INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DO RIO GRANDE DO SUL
CAMPUS RESTINGA

TIAGO GONÇALVES FIGUEIRA

SISTEMA DE INTRANET PARA SERVIÇOS INTERNOS DO CAMPUS RESTINGA

Porto Alegre
2022

TIAGO GONÇALVES FIGUEIRA

SISTEMA DE INTRANET PARA SERVIÇOS INTERNOS DO CAMPUS RESTINGA

Trabalho de conclusão de curso apresentado ao Curso de análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul, como requisito parcial para a Obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Gleison Samuel do Nascimento

Porto Alegre
2022

SISTEMA DE INTRANET PARA SERVIÇOS INTERNOS DO CAMPUS RESTINGA

Trabalho de conclusão de curso apresentado ao Curso de análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul, como requisito parcial para a Obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Gleison Samuel do Nascimento

Aprovado em ___, _____

_____

Nome do Orientador

_____

Membro da Banca – Professor Iuri Albandes Cunha Gomes – IFRS – Campus Restinga

_____

Membro da Banca – Professor Jean Carlo Hamerski – IFRS – Campus Restinga

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO SUL
Reitor: Prof. Júlio Xandro Heck
Pró-Reitora de Ensino: Prof. Lucas Coradini
Diretor-geral do *Campus* Restinga: Prof. Rudinei Müller
Coordenador do CST em Análise e Desenvolvimento de Sistemas: Prof. Iuri Albandes Cunha Gomes
Bibliotecária-chefe do *Campus* Restinga: Paula Porto Pedone

Dedico este trabalho a minha mãe, pilar da minha formação como ser humano. Também dedico à Mariana Farini, que sempre me apoiou nas conquistas e momentos complicados deste período de criação deste trabalho.

AGRADECIMENTOS

Os agradecimentos principais: primeiramente aos professores, que arduamente trabalharam para passar todo seu conhecimento auxiliando na minha jornada acadêmica; aos colaboradores do Campus IFRS - Restinga, que diariamente dão o seu máximo para fazer da instituição um lugar melhor para os frequentadores; aos meus colegas de graduação por todas as atividades em conjunto e troca de conhecimento durante as aulas, a minha amada mãe Maria Cristina, minha namorada Mariana Farini que me deram todo o suporte emocional durante este período e ao meu orientador Prof. Gleison S. do Nascimento, que foi peça ímpar de forma fundamental para a realização deste trabalho de conclusão.

Agradecimentos especiais: são direcionados a minha mãe Maria Cristina e Mariana Farini, pois sem o apoio delas, essa jornada não seria alcançável.

“A melhor forma de prever o futuro é criá-lo.”

(Peter F. Drucker)

Resumo

Este trabalho de conclusão iniciou o desenvolvimento de um sistema de intranet para o IFRS - Campus Restinga, para a centralização das informações dos servidores e estudantes da unidade. A motivação central do sistema é criar uma rotina de trabalho mais leve para os servidores, centralizando as informações necessárias para o trabalho do cotidiano dos diferentes agentes da unidade. O objetivo é realizar a desoneração do trabalho dos funcionários, que diariamente realizam acessos a sistemas que estão espalhados por locais distintos, que precisam ser repetidamente consultados para visualizar informações em muitos casos redundantes entre diferentes sistemas e planilhas. A implantação do sistema unifica estas informações em um único ponto de acesso. Para delimitar o escopo deste projeto, um grupo de trabalho foi criado com setores relacionados, professores e setor de TI para levantamento de informações, fluxos de trabalho e definição de padrões para o funcionamento do sistema. Este grupo de trabalho se reúne semanalmente para discutir e decidir pontos cruciais do sistema. A partir desta rotina, foram criados diagramas de casos de uso das funcionalidades do sistema. Esses diagramas foram detalhados por este trabalho de conclusão, assim como foram criados os demais diagramas necessários para o início da implementação do sistema, como os diagramas de Entidade-Relacionamento e diagrama de classes. A partir dos diagramas iniciou-se a implementação do sistema para um conjunto inicial de casos de uso relacionados à Coordenadoria de Gestão de Pessoas. O sistema foi desenvolvido na arquitetura MVC, usando o framework Spring Boot (Java) para o Back-end da aplicação e o framework ReactJS (HTML, CSS e Javascript) para implementação do Front-end. Para a base de dados foi utilizado o sistema gerenciador de banco de dados MySQL. O presente trabalho de conclusão foi finalizado com a implantação do projeto em um servidor Web Tomcat 9, rodando em uma máquina virtual com Ubuntu 20.04, deixando online a primeira versão do sistema de intranet do Campus Restinga, que será continuada a partir de outros trabalhos de conclusão dos estudantes do curso.

Palavra-chave: Sistema web, intranet, Java, Spring Boot, ReactJS.

Abstract

This conclusion work started the development of an intranet system for IFRS - Campus Restinga, for the centralization of the information of the servers and students of the unit. The central motivation of the system is to create a lighter work routine for the servers, centralizing the information necessary for the daily work of the different agents of the unit. The objective is to relieve the work of employees, who daily access systems that are spread across different locations, which need to be repeatedly consulted to view information that is in many cases redundant between different systems and spreadsheets. The implementation of the system unifies this information in a single access point. To delimit the scope of this project, a working group was created with related sectors, teachers and the IT sector to gather information, work flows and define standards for the system's operation. This working group meets weekly to discuss and decide crucial points in the system. From this routine, diagrams of use cases of the system's functionalities were created. These diagrams were detailed in this conclusion work, as well as the other diagrams necessary for the beginning of the implementation of the system, such as the Entity-Relationship diagrams and the class diagram. From the diagrams, the implementation of the system began for an initial set of use cases related to the People Management Coordination. The system was developed in the MVC architecture, using the Spring Boot framework (Java) for the application's Backend and the ReactJS framework (HTML, CSS and Javascript) for the Frontend implementation. For the database, the MySQL database management system was used. The present conclusion work was concluded with the implementation of the project in a Tomcat 9 Web server, running in a virtual machine with Ubuntu 20.04, leaving online the first version of the Campus Restinga intranet system, which will be continued from other works of students' completion of the course.

Keywords: *Web system, intranet, Java, Spring Boot e React JS.*

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

CSS	Cascading Style Sheets
HTML	HyperText Markup Language
IFRS	Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul
IFSUL	Instituto Federal de Educação, Ciência e Tecnologia Sul-rio-grandense
MVC	Model-View-Controller
SIA	Sistema de Informações Acadêmicas
SIGAA	Sistema acadêmico
SIGPROJ	Sistema de Informação e Gestão de Projetos
SIGRH	Sistema Integrado de Gestão e Recursos Humanos
SQL	Structure Query Language
UML	Linguagem de Modelagem Unificada
CRUD	Create, Read, Update e Delete
GNU	General Public License
SIG	Sistema Integrado de Gestão
COTIL	COLÉGIO TÉCNICO DE LIMEIRA; DINF – DEPARTAMENTO DE INFORMÁTICA
PHP	Hypertext Processor
LDAP	Lightweight Directory Access Protocol
DOCKER	Contêiners de software
DEPLOY	Implantação de sistemas
MYSQL	Sistema gerenciador de banco de dados objeto relacional
PostgreSQL	Sistema gerenciador de banco de dados objeto relacional
SGBD	Sistema gerenciador de banco de dados
GNU	Sistema operacional tipo Unix
HTTP	Hypertext Transfer Protocol
API	Application Programming Interface
BUILD	Arquivo de construção

SUMÁRIO

1 INTRODUÇÃO

De acordo com Lawton (1995, p 20, apud Nunes, 2016) intranet é uma rede de computadores de acesso exclusivo de uma empresa ou corporação. A intranet é usada pelas empresas para armazenamento de informações e é um importante veículo de comunicação entre seus funcionários.

Uma intranet deve ser concebida de acordo com as necessidades da empresa ou da organização (ao nível dos serviços a implementar). Assim, a intranet não deve ser concebida só pelos analistas de tecnologia da empresa, mas de acordo com um projeto que tem em conta as necessidades de todas as partes que constituem a empresa (Muller; Nicolas, 12/08/2010).

Quando alguma informação dessa intranet é aberta a clientes ou fornecedores dessa empresa, essa rede passa a ser chamada de extranet. Se sua empresa tem uma intranet e seu fornecedor também e ambas essas redes privadas compartilham uma rede entre si, para facilitar pedidos, pagamentos e o que mais precisarem, essa rede compartilhada é conhecida como extranet. Ainda, se sua empresa abre uma parte de sua rede para contato com o cliente, ou permite uma interface de acesso dos fornecedores essa rede com ele é chamada de extranet (Assis; Pablo de, 16/04/2009).

Atualmente, o Campus Restinga do IFRS possui diversos sistemas para diferentes assuntos como, sistema de biblioteca (Pergamum), sistemas acadêmicos (SIA e SIGAA), sistema para registros de projetos de pesquisa e extensão (SIGProj), sistema para gestão de pessoas (SIGRH), entre outros. Cada sistema possui uma localização e base de dados própria. Então servidores e alunos necessitam acessar estes sistemas em sua rotina estudantil ou trabalhista, usando credenciais diferentes para autenticação e informações distintas que muitas vezes precisam ser consolidadas manualmente, já que tais sistemas não conversam entre si. Esta rotina, principalmente para os servidores do Campus, que diariamente necessitam realizar procedimentos como geração de documentos aos alunos, servidores, coordenações de curso, entre outros, torna os fluxos de trabalho cansativos e algumas vezes incorretos, pois trabalham com informações divergentes entre os diversos sistemas do Instituto.

Sendo assim, este trabalho de conclusão busca implantar um sistema de intranet web para a centralização de informações e serviços do Campus Restinga. O sistema tem o objetivo de facilitar o uso dos serviços diários do Campus, onde o usuário realizará uma única autenticação, e obtendo as informações consolidadas das diferentes bases de dados dos diversos sistemas utilizados no Campus.

As implementações realizadas neste período foram focadas na parte administrativa focando nas tarefas que os funcionários farão diariamente, como cadastro de setores, cadastro de cargos, a visualização de dados de estudantes em conjunto com sua edição, os

dados de servidores também podem ser manipulados e acessar demais informações, como formações escolares, formações complementares e progressões que é um módulo que ainda não estava automatizado para ser manipulado.

Ainda falta realizar a inserção de mais dados auxiliares que completam os dados de servidores, que serão adicionados futuramente no prosseguimento do trabalho por outro estudante, como também a inserção do módulo de usuários para autenticação da intranet. A intranet teve 8 casos de uso implementados em um total de 20, o que indica 40% de implementação.

Um desafio para o sucesso deste projeto é a adesão dos usuários ao serviço, pois intranets costumam entregar informações sobre a empresa de forma totalmente fria, onde em alguns casos são só páginas web que expõe o aglomerado de sistemas que a empresa ou instituição possui, para que o funcionário utilize sem mais atrativos como informações e novidades sobre o Campus. Sendo assim, a intranet do Campus deve abordar um cenário atrativo para seu uso, onde todos se sintam bem ao acessar e se desfaçam de rotinas onerosas de trabalho, aumentando a agilidade nos processos internos.

1.1 Objetivo Geral

Iniciar o desenvolvimento de um sistema de intranet para o Campus Restinga, através da implementação de funcionalidades para centralização de informações relativas à Coordenadoria de Gestão de Pessoas da unidade.

1.2 Objetivos Específicos

1. Criar o modelo de dados (entidade-relacionamento) inicial do sistema de intranet do Campus Restinga;
2. Definir e documentar o módulo de usuários do sistema de intranet do Campus Restinga;
3. Implementar as funcionalidades de acesso a informações pessoais e funcionais dos servidores para uso da coordenadoria de gestão de pessoas do Campus e outros setores vinculados;
4. Configurar o servidor web e instalar a primeira versão do sistema de intranet.

O restante deste trabalho de conclusão de curso está organizado em uma estrutura, que segue o seguinte formato: o capítulo 2 contempla os trabalhos relacionados; o capítulo 3 apresenta a solução conceitual; o capítulo 4 cita o modelo conceitual, o capítulo 5 contempla o desenvolvimento realizado; o capítulo 6 é a conclusão deste trabalho; seguido das referências e os apêndices que são todos os casos de uso deste trabalho.

2 TRABALHOS RELACIONADOS

Para contextualização do assunto que o trabalho irá abordar, é importante uma melhor compreensão sobre ele. Desta forma um estudo foi executado para encontrar referências bibliográficas e assim, incrementar de forma sintética neste trabalho todo o estudo realizado. A revisão da literatura buscou trabalhos relacionados sobre intranets, procurando relacionar estes trabalhos com a questão de pesquisa elaborada para este trabalho de conclusão. A busca foi realizada no sistema de pesquisa Google Acadêmico. Através desta plataforma foram encontrados três trabalhos que pertencem ao mesmo contexto que a intranet para o IFRS Campus Restinga.

