



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

SIAPE - Sistema Integrado para Apoio a Projetos de Extensão

Emanoel J. C. Lazaro e Gabriel P. Santos

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dra. Carla Cavalcante Koike

Brasília
2023



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

SIAPE - Sistema Integrado para Apoio a Projetos de Extensão

Emanoel J. C. Lazaro e Gabriel P. Santos

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Dra. Carla Cavalcante Koike (Orientador)
CIC/UnB

Prof. Dr. Guilherme N. Ramos Prof. Dr. Genaina Nunes Rodrigues
Universidade de Brasília Universidade de Brasília

do Bacharelado em Ciência da Computação

Brasília, 11 de agosto de 2023

Resumo

Este trabalho apresenta o desenvolvimento de um software para gerar indicadores e estatísticas de extensão universitária, visando permitir a análise, o planejamento e a tomada de decisões. A proposta é que o software reúna as informações e dados relacionados aos projetos de extensão, permitindo um acesso facilitado e a gestão desses dados de forma eficiente. Além disso, o software disponibiliza ferramentas para acompanhamento dos indicadores dos projetos, o que permitirá uma visão mais clara e precisa do desenvolvimento da extensão e auxiliará na tomada de decisões. Este trabalho abordará questões técnicas, de gestão de projetos e também irá contribuir para a melhoria do processo de gestão de projetos de extensão universitária.

Palavras-chave: Extensão universitária, Indicadores, Acompanhamento

Abstract

This work presents the development of a software for managing university extension projects, aiming to optimize the work of the employees involved in these activities. The proposal is for the software to gather all information and data related to extension projects in one place, allowing easier access and efficient management of this data. In addition, the software will provide tools for monitoring the progress of projects, which will allow a clearer and more accurate view of the development of each project and will help in decision-making. This work will address technical and project management issues and will also contribute to the improvement of the university extension project management process.

Keywords: LaTeX, scientific method, thesis, sigaa, extension

Sumário

1	Introdução	1
1.1	Objetivos	3
1.2	Organização do Trabalho	3
2	Referencial Teórico	4
2.1	Indicadores	4
2.2	Visualização de Dados	5
2.3	Acessibilidade de Informações	5
2.4	Tecnologias Utilizadas	6
2.5	Metodologia Ágil	8
3	Requisitos do Sistema	10
3.1	Necessidades	10
3.1.1	Construção de Indicadores	12
3.1.2	Indicadores do Projeto	13
3.2	Restrições e limitações	15
3.3	Requisitos funcionais	16
3.4	Requisitos de sistema	17
4	Desenvolvimento	19
4.1	Script	21
4.1.1	Caminho para os dados pelo SIGAA	22
4.1.2	Coletando os dados	24
4.1.3	Objeto Coletado	25
4.1.4	Testes Unitários	27
4.2	Back-end	28
4.2.1	Node.js	29
4.2.2	Modelo MVC	29
4.2.3	Organização/Estrutura de código	30
4.2.4	Banco de dados	31

4.2.5	JSON	32
4.2.6	Autenticação	32
4.3	Organizando demandas (KanBan)	33
4.4	Interface Front-End	34
4.4.1	Utilização do Next.js	34
4.4.2	UX Design	36
4.4.3	Organização/Estrutura de código	37
4.4.4	Autenticação	39
4.4.5	Google Charts	40
4.4.6	Exportação de Dados	41
5	Resultados	43
5.1	Tela inicial	44
5.2	Resultados indicador de docentes	45
5.3	Resultados - Indicador de Discentes	47
5.4	Resultados - Indicador de Externos	49
5.5	Resultados - Indicador de Público	50
5.6	Resultados - Indicador de Financiamento	51
5.7	Evolução dos projetos	52
5.8	Tempo de execução do <i>script</i>	53
6	Conclusão	54
	Referências	56

Lista de Figuras

3.1	Exemplo da planilha de indicadores fornecidos pelo DEX.	12
4.1	Menu inicial SIGAA.	22
4.2	Filtros para selecionar projetos/eventos de extensão.	23
4.3	Lista de projetos de extensão do SIGAA.	23
4.4	Parte do objeto coletado pelo script.	26
4.5	Estrutura principal do código de <i>back-end</i>	30
4.6	Tabela de usuários simples, para que o usuário possa realizar o login. . . .	31
4.7	KanBan do projeto.	33
4.8	Página de docentes utilizando-se de SSR.	35
4.9	Página inicial.	36
4.10	Tela de login.	37
4.11	Tela de métricas gerais.	37
4.12	Estrutura principal de código.	38
4.13	Conteúdo que aparece ao clicar em "Gerar PDF".	41
4.14	PDF exemplo gerado.	42
5.1	Tela de apresentação.	43
5.2	Dashboard inicial.	44
5.3	Tela de docentes.	46
5.4	Seleção de departamentos.	46
5.5	Evolução de docentes em projetos de extensão.	47
5.6	Indicador de Discentes.	48
5.7	Gráfico evolução de Discentes.	48
5.8	Indicador de Externos.	49
5.9	Gráfico evolução de Externos.	50
5.10	Métricas de público.	50
5.11	Gráfico público.	51
5.12	Gráfico de tipos de financiamentos (projetos de extensão).	52

