocação de custos para empresas que podem mudar de um modelo baseado em custo de capital o *on-premise* para um modelo baseado em despesas operacionais [15].

Conforme observado por Carissimi [16], um dos avanços tecnológicos mais significativos na área da computação foi a virtualização, que deu origem aos pilares fundamentais da computação em nuvem. A computação em nuvem é um modelo de serviço público que oferece um modelo de negócio no qual os usuários pagam apenas pelos recursos que consomem, seguindo o princípio do "pagar pelo uso" (*pay-as-you-go*). Nesse modelo, os provedores de serviços são responsáveis por manter a infraestrutura física, conhecida como *data center*, compartilhando e alugando seus recursos para diferentes usuários. Essa abordagem permite que os provedores de serviços distribuam os custos, manutenção e investimentos associados à infraestrutura, beneficiando tanto os provedores quanto os usuários. Os usuários têm a flexibilidade de utilizar recursos conforme necessário, sem a necessidade de investir em infraestrutura própria, enquanto os provedores de serviços podem otimizar o uso dos recursos disponíveis.

A computação em nuvem traz vários benefícios, o provedor *Amazon Web Services* (AWS) elenca alguns [13] :

- Agilidade: Oferece grande variedade de tecnologias e podem ser provisionados recursos conforme necessidade, de serviços, de infraestrutura, como computação, armazenamento e bancos de dados até Internet das Coisas, *machine learning*, *data lakes*, análises de dados e entre outros recursos;
- Elasticidade: Funcionalidade que permite provisionar recursos conforme necessidade, sendo possível aumentar ou diminuir instantaneamente a escala desses recursos para se ajustar a cada contexto;
- Economia de custo: A nuvem permite que sejam trocadas despesas fixas (*datacenters* e servidores físicos) por despesas variáveis e pagando apenas pela TI consumida;
- Implantação global em questão de minutos: Benefício de implantação global em poucos minutos e possibilidade de implantação em vários locais físicos diferentes, permitindo a redução de latência melhorando assim a experiência do usuário.

A nuvem possui ainda alguns modelos de serviço que permitem a escolha do nível de controle sobre as informações e tipo de serviço a ser fornecido. Segundo Borges [17] "O modelo conceitual encontrado com maior frequência na literatura é composto por três camadas, este modelo define um padrão arquitetural para soluções em computação em nuvem". Como pode ser visualizado na figura 5 com os tipos de camadas.

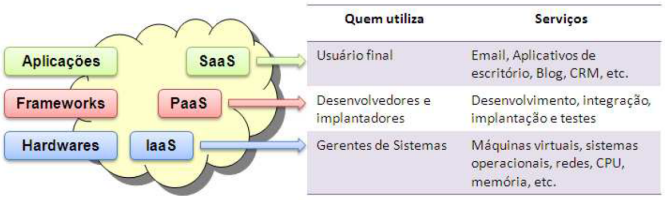

Figure 5: Modelo de serviço. Autor: Borges, Hélder Pereira

Esses modelos de serviços são fundamentais para melhor conhecimento da nuvem e seus recursos, para isso segue uma breve descrição sobre essas camadas[18]:

- Infraestrutura como serviço (IaaS): Categoria mais básica usada para acesso a recursos de computação e armazenamento baseados na Internet. Permite que seja alugada uma infraestrutura de TI (servidores e máquinas virtuais, armazenamento, redes e sistemas operacionais) de um provedor de nuvem em uma base de pagamento conforme o uso;
- Plataforma como serviço (PaaS): Dá aos desenvolvedores as ferramentas necessárias para criar e hospedar aplicativos *Web*. Proporciona aos usuários o acesso aos componentes necessários para desenvolver e operar rapidamente aplicativos *Web* ou móveis na Internet,

sem se preocupar com a configuração ou gerenciamento da infraestrutura subjacente dos servidores, armazenamento, redes e bancos de dados;

- *Software* como serviço (SaaS): Usado para aplicativos baseados na *Web*, com método de entrega de aplicativos de *software* na Internet, no qual os provedores de nuvem hospedam e gerenciam os aplicativos de *software*, fazendo com que seja simples ter o mesmo aplicativo em todos seus dispositivos de uma só vez por meio da nuvem .

Vários serviços são disponibilizados seguindo os modelos que foram especificados acima fazendo com que se tenha uma gama de serviços com o intuito de atingir de forma mais ampla clientes de diversos tamanhos e contextos, alguns exemplos de serviços compatíveis com cada camada são:

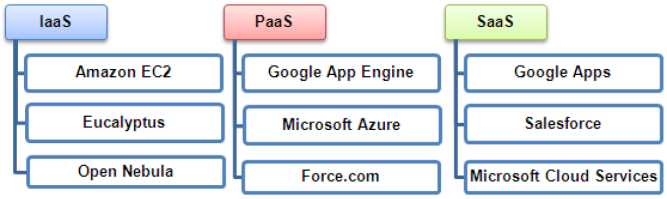

Figure 6: Exemplos de serviços. Autor: Borges, Hélder Pereira

O avanço da computação em nuvem proporcionou a criação de vários modelos, tipos e serviços que foram evoluindo para atender a necessidades tecnológicas das organizações. Há três maneiras de implantação de serviços de nuvem, são elas:

- Nuvem pública: É a forma mais comum e a mais barata, os recursos como servidor e armazenamento pertencem ao provedor de nuvem terceirizado, são operados e entregues pela internet. Nessa modalidade há um compartilhamento dos mesmos dispositivos de *hardware*, armazenamento e de rede com outras organizações / locatários. As vantagens das nuvem públicas são a redução de custos já que o pagamento é feito pelo serviço usado, não é necessária a manutenção já que isso é responsabilidade do provedor, escalabilidade quase ilimitada já que os recursos são sob demanda e podem ser adequados a necessidade real da organização, alta confiabilidade com ampla rede de servidores com redundância para evitar falhas [19];
- Nuvem privada: Como o próprio nome indica, consiste em recursos direcionados usados exclusivamente para uma única empresa ou organização, podendo estar fisicamente no *datacenter* local ou em uma estrutura hospedada pelo provedor de serviços tercerizado, nesse caso os serviços e infraestrutura são exclusivos para a organização. Esse tipo de nuvem é usado por órgãos governamentais, instituições financeiras e outras organizações de grande porte com operações críticas para os negócio ou que por conta de sanções jurídicas devem guardar seus dados de forma individualizada. As vantagens são a maior flexibilidade personalizando ambiente em nuvem para atender necessidades específicas, maior controle tendo mais controle e privacidade com recursos direcionados a sua organização e o benefício já descrito acima, a escalabilidade [19];
- Nuvem híbrida: Combina a infraestrutura local ou uma nuvem privada com uma nuvem pública, permitindo com que os dados e aplicativos se movam entre esses dois ambientes. A escolha desse modelo se dá em muitos casos quando as organizações necessitam atender a requisitos regulatórios e de soberania de dados, aproveitando ao máximo investimento em tecnologia local ou solução de problemas de baixa latência. A nuvem híbrida possui a vantagem de controle já que a organização pode manter uma infraestrutura privada dos artigos confidenciais ou cargas de trabalho de baixa latência, flexibilidade podendo usufruir de recursos adicionais quando for preciso, custo-benefício em escalar para a nuvem e pagar apenas o usar e facilidade para a transição para a nuvem de forma gradual [19].

III. Trabalhos relacionados

Este capítulo apresenta uma análise de algumas propostas de sistema de gestão que de alguma maneira estão relacionados ao sistema de gerenciamento de objetos.

1) Sistema de empréstimo de objetos digitais (SEOD) [20]: Voltado para uso em ambientes restritos como, por exemplo, bibliotecas de instituições de ensino ou empresas, quando se tem acesso apenas para usuários cadastrados no sistema bibliotecário. O autor criou uma modelagem e um protótipo para o gerenciamento de empréstimos e devolução de obras em formato digital. O sistema prevê um conjunto de sistemas, um sistema bibliotecário (SB) e um sistema cliente (SC) com as funcionalidades descritas abaixo:

Sistema Bibliotecário:

- Manter atualizado o catálogo bibliográfico;
- Manter um cadastro de clientes;
- Verificar login e senha do usuário;
- Verificar informações do objeto digital cujo empréstimo está sendo solicitado;
- Controlar a disponibilização de licenças de uso do objeto digital;
- Controlar a utilização da cota de empréstimos do usuário;
- Montar, desmontar e tratar os dados recebidos através dos pacotes de dados;
- Proteger os dados a serem transmitidos.

Sistema Cliente:

- Manter registro dos livros em poder do usuário;
- Gerenciar a utilização dos livros conforme seus prazos de devolução;
- Eliminar livros que já ultrapassaram o prazo de devolução;
- Montar, desmontar e tratar os dados recebidos através dos pacotes de dados;
- Proteger os dados a serem transmitidos.

2) Sistema web para o gerenciamento dos materiais de laboratório de computação [21]: Trata-se de um sistema de controle de entradas e saídas de componentes eletrônicos e equipamentos disponíveis na sala de apoio do Departamento Acadêmico de Informática (DAINF) da Universidade Tecnológica Federal do Paraná, Campus Pato Branco (UTFPR), para utilização em aulas e projetos, aplicação desenvolvida com as tecnologias Java para o *back-end* e Angular para o *front-end*. O sistema se dispõe a:

- Gerenciar os componentes eletrônicos e equipamentos de 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.

Apesar dos dois trabalhos relacionados acima serem voltados para a gestão de objetos, o Sistema de empréstimo de objetos digitais (SEOD) é apenas uma modelagem e protótipo e se dispõe apenas a objetos digitais voltados para ambientes restritos como bibliotecas de instituições de ensino ou empresas, fazendo o gerenciamento de empréstimos e devolução de obras em formato digital para bibliotecas. Já o Sistema de gerenciamento dos materiais de laboratório de computação, foca em equipamentos eletrônicos e equipamentos disponíveis na sala de apoio específicos da Universidade Tecnológica Federal do Paraná UTFPR, o sistema possui grande foco no estoque e compra de itens. Ele está estruturado para atender apenas o contexto do Departamento Acadêmico de Informática (DAINF) e suas especificidades.