2.1 INTRANET INSTITUCIONAL E SISTEMA DE PEDIDOS

O objetivo deste trabalho é a criação de um framework em PHP, onde ele disponibiliza duas aplicações para os servidores do IFSul (MELLO, J; SICCA, W, 2012). Os sistemas são uma intranet e um gerenciador de pedidos de serviço. Nos concentramos na análise da Intranet, por se tratar do foco desse trabalho de conclusão.

A intranet se comporta como um ambiente institucional, com alguns módulos que seus servidores possam utilizar. Possui módulos para gerenciamento de documentos e formulários, criação de pedidos de serviço para a infraestrutura em geral do Campus, integração com o SIGA-ADM e processamento do controle de frequência dos servidores.

O framework foi gerado para reutilização de código no Campus, onde será facilitado a integração entre os sistemas e inclusão de melhorias de modo mais dinâmico. O mesmo, fará a desoneração de trabalho dos servidores, buscando em cada módulo uma melhor estrutura de trabalho e facilitação de uso.

2.2 PROJETO BULLYING E CYBERBULLYING LITORAL DO PARANÁ

Para combater o bullying e cyberbullying nas escolas e demais locais da região envolvida, o projeto entrega um banco de dados com informações públicas com estatísticas deste projeto, para gerar, sistematizar e refletir sobre os impactos que o assunto provoca, facilitando uma estratégia de combate. O sistema basta ter internet e o acesso padrão do sistema para informar ou analisar dados. (GONÇALVES, A. 2015).

O bullying, prática criminosa, é um problema que a sociedade vem sofrendo de uns anos para cá, como forma explicitamente citada, pois as práticas que envolvem o bullying acontecem desde sempre, infelizmente a mídia em cima do assunto só tem tomada tamanha proporção na mesma época em que ela teve esta denominação. O Cyberbullying nada mais é do que a extensão do bullying, onde ela tem como vantagem a “distância” entre o praticante

e a vítima, pois em muitos casos o praticante não revela sua identidade real, utilizando perfis falsos. Então, este trabalho relacionado tem como objetivo combater ambas as práticas, revelando sensos próprios baseados nas pesquisas que foram realizadas em escolas e os demais que já responderam e tem os mesmos publicados na intranet.

A intranet foi implementada na linguagem de programação PHP. O front-end foi desenvolvido com HTML e CSS, para que seja apresentado ao usuário final os dados criados foi utilizado o MYSQL.

2.3 SISTEMA DE INTRANET - WATS

O objetivo deste sistema é realizar a centralização dos serviços internos que o COLÉGIO TÉCNICO DE LIMEIRA necessita. O sistema foi criado por alunos da Unicamp, acessando via login único através da autenticação LDAP, para que docentes e funcionários possam utilizar, facilitando o acesso dos usuários para os sistemas mais utilizados.

O Sistema desenvolvido utiliza tecnologias atuais para implantação, com disponibilização gratuita e potencial escalabilidade ao seu código-fonte. A implementação usa na infraestrutura um servidor novo CentOS Linux e a ferramenta Docker otimização de deploys desonerando a distribuição das versões do sistema. A interface do usuário (sistema web) foi baseada no framework Bootstrap, enquanto o back-end utiliza o framework Django e linguagem de programação Python, com persistência dos dados no sistema gerenciados de banco de dados PostgreSQL.

O WATZ trouxe consigo além da inovação tecnológica na estrutura que foi implementada, também acarretou o desenvolvimento de processos internos, trazendo uma integração mais objetiva ao ambiente, habilitando também a inclusão de novos sistemas e aumentou a qualidade interna da jornada profissional. (GODOY, A.; UCELLI, J.; ROCHA, T.; PAIVA, W., 2019).

2.4 Comparativo dos trabalhos

A tabela 1 apresenta-se uma tabela comparativa entre os trabalhos, mostrando as diferenças das tecnologias utilizadas na conclusão deles, onde podemos identificar a repetição de algumas informações em alguns pontos como a disponibilidade, linguagem e sistemas externos relacionados.

Tabela 1 - Comparativo entre os sistemas

Padrões	Intranet Campus Restinga	Intranet Institucional e sistema de pedidos	Projeto Bullying e Cyberbullying litoral do Paraná	WATS
Arquitetura de Sistema	Web	Web	Web	Web
Banco de dados	Mysql	PostgreSQL	Mysql	PostgreSQL
Linguagem de Programação	Java	PHP	PHP	Python
Arquitetura de Desenvolvimento	MVC	Não Informado	Não Informado	Não Informado
Framework Front-end	Bootstrap	Não Informado	Não Informado	Bootstrap
Sistemas externos relacionados	Sim	Sim	Não	Sim

Fonte: Próprio autor

3 SOLUÇÃO CONCEITUAL

Com os tópicos levantados anteriormente, foi possível mapear e decidir quais ferramentas serão utilizadas para efetivar a criação da intranet para o Campus Restinga. Será criado um sistema web que trará acessibilidade dos dados via navegador, tanto para computadores portáteis, smartphone ou tablets.

Para codificação do sistema será usada a linguagem de programação Java em conjunto com o framework Spring Boot, que tem distribuição livre e boa aceitação no mercado por permitir a implementação de uma infraestrutura de serviços que pode ser usada para construção de sistemas multiplataformas (por exemplo, os mesmos serviços podem ser empregados para desenvolvimento web e para a construção de um aplicativo de celular).

Para o armazenamento de dados, será utilizado o sistema gerenciador de banco de dados MySQL, através do Mysql Workbench, por ter distribuição gratuita e implementação simples, que garante maior desempenho nas transações SQL, sendo um banco muito usado para implementações de sistemas web.

Para a interação do usuário (front-end) será utilizado o HTML, como linguagem de hipertexto, o Bootstrap como framework baseado em CSS e o framework ReactJS para implementações de lógica em JavaScript. Da mesma forma que o Spring Boot para o back-end, o framework ReactJS permite o uso da mesma ferramenta para desenvolvimento web e aplicativos de celular, possibilitando assim a implementação futura uma interface do sistema de intranet para celulares.

3.1 Requisitos

Antigamente dizia-se que requisitos de software eram sinônimos de funções, ou seja, tudo que o software deveria fazer funcionalmente. No entanto, atualmente assumiu-se que os requisitos de software é muito mais do que apenas funções. Requisitos são, além de funções, objetivos, propriedades, restrições que o sistema deve possuir para satisfazer contratos, padrões ou especificações de acordo com o(s) usuário(s). De forma mais geral um requisito é uma condição necessária para satisfazer um objetivo.

Portanto, um requisito é um aspecto que o sistema proposto deve fazer ou uma restrição no desenvolvimento do sistema, Vale ressaltar que em ambos os casos devemos sempre contribuir para resolver os problemas do cliente e não o que o programador ou um arquiteto deseja. Dessa forma, o conjunto dos requisitos como um todo representa um acordo negociado entre todas as partes interessadas no sistema. Isso também não significa que o programador, arquiteto ou um analista bem entendido no assunto de tecnologia não possam contribuir com sugestões e propostas que levem em conta o desejo do cliente (DevMedia, 2013).

3.1.1 Funcionais

Eles dizem respeito às funções e informações que o software deve possuir, ou seja, ao seu comportamento: a como ele deve reagir a entradas específicas, como ele irá se portar em determinadas situações, e até mesmo declarar o que o sistema não deve fazer (Rossetti, Micaela, L. 17 jun. 2021).

3.1.2 Não funcionais

Eles, por sua vez, podem referir-se aos critérios que qualificam os requisitos funcionais: estão relacionados a qualidades específicas e restrições que o software deve atender, ou seja, não se referem a funcionalidades em si, mas fazem parte do escopo do produto (Rossetti, Micaela, L. 17 jun. 2021).

3.2 Padrões de projeto

Conceito extremamente citado em disciplinas de Engenharia de software e que hoje regem muitos projetos empresariais para desenvolvimento de sistemas. Seguir um padrão de projeto é uma medida que agrega muito valor ao desenvolvimento do projeto, onde se foca pontos de extensibilidade e reusabilidade de código. Através dos padrões de projeto também é possível detalhar melhor os requisitos do projeto, para minimizar os problemas de implementação do sistema. (LEITE, A., 2005)

3.2.1 MVC

Para este projeto será empregado o padrão MVC. Esse padrão é utilizado em muitos projetos devido a sua arquitetura, que possibilita a divisão do projeto em camadas muito bem definidas. Cada uma delas, executa uma tarefa distinta no sistema, conforme a Figura 1 logo abaixo (DevMedia, 2013).

A utilização do padrão MVC traz como benefício o isolamento das regras de negócios da lógica de apresentação, que é a interface com o usuário. Isto possibilita a existência de várias interfaces com o usuário que podem ser modificadas sem a necessidade de alterar as regras de negócios, proporcionando muito mais flexibilidade e oportunidades de reuso de código. Uma das características de um padrão de projeto é poder aplicá-lo em sistemas distintos.

O padrão MVC pode ser utilizado em vários tipos de projetos como, por exemplo, desktop, web e mobile.

A comunicação entre interfaces e regras de negócios é definida através de um controlador, que separa as camadas. Quando um evento é executado na interface gráfica, como um clique em um botão, a interface se comunicará com o controlador, que por sua vez se comunica com as regras de negócios. Desta forma, nosso projeto terá três camadas:

- Model – Nessa camada são implementadas as regras de negócio, além dos serviços de acesso e manipulação dos dados no software.
- View – É a camada responsável pela construção da interface gráfica do usuário, onde são apresentadas as informações do sistema aos usuários.
- Controller – Camada responsável pela comunicação entre a interface gráfica (View) e a camada de regras de negócio (Model), desencadeando a troca de dados entre as camadas citadas.

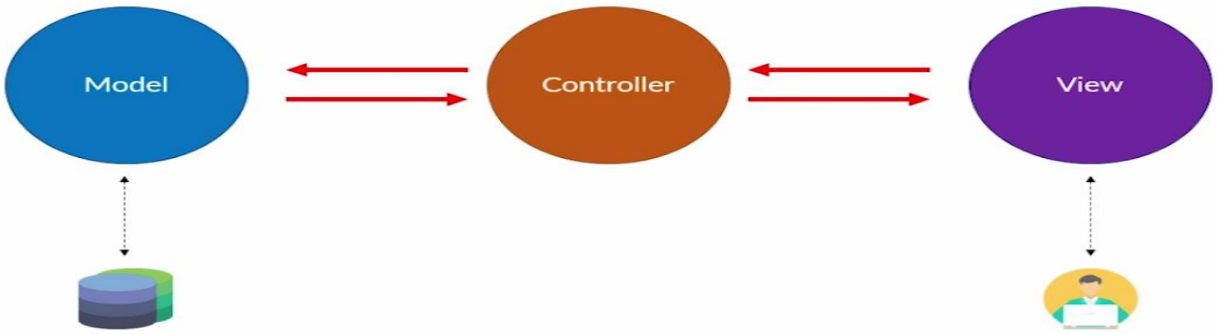

Figura 1: Padrão MVC.
Fonte: DevMedia.

3.3 Java

Para implementação das camadas de modelo e controle foi escolhida a linguagem de programação Java, definida através do grupo de trabalho constituído no Campus Restinga para delimitar e definir as funcionalidades do sistema de intranet da instituição. A escolha foi feita por ser uma linguagem multiplataforma, gratuita e ser a linguagem empregada no curso de Análise e Desenvolvimento de Sistemas, possibilitando trabalhar a implementação do sistema através dos estudantes do curso. O Java atualmente está na versão estável 11, possui diversos pontos fortes:

1. A orientação a objetos, que é um método de codificação que implica na utilização de objetos e suas relações a fim de descrever, de forma programática, o problema a ser resolvido;
2. A gratuidade da linguagem, que é totalmente gratuita, onde podemos gerar quaisquer tipos de projetos em ambientes de produção que também são gratuitos como o Eclipse, o JCreator e o Netbeans;
3. A usabilidade do Java tem como principal diferencial a amplitude de sistemas que aceitam esta linguagem, seja ela desktop, celular, relógio, televisores e afins.

Para agilizar o desenvolvimento do back-end da aplicação (camadas de modelo e controle), será empregado o framework Spring Boot.

O Spring Boot é um projeto da Spring, um facilitador para a etapa de configuração e publicação das aplicações desejadas. Com isso, o objetivo de ter os projetos rodando de forma rápida é atingida. A ferramenta favorece o padrão a seguirmos para configuração, cite

os módulos que deseja utilizar no projeto via arquivo pom.xml que serão reconhecidos e configurados. (AFONSO, A, 2017).

3.4 HTML e CSS

HTML (Hypertext Markup Language) e CSS (Cascading Style Sheets) são duas das principais tecnologias para a construção de páginas da Web. HTML fornece a estrutura da página, CSS o layout (visual e auditivo), para uma variedade de dispositivos. Junto com gráficos e scripts, HTM e CSS são a base da construção de páginas da Web e aplicativos da Web. (W3C)

3.4.1 O que é HTML?

O HTML é a linguagem de marcação responsável pela construção de todo e qualquer conteúdo de uma página web, onde se estrutura ela inserindo elementos, texto e mídias. Ao seguir os padrões atuais de codificações da linguagem, o resultado é a renderização em tela dos pontos acessados em uma página. (W3C)

3.4.2 O que é **CSS?**

CSS ou folhas de estilo em cascata, teve sua criação executada em 1996 pela W3C, pela lógica de leitura que a linguagem adota para implementar seus códigos, pois se inicia de cima para baixo. Apesar que mesmo nesta leitura, há regras de prioridade, então não basta somente o posicionamento em que o a tag foi criada, mas parâmetros estratégicos também influenciam na leitura de tags. (Rock Content, 2019).