5.13 Gráfico evolução de projetos. 53

Capítulo 1

Introdução

No âmbito universitário, a extensão é uma iniciativa que tem se destacado como uma importante forma de aproximação entre a instituição de ensino e a comunidade externa. A extensão universitária se caracteriza por ações que envolvem o ensino, a pesquisa e a prestação de serviços à sociedade, buscando uma interação mais próxima entre a academia e a comunidade[1]. Essas ações são chamadas ações de extensão e podem envolver diversas áreas do conhecimento e ter objetivos variados, como a promoção do desenvolvimento regional, a melhoria da qualidade de vida das pessoas, entre outros.

A extensão universitária desempenha um papel fundamental, sendo uma iniciativa que estabelece uma ponte significativa entre a universidade e a comunidade externa e interna. Em um contexto em que a interação e o envolvimento com a sociedade são cada vez mais valorizados [2], a extensão se destaca como uma forma essencial de colocar em prática o conhecimento acadêmico, levando-o além dos muros da instituição de ensino. Por meio de ações de extensão, a universidade consegue promover ações que beneficiam a comunidade, seja através de serviços prestados, solução de problemas locais ou promoção do desenvolvimento regional. Dessa forma, a extensão universitária é uma iniciativa que demonstra o compromisso da universidade em estabelecer uma relação de reciprocidade e troca de conhecimentos com a sociedade, buscando um impacto positivo e uma transformação social significativa.

Em dezembro de 2018, foi estabelecido que as atividades de extensão devem compor, no mínimo, 10% (dez por cento) do total da carga horária curricular estudantil dos cursos de graduação, as quais deverão fazer parte da matriz curricular dos cursos [3], e a partir de janeiro de 2023, essa resolução foi aplicada na Universidade de Brasília. Essa determinação visa atender ao princípio constitucional da indissociabilidade entre ensino, pesquisa e extensão, assegurando que as instituições cumpram seu papel de promover a integração e interação entre essas três importantes dimensões acadêmicas [4]. Além disso, o Decanato de Extensão (DEX) da Universidade de Brasília anunciou a abertura do edital para o

Programa Institucional de Bolsas de Extensão (Pibex), promovendo a disponibilidade de diversas bolsas para incentivar a participação de estudantes em Projetos de Extensão de Ação Contínua (Peac) [5]. Esse número representa um aumento significativo de mais de 15% em comparação com o ano anterior (2022). A iniciativa tem como objetivo atender a todas as ações e programas atualmente cadastrados no SIGAA (Sistema Integrado de Gestão de Atividades Acadêmicas), possibilitando a concessão de pelo menos uma bolsa de extensão para cada um deles.

Diante das mudanças atuais mencionadas, a procura crescente por atividades de extensão será iminente, demandando uma gestão eficiente por parte dos responsáveis pelas ações e recursos. Nesse contexto, é imprescindível que os gestores tenham à disposição ferramentas eficazes para auxiliá-los na administração dessas atividades. A tecnologia surge como uma importante aliada na concepção de soluções capazes de atender a essas demandas.

Os gestores e tomadores de decisão responsáveis pela extensão na Universidade de Brasília (UnB) enfrentam dificuldades em monitorar, organizar e analisar os dados da universidade, devido às limitações das ferramentas disponíveis atualmente. Essas limitações resultam em trabalhos cansativos e na falta de acesso a dados básicos essenciais para obter uma visão abrangente do desenvolvimento da extensão e tomar decisões informadas. Como consequência, a análise de dados relacionados à extensão universitária é prejudicada por lacunas na coleta de informações através do sistema SIGAA.

O software desenvolvido visa auxiliar os gestores responsáveis pelo gerenciamento de ações de extensão universitária, oferecendo uma ferramenta eficiente e prática para o acompanhamento de atividades e recursos, esse acompanhamento é realizado através dos indicadores. Os indicadores são métricas que são retiradas de uma análise geral das ações de extensão, tais como fonte de financiamento, porcentagem de ações para público externo e interno, quantidade de ações por departamentos e muitos outros indicadores. Os funcionários sofrem de grande dificuldade para o levantamento desses indicadores com o SIGAA, pois exige trabalho manual repetitivo por conta da necessidade de acessar as diversas ações individualmente, obter os dados e calcular os indicadores

Com a utilização do Next.js, que é uma estrutura de desenvolvimento web baseada em React, é possível construir interfaces de usuário modernas e responsivas, que se adaptam a diferentes dispositivos e tamanhos de tela. Já o uso de *scripts* Python com Selenium permite a coleta automatizada de dados do SIGAA, o que pode ser muito útil para a obtenção de informações relevantes para o gerenciamento das ações de extensão. A partir da construção do software, espera-se contribuir para a melhor gestão das ações de extensão universitária, promovendo facilitação em relação à integração entre a universidade e a comunidade externa. Além disso, a utilização de tecnologias modernas e práticas de

desenvolvimento de software pode servir como uma referência para outras instituições de ensino que também enfrentam desafios semelhantes na gestão de ações de extensão universitária.