O GEIFBA é um sistema mais abrangente que oferece diversas possibilidades de cadastro de objetos, bem como um acompanhamento detalhado e a opção de solicitação de empréstimo tanto pelos usuários que desejam pegar um objeto emprestado, quanto pelo administrador do sistema. Além disso, o sistema envia e-mails com notificações sobre as movimentações de empréstimos, proporcionando uma visão completa das atividades relacionadas.

IV. GEIFBA

O presente projeto tem como objetivo geral gerenciar os empréstimos de objetos do Instituto Federal da Bahia (chaves de setores, projetores, computadores, etc). Tendo como objetivos específicos o auxílio na gestão dos empréstimos realizados no IFBA no campus Salvador, o desenvolvimento das funcionalidades de cadastro, edição e remoção de usuários, grupos, itens, tornando o processo mais visual, identificando padrões e necessidade de ampliação de ativos físicos dos empréstimos, essa aplicação é o Sistema para Gerenciamento de Empréstimos de Objetos (GEIFBA).

A. Metodologia

Foi realizado um levantamento detalhado das necessidades relacionadas ao processo de empréstimo, buscando soluções personalizadas para o problema. O desenvolvimento foi conduzido de forma sequencial, seguindo a ordem relacionada abaixo:

- Levantamento de requisitos: Os dados foram obtidos a partir de reuniões entre o orientador deste trabalho que elucidou dúvidas referentes aos processo;
- Análise do projeto e viabilidade: Foi modelada a estrutura do projeto, diagrama de classes, diagrama de casos de uso e revisão dos requisitos que melhor se adequavam ao contexto do problema;
- Desenvolvimento: Inicialmente ocorreu a escolha das tecnologias e o projeto foi iniciado pelo *backend*, com o *Create, Read, Update, Delete* (CRUD) de empréstimo, item, pessoa e usuário, além dos mecanismos de autenticação usando e autorização de recursos conforme perfil. Posteriormente, foi realizado o refinamento das rotas visando facilitar as buscas e paginação das listas no *backend*. Nessa etapa também foi gerada a documentação da aplicação através com o *Swagger* que auxilia a descrição, consumo e visualização de serviços de uma API REST, foram construídos testes com as chamadas da API (*Application Programming Interface*) usando a ferramenta Postman que facilita a verificação se a cada ajuste o sistema era afetado de alguma maneira, trazendo maior segurança ao processo de desenvolvimento. Na sequência, foi iniciado o projeto *frontend* com a criação dos *services* na conexão das chamadas do *backend* e foram feitas as telas de cadastros de pessoa, usuário, objeto e empréstimo, tratativa da autenticação do sistema com guarda de rotas e menu personalizado dos perfis do sistema.
- Testes: Foi realizado os testes da API no Postman que é uma plataforma de API que desenvolvedores podem projetar, construir e testar suas APIs. Os testes foram feitos em forma de *scripts* e a medida que o desenvolvimento foi seguindo foi possível gerar a bateria de testes e verificar se as funcionalidades implementadas estavam íntegras.
- Implantação: Ocorreu no provedor AWS, inicialmente foi feito o *deploy* do *backend* no serviço EC2. Já o *frontend* da solução foi implantado no *bucket* do S3, dessa forma o sistema encontra-se disponível usando os recursos de nuvem.

B. Escopo do sistema

A aplicação tem como objetivo realizar de forma eficiente a gestão do fornecimento de objetos no Instituto dos alunos, monitores, professores ou técnicos centralizando na aplicação esse processo de cadastro, aprovação e acompanhamento dos empréstimos.

O processo é iniciado com o cadastro de pessoa, com esse cadastro o usuário já pode ter a operação de empréstimo desde que seja feita pela interface do administrador. Na criação do usuário deve ser cadastrado o *login* e a senha de acesso ao sistema e seu perfil.

Após o *login* no sistema, as operações estão disponíveis a todos os usuários. O sistema conta com dois perfis: o perfil administrador, destinado aos servidores responsáveis por cadastrar, excluir ou editar objetos, aprovar solicitações de empréstimo, registrar a devolução dos objetos, criar empréstimos

quando necessário e cancelar empréstimos. Por outro lado, o perfil usuário permite que os dados pessoais do usuário logado sejam visualizados e editados, além de possibilitar solicitações de empréstimo e o cancelamento de pedidos de empréstimo.

Durante o processo de cadastro do objeto, é necessário selecionar a qual grupo de permissões ele pertence, que varia de aluno a técnico, seguindo uma ordem hierárquica do menor para o maior. Conforme figura 7, o aluno possui a menor permissão, podendo pegar emprestado apenas se o objeto pertencer ao mesmo grupo que ele, já todos os demais grupos monitor, professor e técnico estão habilitados para empréstimos do objeto do grupo aluno, pois a permissão deste grupo é a mais baixa.

Grupo de permissões				
	Aluno	Monitor	Professor	Técnico
Aluno	x			
Monitor	x	x		
Professor	x	x	x	
Técnico	x	x	x	x

Figure 7: Grupo de permissões da solução. Fonte: próprio autor.

Algumas funcionalidades do sistema são:

1) Cadastro de pessoa: O administrador deve registrar as pessoas que irão utilizar o sistema.

2) Cadastro de usuário: os administradores podem cadastrar os usuários informando a pessoa, *login* e senha.

3) Cadastro de objeto: Os administradores podem cadastrar os objetos informando o nome, descrição e grupo de permissão.

4) Cadastro de empréstimo: Os usuários podem solicitar o empréstimo o pedido é direcionado para a tela do administrador que pode aprová-lo ou nega-lo. Os administradores podem iniciar um empréstimo imediatamente, sem precisar da etapa de aprovação selecionando a pessoa e o objeto a ser emprestado.

5) Lista de empréstimos pendentes de aprovação: Os administradores após logar no sistema tem como tela inicial a lista dos empréstimos pendentes de aprovação.

6) Lista de empréstimos em andamento: Os administradores podem visualizar a lista dos empréstimos em andamento, ou seja os objetos que estão emprestados.

7) Lista de empréstimos: Os administradores podem visualizar a lista dos empréstimos de todos os empréstimos. Os usuários podem visualizar a lista de todos os empréstimos existentes no seu cadastro com seu respectivos status.

C. Requisitos

No processo de criação de software, é fundamental considerar os requisitos funcionais e não funcionais. Os requisitos funcionais descrevem as funções e características que o sistema deve possuir, além de especificar o comportamento esperado em situações específicas. Por outro lado, os requisitos não funcionais abrangem aspectos que não estão diretamente ligados a funcionalidades específicas, mas sim a condições gerais necessárias. Isso inclui requisitos como confiabilidade, desempenho, portabilidade, entre outros. Dessa forma, os requisitos funcionais estabelecem as funcionalidades que o sistema deve oferecer, enquanto os requisitos não funcionais definem as características e condições mais abrangentes que o sistema precisa atender e ser considerado adequado.

Foi feito o levantamento dos requisitos buscando-se facilitar a identificação de quais as necessidades e cenários a aplicação busca atender. Foram separados conforme literatura em dois grupos: requisitos funcionais e requisitos não funcionais.

Os requisitos funcionais do Sistema de gerenciamento de empréstimos apresentados na figura 9 descrevem o que o usuário com o perfil denominado administrador (ADMIN) pode realizar, sendo possível cadastrar usuários, itens, empréstimos, editar pessoas, usuários, empréstimos e itens. Também é possível visualizar lista de empréstimos pendentes de aprovação, lista de empréstimos em andamento e a lista completa de todos os empréstimos realizados. O usuário com perfil denominado usuário (USER) tem uma visão mais restrita do sistema, sendo possível recuperar a senha em caso de esquecimento, editar dados relacionados ao seu perfil, solicitação de empréstimo, lista com os empréstimos relacionados ao seu usuário, além disso o sistema notifica no e-mail cadastrado as operações de empréstimo e recuperação de senha.

Os requisitos não funcionais especificados na figura 8 estão relacionados aos atributos de qualidade necessários, buscando garantir que o sistema seja adequado à sua finalidade, sendo escalável, seguro e confiável. Na solução apresentada, o objetivo é simplificar o processo ao máximo. Em prol disso, o requisito de portabilidade é utilizado, permitindo que o GEIFBA seja acessado diretamente pelo navegador, sem a necessidade de instalação no dispositivo, apenas exigindo um navegador de internet.

A usabilidade é um aspecto importante, proporcionando ao usuário uma fácil visualização do foco da aplicação, que é o empréstimo, sem exigir longas horas de treinamento na realização dessa tarefa. A aplicação foi construída de forma desacoplada e modularizada, o que facilita a manutenção e a adição de novas funcionalidades.

Quanto à disponibilidade, o sistema está hospedado na nuvem, e espera-se que ele seja altamente disponível. Em relação à segurança, todas as rotas do sistema são autenticadas, exigindo o login inicial para realizar qualquer ação. Além disso, são aplicadas permissões de acordo com o usuário, abrangendo operações como empréstimos, visualização de informações, cadastro de objetos, cadastro de usuários, entre outras.

Requisito	Descrição
Portabilidade	O sistema deve ser acessado diretamente no browser ou navegador, podendo usado em diversos dispositivos;
Usabilidade	O sistema deve ter uma interface que permita ao usuário ter uma facilidade de uso e não deve despender de muitas horas de treinamento;
Manutenibilidade	O sistema deve ser criado de forma a facilitar futuros reparos e criação de novas funcionalidades;
Disponibilidade	O sistema deve ser hospedado de forma a ter maior disponibilidade e diminuição de ocorrência de falha;
Segurança	O sistema deve checar com senha e outros meios de checagem de identidade dos usuários e suas permissões.

Figure 8: Requisitos não funcionais da solução. Fonte: próprio autor.