Para agilizar o processo de construção dos estilos em CSS, usaremos o frameowrk BootStrap. O Bootstrap é um framework com finalidade em frontend utilizado por maioria dos desenvolvedores web do mundo, onde ao implementado traz consigo agilidade no desenvolvimento em conjunto com dinamismo, pois há diversos elementos prontos. Ele é gratuito, onde possui código fonte aberto e foi criado em 2010 pelos engenheiros Jacob Thorton e Mark Otto. (Bootstrap)

3.5 JAVASCRIPT

O Javascript é uma linguagem que contempla a criação de funções complexas para páginas HTML, envolvendo o HTML e CSS, onde ele completa uma trinca completa o suficiente para criar elementos HTML, estilização e animações em página web, onde o conteúdo afetado pode ser atualizado a maneira que deseja de forma dinâmica. (Mozila).

Para agilizar o desenvolvimento do front-end será usado a biblioteca React JS, criado e mantido pelo Meta Platforms, o React é uma biblioteca open-source com foco na criação de interfaces de usuário em sistemas web, baseados em componentes. (Meta Platforms).

3.6 Banco de Dados

O SQL, é uma linguagem base para trabalhar com bancos de dados relacionais, a sintaxe não é complexa então a aprendizagem é extremamente fácil, onde rapidamente qualquer pessoa pode escrever códigos em pouco tempo. O SQL possui alguns gerenciadores de bancos de dados com versões pagas e gratuitas, onde a Mysql e PostgreSQL são as mais conhecidas e mencionadas nos trabalhos relacionados. (SILVEIRA, P, 2019).

O Mysql foi escolhido por obter vantagens que se enquadram nos padrões do Campus Restinga, onde abaixo podemos encontrar algumas delas:

1. Desempenho - O Mysql é considerado o banco de dados mais rápido, no âmbito da internet a agilidade tem se tornado relevante ao extremo em sistemas. A cada dia que passa, a equipe de desenvolvimento do MySql tem se dedicado à performance. Esse fato, tende de que o SGBD mais rápido potencializará ainda mais a agilidade e eficácia.
2. Segurança - Devido a diversificação de tipos de tabelas, que é característica única do MySql, há possibilidade de ter um banco de dados seguro e estável, que contempla integridade referencial, backup e restore de database, controle de usuários e verificação e correção de corrompimento de tabelas.
3. Aplicabilidade - A possibilidade de uso do Mysql em sistemas Web e Desktop, apontam como uma grande vantagem, pela possibilidade de uso em sistemas corporativos. Devido ao suporte para diversas linguagens de programação com Delphi e Java, ambos podem acessar o Mysql pelos drivers ODBC e JDBC, disponibilizados no site do produto: http://www.mysql.com/
4. Distribuição - O Mysql é gratuito, onde também está listado no tipo de licenciamento GNU. Lembrando que ao disponibilizar seu sistema que utiliza este SGBD, o pacote de instalação dele deve ser disponibilizado a parte do seu sistema ou adquira o licenciamento GNU para disponibilização conjunta.

3.7 Esquema Conceitual

A Figura 2 mostra o esquema conceitual que será empregado na construção da intranet do Campus Restinga, apresentando as diferentes camadas do padrão MVC e as tecnologias que serão adotadas nas respectivas implementações.

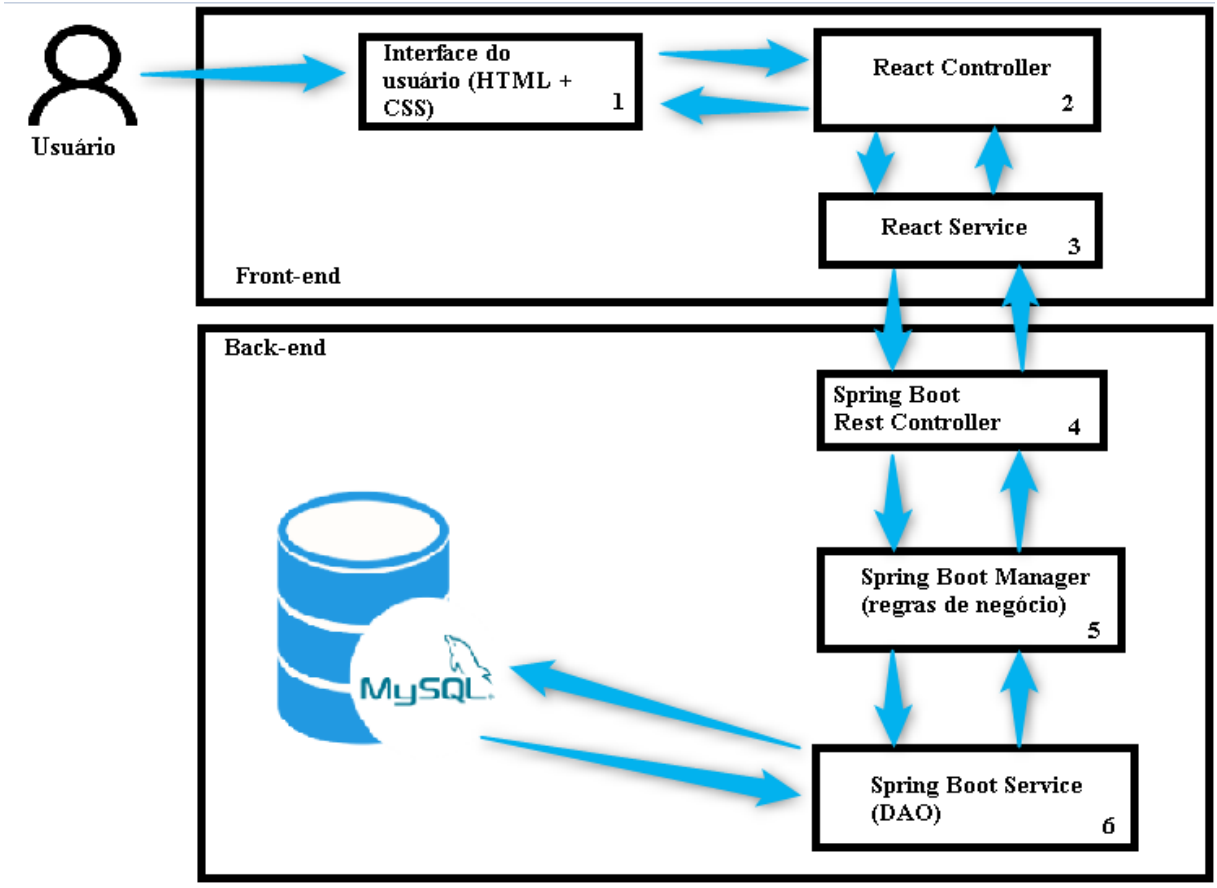

Figura 2: Arquitetura da intranet.
Fonte: Próprio autor.

3.7.1 Front-end

1. **Interface do usuário:** As páginas web do sistema. É responsável por apresentar a interface ao usuário. Utiliza tecnologias como HTML, CSS e JavaScript.
2. **React JS:** Devido ao seu padrão de componentização, o React consegue realizar as operações de forma síncrona ou assíncrona em seus projetos, onde a interface de usuário se comunica com o Back-end via a ferramenta axios, por métodos HTTP que estão criados no back-end para gerar a massa de dados que a intranet necessita.

3.7.2 Back-end

1. **Spring Boot Controller:** Recebe as requisições do front-end, e as encaminha para os gerenciadores de regras de negócio.
2. **Spring Boot Manager:** Aqui é realizado todo o processamento dos dados. Validações, filtros, tratamento de exceções e todas as regras de negócio são feitas nesta camada da aplicação.
3. **Spring Boot Service:** Responsável pela persistência e recuperação dos dados dos usuários. Nessa camada será utilizado um framework de mapeamento objeto-relacional.

4. **APIs de terceiros:** APIs de autenticação e recuperação de informações dos usuários em sistemas de terceiros.
    - Acesso gov.br - A conta gov.br é um meio de acesso digital para os cidadãos com finalidade de acesso aos serviços públicos digitais, de forma única por usuário; (GOV.BR, 2020)
    - SIG - Sistema integrador de dados e processos das áreas acadêmica, administrativa, gestão de recursos humanos, planejamento e projetos, gestão eletrônica de documentos e administração e comunicação do IFRS. (IFRS, 2018)
5. **Banco de dados:** banco de dados relacional onde as informações e configurações dos usuários são persistidas e manipuladas.

3.7.3 APIs

São mecanismos que permitem que dois componentes de software se comuniquem usando um conjunto de definições e protocolos. Por exemplo, o sistema de software do instituto meteorológico contém dados meteorológicos diários. O aplicativo em seu telefone "fala" com este sistema por meio de APIs e mostra atualizações meteorológicas diárias no seu telefone. A interface pode ser pensada como um contrato de serviço entre duas aplicações. Esse contrato define como as duas se comunicam usando solicitações e respostas. A documentação de suas respectivas APIs contém informações sobre como os desenvolvedores devem estruturar essas solicitações e respostas.

Infelizmente não foi possível realizar a implementação da API gov.br para autenticação dos usuários, devido à forte exigência documental que o gov solicita de informações para que possamos nos comunicar com a API deles. A comunicação também com o SIG foi afetada, devido ao tempo para implementar a comunicação, onde ela não foi realizada (AWS).

4 MODELO CONCEITUAL

Como boa prática de projeto, trabalhar com diagramas em suas diversas opções sempre auxilia o desenvolvedor a chegar ao melhor resultado em uma aplicação.

Para a modelagem do sistema, foi criado um grupo de trabalho no Campus Restinga, com representantes da TI, professores da área de computação e servidores chaves da área de negócio. O grupo reúne-se semanalmente, para definir as funcionalidades a serem implementadas. São destas reuniões que foram extraídos os diagramas que seguem.

4.1 Diagrama de casos de uso

Na linguagem de modelagem unificada (UML), o diagrama de caso de uso faz de forma resumida, o detalhamento do sistema usando o conceito de atores para os usuários-chave do sistema, permitindo visualizar as funcionalidades do sistema no âmbito de cada usuário-chave. A Figura 3 mostra as funcionalidades do sistema de intranet do Campus Restinga que serão implementadas no âmbito deste trabalho de conclusão, assim como os usuários-chave que serão beneficiados nessa primeira versão do sistema.

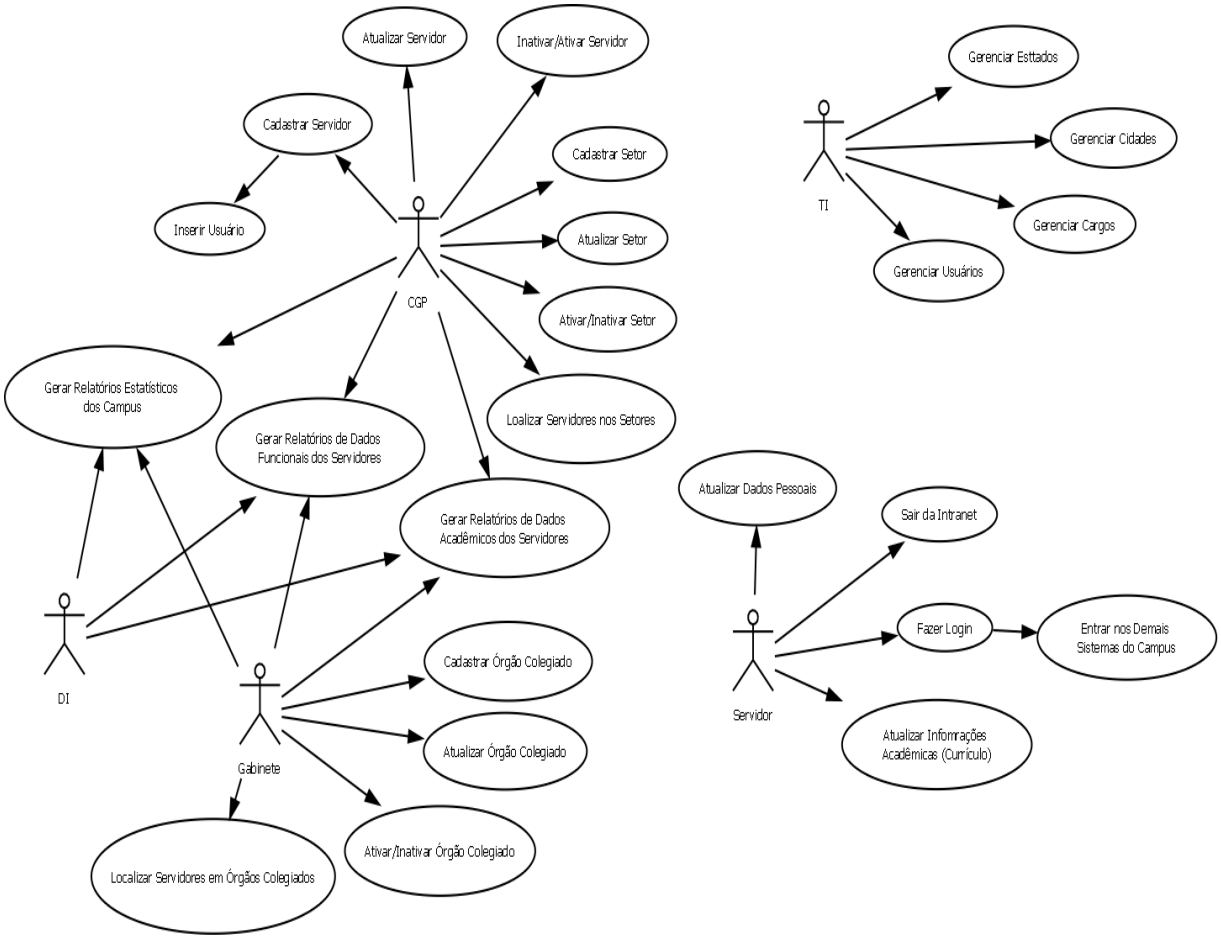

Figura 3: Diagrama Casos de uso.
Fonte: Próprio autor.