Espera-se que a construção desse software contribua com a gestão e entrega dos indicadores para os responsáveis ou a quem interessa os dados, facilitando a integração entre a universidade e a comunidade externa. Além disso, o uso de tecnologias modernas e práticas no desenvolvimento de software pode servir como referência para outras instituições de ensino que enfrentam desafios semelhantes na gestão de ações de extensão universitária.

1.1 Objetivos

Esse trabalho tem como objetivo a construção de um software capaz de construir e apresentar dados importantes para os usuários, criados a partir de indicadores, com o intuito de melhorar a experiência desses usuários e fornecer facilidades na análise de dados. O *software* tem como fundamento padrões de UX (*User Experience*) e foco no usuário, a fim de produzir uma interface limpa e clara, com padrões estéticos agradáveis, dessa forma a experiência desses usuários pode ser boa ao utilizar o sistema proposto, reconhecendo os dados de maneira clara e intuitiva, melhorando assim o cotidiano desses profissionais.

1.2 Organização do Trabalho

Este artigo está estruturado em cinco seções principais, em cada uma será apresentado pontos importantes para o projeto desenvolvido. Na primeira seção, Referencial Teórico, serão apresentados conceitos e técnicas e conhecimentos relevantes que foram utilizados na construção desse projeto. Na segunda seção, Requisitos do Sistema, todas as necessidades do sistema serão apresentadas, bem como limitações localizadas e exigências. Na terceira seção, Desenvolvimento, uma descrição de todo o processo de construção do *software* será apresentada, destacando as decisões tomadas. Com isso, na seção de Resultados todos os dados gerados pelo trabalho serão apresentados e explicados, assim sustentando todo o processo apresentado nas seções anteriores. Por fim, na seção de conclusão, as considerações finais do trabalho estão apresentadas, dando assim um encerramento apropriado e explicativo sobre o projeto

Capítulo 2

Referencial Teórico

O referencial teórico é uma seção fundamental para embasar o desenvolvimento do projeto, fornecendo suporte conceitual e teórico para todas as seções posteriores. Nesta seção, serão apresentados conceitos e teorias relacionados ao desenvolvimento do projeto, que foca em aspectos de projetos de extensão universitária, bem como às tecnologias utilizadas no desenvolvimento do sistema.

2.1 Indicadores

No contexto da extensão universitária, a busca por métodos para avaliar e comunicar o engajamento das ações se torna crucial para os tomadores de decisão [6]. Nesse sentido, a utilização de indicadores é uma ferramenta indispensável, eles não apenas informam dados genéricos, mas também possibilitam o entendimento de cenários mais específicos. Por meio da representação de dados quantitativos e qualitativos, os indicadores permitem uma análise mais profunda e informada das contribuições da extensão universitária para a sociedade e o crescimento institucional. O objetivo desta seção é apresentar o papel crucial dos indicadores para a avaliação das iniciativas de extensão.

Indicadores, como instrumentos de avaliação, desempenham um papel crucial na descrição e medição de aspectos diversos. São abstrações ou parâmetros representativos que quantificam características, resultados ou desempenho do que se tem interesse em avaliar. No contexto da extensão universitária, os indicadores têm sido buscados para mensurar efetivamente a amplitude das ações e os impactos por elas causados, possibilitando um maior entendimento do cenário de extensão na universidade em que são aplicados. Desta forma, essas informações podem auxiliar o trabalho de tomadores de decisão da área.

A busca por indicadores apropriados tem enfrentado desafios, uma vez que não existe um método universal ou padrão único de construção. O Fórum de Pró-Reitores de Extensão das Instituições Públicas de Educação Superior Brasileiras (FORPROEX) desempe-

nhou um papel central nesse processo [7], estabelecendo grupos de trabalho para discutir a avaliação da extensão e sua consequente instrumentalização por meio de indicadores. Esse caminho de discussões resultou em iniciativas como o Grupo de Trabalho Interinstitucional sobre Indicadores Brasileiros de Extensão Universitária (GT IBEU), que buscou validar indicadores que atendessem aos diferentes contextos e regiões do Brasil, promovendo assim a avaliação e gestão da Extensão Universitária em nível nacional.

2.2 Visualização de Dados

A visualização de dados desempenha um papel fundamental na comunicação eficaz de informações complexas. Através da representação visual de dados por meio de gráficos, tabelas e outros elementos visuais, é possível identificar padrões, tendências e *insights* relevantes. Explorar os princípios e as melhores práticas nesse campo permite criar visualizações impactantes e significativas.