D. Diagramas

Os diagramas são muito importantes na representação sistema e suas particularidades, sendo uma comunicação clara e visual das diferentes partes de um sistema.

O diagrama de caso de uso do sistema possui dois atores, que são denominados Usuário (User) e Administrador (Admin). Cada ator possuirá um fluxo dentro da aplicação conforme apresentado a seguir.

Visando compreender melhor o funcionamento do fluxo de empréstimo, é essencial mencionar os diferentes tipos de status e quando eles ocorrem. A Figura 10 apresenta cada um dos status existentes no sistema, juntamente com suas descrições. É por meio dessa figura que é possível entender e acompanhar os diferentes estágios pelos quais um empréstimo pode passar, fornecendo um contexto claro e detalhado do status em cada etapa do processo.

A figura 11 retrata o diagrama de caso de uso do sistema do perfil User na aplicação. Os usuários com esse perfil podem solicitar o empréstimo tornando o status do empréstimo como início pendente, esse status se dá por conta da pendência de aprovação que é dada apenas pelo usuário com perfil Admin. Além do status início pendente o empréstimo pode ter o status iniciado, cancelado e finalizado conforme explicitado na tabela de status dos empréstimos na figura 10. Conforme diagrama após efetuar o *login* é possível verificar o histórico dos empréstimos relacionados ao seu usuário, editar perfil alterando seus dados, solicitar um empréstimo de objeto, em caso de desistência o usuário pode cancelar o pedido de empréstimo. Nas operações de cancelamento, aprovação e negativa de empréstimo, o usuário recebe um informe no seu e-mail da operação que é feita trazendo maior segurança do processo.

ID	Requisito funcional
RF1	O sistema deve permitir que o usuário administrador autenticado associe uma pessoa a um grupo de aluno, monitor, professor ou técnico;
RF2	O sistema deve permitir que o usuário administrador autenticado cadastre novos usuários;
RF3	O sistema deve permitir que o usuário administrador autenticado cadastre e exclua itens do empréstimo;
RF4	O sistema deve permitir que o usuário administrador autenticado cadastre e exclua usuário;
RF5	O sistema deve permitir que a senha do usuário seja recuperada por e-mail;
RF6	O sistema deve permitir o cadastramento e exclusão de item de empréstimo;
RF7	O sistema deve gerar comprovante para os empréstimos;
RF8	O sistema deve gerar comprovante para as devoluções;
RF9	O sistema deve permitir que seja filtrado por data de empréstimo;
RF10	O sistema deve permitir que seja filtrado por data de devolução do item;
RF11	O sistema deve permitir que seja filtrado por usuário;
RF12	O sistema deve permitir que seja filtrado por servidor responsável pelo empréstimo;
RF13	O sistema deve permitir que os itens sejam filtrados por status.

Figure 9: Requisitos funcionais da solução. Fonte: próprio autor.

O diagrama de caso de uso perfil Admin está apresentado na figura 12, ao efetuar o login no sistema, possui uma visão geral das funcionalidades relacionadas a objetos, usuários e empréstimos. O Admin tem acesso a uma lista completa de empréstimos, incluindo aqueles pendentes e em andamento. No caso dos empréstimos pendentes, o Admin tem a capacidade de autorizar ou negar os pedidos de empréstimo. Quando autorizado e realizado o empréstimo, o Admin pode finalizá-lo, registrando a devolução do objeto, ou cancelar o empréstimo.

No perfil Admin, é oferecida uma funcionalidade adicional de inclusão de empréstimos. Nessa opção, o sistema apresenta um formulário que permite ao Admin selecionar o objeto disponível, escolher o usuário que irá pegá-lo emprestado e realizar a inclusão do empréstimo. Após essa ação, o status do empréstimo é definido como "iniciado" e o usuário envolvido recebe um e-mail de confirmação sobre essa transação. É importante ressaltar que o perfil Admin possui todas as funcionalidades do perfil User.

Status empréstimo	
Tipo	**Descrição**
Início pendente	Quando o empréstimo é solicitado pelo perfil User e está pendente de aprovação
Iniciado	Ocorre quando o empréstimo é aprovado pelo perfil Admin ou Ocorre quando o empréstimo é cadastrado diretamente pelo perfil Admin selecionando a pessoa que pegou emprestado o objeto.
Cancelado	Ocorre em caso de não aprovação do pedido de empréstimo feito pelo perfil User ou Ocorre quando o perfil User desiste do pedido de empréstimo
Finalizado	Quando o objeto é devolvido e é feita a baixa do empréstimo no sistema

Figure 10: *Status* empréstimo do GEIFBA. Fonte: próprio autor.

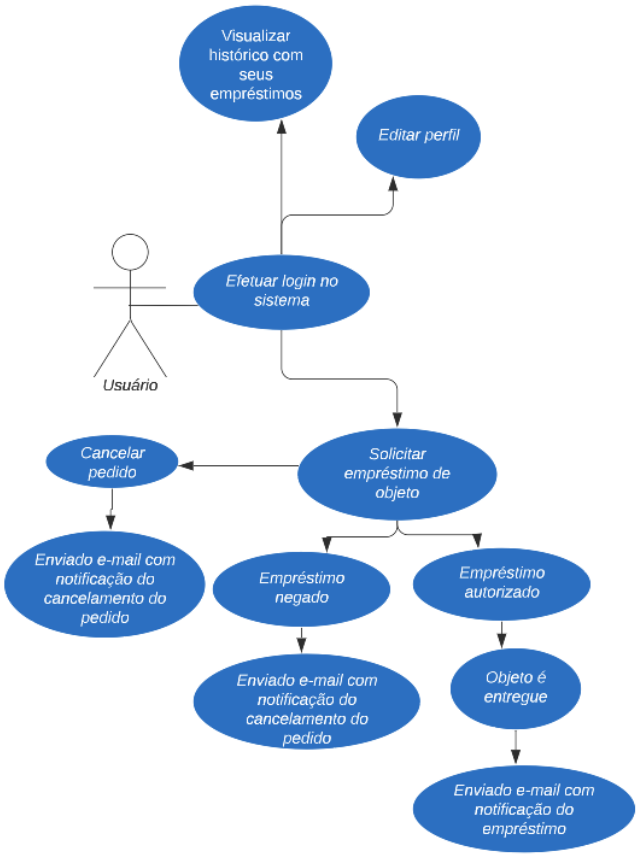

Figure 11: Diagrama de caso de uso perfil User do GEIFBA. Fonte: próprio autor.

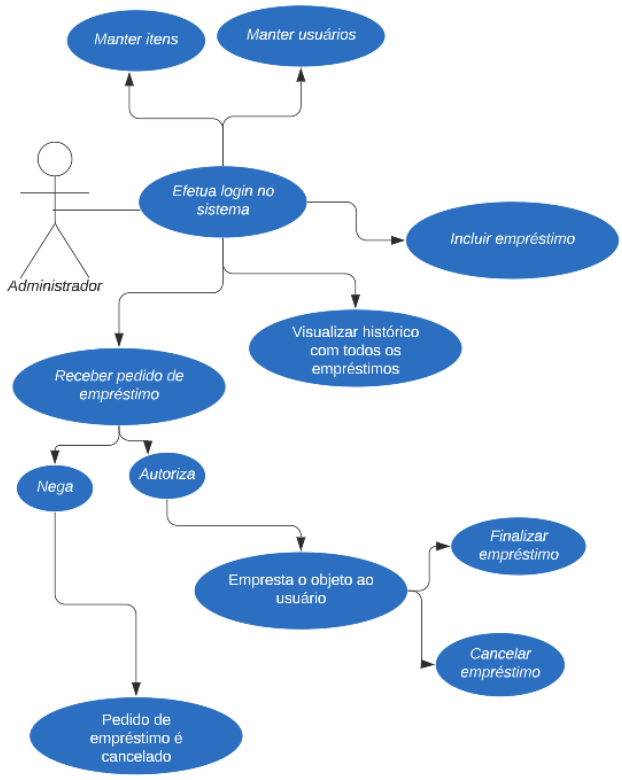

Figure 12: Diagrama de caso de uso perfil Admin do GEIFBA. Fonte: próprio autor.

O diagrama de classes simboliza como foi estruturado o GEIFBA, na figura 13 apresenta a estruturação do modelo ou classe domínio que são a representação de uma entidade do negócio como por exemplo o empréstimo, item, pessoa e usuário.

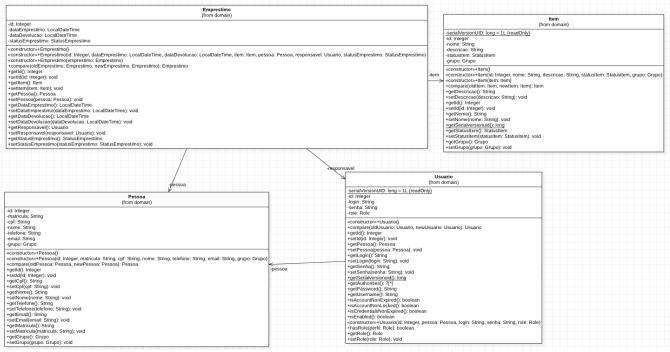

Figure 13: Diagrama de classe domínio da solução. Fonte: próprio autor.

A figura 14 ilustra o diagrama de algumas configurações que foram utilizadas no sistema através do *spring boot* e que foram fundamentais no processo de desenvolvimento da API.

Já no diagrama de classes de recurso da figura 15 são apresentados os recursos da aplicação que são um elo entre a classe de domínio e as de serviços, apresentando os *endpoints*

Figure 14: Diagrama de classe de configuração da solução. Fonte: próprio autor.

que foram criados, como *findAll* que retorna uma lista com todos os cadastrados ou *findById* que retorna apenas do id passado como parâmetro da chamada.