Após a construção do diagrama, os casos de uso foram detalhados textualmente apresentado o fluxo de execução da funcionalidade e as regras de negócio necessárias para sua implementação. A Figura 4 mostra um exemplo da descrição do caso de uso ‘Gerenciar usuários’. A documentação contempla dos demais casos de uso do sistema de intranet estão disponíveis no Apêndice A.

Nome:	UCTI04 – Gerenciar usuários
Descrição:	Este caso de uso é utilizado para realizar o gerenciamento de usuários no sistema.
Atores Envolvidos:	TI
Pré-condição:	O TI deve estar autenticado no sistema.
Pós-condição:	Os usuários inseridos devem estar disponíveis para alterações.
Fluxo da tarefa principal:	1. O usuário clica na opção de “Usuários” no site. 2. O usuário deve selecionar o usuário desejado. 3. O usuário realiza as alterações que necessita, como inativar, alterar senha, alterar permissões. 4. O usuário deve salvar as alterações 5. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera algum campo com informações inválidas ou deixa algum campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Figura 4: Detalhamento do Caso de Uso Gerenciar usuários
Fonte: Próprio autor.

4.2 Diagrama de Classes

A partir das descrições dos casos de uso criamos o diagrama de classes da aplicação, sempre considerando o escopo deste trabalho de conclusão. As figuras 5, 6 e 7 mostram o diagrama de classes implementado atualmente no sistema de intranet. Futuramente esse diagrama será ampliado com a inclusão de novas funcionalidades no sistema.

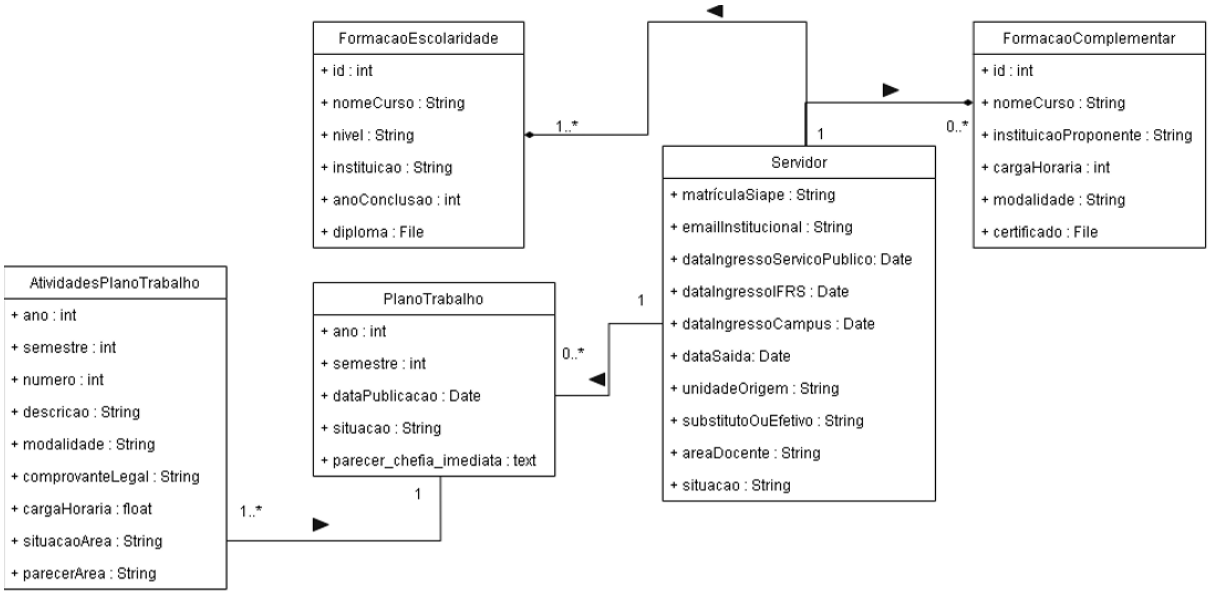

Figura 5: Diagrama de Classes (Servidor, Escolaridade, Formação complementar, Plano de trabalho e Atividades do Plano de Trabalho).
Fonte: Próprio autor.

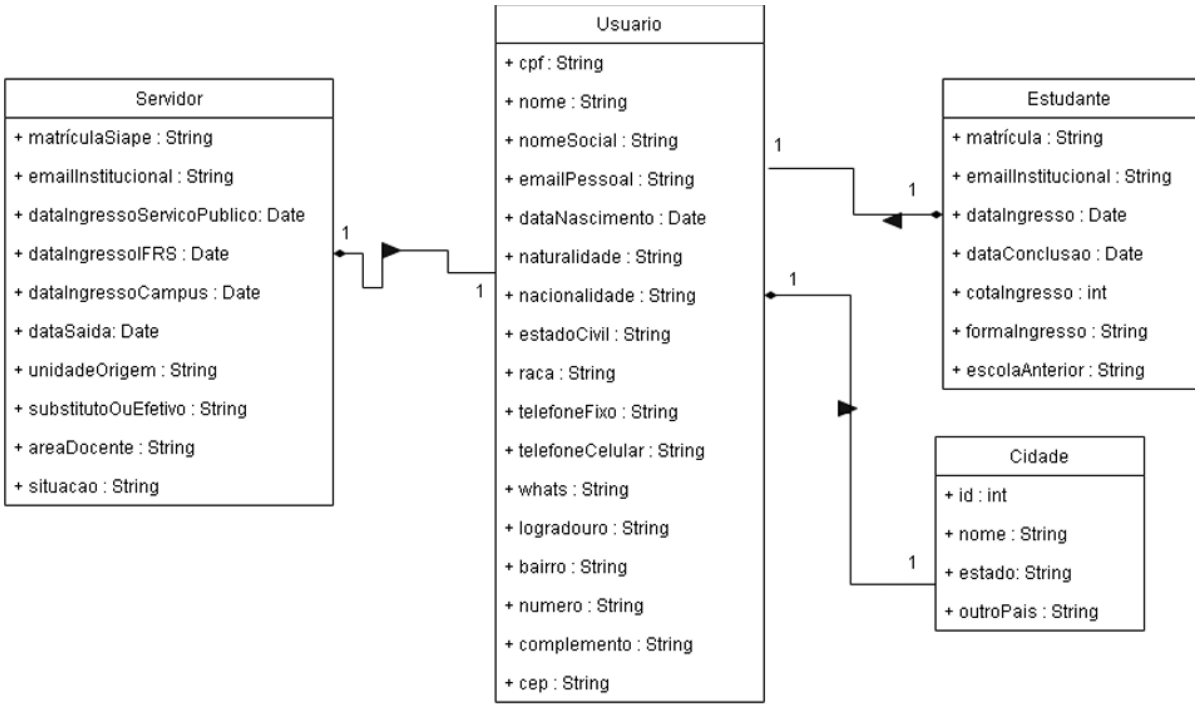

Figura 6: Diagrama de Classes (Usuário, Servidor, Estudante e Cidade).
Fonte: Próprio autor.

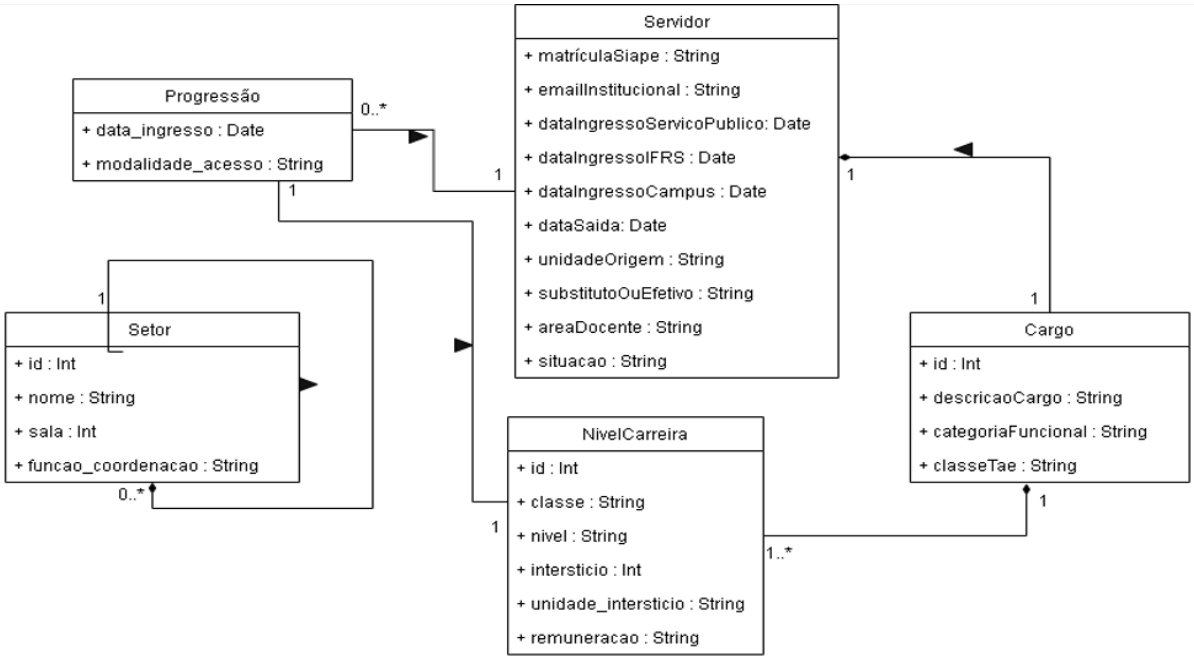

Figura 7: Diagrama de Classes (Servidor, Progressão, Setor, Cargo e Nível de Carreira). Fonte: Próprio Autor.

4.3 Diagrama Entidade Relacionamento

Este diagrama refere-se as ligações definidas para as classes mapeadas na intranet, a Figura 8 e 9 detalham o relacionamento apontando ligações de acordo com o mapeamento objeto-relacional.

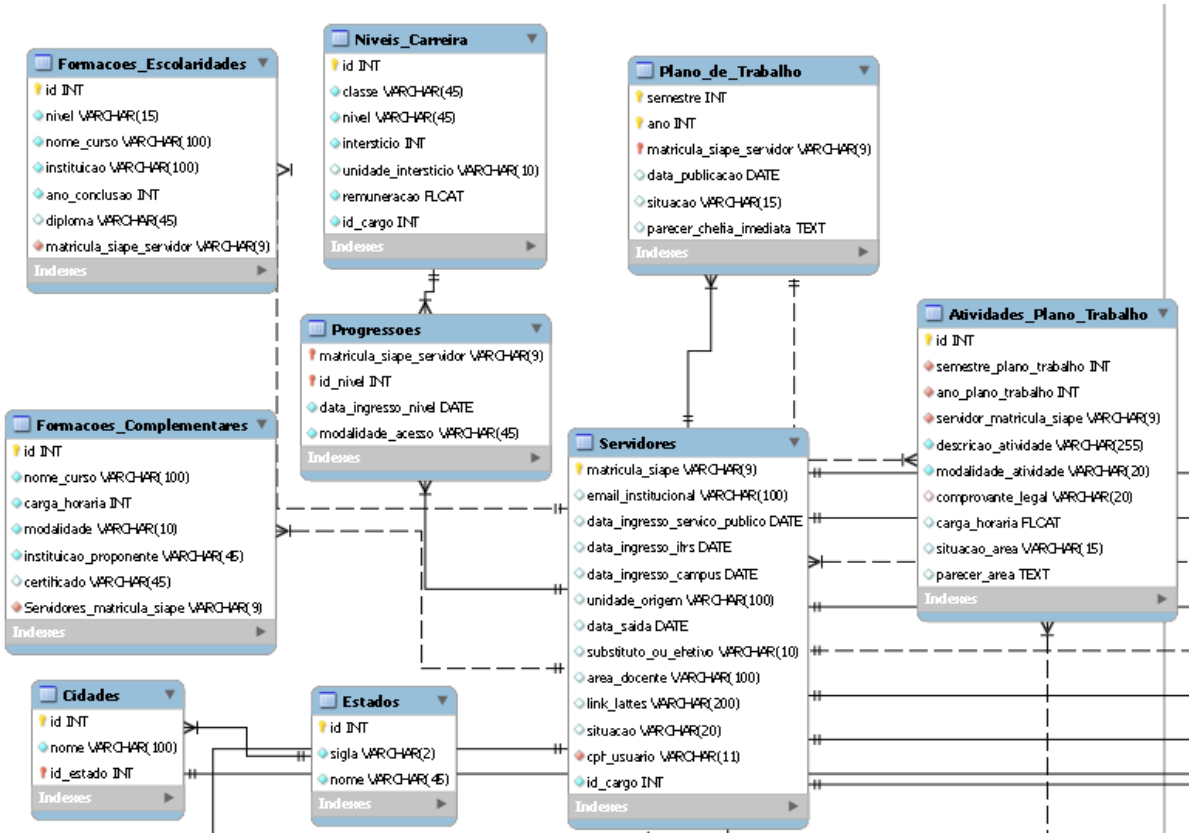

Figura 8: Diagrama ER (Servidores, Cidades, Estados, Progressões, Plano de Trabalho, Atividades Plano de Trabalho, Formações Complementar e Escolar)
Fonte: Próprio autor.