Os gráficos escolhidos para a plataforma têm como base as ideias de Edward Tufte, um pioneiro na visualização de dados. Em seu livro *The Visual Display of Quantitative Information* [8], Tufte destaca a importância da clareza, precisão e efetividade na comunicação de informações por meio de gráficos. Ele enfatiza que a simplicidade e a elegância são elementos essenciais para uma visualização bem-sucedida, ressaltando que os gráficos mais simples tendem a ser os mais eficazes. Assim, ao desenvolver a plataforma, foi priorizado o uso de gráficos de torta, bem como gráficos de barras horizontais e verticais, pois essas representações visuais proporcionam uma apresentação clara e concisa dos dados, permitindo uma fácil interpretação e análise dos resultados.

Outra referência valiosa no campo da visualização de dados é o livro *Data Visualization: A Practical Introduction* de Kieran Healy [9]. Healy apresenta um guia abrangente sobre os conceitos fundamentais da visualização de dados. Ele explora a seleção de gráficos apropriados para diferentes tipos de dados, a importância da escolha de cores adequadas, a construção de narrativas visuais envolventes e a consideração do *design* responsivo. Ao incorporar esses princípios na plataforma, foi tomado como objetivo oferecer uma interface intuitiva e atraente para apresentar os dados aos usuários.

2.3 Acessibilidade de Informações

A acessibilidade de informações é um aspecto crucial ao projetar uma plataforma que visa facilitar o acesso e a compreensão dos dados. Ao aplicar os conceitos de arquitetura de informações e usabilidade, a plataforma foi desenvolvida com o objetivo de garantir que os usuários possam acessar os dados de forma intuitiva e eficiente.

No livro *Don't Make Me Think: A Common Sense Approach to Web Usability* [10] de Krug, é enfatizada a importância de criar interfaces intuitivas e fáceis de usar. Ao implementar os princípios de usabilidade propostos por Krug, como a clareza das informações, a simplicidade da navegação e a redução da carga cognitiva, a plataforma se torna mais acessível e amigável para os usuários. Um exemplo disso foi a redução do número de menus, simplificando a interface e tornando as informações disponíveis de forma mais acessível.

As referências citadas fornecem orientações sobre como representar visualmente os dados de maneira clara e eficaz. Ao utilizar técnicas gráficas apropriadas, seleção adequada de gráficos, escolha de cores relevantes e *design* responsivo, é possível que os dados sejam apresentados de forma compreensível e atraente. Isso permite que os usuários explorem visualmente os dados, identifiquem padrões e tomem decisões informadas com base nas informações apresentadas.

2.4 Tecnologias Utilizadas

No desenvolvimento da plataforma, são utilizadas tecnologias como Node.js, Python, Selenium, React e Next.js. Para compreender essas tecnologias e seu papel no projeto, as seguintes referências são relevantes:

Para o *back-end* foi utilizado o Node.js, um moderno *framework* Javascript. Foi escolhida essa linguagem para a construção do *back-end* por conter muitas referências teóricas e padrões de projeto já consolidados no desenvolvimento de uma API. Uma API (*Application Programming Interface*) é um componente essencial no desenvolvimento de aplicações modernas, permitindo a integração e comunicação entre diferentes sistemas de forma padronizada e segura. No âmbito do desenvolvimento web, uma API construída utilizando Node.js desempenha um papel fundamental ao servir como uma ponte de conexão entre o *front-end* e o *back-end* de um aplicativo. Por meio dessa API, o *front-end* pode enviar requisições ao servidor, solicitando dados ou a execução de determinadas ações, e receber respostas correspondentes [11]. A utilização de Node.js como tecnologia para a construção da API oferece uma abordagem eficiente e escalável, possibilitando a implementação de funcionalidades complexas e aprimorando a experiência do usuário na interação com a aplicação web.

O livro *Node.js in Action* [12] explora o ecossistema do Node.js, abordando desde os conceitos básicos até tópicos avançados, como criação de APIs e gerenciamento de dependências. Essa obra fornece uma base sólida para o entendimento e a aplicação efetiva do Node.js no desenvolvimento da plataforma. Além do motivo da quantidade de

informação disponível sobre a tecnologia, foi escolhido o Node.js por conta da familiaridade da equipe com a linguagem Javascript.

Foi utilizado Python para desenvolvimento do *script* que captura os dados do SIGAA, e foi escolhido por ser popular para *scripts* de automatização devido à sua simplicidade, legibilidade, bibliotecas padrões consolidadas, e integração com outras linguagens e sistemas. Sua popularidade e a comunidade ativa de desenvolvedores também contribuem para a disponibilidade de recursos e bibliotecas especializadas em automação, em conjunto com a biblioteca Selenium, que simula a interação do usuário com páginas web de forma . Em relação ao Python e ao Selenium, o livro *Automate the Boring Stuff with Python* de Sweigart [13] é uma referência útil. Ele apresenta técnicas de automação de interfaces web e apresenta tutoriais da melhor forma de se fazer esse tipo de automação, e foi utilizado para compreender todos os comandos e padrões de código que o *Selenium* possui e recomenda.