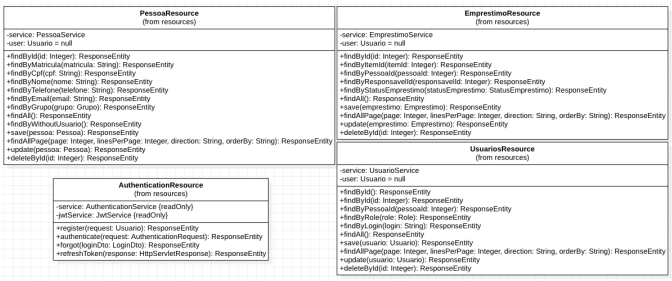

Figure 15: Diagrama de classe resource da solução. Fonte: próprio autor.

Em benefício de um bom funcionamento de uma API é fundamental as tratativas de exceções, a figura 16 retrata o digrama de classe das exceções da solução, a exemplo do *objectNotFoundException* que é gerada quando o objeto solicitado não é encontrado no repositório, na *AccessDeniedException* é lançada quando não possui uma autoridade necessária, entre outros conforme diagrama.

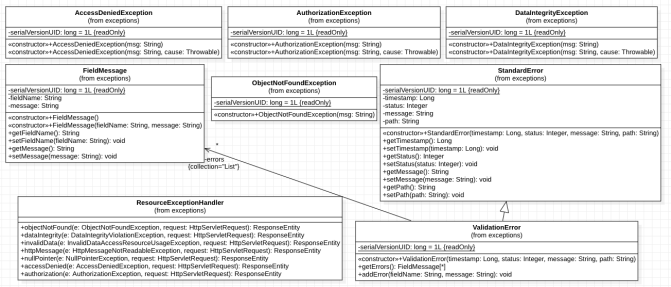

Figure 16: Diagrama de classe com as exceções da solução. Fonte: próprio autor.

E. Arquitetura do sistema

Apresentando a Arquitetura de Sistema na figura 17. Essa arquitetura de sistema combina diversas tecnologias e serviços poderosos em prol de oferecer um sistema robusto, escalável e seguro. Com a combinação do Angular, Java, PostgreSQL, Spring Boot, JWT, Swagger e os serviços *Amazon S3* e *Amazon EC2*, visando fornecer uma solução de alta qualidade que atenda às necessidades do projeto.

F. Apresentação do sistema

Na apresentação da solução, serão demonstradas algumas telas e funcionalidades do Sistema de Gerenciamento de Em-

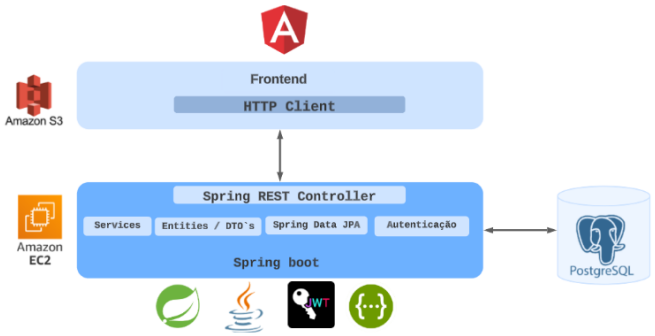

Figure 17: Arquitetura do GEIFBA. Fonte: próprio autor.

préstimos de Objetos (GEIFBA).

1) Empréstimos: Estão divididos conforme abaixo, sendo o primeiro disponível nos perfis administrador e usuário e os demais visíveis apenas no perfil administrador:

- Meus empréstimos: Lista os empréstimos do usuário logado;
- Empréstimos pendentes: Lista os empréstimos pendentes de aprovação;
- Empréstimos em andamento: Lista os empréstimos que foram iniciados e ainda não constam como devolvidos;
- Todos os empréstimos: Lista todos os empréstimos realizados no sistema e seus dados.

*a) **Meus empréstimos**:* Após o usuário ser autenticado é exibida a tela "Meus empréstimos", conforme figura 18, nesta tela são exibidos todos os empréstimos do usuário logado e seus status.

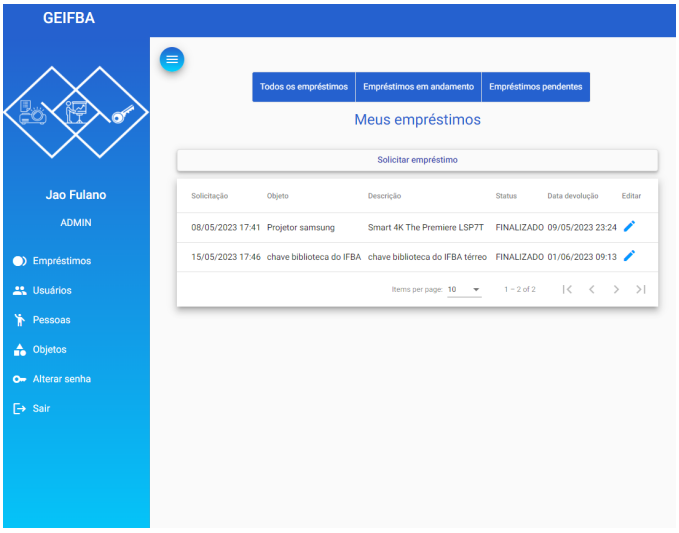

Figure 18: Tela inicial meus empréstimos do GEIFBA. Fonte: próprio autor.

Para solicitar empréstimo o usuário clica no botão “Solicitar empréstimos” conforme figura 19, a funcionalidade está

disponível em ambos perfis, administrador e usuário. A seguir, será apresentado formulário que já possui o nome do usuário logado carregado conforme figura 20 e deve ser selecionado o objeto do pedido de empréstimo. O usuário só visualiza os empréstimos que o mesmo pode pegar emprestado de acordo com o seu grupo.

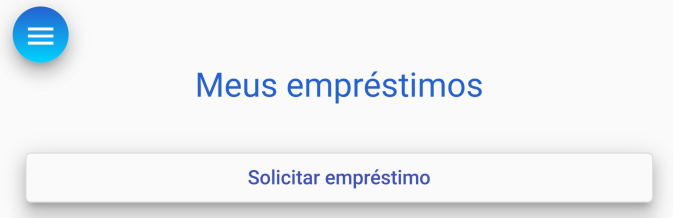

Figure 19: Solicitação de empréstimo usuário logado no GEIFBA. Fonte: próprio autor.

Figure 20: Formulário de empréstimo usuário logado no GEIFBA. Fonte: próprio autor.

O status do objeto ficará como “EMPRESTADO”, com a finalidade de que outro usuário não possa solicitar o mesmo objeto.

O empréstimo será enviado e o usuário com o perfil Administrador deve aprovar ou rejeitar. O usuário recebe um e-mail quando ocorre o início ou o cancelamento do pedido de empréstimo.

Caso o usuário desista do empréstimo o mesmo pode solicitar o cancelamento, caso ainda não tenha sido aprovado.

b) ***Empréstimos pendentes****:* Nessa tela, é possível aprovar ou rejeitar as solicitações de empréstimo solicitadas pelos usuários.

Ao clicar no botão vermelho “Rejeitar” o usuário será sinalizado no e-mail cadastrado do cancelamento do empréstimo e o pedido não constará mais na lista.

Ao clicar no botão verde “Aprovar” o usuário será sinalizado no e-mail cadastrado do início do empréstimo e o objeto selecionado se mantém com o status “emprestado” e poderá ser acompanhado através do menu “Empréstimos em andamento”.

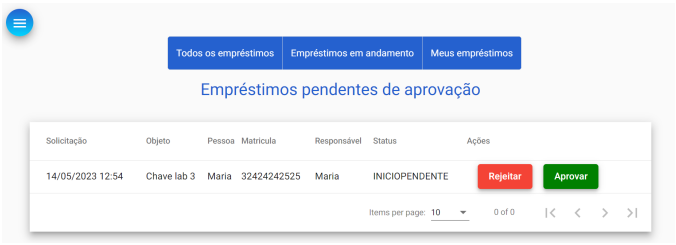

Figure 21: Empréstimos pendentes de aprovação do GEIFBA. Fonte: próprio autor.

c) ***Empréstimos em andamento****:* Nessa tela, é possível acompanhar os empréstimos em andamento, ou seja, que foram iniciados e o objeto ainda não foi devolvido conforme figura 22.

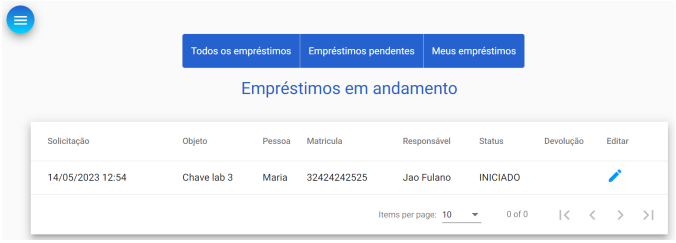

Figure 22: Empréstimos em andamento do GEIFBA. Fonte: próprio autor.

d) ***Todos os empréstimos****:* Nessa tela, é possível visualizar uma lista com todos os empréstimos realizados no sistema e seus dados, conforme figura 23 Nesse módulo do sistema o administrador pode criar um empréstimos selecionando uma pessoa do sistema, conforme exemplificado na figura 24, a solicitação pode ser feita através do menu "Todos os empréstimos" clicando no botão novo empréstimo, será aberto o formulário figura 25, e basta selecionar na lista o usuário e objeto a ser emprestado, e clicar em “Cadastrar”. Com esse procedimento o empréstimo já é iniciado e caso a pessoa possua usuário vai receber notificação no e-mail cadastrado.

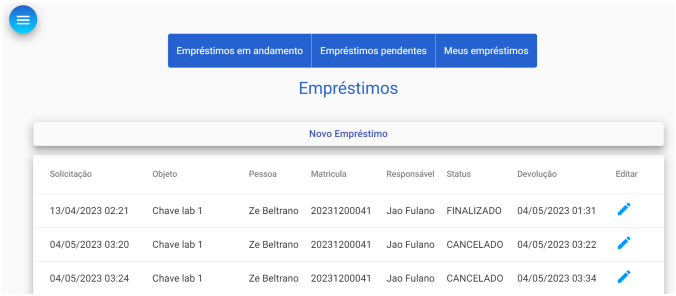

Figure 23: Lista com todos os empréstimos realizados no GEIFBA. Fonte: próprio autor.