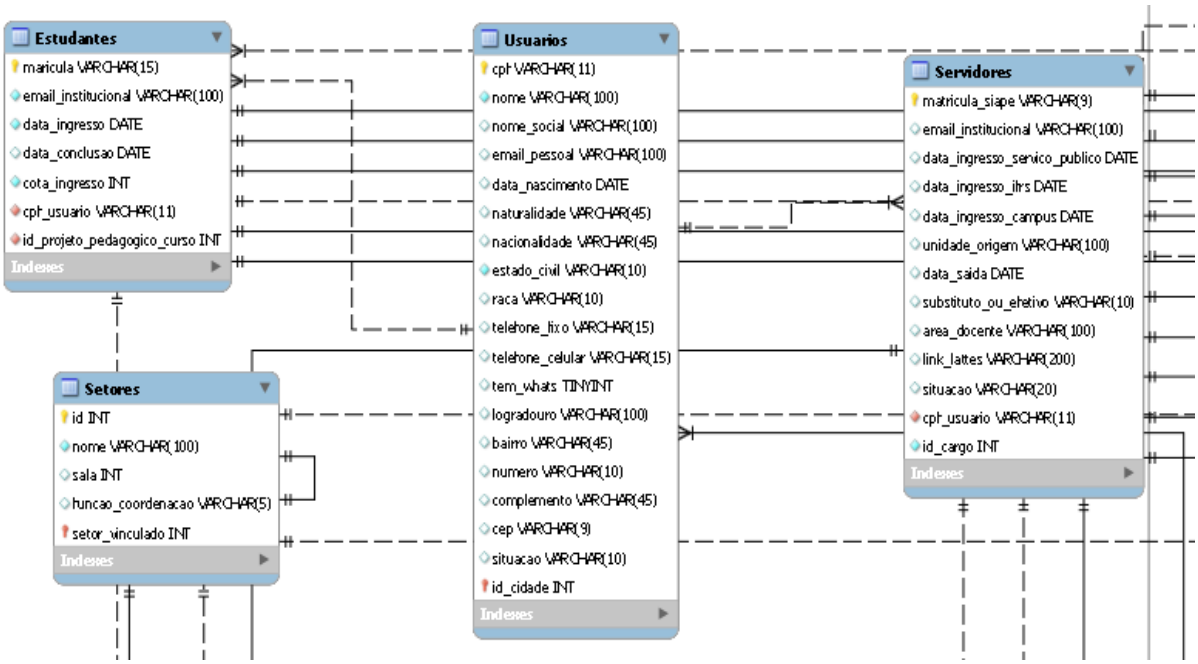

Figura 9: Diagrama ER (Usuários, Servidores, Estudantes e Setores).
Fonte: Próprio autor.

Como o diagrama de Entidade Relacionamento já está pronto, pensado em muitas classes à frente do que já foi implementado, é o motivo para terem muitas ligações dentro das figuras 8 e 9, pois estão interligadas a classes que ainda serão implementadas. A divisão se faz necessária para a melhor legibilidade das figuras.

5 DESENVOLVIMENTO

Como definido na solução conceitual, capítulo 3, o desenvolvimento do sistema foi dividido em duas etapas:

- Back-end: Esta etapa foi responsável por criarmos as camadas de Controle, Modelo e Visualização, onde o é possível a partir da comunicação entre eles gerar dados e disponibilizar via http para o nosso cliente front-end;
- Front-end: Esta etapa é responsável por apresentar visualmente o sistema, onde por ele podemos realizar os procedimentos CRUD que manipulará nosso back-end para gerar os dados enviados pelo nosso cliente.

Realizando a captura de arquivos totais que foram construídos durante o projeto, tivemos 37 arquivos no Back-end (37 classes Java, 3 arquivos de configuração). Já para o Front-end tivemos o total de 44 arquivos (42 arquivos JavaScript, 2 arquivos de configuração).

5.1 Integrações com SIG

A comunicação com os sistemas entrelaçados com o SIG é de suma importância, pois pelo objetivo de concentrarmos as informações dos diversos sistemas do Campus em um só lugar, a acessibilidade das informações de forma organizada, única e facilitada traz consigo o fator decisivo de adesão a esta intranet, tornando o sistema relevante para uso de sistemas já utilizados em conjunto com possíveis automatizações de processos manuais.

A intranet em sua implementação por não ter comunicação com o SIG, teve em sua jornada de criação do sistema o uso de dados fictícios, onde via postman eram feitas as requisições HTTP para manipular o back-end até que as funcionalidades no front-end fossem concluídas, assim testando todo o sistema diretamente pela interface do usuário.

5.2 DESENVOLVIMENTO BACK-END

A aplicação do lado back-end teve sua implementação realizada pelo framework Spring Boot, citado no capítulo 3. Um dos seus princípios é a organização do projeto, onde ele apresenta as pastas a partir da raiz “src”, separando-as de acordo com cada responsabilidade da classe no projeto. O exemplo da Figura 10 mostra as pastas criadas para o projeto:

- Config: pasta onde contempla o arquivo CorsConfiguration, que realiza a permissão entre domínios que é o caso do projeto, que possui um host diferente para o back-end e outro para o front-end;
- Controller: onde fica as classes responsáveis pela implementação da camada rest Controller apresentada no esquema conceitual da figura 2;
- Model: onde se encontra as entidades que encorpam a aplicação, contendo os atributos de cada classe, por exemplo as classes de usuários, cargos e setores;
- Repository: onde fica as classes do tipo interface, que contemplam os métodos que fazem os métodos que acessam o banco de dados, como select e delete;

- Service: pasta que contempla os serviços que manipulam via métodos as alterações das classes modelos.

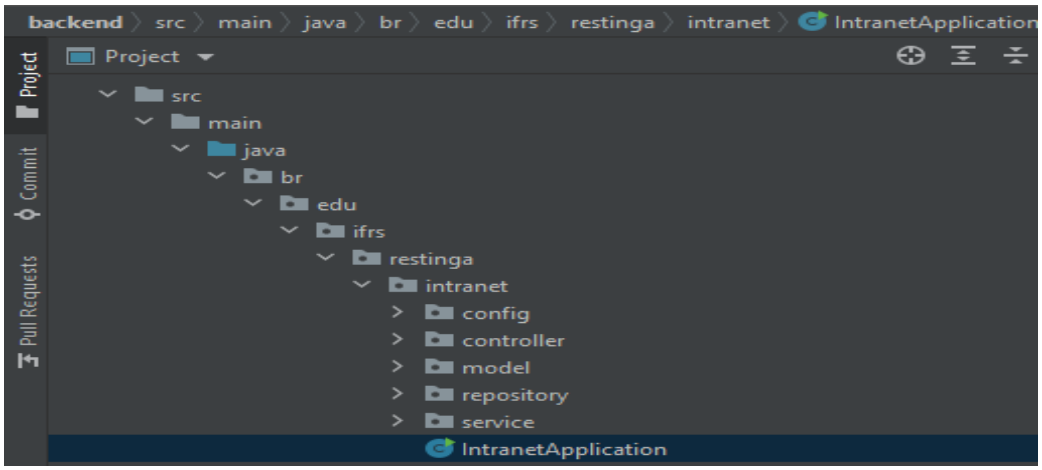

Figura 10: Estrutura do código fonte back-end.
Fonte: Próprio autor.

Dentro da pasta src, também existe a pasta resources que contém o arquivo application.properties para guardar configurações do banco de dados que o projeto irá utilizar e em que porta será disponibilizado a comunicação para métodos HTTP. A Figura 11 apresenta a configuração para o ambiente local. O endereço real onde a aplicação irá rodar, possui outras informações de endereços e credenciais de banco de dados.

```
1  spring.datasource.url=jdbc:mysql://localhost/intranet?createDatabaseIfNotExist=true&useSSL=false
2  spring.jpa.properties.hibernate.jdbc.time_zone=America/Sao_Paulo
3  server.port=8080
4  spring.datasource.username=root
5  spring.datasource.password=
6  spring.jpa.hibernate.ddl-auto=update
7  spring.jpa.database-platform = org.hibernate.dialect.MySQL5InnoDBDialect
8  spring.jpa.properties.hibernate.show_sql= true
```

Figura 11: Arquivo application.properties.
Fonte: Próprio autor.

Importante citar o arquivo pom.xml da Figura 12, que traz de forma organizada as dependências que o projeto Spring utilizará para o funcionamento. Ele cita versão do Java, a extensão do arquivo de build que o projeto terá para disponibilizar deploy, dependências como o lombok que é um facilitador de escrita, onde é possível desonerar a criação de métodos getter e setter de forma manual. A dependência starter-validation que auxilia nas validações de métodos que relacionam o model em sua criação e alteração pelo controller.

```xml
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.6.2</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>

    <groupId>br.edu.ifrs.restinga</groupId>
    <artifactId>intranet</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <!--<packaging>war</packaging> -->
    <name>intranet</name>
    <description>Intranet project for Spring Boot</description>
    <properties>
        <java.version>11</java.version>
        <!-- <skipTests> true </skipTests> -->
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-validation</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
```

Figura 12: Dependências do pom.xml.
Fonte: Próprio autor.

Para exemplificar o código de um Controller seguindo o padrão MVC, a Figura 13 traz estes CRUDS que estão manipulando os atributos na classe modelo de setores. A organização seguida contempla as operações esperadas para o ciclo de vida de um setor.

```java
    @RequestMapping(method = RequestMethod.GET)
    public ResponseEntity<List<Setor>> all() {
        List<Setor> setores = this.setorService.buscarSetores();
        return ResponseEntity.ok().body(setores);
    }

    @RequestMapping(value = "/{idSetor}", method = RequestMethod.GET)
    public ResponseEntity<Setor> setorPorId(@PathVariable("idSetor") Long idSetor) {
        return ResponseEntity.ok().body(this.setorService.buscarSetorPorId(idSetor));
    }

    @RequestMapping(method = RequestMethod.POST)
    public ResponseEntity<Void> criarSetor(@RequestBody @Valid Setor setor) {
        Setor obj = this.setorService.cadastrarSetor(setor);
        URI uri = ServletUriComponentsBuilder.fromCurrentRequest()
                .path("/{idSetor}").buildAndExpand(obj.getIdSetor()).toUri();
        return ResponseEntity.created(uri).build();
    }

    @RequestMapping(value = "/{idSetor}", method = RequestMethod.PUT)
    public ResponseEntity<Void> atualizarSetor(@RequestBody @Valid Setor setor,
            @PathVariable Long idSetor) {
        this.setorService.editarSetor(setor, idSetor);
        return ResponseEntity.noContent().build();
    }

    @RequestMapping(value = "/{idSetor}", method = RequestMethod.DELETE)
    public ResponseEntity<Void> deletarSetor(@PathVariable Long idSetor) {
        this.setorService.excluirSetor(idSetor);
        return ResponseEntity.noContent().build();
    }
```

Figura 13: Classe Controller setores.
Fonte: Próprio autor.

5.3 DESENVOLVIMENTO FRONT-END

Para a camada View, que já é o projeto front-end, a estrutura das pastas ficou separada de acordo com a estrutura do nosso diagrama de classes. Cada classe tem uma pasta. Por exemplo, a classe Servidor é implementada na pasta Usuário, conforme a hierarquia estabelecida por essa especialização. Da mesma forma, as classes de formação complementar, formação escolar e progressões estão relacionadas a servidores, por isso implementadas com subpastas de servidor, como mostra o exemplo na figura 14.