Pelos mesmos motivos da escolha de *frameworks* consolidados para serem utilizados neste projeto, foi escolhido o React para ser a base do *front-end*, pois possui amplos tutorias e padrões de código já conhecidos. O livro *Learning React: Functional Web Development with React and Redux* de Banks e Porcello) [14] é uma referência abrangente que foi utilizada durante todo o desenvolvimento, onde aborda conceitos fundamentais do React e explora a criação de componentes reutilizáveis e o gerenciamento de estados globais com o uso de *Context*, que serão melhor explicados na seção de desenvolvimento. Ainda sobre o desenvolvimento do *front-end*, foi utilizado também o livro *Next.js Quick Start Guide* de Qian [15], que fornece um guia prático para o uso do Next.js no desenvolvimento de aplicações web, no qual explora recursos como roteamento, renderização do lado do servidor e otimizações de desempenho, que foram muito necessárias para o desenvolvimento do projeto, devido ao grande número de dados a serem renderizados e processados.

Além disso, foram utilizadas bibliotecas que facilitaram e aceleraram o desenvolvimento, como o Material-UI, que funciona como um conjunto de componentes que se acoplam na aplicação, tornando a interface do usuário mais amigável e intuitiva. A combinação dessas referências teóricas e práticas contribuiu para o desenvolvimento da plataforma, abordando aspectos de acessibilidade de informações, usabilidade, visualização de dados e aplicação eficaz das tecnologias escolhidas. Ao utilizar esses recursos, a plataforma será capaz de fornecer aos usuários uma experiência de uso intuitiva e facilitada na obtenção e análise dos dados, contribuindo para uma melhor tomada de decisões e aprimoramento do cenário de extensão na Universidade de Brasília.

2.5 Metodologia Ágil

A aplicação dos conceitos de engenharia de *software* desempenhou um papel fundamental no desenvolvimento da plataforma, proporcionando um processo de desenvolvimento organizado, de fácil manutenção e escalabilidade. Ao adotar os princípios e práticas da engenharia de *software*, busca-se garantir a qualidade, eficiência e escalabilidade do *software* desenvolvido. Nessa seção serão apresentados os métodos utilizados para a produção do sistema.

Durante o desenvolvimento do projeto, foram aplicados diversos processos de engenharia de *software* que contribuíram para o sucesso da plataforma. Desde o início da construção do projeto, foi utilizada a abordagem ágil de desenvolvimento de *software*, que se baseia na colaboração e adaptação contínua. Isso permitiu responder de forma rápida e eficiente as mudanças e necessidades do projeto, resultando em uma solução mais flexível e alinhada às demandas dos usuários.

A utilização de metodologias ágeis, como Scrum e Kanban, desempenhou um papel fundamental no desenvolvimento do *software*. O Scrum, descrito por Schwaber e Sutherland [16], é uma metodologia de trabalho que divide o projeto em *sprints*, permitindo um planejamento mais eficiente, maior transparência e maior flexibilidade para lidar com mudanças durante o desenvolvimento. Através do Scrum, a equipe foi capaz de estabelecer um ritmo de trabalho consistente, realizando entregas incrementais ao final de cada *sprint*, o que facilitou a identificação e resolução de problemas de forma rápida. Além disso, o Kanban, que é baseado nos princípios da gestão visual, proporcionou um fluxo contínuo de trabalho, permitindo visualizar o progresso das tarefas em um quadro e facilitando a distribuição equilibrada das atividades entre os membros da equipe. Para a organização e definição das tarefas, foi utilizado o quadro Scrum disponibilizado no Teams, garantindo uma melhor colaboração e acompanhamento do progresso do projeto.

Além disso, foi utilizado o conceito de *pair programming* ao longo de todo o projeto. Essa prática envolveu a colaboração entre dois programadores trabalhando juntos no mesmo código. O *pair programming* promoveu a troca constante de conhecimento, a revisão contínua do código e a busca por soluções mais eficientes e elegantes. Essa abordagem resultou em um código de melhor qualidade, reduzindo erros e facilitando a manutenção futura.

A gestão de projetos também desempenhou um papel crucial no desenvolvimento da plataforma. Foram utilizadas técnicas de gestão de projetos, como a definição de escopo, alocação de recursos e acompanhamento do progresso, para garantir que o projeto fosse entregue dentro dos prazos e orçamentos estabelecidos. A utilização de ferramentas de gestão, como quadros Kanban e reuniões diárias de acompanhamento, facilitou a comu-

nicação entre os membros da equipe e permitiu um controle mais eficiente do progresso do projeto.

Além dos processos mencionados, também foram aplicadas boas práticas de codificação, como revisões de código, testes automatizados e o uso de padrões de projeto. Essas práticas ajudaram a garantir a qualidade do *software*, reduzindo erros e facilitando a manutenção. Também houve uma preocupação com a organização do código fonte, seguindo uma estrutura componentizada (isto é, que utiliza componentização em sua organização) e utilizando pastas *src* tanto no React quanto no Node.js. Isso permitiu uma melhor organização e reutilização de código, evitando a criação de um monólito (sistema único, não dividido, que roda em um único processo) e facilitando a adição de novos recursos e funcionalidades no futuro.