2) Objetos: A visualização do sistema é diferenciada conforme perfil autenticado.
O menu Objetos do perfil Usuário é apresentada uma lista apenas com os objetos que o usuário pode pegar “emprestado”

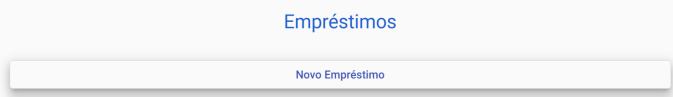

Figure 24: Botão de solicitação de empréstimo selecionando pessoa cadastrada GEIFBA. Fonte: próprio autor.

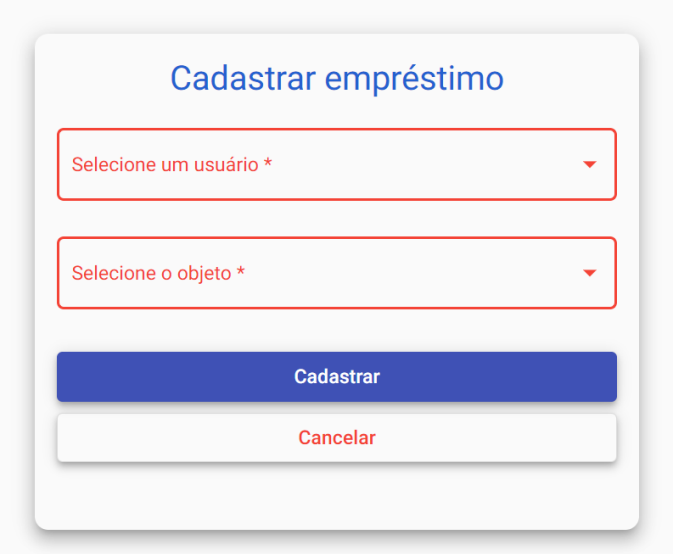

Figure 25: Solicitação de empréstimo selecionando pessoa cadastrada no GEIFBA. Fonte: próprio autor.

com o nome, descrição, status e grupo de cada objeto.
No menu Objetos do perfil Administrador é apresentada uma lista com todos os objetos e seus dados: nome, descrição, status e grupo de cada objeto e as ações que podem ser feitas de editar e excluir (caso não possua empréstimo vinculado), conforme figura 26.

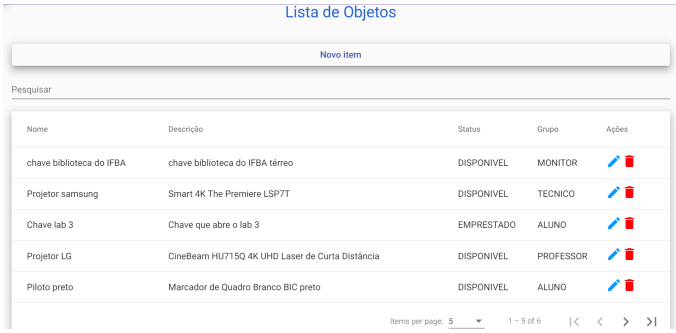

Figure 26: Lista de Objetos do GEIFBA. Fonte: próprio autor.

O cadastro de objetos apenas pode ser feito com o perfil administrador, ao clicar em "Novo item" conforme figura 27, posteriormente será exibido o formulário abaixo devendo ser feito o preenchimento de dados referentes ao objeto que será cadastrado e ser selecionado o grupo ao qual o objeto pertence, posteriormente conforme figura 28.

Com o objetivo de proporcionar uma compreensão mais clara do grupo de permissões, é relevante ressaltar que o aluno

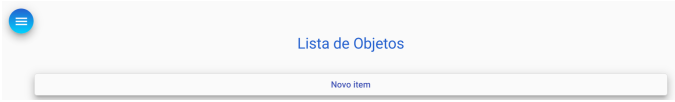

Figure 27: Inclusão de novo objeto GEIFBA. Fonte: próprio autor.

Figure 28: Formulário cadastrar objeto do GEIFBA. Fonte: próprio autor.

possui o nível de permissão mais baixo, enquanto o técnico detém o mais elevado, conforme ilustrado na figura 29.

ALUNO	MONITOR	PROFESSOR	TÉCNICO

Figure 29: Inclusão de novo objeto GEIFBA. Fonte: próprio autor.

Sendo a seguinte regra de permissão:

- Aluno: os objetos cadastrados com esse grupo podem ser emprestados a todas as pessoas;
- Monitor: Os objetos cadastrados com esse grupo podem ser emprestados ao perfil monitor, professor ou técnico;
- Professor: Os objetos cadastrados com esse grupo podem ser emprestados ao perfil professor ou técnico;
- Técnico: Os objetos cadastrados com esse grupo só podem ser emprestados às pessoas com o perfil técnico.

3) Pessoas: O módulo pessoa condensa informações cadastrais das pessoas vinculadas ao sistema. Além disso é possível

editar os dados conforme ícone em azul como pode ser visualizado na figura 30. Nessa mesma tela é possível incluir

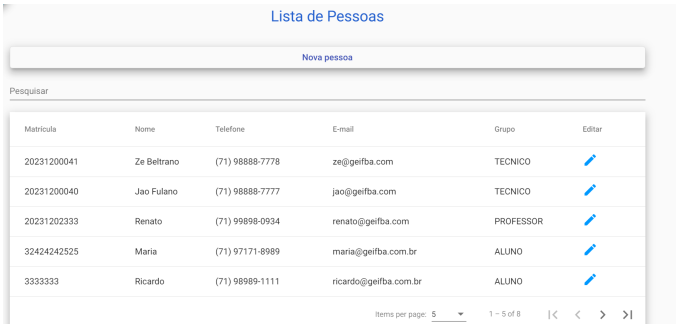

Figure 30: Tela com lista de pessoas cadastradas no sistema. Fonte: próprio autor.

pessoa informando os dados de matrícula, nome, CPF, telefone, e-mail e grupo. Com a pessoa cadastrada, é possível criar empréstimo no menu meus empréstimos (perfil administrador) e a pessoa recebe no e-mail cadastrado as notificações.

A fim de permitir o acesso ao sistema e a solicitação de empréstimos individuais, é necessário que o usuário seja devidamente cadastrado, conforme será detalhado a seguir.

4) Usuários: No menu Usuários é apresentada uma lista de todos os usuários cadastrados no sistema, também é possível editar os dados conforme ícone em azul como pode ser visualizado na figura 31.

Para cadastrar os usuários é solicitado o *login*, a senha. É necessário selecionar o perfil desejado se administrador ou usuário. Com isso, o acesso e solicitação de empréstimos pode ser solicitado e acompanhado pelo próprio usuário.

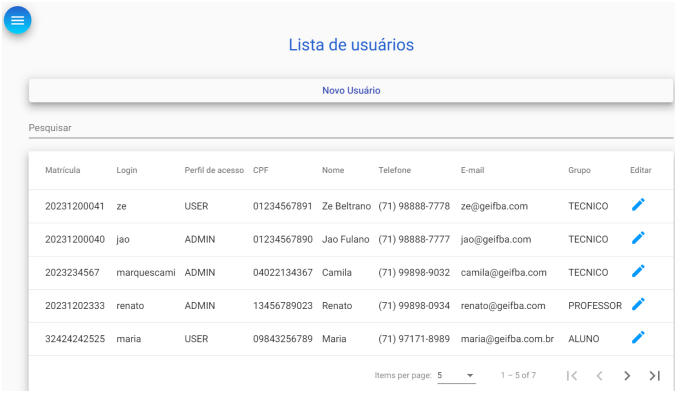

Figure 31: Tela com lista de usuários cadastrados no sistema. Fonte: próprio autor.

G. Notificações do sistema

A fim de evitar a necessidade de imprimir comprovantes de empréstimos, devoluções e cancelamentos, o GEIFBA envia por e-mail para o endereço cadastrado conforme abaixo:

- Cancelamento do empréstimo efetuado: Quando ocorrer o cancelamento seja por parte do próprio solicitante ou se o empréstimo for rejeitado o solicitante recebe, em seu e-mail cadastrado notificação informando do cancelamento, conforme figura 32.

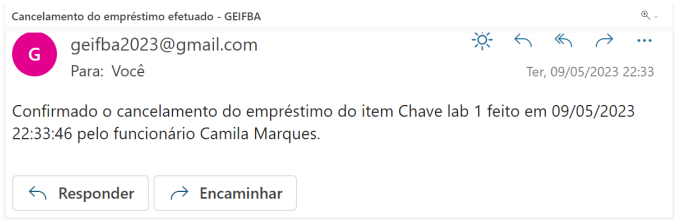

Figure 32: E-mail do GEIFBA notificando do cancelamento do empréstimo. Fonte: próprio autor.

- Devolução do empréstimo efetuado: Quando uma devolução for efetuada o usuário será notificado no *e-mail* cadastrado, conforme figura 33.

Figure 33: E-mail do GEIFBA notificando a devolução do objeto emprestado. Fonte: próprio autor.

- Empréstimo efetuado: Quando um empréstimo é iniciado o solicitante recebe em seu e-mail cadastrado a sinalização, conforme figura 34.

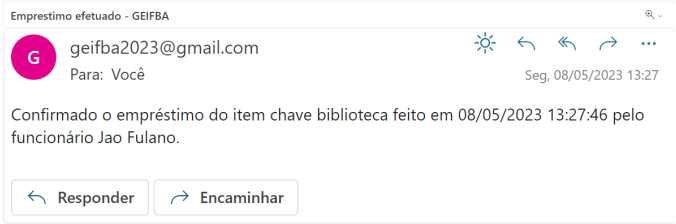

Figure 34: E-mail do GEIFBA notificando do empréstimos efetuado. Fonte: próprio autor.