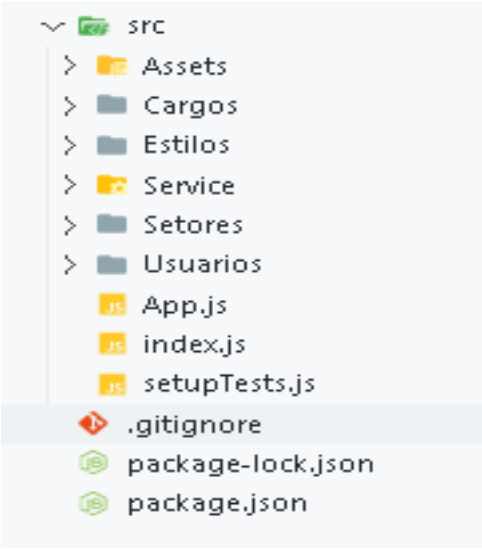

Figura 14: Raiz de pastas no front-end.
Fonte: Próprio autor.

No front-end, também foram necessários alguns arquivos auxiliares para ajustar algumas configurações do projeto, como por exemplo o endereço do banco de dados onde o front-end irá ler as informações e realizar requisições. Essa configuração encontra-se no arquivo api.js da Figura 15. O arquivo contém o endereço do servidor, onde com o auxílio do axios (cliente http) faremos as requisições http para o nosso back-end.

Figura 15: Arquivo api.js.
Fonte: Próprio autor.

Pelo fato deste projeto estar sendo planejado para uma utilização a longo prazo, onde o objetivo dele impactará muitas áreas do Campus, foi implantado dentro do front-end uma estratégia de roteamento, a partir da importação do React-Router-Dom.

O roteamento das urls do projeto, ficaram dentro do app.js, onde é possível encontrar as rotas principais. As rotas ficam visíveis no menu para todo o sistema e a lógica das rotas principais são apresentadas na Figura 16.

```
<Router>
  <header>
    <div className="navbar navbar-expand-lg navbar-light menu">
      <div className="collapse navbar-collapse" id="navbarNavAltMarkup">
        <div className="navbar-nav">
          <img src={logo} width="150px" height="150px" className="d-inline-block"></img>
          <Link className="nav-link active" aria-current="page" to='/'>Home</Link>
          <Link className="nav-link active" aria-current="page" to='/pesquisa-estudantes'>Estudantes</Link>
          <Link className="nav-link active" aria-current="page" to='/pesquisa-servidores'>Servidores</Link>
          <Link className="nav-link active" aria-current="page" to='/cargos'>Cargos</Link>
          <Link className="nav-link active" aria-current="page" to='/setores'>Setores</Link>
        </div>
      </div>
    </div>
  </header>
```

Figura 16: Roteamento das principais páginas.
Fonte: Próprio autor.

Já para as demais rotas, que lidam com subpáginas, como acessar as progressões de um servidor, fica no conteúdo do app.js, onde foi mapeado qual dado será mostrado de um servidor ou estudante em uma classe auxiliar, para que fique em um padrão de dados compreensível aos usuários. A Figura 17 mostra os roteamentos auxiliares.

```
<div className="content">
    <Switch>
      <Route exact path='/pesquisa-estudantes' component={PesquisaView} />
      <Route path='/pesquisa-estudantes/detalhes/:matricula' component={DetalhesView} />
      <Route path='/pesquisa-estudantes/fichafuncional/:matricula' component={FichaFuncionalView} />
      <Route path='/pesquisa-estudantes/fichapessoal/:matricula' component={FichaPessoalView} />

      <Route exact path='/pesquisa-servidores' component={PesquisaViewServ} />
      <Route path='/pesquisa-servidores/detalhes/:matriculaSiape' component={DetalhesViewServ} />
      <Route path='/pesquisa-servidores/fichafuncional/:matriculaSiape' component={FichaFuncViewServ} />
      <Route path='/pesquisa-servidores/fichapessoal/:matriculaSiape' component={FichaPessoalViewServ} />
      <Route path='/pesquisa-servidores/:matriculaSiape/escolaridades' component={EscolarView} />
      <Route path='/pesquisa-servidores/:matriculaSiape/formacoescomplementares' component={ComplementarView} />
      <Route path='/pesquisa-servidores/:matriculaSiape/progressoes' component={ProgressaoView} />

      <Route exact path='/cargos' component={CargosView} />
      <Route path='/cargos/:idCargo/niveis' component={NivelView} />

      <Route exact path='/setores' component={SetorView} />
    </Switch>
```

Figura 17: Roteamento de subpáginas.
Fonte: Próprio autor.

Uma implementação que pode ser citada, é a de cadastro de setores que é uma classe pouco complexa e simples implementação, devido ao seu tamanho que é pequeno, comparado as demais do projeto. A renderização dela traz o seu formulário de cadastro e atualização em conjunto com a tabela de campos que esta classe possui, para auxiliar na visualização dos setores cadastrados anteriormente, conforme a Figura 18.

```
render() {
    return <div>
        <SetorForm
            key={this.state.itemEditar ? this.state.itemEditar.idSetor : "novo"}
            itemEditar={this.state.itemEditar}
            onCadastrar={(setor) => {
                return this.cadastrarSetor(setor);
            }}
            onAtualizar={(setor) => this.atualizarSetor(setor)}
            onCancelar={() => this.setState({ itemEditar: null })}
        />

        {this.state.carregar ? "Carregando" :
            <SetorTable
                itens={this.state.setores}
                onEditar={(setor) => this.editarSetor(setor)}
                onDeletar={(setor) => this.deletarSetor(setor)}
            />
        }
    </div>
}
```

Figura 18: Código fonte do View de setores.
Fonte: Próprio autor.

Complementando a implementação do cadastro de setores, é importante mostrar como está feito a requisição http para o servidor da intranet, onde temos para cada requisição o auxílio do axios na Figura 19.

src > Setores > Pages > SetorView.js > SetorView > deletarSetor

```
14
15      async listarSetores() {
16          await api.get(`/intranet/setores`).then(
17              (retorno) => this.setState({
18                  carregar: false,
19                  setores: retorno.data
20              })
21          ).catch((err) => {
22              console.log("Ops! Ocorreu um erro" + err);
23          })
24      }
25
26      async deletarSetor(setor) {
27          try {
28              await api.delete(`/intranet/setores/${setor.idSetor}`);
29              this.listarSetores();
30          } catch (erro) {
31              this.setState({
32                  erro: erro
33              });
34              return false;
35          }
36      }
```

src > Setores > Pages > SetorView.js > SetorView > atualizarSetor

```
38      async cadastrarSetor(setor) {
39          this.setState({ erro: null });
40          try {
41              await api.post("/intranet/setores", setor);
42              await this.listarSetores();
43              return true;
44          } catch (erro) {
45              this.setState({  erro: erro });
46              return false;
47          }
48      }
49
50      async atualizarSetor(setor) {
51          this.setState({ erro: null });
52          try {
53              await api.put(`/intranet/setores/${setor.idSetor}`, setor);
54              await this.listarSetores();
55              this.setState({ itemEditar: null });
56              return true;
57          } catch (erro) {
58              this.setState({ erro: erro });
59              return false;
60          }
```

Figura 19: Requisições http de setores.
Fonte: Próprio autor.

5.4 Prototipagem em React

Nesta seção, será mostrado algumas telas com cadastros criados em um ambiente local que não condiz com dados reais de servidores ou estudantes. Estas telas, tiveram a implementação simples devido as prioridades definidas pelo grupo de trabalho, que buscou estruturar os dados/classes que deveriam ser implementadas para este trabalho, a partir da Figura 20, logo abaixo.

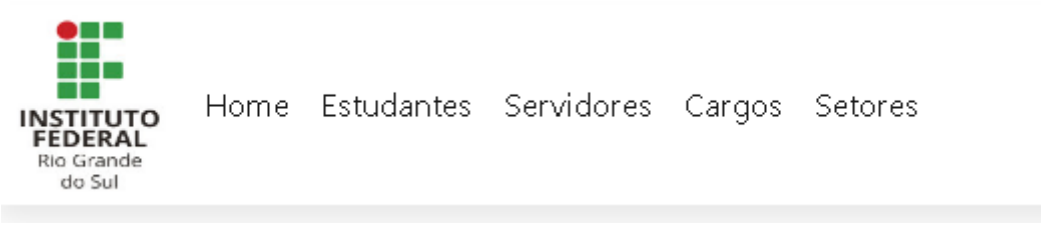

Figura 20: Menu de páginas principais.
Fonte: Próprio autor.

A Figura 21 mostra o cadastro de setores que foi implementado na intranet, resultado dos exemplos de codificação apresentados nas seções anteriores. Foram criados dois registros para testes de edição e exclusão de setores, com salas no padrão do IF e funcionalidades passadas pelo grupo de trabalho.

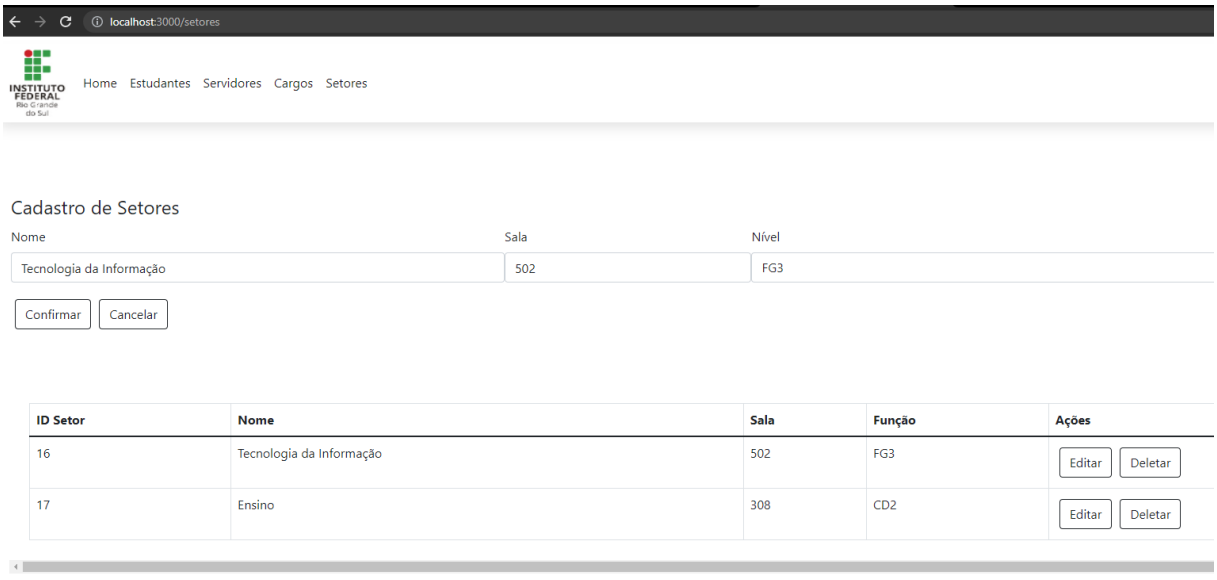

Figura 21: Página de setores.
Fonte: Próprio autor.

5.5 Execução da intranet

A intranet está em fase inicial, então alguns pontos foram implementados neste trabalho, mas o mesmo, terá sua implementação continuada por outros estudantes, onde o grupo de trabalho irá prosseguir encorpando o escopo do projeto, apresentando os pontos a serem implementados em conjunto com o que se espera de cada funcionalidade, deixando claro como funciona cada ponto.

Nesta fase, a intranet não possui uma autenticação, então ao acessar o endereço principal, será possível visualizar a intranet.

5.5.1 Usuários Estudantes e Servidores

Para o acesso aos dados de estudantes, basta escolher no menu a opção estudantes, que trará os estudantes cadastrados com a possibilidade de pesquisa pelo seu nome e sendo possível a visualização dos dados deste estudante, como também a edição dos dados pessoais deste estudante como também os dados institucionais, conforme a Figura 22.

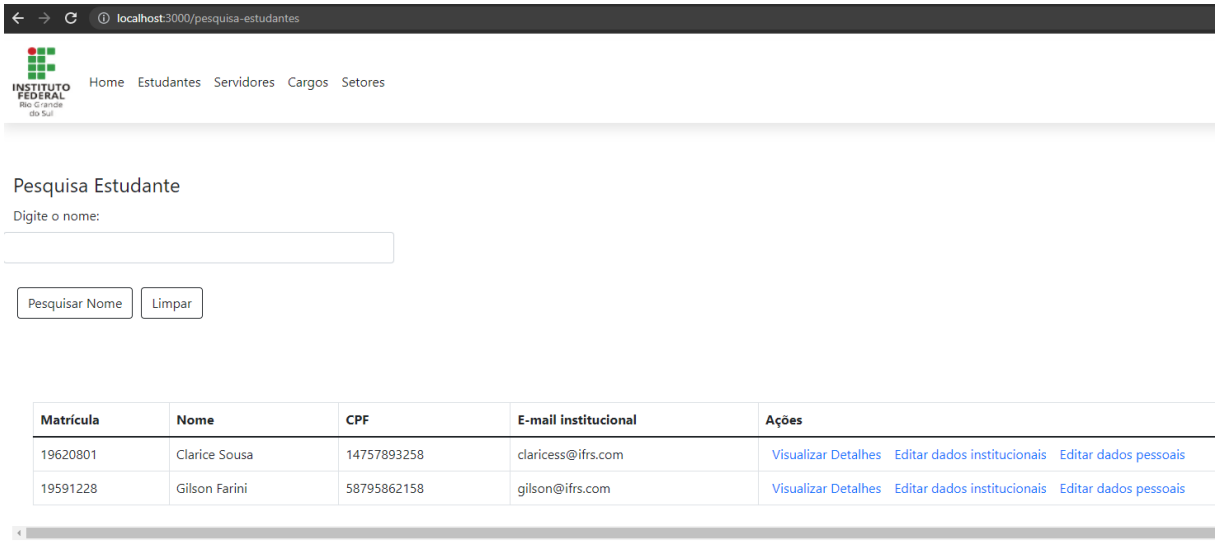

Figura 22: Página de estudantes.
Fonte: Próprio autor.