Em suma, a aplicação dos processos de engenharia de *software*, como a abordagem ágil, o modelo MVC, o *pair programming*, a gestão de projetos e as boas práticas de codificação, foram fundamentais para o sucesso do desenvolvimento da plataforma de acompanhamento de ações de extensão universitária. Esses processos contribuíram para a criação de uma solução de alta qualidade, eficiente, flexível e fácil de manter.

Por fim, é importante destacar a relevância da análise de dados no contexto da extensão universitária. Diversos estudos têm mostrado que a análise de dados pode ajudar na tomada de decisão e na melhoria da gestão de projetos de extensão universitária. Nesse sentido, a utilização de gráficos e tabelas pode ser uma forma eficiente de apresentar dados de forma clara e objetiva, facilitando a compreensão e tomada de decisão por parte dos gestores e envolvidos no processo de extensão universitária.

Nesta seção foram apresentados todos os materiais teóricos necessários para a construção do sistema, bem como a metodologia de desenvolvimento aplicada ao projeto, ao fim desta seção é possível compreender quais foram as tecnologias empregadas, bem como a explicação de sua utilização.

Capítulo 3

Requisitos do Sistema

O levantamento de requisitos é uma etapa crucial no processo de desenvolvimento de *software*, pois é nesse momento que as necessidades, expectativas e restrições dos *stakeholders* são identificadas e documentadas. Essa atividade é essencial para o sucesso do projeto, uma vez que define a base para o planejamento, a implementação e a entrega de um sistema que atenda às demandas dos usuários e aos objetivos estabelecidos.

Nessa seção, inicialmente serão abordados os seguintes tópicos:

- **Compreensão das necessidades dos usuários:** Entender as necessidades dos usuários finais, que serão os coordenadores de extensão e gestores do DEX, seus objetivos e os problemas que estão enfrentando. Isso garantirá que o *software* seja projetado e desenvolvido para atender a essas necessidades específicas, aumentando a satisfação do usuário.
- **Definição clara do escopo do projeto:** O levantamento de requisitos ajuda a estabelecer o escopo do projeto, ou seja, o conjunto de funcionalidades e características que serão incluídas no *software*, evitando um escopo mal definido.
- **Identificação de restrições e limitações:** Durante o levantamento de requisitos, são identificadas as restrições e limitações que podem impactar o desenvolvimento e a implementação do *software*. Ao considerar essas restrições desde o início, é possível tomar decisões informadas e realistas ao planejar o projeto.

3.1 Necessidades

Para o possível desenvolvimento da aplicação, primeiro foi iniciado um estudo focado no usuário, onde através de reuniões mediadas pela professora orientadora do trabalho de conclusão, com possíveis usuários integrantes do DEX, estes que são o diretor técnico de extensão e membros da comissão de indicadores de extensão, que através do contato

diretamente com a orientadora deste trabalho, pode-se obter algumas respostas de início. Foi notado através desses *feedbacks*, que o usuário tinha uma dificuldade preocupante em analisar e obter os dados de extensão que ocorrem na Universidade de Brasília (UnB), tais dados não eram disponibilizados e apresentados de maneira eficaz para o usuário, assim dificultando uma certa monitoração e possivelmente identificação de pontos importantes que envolvam correções. Com isso foram avaliadas, de início, as principais dores do usuário, que envolvem a falta de dados disponibilizados na plataforma SIGAA, a dificuldade de análise dos dados e a pouca intuitividade dos dados apresentados.

Após a obtenção dos requisitos passados pelos usuários, foi iniciada a produção de um projeto para atender essas necessidades e saná-las: um sistema capaz de capturar dados do SIGAA, preparar esses dados da melhor forma, e apresentar estes para os usuários da maneira mais clara e intuitiva possível, na forma de indicadores. Com isso estão destacados os requisitos mais práticos que estariam envolvidos no desenvolvimento do sistema, necessários para a construção do *front-end*, *back-end* e armazenamento dos dados.

Para esse projeto, foi tomado como ponto de partida uma planilha desenvolvida por alguns dos funcionários do DEX, contendo diversos indicadores relevantes para a gestão dos projetos de extensão. A planilha de indicadores contém um conjunto de métricas necessárias para as atividades de extensão realizadas pela universidade, como o percentual anual de ações de extensão com financiamento interno em relação às ações autofinanciadas, a quantidade de ações de extensão por unidade, o número de estudantes extensionistas por projeto, o total de pessoas diretamente atendidas pelos programas e projetos de extensão, o número de pessoas externas à UnB que atuaram como membros da equipe das ações de extensão, e diversas outras métricas gerais. Essas métricas são essenciais para a análise e tomada de decisões relacionadas às atividades de extensão e desejam tê-las de forma acessível e facilitada.