Além dos e-mails acima, o sistema notifica caso o usuário esqueça a senha e deseje uma nova senha de acesso, basta clicar em esqueci a senha conforme figura 35. Depois será exibida a tela de recuperação de senha, onde deve ser informado o usuário conforme figura 36. Será enviada nova senha para o e-mail cadastrado, conforme figura 37.

H. Banco de dados

No projeto foi escolhido o PostgreSQL, banco de dados objeto relacional de código aberto, gratuito que estende a linguagem SQL e é bastante usado pelo mercado.

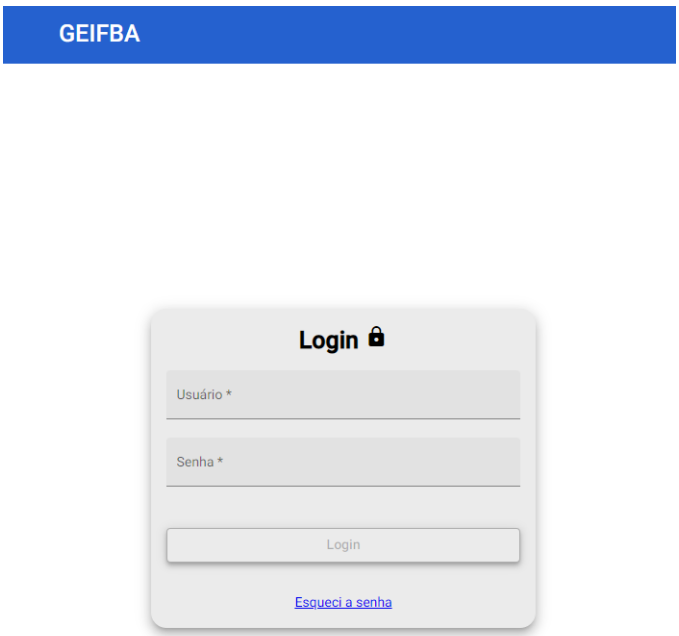

Figure 35: Tela esqueci a senha do GEIFBA. Fonte: próprio autor.

Figure 36: Tela recuperar a senha do GEIFBA. Fonte: próprio autor.

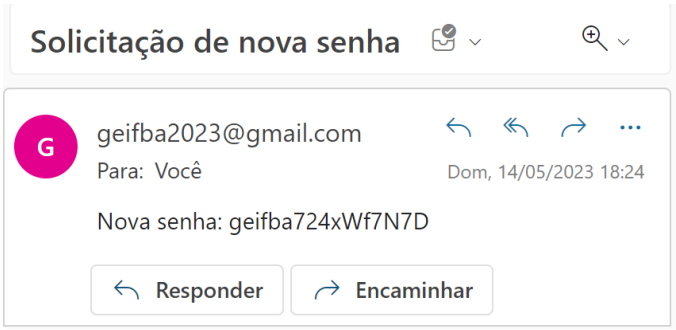

Figure 37: E-mail de solicitação de senha do GEIFBA. Fonte: próprio autor.

A modelagem foi realizada conforme figura 38, sendo a tabela empréstimo a maior agregadora já que possui a união de item, pessoa e usuário. A tabela pessoa e usuário possuem uma relação de dependência, pois a existência de um usuário requer a existência prévia de uma pessoa. Por outro lado, o item representa o objeto a ser emprestado, incluindo seus atributos.

I. Ferramenta para implementação, implantação e testes

No desenvolvimento foi usado o IntelliJ IDEA que é um ambiente de desenvolvimento integrado bastante usado no desenvolvimento de aplicações em várias linguagens.

Foi usado o pgAdmin na administração do banco de dados, trata-se de um *Open Source* administrador e plataforma de desenvolvimento do PostgreSQL.

No mapeamento das rotas que foram criadas foi utilizado o Swagger que permite descrever a estrutura das APIs para que as máquinas possam lê-las, criando uma documentação de forma automática e padronizada. É uma linguagem de descrição de interface que descreve APIs RESTful expressas usando JSON, conforme apresentado a partir da figura 39 até a figura 42 com a representação das rotas do sistema.

Durante o desenvolvimento do *backend* o Postman foi bastante usado, trata-se de uma API Client que permite salvar solicitações HTTP e HTTPs ler suas respostas. Durante todo o processo de desenvolvimento ocorreu o uso da *collection* com as requisições do sistema, outro uso feito no Postman foram os *scripts* de testes conforme figura 44, que a medida que o sistema foi sendo desenvolvido viabilizou que as funcionalidades implementadas pudessem ser checadas a cada ajuste conforme figuras 43 e 45, tendo uma visão ampla do que era impactado na aplicação.

J. Tecnologias Utilizadas

Foi desenvolvida uma aplicação Web, a solução permite que o acesso seja diretamente no browser ou navegador não necessitando de nenhuma instalação na máquina do usuário.

- **Angular**: Como interface gráfica da aplicação foi utilizado este framework que otimiza as aplicações das páginas web através do conceito de componentes, facilitando o desenvolvimento e consequentemente

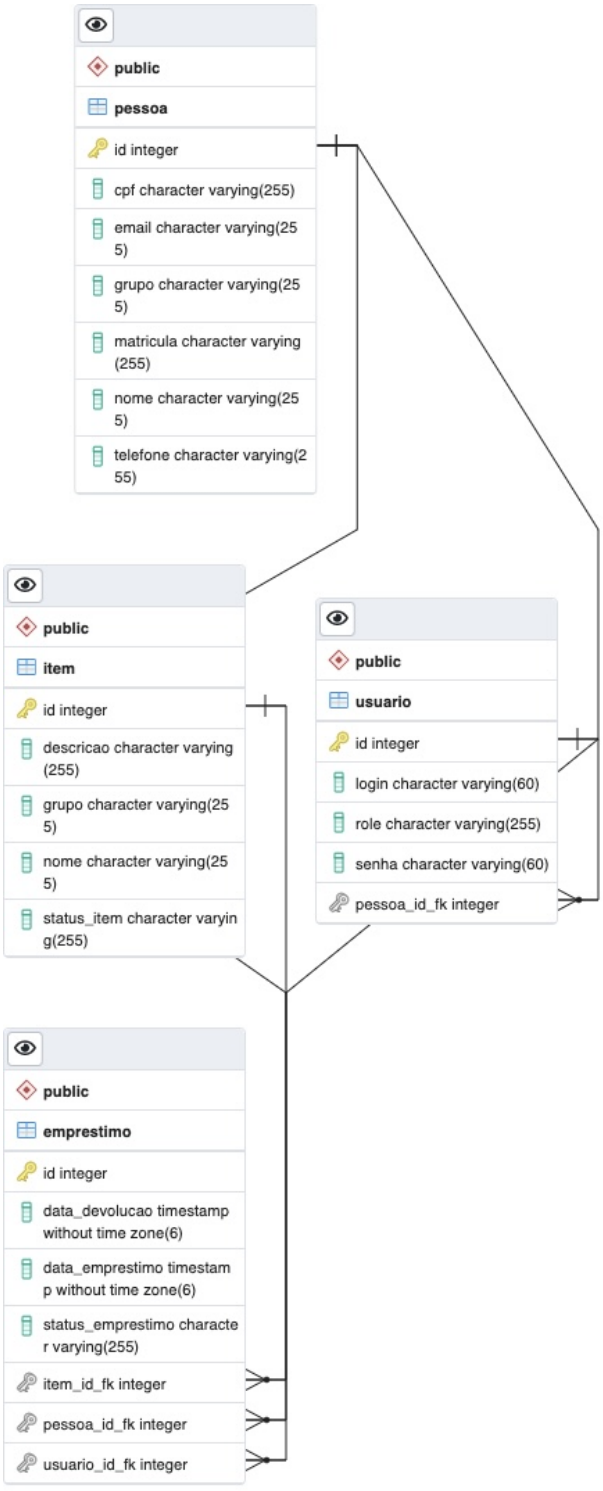

Figure 38: Modelo de entidade de relacionamento GEIFBA. Fonte: próprio autor.

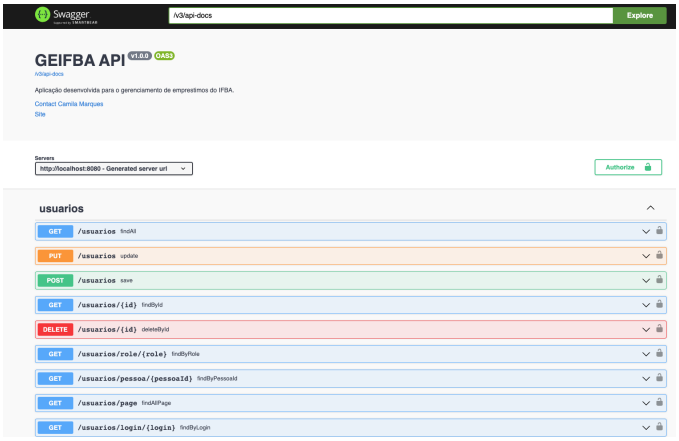

Figure 39: Swagger rota usuários. Fonte: próprio autor.

Figure 40: Swagger rota pessoas. Fonte: próprio autor.

aumentando a produtividade. Em estudo realizado Bagliotti [22] concluiu que o desenvolvimento com o *framework* Angular seja em pequenos ou em gigantescos projetos torna o trabalho mais prático, eficiente e garante os esforços na implementação dos requisitos oferecida pela base do *framework* devido a reusabilidade que é proporcionada, reduzindo o tempo de desenvolvimento e facilitando as manutenções e correções.

- **Java**: Foi utilizado no *backend* do sistema, trata-se de uma linguagem orientada a objetos que foi criada em 1991 pela *Sun Microsystems*, tendo como seu objetivo-chave ser capaz de escrever programas

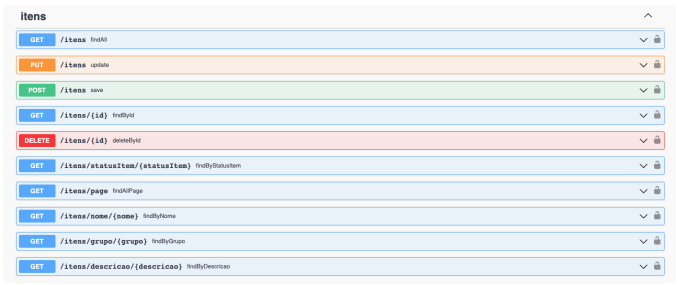

Figure 41: Swagger rota itens. Fonte: próprio autor.