O padrão de página citado acima, também serve para os servidores, onde é possível acessar conforme menu principal este módulo na Figura 23, onde podemos pesquisar, visualizar os detalhes de seu cadastro, edição dos dados institucionais e pessoais, junto com as formações e progressões do servidor.

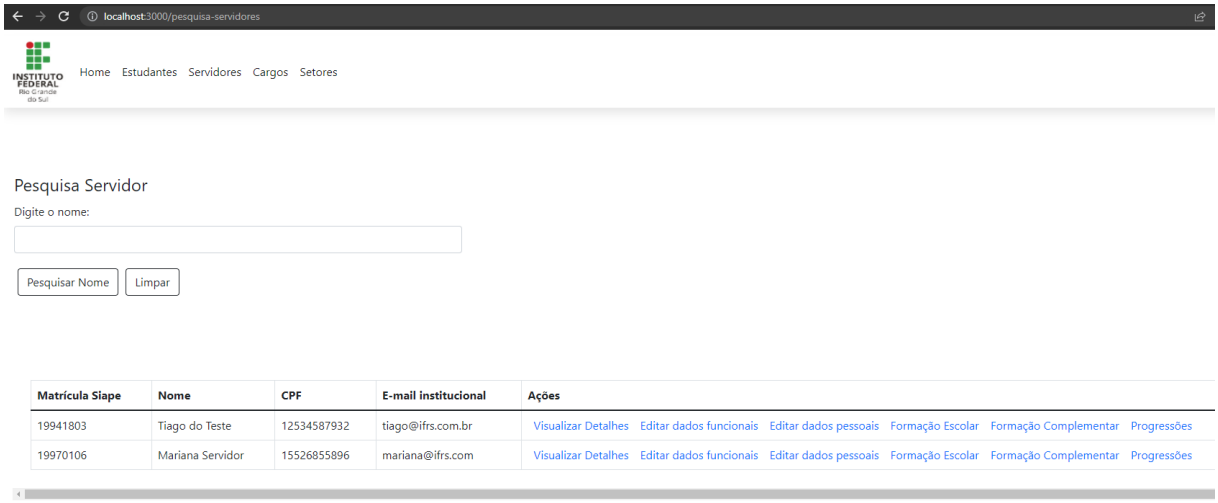

Figura 23: Página de servidores.
Fonte: Próprio autor.

5.5.2 Cargos

O módulo de cargos, é uma das funcionalidades solicitadas para este trabalho, onde tivemos os dados de categoria e classe passados para a implementação mesmo que em fase inicial já possuam alguns dados reais, para que em um novo cadastro de setor, fosse inserido corretamente estas informações para posteriormente ligarmos aos servidores. Lembrando que é possível acessarmos os níveis de carreira que este cargo tem, como por exemplo níveis como: Júnior, Pleno e Sênior, onde é possível fazermos também o CRUD completo dos níveis. Exemplo na Figura 24, logo abaixo.

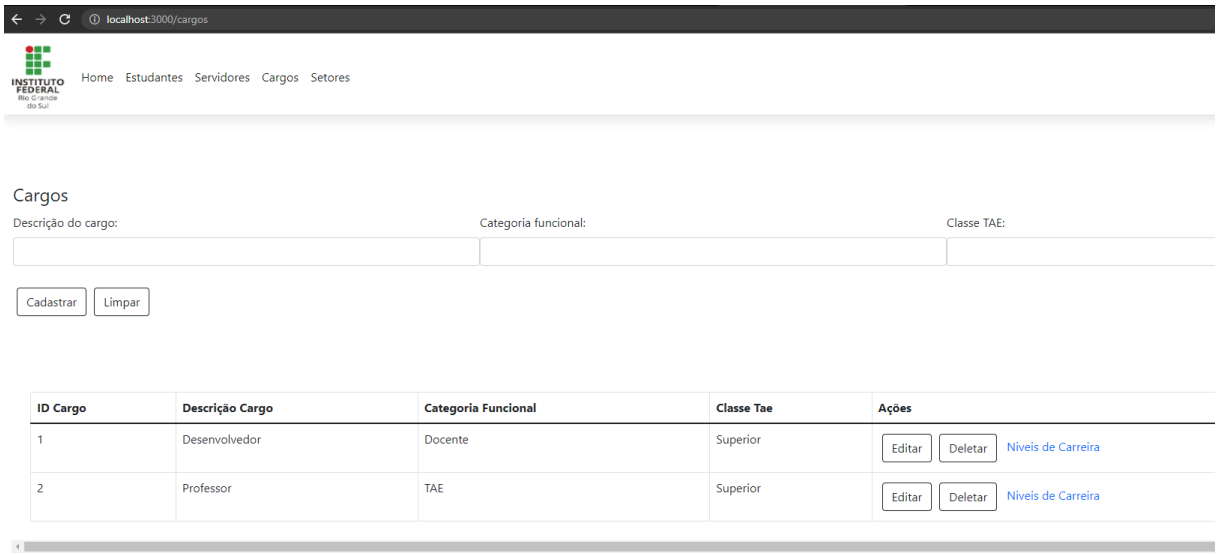

Figura 24: Página de cargos.
Fonte: Próprio autor.

5.5.3 Setores

Para os setores do Campus, o último módulo na ordem do menu principal também ao visualizar este módulo, há possibilidade de realizar CRUD, com setores baseados no padrão estrutural que o Campus possui, onde os dados da Figura 21 tiveram uso de dados reais do Campus com cargos fictícios para finalidade de teste.

6 CONCLUSÃO

O sistema de intranet para serviços internos do Campus Restinga viabilizará o acesso aos sistemas vinculados à jornada de trabalho dos servidores do Campus, permitindo que a sua rotina seja menos onerosa em relação a encontrar informações e acessar as mesmas. Esse sistema unirá todas as informações em um só local e descomplicando fluxos de trabalho que hoje são implementados em planilhas ou formulários do Google-Drive.

Para trabalhos futuros sugere-se: prosseguir com a implementação das funcionalidades do sistema automatizando processos de trabalho usados no Campus; implementar novo layout para o sistema para que o visual da intranet tenha a identidade do IFRS; incluir login único permitindo que os sistemas que serão vinculados a intranet não precisem realizar uma segunda autenticação; e integrar informações dos servidores e estudantes com os sistemas governamentais e sistemas acadêmicos do Instituto, utilizando de uma comunicação com os módulos do SIG.

A intranet teve um percentual de 40% das funcionalidades criadas em caso de teste implementadas, onde apresentamos o módulo de estudantes e edição de seus dados. Também foi apresentado neste início de trabalho a edição dos dados dos servidores (funcionários) do Campus e suas informações complementares como cargo, setor, formações escolares e complementares.

A elaboração dos documentos oriundos da engenharia de software como a elaboração do diagrama dos casos de teste, o detalhamento de cada caso criado de acordo com sua função e a criação do diagrama entidade-relacionamento foram desafios concluídos sem dificuldades, pois, a jornada que esta Graduação proporciona cria profissionais competentes e preparados para o mercado de trabalho, encarando e resolvendo problemas do cotidiano da tecnologia.

REFERÊNCIAS

Bootstrap 5.1. Disponível em: https://getbootstrap.com/docs/5.1/getting-started/introduction/. Acesso em: 10 dez. 2021.

Cidadão Login único. Disponível em: https://www.gov.br/conecta/catalogo/apis/brasil-cidadao-login-unico. Acesso em: 18 mai. 2022.

GODOY, A. R. L. de; UCELLI, J. M.; ROCHA, T.; PAIVA, W. R. de. WATZ: Sistema de Intranet. Sínteses: Revista Eletrônica do SimTec, Campinas, SP, n. 7, p. e019108, 2019. DOI: 10.20396/sinteses.v0i7.11561. Disponível em: https://econtents.bc.unicamp.br/inpec/index.php/simtec/article/view/11561. Acesso em: 13 dez. 2021.

GONÇVALES, André Luiz. SISTEMA DE INTRANET PARA TABULAÇAO DE DADOS COMO MATERIAL DE APOIO AO PROJETO BULLYING E CYBERBULLYING LITORAL DO PARANÁ. 2015. Matinhos. Monografia (Bacharel em Informática e Cidadania) – Universidade Federal do Paraná – Setor Litoral. Disponível em: https://core.ac.uk/download/pdf/147520642.pdf. Acesso em: 11 dez. 2021.

HTML + CSS. Disponível em: https://www.w3.org/standards/webdesign/htmlcss. Acesso em: 17 mai. 2022.

Introdução a requisitos de software. Disponível em: https://www.devmedia.com.br/imagens/fotoscolunistas. Acesso em 12 jul. 2022.

Leite, Alessandro. Conheça os Padrões de projeto. Acre: DevMedia, 2005. Disponível em: https://www.devmedia.com.br/conheca-os-padroes-de-projeto/957. Acesso em 15 mar. 2022.

MELLO, J. M de; SICCA, W. dos S. Intranet institucional e sistema de pedidos. Disponível em: https://computacao.ucpel.edu.br/wp-content/uploads/2016/06/2012-1_3.pdf. Acesso em: 12 dez. 2021.

Muller, Nicolas. O que é uma intranet e para que serve? Disponível em: https://www.oficinadanet.com.br/artigo/intranet/o_que_e_uma_intranet_e_pra_que_serve. Acesso em: 13 dez. 2021.

NUNES, Paulo. Citação do conceito de Intranet. Knoow Enciclopédia temática, 2016, ago. Disponível em: https://knoow.net/en/economics-business/management/intranet/. Acesso em: 15 mai. 2022.

Assis, Pablo, de. O que é intranet e extranet? Disponível em: https://www.tecmundo.com.br/conexao/1955-o-que-e-intranet-e-extranet-.htm. Acesso em: 12 jul. 2022.

O que é JavaScript. Disponível em: https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps/What_is_JavaScript. Acesso em: 13 dez. 2021.

O que é uma API? Disponível em https://aws.amazon.com/pt/what-is/api/. Acesso em: 12 jul. 2022.

Rossetti, Micaela, L. Requisitos de software: Funcionais e Não-Funcionais. Disponível em: https://softdesign.com.br/blog/requisitos-de-software-funcionais-e-nao-funcionais. Acesso em: 12 jul. 2022.

SILVEIRA, Paulo. O que é SQL? 2019 Disponível em: https://www.alura.com.br/artigos/o-que-e-sql. Acesso em: 17 mai. 2022.

Sistemas integrados – SIG. Disponível em: https://ifrs.edu.br/tecnologia-da-informacao/sistemas-integrados/sig/. Acesso em: 18 mai. 2022.

APÊNDICES

APÊNDICE A - CASOS DE USO

Um detalhamento dos casos de usos criados para a intranet.