Com base nesses indicadores, é possível identificar as necessidades e demandas dos usuários do *software*, ou seja, dos funcionários que fazem o gerenciamento dos projetos de extensão. Por exemplo, o indicador referente à quantidade de ações de extensão por unidade pode indicar a necessidade de um filtro de busca que permita aos usuários selecionarem e visualizarem apenas os projetos de extensão de determinada unidade. Já o indicador sobre o número de estudantes extensionistas por projeto pode indicar a necessidade de uma funcionalidade que permita aos usuários gerenciar e atualizar a lista de estudantes envolvidos em cada projeto.

Assim, a planilha de indicadores se torna uma importante ferramenta para a definição dos requisitos do *software* e para garantir que ele atenda às necessidades dos usuários de forma eficiente e eficaz. Dentre estes indicadores, foi destacado a quantidade de docentes por departamento, que foi selecionado como um dos primeiros a ser implementado. Para

	Nome do indicador	Possível encontrá-lo?	Onde?	OBSERVAÇÕES
3	Índice de ações institucionalizadas no SIGAA em relação ao ano anterior	Sim	Portal do Discente > Consultar Ações de Extensão > Busca por Ações de Extensão > Ano	O próprio SIGAA informa que a consulta retorna muitos resultados, e é necessário restringir mais a busca. Portanto, como devemos fazer essa restrição? Por Tipo da Ação? Sugestão: área CNPQ
4	Índice do Total de ações institucionalizadas no ano / total de servidores/as ativos do DEX	Provável que sim com outro perfil	Referente ao total de ações deve-se seguir o mesmo caminho acima. Falta encontrar o caminho do total de servidores ativos no DEX	O problema encontra-se no que tange como encontrar o total de servidores ativos no DEX apenas com o perfil de aluno. Mas, em outro perfil deve ser possível encontrar esse dado facilmente
5	Índice do Percentual anual de ações de extensão com financiamento interno (aprovadas em editais do DEX) em relação às ações autofinanciadas	Provável que sim	Portal do Discente > Visualização da Ação de Extensão > DADOS DA AÇÃO DE EXTENSÃO > Fonte de Financiamento	Como vamos fazer para diferenciar as ações que receberam financiamento externo das que foram contempladas em edital? Na aba de fonte de financiamento nas que tem edital eles colocam essa informação em parênteses, então, talvez seja a única solução

Figura 3.1: Exemplo da planilha de indicadores fornecidos pelo DEX.

tal, foi desenvolvido um *script* em Python utilizando o *Selenium* para coletar os dados diretamente do sistema de gestão acadêmica da universidade (SIGAA). Com os dados em mãos, foram criados gráficos e tabelas no *front-end* da plataforma, que permitiram uma visualização mais clara e intuitiva da quantidade de docentes por departamento, além de outras informações estatísticas relevantes. Esse processo de busca e integração de dados foi essencial para garantir a qualidade e relevância das informações apresentadas na plataforma, tornando-a mais útil e efetiva para os usuários.

3.1.1 Construção de Indicadores

Como descrito, o presente projeto foi desenvolvido para atender aos indicadores, estes que foram criados em sua maioria pelo Fórum de Pró-Reitores de Extensão das Instituições Públicas de Educação Superior Brasileiras (FORPROEX) [7], enquanto outros indicadores foram criados pela comissão de indicadores do decanato de extensão, onde foi seguida o seguinte método proposto pelo artigo, que consiste em oito passos para a construção de indicadores:

1. Avaliar objetivos e metas;

2. Identificar os interessados;
3. Definir tipos de indicadores;
4. Definir critérios de seleção;
5. Mapear indicadores candidatos;
6. Realizar análise de *trade-off*;
7. Validar indicadores;
8. Cadastrar indicadores;

3.1.2 Indicadores do Projeto

Assim como já tratado nas seções anteriores, durante as reuniões de projeto, foram selecionados indicadores retirados do FORPROEX [7], estes que foram escolhidos por conta de sua simplicidade e objetividade, para servir como prova de conceito que outros indicadores mais complexos também poderiam ser implementados. Os resultados informados por esses indicadores serão compartilhados na seção de resultados.

Os indicadores selecionados do FORPROEX são:

- PA06 - Estudantes na extensão.
- PA08 - Docentes na extensão.
- PG11 - Recursos do orçamento anual público voltado para extensão.
- RUS7 - Público alcançado por programas e projetos.

A seguir uma listagem dos indicadores inspirados no FORPROEX para implantação nesse projeto:

- Índice do Total de docentes/discentes/externos em projetos institucionalizados no ano.
- Índice do Percentual anual de projetos de extensão com financiamento interno (aprovadas em editais do DEX).
- Índice do Percentual anual de projetos de extensão com financiamento externo (aprovadas em editais do DEX).
- Índice do Percentual anual de projetos de extensão autofinanciados.
- Índice do Percentual de público participante de projeto de extensão em cada ano (público interno, externo, estimado e atingido).