Figure 42: Swagger rotas empréstimos e auth. Fonte: próprio autor.

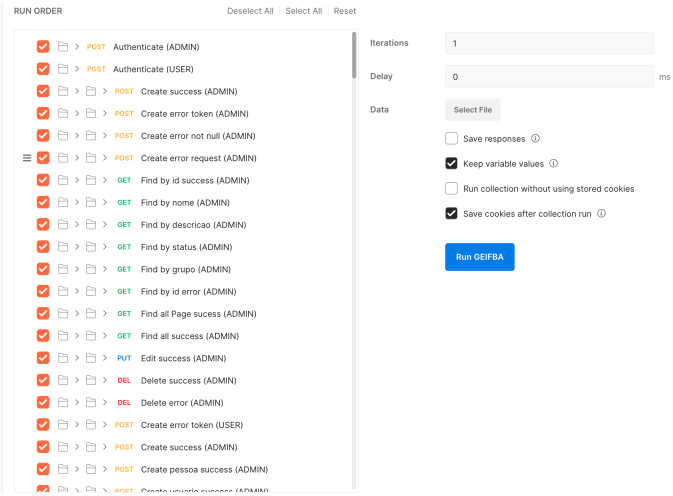

Figure 43: Postman testes GEIFBA. Fonte: próprio autor.

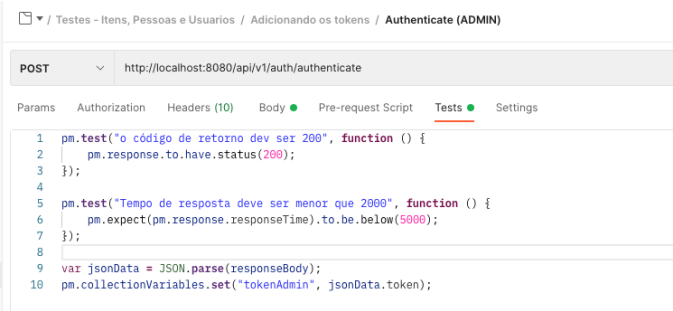

Figure 44: Postman exemplo de script teste autenticação GEIFBA. Fonte: próprio autor.

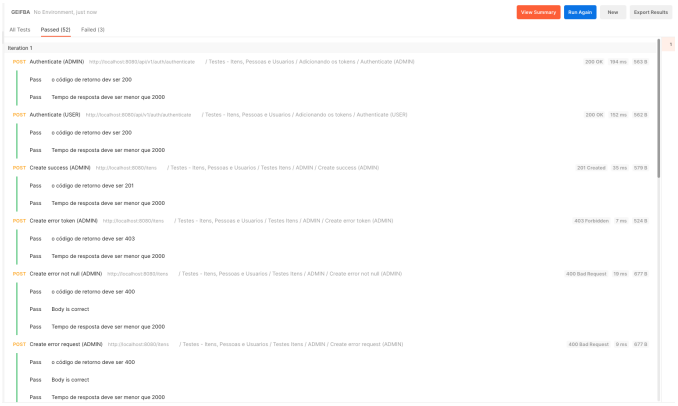

Figure 45: Postman testes executados GEIFBA. Fonte: próprio autor.

a serem executados em uma grande variedade de sistemas computacionais e dispositivos controlados por computador. Isso às vezes é chamado de "escreva uma vez, execute em qualquer lugar". [23]

- **Spring:** *Framework* que torna mais fácil criar aplicações autosuficientes e robustas, suas facilidade se dá principalmente na etapa de configuração do projeto com o *Spring Initializr* a criação do projeto é simplificada poupando tempo e aplicando padrão de arquitetura [24]. O uso do *Spring* facilitou a implementação da aplicação, pois por meio das anotações foi possível utilizar vários serviços, a exemplo do *SimpleMailMessage* que usado na geração dos dados de envio do e-mail.
- **JWT:** *JSON Web Token* é um padrão aberto que define uma maneira compacta e independente de transmitir informações com segurança entre as partes como um objeto JSON [25]. O acesso da aplicação é feito através de token definido que norteia as permissões através do *login*, tipo de perfil e período em que o usuário pode ficar logado.

K. Implantação do software

O processo de implantação é uma etapa muito importante do desenvolvimento de software. O processo de *deploy* foi feito usando *Cloud Computing* ou Computação em Nuvem com o provedor *Amazon Web Services (AWS)*. O serviço escolhido de hospedagem do *backend* da aplicação foi o *Amazon EC*2 que possui capacidade de computação confiável e escalável sob demanda, o provedor promete compromisso de *SLA* de 99,99% de disponibilidade [26].

No *frontend* foi utilizado o *Amazon Simple Storage Service (Amazon S3)* que é um sistema de armazenamento de objetos que oferece escalabilidade, disponibilidade de dados, segurança e performance. A aplicação em angular foi armazenada em um *bucket* do S3 que é um contêiner de objetos do provedor AWS.

V. Estudo Preliminar de Validação GEIFBA

O objetivo deste estudo foi o de ter uma percepção inicial sobre a visão dos usuários utilizando o sistema, conforme

detalhado a seguir:

1) Inicialmente foi feito um script com a descrição do que deveria ser feito:
    - Perfil User: Solicitar empréstimo, cancelar pedido de empréstimo, visualizar objetos, recuperar senha, entrar com nova senha que recebeu no e-mail e por fim alterar senha.
    - Perfil Admin: Cadastrar nova pessoa, cadastrar usuário, solicitar empréstimo para seu usuário, aprovar empréstimo, solicitar empréstimo selecionando um outro usuário e por fim cadastrar objeto.
2) Posteriormente, foram cadastrados dez usuários no sistema, sendo divididos entre os dois perfis acima descritos e foi enviado o *login* e senha do sistema juntamente com o respectivo *script*, juntamente com o link da pesquisa online a ser realizada no final. O prazo estipulado de finalização foi de três dias e o perfil dos participantes da pesquisa foi o de estudante do IFBA.

A. *Resultados*

Ao final do prazo foram coletados os resultados da pesquisa.

1) Referente a navegabilidade do sistema 9 pessoas classificaram como 5 excelente e uma pessoa classificou com 4, conforme imagem 46.

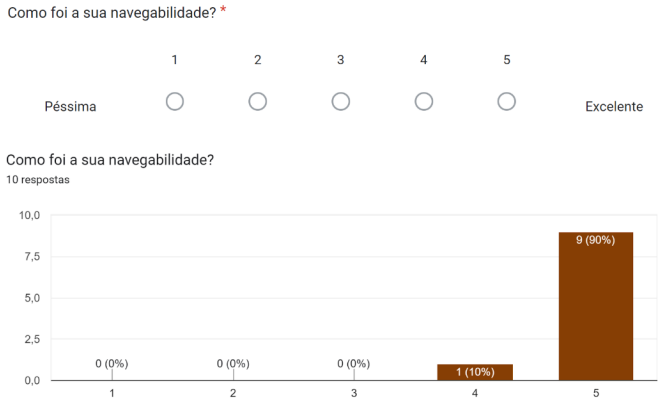

Figure 46: Pergunta sobre navegabilidade do GEIFBA. Fonte: próprio autor.

2) Referente ao compreensão do sistema 9 pessoas classificaram como 5 excelente e uma pessoa classificou com 4, conforme imagem 47.
3) Sobre a utilidade do sistema foi questionado se acreditava-se que o sistema seria útil para os envolvidos como substituto a forma manual que é feita hoje, todos os questionados responderam que sim, conforme imagem 48.
4) Foi solicitado que classificasse o sistema de maneira geral com a nota de 1 a 5 e todos responderam 5 excelente, conforme imagem 49.
5) Foi questionado o que os usuários do sistema mais gostam e foram coletadas as respostas abaixo:

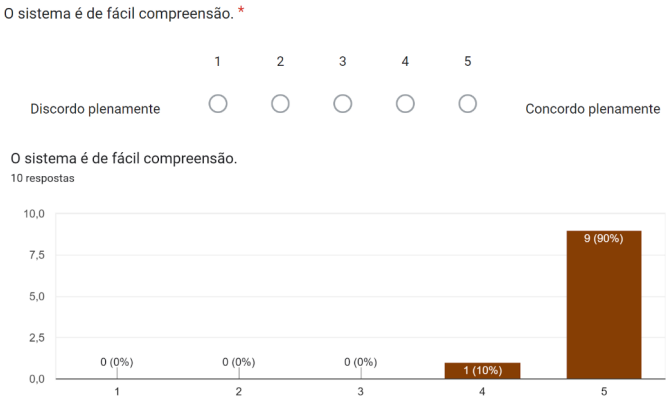

Figure 47: Pergunta sobre compreensão sobre o GEIFBA. Fonte: próprio autor.

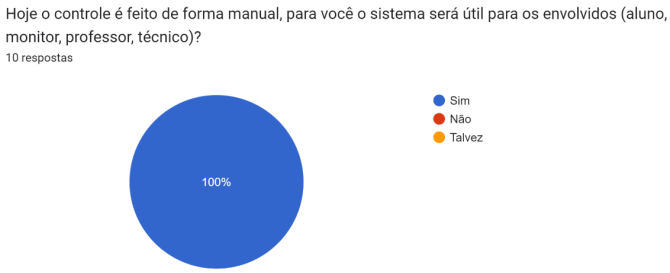

Figure 48: Pergunta sobre utilidade do GEIFBA. Fonte: próprio autor.