Casos de uso TI

Nome:	UCTI01 – Gerenciar estados
Descrição:	Este caso de uso é utilizado para realizar o gerenciamento de estados do país Brasil no sistema.
Atores Envolvidos:	TI
Pré-condição:	O TI deve estar autenticado no sistema.
Pós-condição:	Os estados inseridos devem estar disponíveis para alterações.
Fluxo da tarefa principal:	1. O usuário clica na opção de “Estados” no site. 2. O usuário deve selecionar o estado desejado. 3. O usuário realiza as alterações que necessita. 4. O usuário deve salvar as alterações 5. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera o campo com informações inválidas ou deixa o campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCTI02 – Gerenciar cidades
Descrição:	Este caso de uso é utilizado para realizar o gerenciamento de cidades do país Brasil no sistema.
Atores Envolvidos:	TI
Pré-condição:	O TI deve estar autenticado no sistema.
Pós-condição:	As cidades devem estar disponíveis para alterações.
Fluxo da tarefa principal:	1. O usuário clica na opção de “Cidades” no site. 2. O usuário deve selecionar a cidade desejada. 3. O usuário realiza as alterações que necessita. 4. O usuário deve salvar as alterações 5. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera o campo com informações inválidas ou deixa o campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCTI03 – Gerenciar cargos
Descrição:	Este caso de uso é utilizado para realizar o gerenciamento de cargos da instituição no sistema.
Atores Envolvidos:	TI
Pré-condição:	O TI deve estar autenticado no sistema.
Pós-condição:	Os cargos inseridos devem estar disponíveis para alterações.
Fluxo da tarefa principal:	1. O usuário clica na opção de “Cargos” no site. 2. O usuário deve selecionar o cargo desejado. 3. O usuário realiza as alterações que necessita. 4. O usuário deve salvar as alterações 5. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera algum campo com informações inválidas ou deixa algum campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCTI04 – Gerenciar usuários
Descrição:	Este caso de uso é utilizado para realizar o gerenciamento de usuários no sistema.
Atores Envolvidos:	TI
Pré-condição:	O TI deve estar autenticado no sistema.
Pós-condição:	Os usuários inseridos devem estar disponíveis para alterações.
Fluxo da tarefa principal:	1. O usuário clica na opção de “Usuários” no site. 2. O usuário deve selecionar o usuário desejado. 3. O usuário realiza as alterações que necessita, como inativar, alterar senha, alterar permissões. 4. O usuário deve salvar as alterações 5. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera algum campo com informações inválidas ou deixa algum campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Casos de uso Servidor

Nome:	UCSRV01 – Atualizar informações acadêmicas (currículo)
Descrição:	Este caso de uso é utilizado para atualizar o currículo acadêmico do servidor
Atores Envolvidos:	Servidor
Pré-condição:	O servidor deve estar autenticado no sistema.
Pós-condição:	As alterações realizadas devem ser atualizadas no currículo do servidor.
Fluxo da tarefa principal:	1. O usuário clica na opção de “Currículo” no site. 2. O usuário deve selecionar o usuário desejado. 3. O usuário realiza as alterações que necessita, como inativar, alterar senha, alterar permissões. 4. O usuário deve salvar as alterações 5. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera algum campo com informações inválidas ou deixa algum campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCSRV02 – Atualizar dados pessoais
Descrição:	Este caso de uso é utilizado para atualizar os dados pessoais do servidor.
Atores Envolvidos:	Servidor
Pré-condição:	O servidor deve estar autenticado no sistema.
Pós-condição:	As alterações realizadas devem ser atualizadas nos dados pessoais do servidor.
Fluxo da tarefa principal:	1. O usuário clica na opção de “Dados pessoais” no site. 2. O usuário altera os campos que deseja. 3. O usuário deve salvar as alterações. 4. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera algum campo com informações inválidas ou deixa algum campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Casos de uso Gabinete

Nome:	UCGBN01 – Cadastrar órgão colegiado
Descrição:	Este caso de uso é utilizado para realizar a criação de órgão colegiado no sistema.
Atores Envolvidos:	Gabinete
Pré-condição:	O responsável no gabinete deve estar autenticado no sistema.
Pós-condição:	O cadastro deve inserir no sistema este novo órgão colegiado para que possa desenvolver suas atividades.
Fluxo da tarefa principal:	1. O usuário clica na opção “Órgão colegiado” no site. 2. O usuário preenche os campos requeridos. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a criação.
Fluxo alternativo:	1. O usuário altera algum campo com informações inválidas ou deixa algum campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de criação.

Nome:	UCGBN02 – Atualizar órgão colegiado
Descrição:	Este caso de uso é utilizado para realizar alterações em órgãos colegiados no sistema.
Atores Envolvidos:	Gabinete
Pré-condição:	O responsável no gabinete deve estar autenticado no sistema.
Pós-condição:	O cadastro alterado deve ter seus dados alterados disponíveis para visualização.
Fluxo da tarefa principal:	1. O usuário clica na opção “Órgão colegiado” no site. 2. O usuário altera os campos desejados. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera algum campo com informações inválidas ou deixa algum campo em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCGBN03 – Ativar/inativar órgão colegiado
Descrição:	Este caso de uso é utilizado para ativar ou inativar um órgão colegiado.
Atores Envolvidos:	Gabinete
Pré-condição:	O responsável no gabinete deve estar autenticado no sistema.
Pós-condição:	O cadastro escolhido deve sofrer alteração em seu status de cadastro para ativo ou inativo.
Fluxo da tarefa principal:	1. O usuário clica na opção “Órgão colegiado” no site. 2. O usuário altera o campo status. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera o campo status sem salvar. 2. Usuário sai da tela de cadastro e realiza teste na recente alteração. 3. Não vai haver alteração, pois a mesma não foi salva. 4. O usuário deverá repetir a alteração e salvar. 5. O sistema confirma a alteração realizada.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCGBN04 – Localizar servidores em órgãos colegiados
Descrição:	Este caso de uso é utilizado para realizar a busca por servidores em órgãos colegiados.
Atores Envolvidos:	Gabinete
Pré-condição:	O responsável no gabinete deve estar autenticado no sistema.
Pós-condição:	Os dados inseridos na pesquisa, devem trazer tudo relacionado a palavra-chave utilizada.
Fluxo da tarefa principal:	1. O usuário clica na opção “Órgão colegiado” no site. 2. O usuário preenche o campo de pesquisa. 3. O usuário deve clicar em pesquisar. 4. O sistema deve trazer todos os dados de acordo com a pesquisa.
Fluxo alternativo:	1. O usuário deixa o campo em branco ou informa caracteres inválidos. 2. O sistema deve informar que o campo está em branco ou que os dados digitados não correspondem a um cadastro no sistema. 3. O usuário deve corrigir o campo de pesquisa e buscar novamente. 4. O sistema deve trazer os dados solicitados.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Casos de uso CGP

Nome:	UCCGP01 – Cadastrar servidor
Descrição:	Este caso de uso é utilizado para realizar o cadastro de servidores no sistema.
Atores Envolvidos:	CGP
Pré-condição:	O responsável no CGP deve estar autenticado no sistema.
Pós-condição:	O novo cadastro deve estar disponível no sistema e o servidor deve estar apto a usar a aplicação.
Fluxo da tarefa principal:	1. O usuário clica na opção “Servidores” no site. 2. O usuário preenche os campos requeridos. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a criação.
Fluxo alternativo:	1. O usuário não preenche corretamente um campo ou deixa em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a criação.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCCGP02 – Atualizar servidor
Descrição:	Este caso de uso é utilizado para realizar alterações no cadastro de servidor no sistema.
Atores Envolvidos:	CGP
Pré-condição:	O responsável no CGP deve estar autenticado no sistema.
Pós-condição:	O cadastro alterado deve ter seus dados alterados disponíveis para visualização.
Fluxo da tarefa principal:	1. O usuário clica na opção “Servidores” no site. 2. O usuário atualiza o campo status. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário não preenche corretamente um campo ou deixa em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCCGP03 – Ativar/inativar servidor
Descrição:	Este caso de uso é utilizado para realizar alteração de status do servidor no sistema.
Atores Envolvidos:	CGP
Pré-condição:	O responsável no CGP deve estar autenticado no sistema.
Pós-condição:	O cadastro escolhido deve sofrer alteração em seu status de cadastro para ativo ou inativo.
Fluxo da tarefa principal:	1. O usuário clica na opção “Servidores” no site. 2. O usuário altera o campo status. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera o campo status sem salvar. 2. Usuário sai da tela de cadastro e realiza teste na recente alteração. 3. Não vai haver alteração, pois a mesma não foi salva. 4. O usuário deverá repetir a alteração e salvar. 5. O sistema confirma a alteração realizada.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCCGP04 – Cadastrar setor
Descrição:	Este caso de uso é utilizado para realizar o cadastro de setores no sistema.
Atores Envolvidos:	CGP
Pré-condição:	O responsável no CGP deve estar autenticado no sistema.
Pós-condição:	O novo cadastro deve estar disponível no sistema e o setor deve estar disponível para quaisquer atividades no sistema.
Fluxo da tarefa principal:	1. O usuário clica na opção “Setores” no site. 2. O usuário preenche os campos requeridos. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a criação.
Fluxo alternativo:	1. O usuário não preenche corretamente um campo ou deixa em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a criação.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCCGP05 – Atualizar setor
Descrição:	Este caso de uso é utilizado para realizar alterações no cadastro de setores no sistema.
Atores Envolvidos:	CGP
Pré-condição:	O responsável no CGP deve estar autenticado no sistema.
Pós-condição:	O setor alterado deve ter seus dados alterados disponíveis para visualização.
Fluxo da tarefa principal:	1. O usuário clica na opção “Setores” no site. 2. O usuário atualiza o campo status. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário não preenche corretamente um campo ou deixa em branco. 2. O sistema deve informar que há irregularidades no cadastro. 3. O usuário deve corrigir as informações 4. O sistema deve confirmar a alteração.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCCGP06 – Ativar/inativar setor
Descrição:	Este caso de uso é utilizado para realizar alteração de status de setores no sistema.
Atores Envolvidos:	CGP
Pré-condição:	O responsável no CGP deve estar autenticado no sistema.
Pós-condição:	O setor escolhido deve sofrer alteração em seu status de cadastro para ativo ou inativo.
Fluxo da tarefa principal:	1. O usuário clica na opção “Setores” no site. 2. O usuário altera o campo status. 3. O usuário deve salvar os dados. 4. O sistema deve confirmar a alteração.
Fluxo alternativo:	1. O usuário altera o campo status sem salvar. 2. Usuário sai da tela de cadastro e realiza teste na recente alteração. 3. Não vai haver alteração, pois a mesma não foi salva. 4. O usuário deverá repetir a alteração e salvar. 5. O sistema confirma a alteração realizada.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UCCGP07 – Localizar servidores em setores
Descrição:	Este caso de uso é utilizado para realizar a busca por servidores em setores.
Atores Envolvidos:	CGP
Pré-condição:	O responsável no CGP deve estar autenticado no sistema.
Pós-condição:	Os dados inseridos na pesquisa, devem trazer tudo relacionado a palavra-chave utilizada.
Fluxo da tarefa principal:	1. O usuário clica na opção “Setores” no site. 2. O usuário preenche os dados do servidor. 3. O usuário deve clicar em pesquisar. 4. O sistema deve trazer todos os dados de acordo com a pesquisa.
Fluxo alternativo:	1. O usuário deixa o campo em branco ou informa caracteres inválidos. 2. O sistema deve informar que o campo está em branco ou que os dados digitados não correspondem a um cadastro no sistema. 3. O usuário deve corrigir o campo de pesquisa e buscar novamente. 4. O sistema deve trazer os dados solicitados.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Casos de uso compartilhados
CGP, D1 e Gabinete

Nome:	UC01 – Gerar relatórios estatísticos dos Campus
Descrição:	Este caso de uso é utilizado para realizar a geração de relatórios estatísticos do Campus.
Atores Envolvidos:	CGP, DI e Gabinete
Pré-condição:	O usuário deve pertencer à pelo o menos um dos cargos citados nos atores e autenticado no sistema.
Pós-condição:	O relatório deve selecionado deve estar disponível para visualização.
Fluxo da tarefa principal:	1. O usuário clica na opção “Relatórios” no site. 2. O usuário seleciona o relatório desejado. 3. O usuário deve clicar em gerar. 4. O sistema deve apresentar em tela o relatório citado
Fluxo alternativo:	1. O usuário informa um período do relatório inválido ou deixa campo em branco. 2. O sistema deve alertar a inconsistência digitada. 3. O usuário deve corrigir os campos e solicitar relatório novamente. 4. O sistema deve apresentar o relatório.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UC02 – Gerar relatórios de dados funcionais dos servidores
Descrição:	Este caso de uso é utilizado para realizar a geração de relatórios dados funcionais dos servidores.
Atores Envolvidos:	CGP, DI e Gabinete
Pré-condição:	O usuário deve pertencer à pelo o menos um dos cargos citados nos atores e autenticado no sistema.
Pós-condição:	O relatório deve selecionado deve estar disponível para visualização.
Fluxo da tarefa principal:	1. O usuário clica na opção “Relatórios” no site. 2. O usuário seleciona o relatório desejado. 3. O usuário deve clicar em gerar. 4. O sistema deve apresentar em tela o relatório citado
Fluxo alternativo:	1. O usuário informa o dado inválido de um servidor. 2. O sistema deve alertar a inconsistência digitada. 3. O usuário deve corrigir o campo e solicitar relatório novamente. 4. O sistema deve apresentar o relatório.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.

Nome:	UC03 – Gerar relatórios de dados acadêmicos dos servidores
Descrição:	Este caso de uso é utilizado para realizar a geração de relatórios dados acadêmicos dos servidores.
Atores Envolvidos:	CGP, DI e Gabinete
Pré-condição:	O usuário deve pertencer à pelo o menos um dos cargos citados nos atores e autenticado no sistema.
Pós-condição:	O relatório deve selecionado deve estar disponível para visualização.
Fluxo da tarefa principal:	1. O usuário clica na opção “Relatórios” no site. 2. O usuário seleciona o relatório desejado. 3. O usuário deve clicar em gerar. 4. O sistema deve apresentar em tela o relatório citado
Fluxo alternativo:	1. O usuário informa o dado inválido de um servidor. 2. O sistema deve alertar a inconsistência digitada. 3. O usuário deve corrigir o campo e solicitar relatório novamente. 4. O sistema deve apresentar o relatório.
Requisitos:	O usuário deve estar autenticado no sistema. O usuário deve possuir permissão para visualizar o módulo. O usuário deve possuir permissão de alteração.