- Gostei do recibo de locação e entrega ser feito de forma eletrônica (por e-mail);
- A possibilidade de saber quem está com o objeto que você precisa;
- Ser fácil de navegar;
- O fato de está bem sucinto;
- Tudo;
- O sistema é bem auto explicativo e acho que será bastante funcional e facilitará muito tanto o registro de empréstimos, como o acompan-

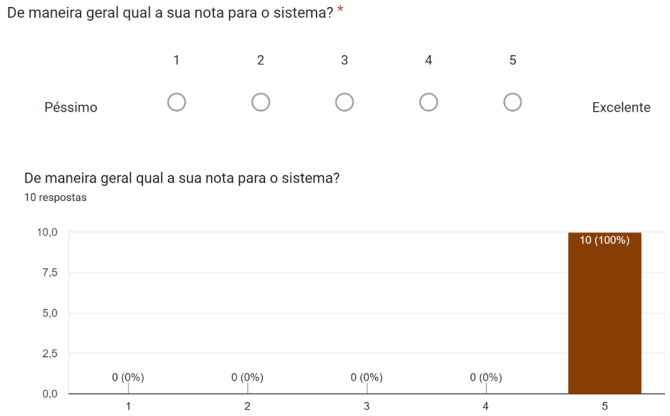

Figure 49: Pergunta sobre nota geral do GEIFBA. Fonte: próprio autor.

hamento deles;
- Simplicidade e responsividade do sistema. A usabilidade foi bastante fluida, sem qualquer tipo de erros e lentidões. Os menus e opções são bastante intuitivos;
- A facilidade de encontrar o que preciso e de ter visibilidade do que está emprestado;
- O sistema tem um ambiente limpo e de fácil compreensão, de maneira geral é bem fluido e intuitivo, apresenta *feedbacks* lúdicos e tem uma *interface* pensada no trabalho continuo;
- O sistema é fluido, tem os *toasts* de respostas em todas as ações, não há travamentos, é simples e direto com todas as funcionalidades.

6) Por fim foi questionado sobre o que pode ser melhorado no sistema, com as respostas abaixo:
   - Acredito que um APP ajudará bastante no uso do sistema pelo celular;
   - Apenas algumas práticas de UI/UX que estão faltando, mas nada que impeça o entendimento do sistema;
   - Talvez um filtro facilite a localização de algum nome mais fácil na tela de empréstimo, usuários e pessoas no modo ADMIN;
   - Na tabela de objetos pode ser inserido uma coluna com "detalhar" nos itens que estão com status emprestado. Por exemplo, ao clicar em detalhar, um modal se abre com as informações do empréstimo (usuário, data...);
   - Adição de um campo para reportar a perda do objeto em posse;
   - Existir uma tela de acompanhamento por período do fluxo de empréstimos, só por forma de análise e registro;
   - Informar no sistema a todos, quem está de posse dos objetos que não venham a comprometer a segurança das instalações ou causar prejuízos financeiros. Ex: Em caso de esquecimento na devolução, o usuário de posse de uma chave do banheiro poderia ser contatado mais rapidamente pelo usuário que esteja em necessidade do objeto;
   - Nos objetos ter a opção de cadastro por número de série do item;
   - Responsividade;
   - O design, é um pouco rústico mas nada que atrapalhe a utilização.

VI. Conclusão e Trabalhos Futuros

O objetivo deste trabalho foi o desenvolvimento de uma aplicação web que pudesse realizar o controle mais eficiente e seguro dos empréstimos realizados no IFBA, tendo em vista que atualmente o controle é feito através de papel gerando vários riscos para o processo. Na criação do projeto o intuito foi o de mesmo informatizando as tarefas que envolvem o processo de empréstimo não tornar esse processo complexo, com *interface* simples e amigável diminuindo a complexidade de acesso e com isso facilitando a adesão dos usuários.

Diversas tecnologias foram utilizadas ao longo de todo o processo de desenvolvimento, como destaque o os *frameworks* *Spring* e Angular. Esses *frameworks* foram muito importantes, pois diminuíram a complexidade da implementação e construção da API.

Foi apresentado ao gestor de TI do IFBA detalhes da estrutura e o sistema em funcionamento, juntamente com o seu manual explicando todos os fluxos e funcionalidades de acordo com cada tipo de perfil. O *feedback* obtido foi positivo, tendo como vantagem a possibilidade de inclusão e gestão de diversos objetos, além da possibilidade de inclusão do empréstimo pelo próprio servidor com o perfil administrador ou pelo usuário que pode verificar os itens disponíveis de acordo com o seu perfil e solicitá-los. O sistema está sendo testado com objetos do setor de audiovisual para que se entenda na prática sobre esse fluxo e os usuários possam testar e se familiarizar com a ferramenta.

Como trabalhos futuros espera-se tornar o sistema totalmente responsivo se adequando aos diversos aparelhos celular, inclusão de filtro em todas as telas para facilitar a localização de informações, possibilidade de inativação de objeto por perda ou dano, geração de relatório de empréstimo por data selecionada ou por usuário.

Além disso foi levantada a possibilidade de realização do cadastro de usuários do sistema através de integração do sistema com a base de usuários que é utilizado no Sistema Unificado de Administração Pública (SUAP). Outra possibilidade é transformar o sistema em um módulo do SUAP que todos os alunos do campus possuem acesso, já que trata-se de um portal onde notas são lançadas, assiduidade, grade curricular, pedido de matrícula, entre outras funções.

References

[1] C. S. ALBANO and G. ZAMBERLAN, "Área temática: Gestão tecnológica. título: Utilização da tecnologia da informação em organizações públicas: Um estudo comparativo entre organizações municipais."

[2] R. Pressman and B. Maxim, "Engenharia de software: uma abordagem profissional-tradução: João eduardo nóbrega tortello," *Revisão técnica: Reginaldo Arakaki, Julio Arakaki, Renato Manzan de Andrade.–8 ed. Porto Alegre: Amgh Editora*, 2016.

[3] "Ifba o instituto," Navegador Google Chrome, acesso em: 28 de fevereiro 2023. [Online]. Available: https://portal.ifba.edu.br/acessoainformacao/institucional

[4] I. SOMMERVILLE, *Engenharia de Software*, 9th ed. Pearson Addison-Wesley, 2011.

[5] M. T. Valente, "Engenharia de software moderna. 2020," *Available on*, 2020.

[6] C. E. Vazquez and G. S. Simões, *Engenharia de Requisitos: software orientado ao negócio*. Brasport, 2016.

[7] G. Booch, J. Jacobson, and J. Rumbaugh, *Uml-Guia do usuário, tradução da segunda edição*. Elsevier Brasil, 2016.

[8] G. T. Guedes, "Uml 2," *Uma Abordagem Prática", São Paulo, Novatec*, p. 32, 2009.

[9] "O que é api restful?" Navegador Google Chrome, acesso em: 21 de abril de 2023. [Online]. Available: https://aws.amazon.com/pt/what-is/restful-api/

[10] M. Brito, *Spring Boot Da API REST aos Microservices*. Project Decoder, 2022.

[11] "O que é api rest?" Navegador Google Chrome, acesso em: 21 de abril de 2023. [Online]. Available: https://www.redhat.com/pt-br/topics/api/what-is-a-rest-api

[12] E. A. G. Junior, R. D. Rocha, and R. de Souza Maciel, "Desenvolvimento de api rest com spring boot," *Revista Rios*, vol. 15, no. 29, pp. 499–525, 2021.

[13] "O que é a computação em nuvem?" Navegador Google Chrome, acesso em: 15 de abril de 2023. [Online]. Available: https://aws.amazon.com/pt/what-is-cloud-computing/

[14] F. R. Sousa, L. O. Moreira, and J. C. Machado, "Computação em nuvem: Conceitos, tecnologias, aplicações e desafios," *II Escola Regional de Computação Ceará, Maranhão e Piauí (ERCEMAPI)*, pp. 150–175, 2009.

[15] M. Veras, "Cloud computing," *Nova Arquitetura da TI*, vol. 3, 2012.

[16] A. Carissimi, "Desmistificando a computação em nuvem," *ROSE, Cesar de*, pp. 3–24, 2015.

[17] H. P. Borges, J. N. d. SOUZA, B. Schulze, and A. R. Mury, "Computação em nuvem," *Instituto Federal de educação, Ciência e Tecnologia do Maranhão*, 2011.

[18] "Quais são os diferentes tipos de serviços de computação em nuvem?" Navegador Google Chrome, acesso em: 15 de abril de 2023. [Online]. Available: https://azure.microsoft.com/pt-br/resources/cloud-computing-dictionary/types-of-cloud-computing

[19] "O que são nuvens públicas, privadas e híbridas?" Navegador Google Chrome, acesso em: 19 de abril de 2023. [Online]. Available: https://azure.microsoft.com/pt-br/resources/cloud-computing-dictionary/types-of-cloud-computing

[20] K. S. Borges, "Bibliotecas digitais: um sistema para o controle de empréstimos e devoluções de objetos digitais," Master's thesis, Pontifícia Universidade Católica do Rio Grande do Sul, 2000.

[21] G. H. L. S. Zaffani, "Sistema web para o gerenciamento dos materiais de laboratório de computação," B.S. thesis, Universidade Tecnológica Federal do Paraná, 2020.

[22] I. R. Bagliotti and D. Gibertoni, "Reusabilidade no desenvolvimento de um sistema web utilizando o framework angular," *Revista Interface Tecnológica*, vol. 17, no. 1, pp. 192–204, 2020.

[23] H. Deitel, C. P. Java, D. HM, and D. PJ, "trad. carlos arthur. lang lisboa.–," *Java como programar, 4º edição,-Porto Alegre, Bookman*, 2003.

[24] "Why spring?" Navegador Google Chrome, acesso em: 03 de abril de 2023. [Online]. Available: https://spring.io/

[25] "Introduction to json web tokens," Navegador Google Chrome, acesso em: 03 de abril de 2023. [Online]. Available: https://jwt.io/introduction

[26] "Amazon ec2," Navegador Google Chrome, acesso em: 15 de abril de 2023. [Online]. Available: https://aws.amazon.com/pt/ec2