- Índice do Percentual anual de projetos de extensão com financiamento externo (aprovadas em editais do DEX) em relação às ações autofinanciadas.
- Índice do Percentual anual de ações de extensão com financiamento externo em relação às ações autofinanciadas.

O primeiro indicador é basicamente descobrir a quantidade de docentes envolvidos em projetos de extensão na Universidade de Brasília ao longo dos anos, sendo possível visualizar de maneira separada (por departamento) e global (todos os departamentos) a quantidade de docentes envolvidos nessas atividades com o passar dos anos, assim o usuário teria em sua posse uma informação não disponibilizada de maneira clara pelo SIGAA, visto que a plataforma da UnB só apresenta os docentes de cada projeto. Na plataforma, a ideia é que filtros interativos dessem ao usuário a liberdade da visualização personalizada dos dados, então o indicador terá de ser capaz de possibilitar visualizações por ano e por departamento, bem como o agrupamento de departamentos. Assim foi construído esse indicador se baseando nessas premissas, tal processo será explicado na seção Script e Back-end, que sucede essa.

Após a formulação do indicador de docentes, a lógica utilizada foi estendida para a implementação dos outros dois, que são os discentes e externos, também escolhidos para em conjunto com os usuários. O indicador de discentes é basicamente o mesmo de docentes, apenas mudando o contexto do objeto observado, pegando como parâmetro os alunos participantes da UnB em projetos de extensão, desta forma será possível analisar o crescimento ou não da participação dos alunos da universidade em projetos de extensão, vale ressaltar que o indicador também deverá suportar os filtros contidos no de docentes. O mesmo vale para o outro indicador, o de externos, neste foi analisado todo o público participante de projetos de extensão na Universidade de Brasília que são externos à mesma.

Tendo em posse esses três indicadores, foi iniciada a busca por mais métricas que seriam interessantes para os profissionais do DEX que seriam possíveis de formular a partir dos dados do SIGAA. A orientadora do projeto ajudou muito neste processo, pois como a mesma também atua em projetos de extensão, o trabalho sempre tem muito contato com as necessidades dos usuário, logo foram levantadas métricas mais específicas que serão abordadas a seguir. Durante as reuniões surgiu a necessidade de disponibilizar para o usuário a porcentagem de público e suas origens em projetos de extensão, para isso foi feita uma pesquisa sobre as informações contidas no SIGAA e assim foi notado que em cada projeto é tratado o público estimado externo, interno e público atingido, logo para a construção desse indicador é necessário coletar todos os dados de público de cada projeto em cada ano, com isso o usuário consegue ter a noção do impacto dos projetos de extensão observados, bem como o conjunto de todos os projetos, podendo saber se a

expectativa de público passa perto ou não do público real. Assim, mais uma métrica foi construída, e esta apresentou resultados bem interessantes, estes serão disponibilizados na seção de resultados.

Nas últimas métricas, que tem como ideia base calcular as origens dos financiamentos em projetos de extensão na UnB, foi feita uma busca no SIGAA para catalogar os tipos de financiamentos adotados nos projetos, com isso foi notado que existem três tipos principais, fonte de financiamento interno, externo e projetos autofinanciados, que coincidem com os levantado pelos usuários. Vale ressaltar que a construção idealizada dos indicadores foram feitas pelos usuários, nesse projeto alguns destes indicadores foram implementados, em conjunto com a orientadora.

3.2 Restrições e limitações

Durante o desenvolvimento do projeto, foi importante levar em consideração as restrições e limitações que podem impactar a implementação e o funcionamento do sistema. Foram encontradas restrições e limitações de natureza técnica, organizacional ou mesmo relacionadas aos recursos disponíveis, principalmente em relação ao tipo de conta para acesso ao sistema SIGAA. Ao identificar e compreender essas restrições desde o início do processo, é possível tomar decisões informadas e realistas, garantindo a viabilidade do projeto.

Uma das principais restrições técnicas é a integração com o sistema existente, o SIGAA. Embora o sistema proposto se baseie na coleta de dados do SIGAA, é importante considerar que o acesso a esses dados deve ser feito de acordo com as políticas de segurança e privacidade estabelecidas pela universidade. Será necessário garantir a conformidade com as diretrizes institucionais, bem como a integridade e confidencialidade dos dados dos usuários.

Durante o processo de desenvolvimento do *software* foi notado que o SIGAA altera alguns identificadores dos elementos de sua plataforma, assim, o sistema possui algumas limitações em seu funcionamento, visto que ao haver mudanças na plataforma SIGAA o *software* em desenvolvimento tem que passar por manutenção para se adequar. É possível localizar esses identificadores através da ferramenta inspecionar, disponível em navegadores de internet.

Outra restrição a ser considerada é a compatibilidade com diferentes dispositivos e navegadores. O sistema deve ser desenvolvido levando em conta a variedade de dispositivos utilizados pelos usuários, como desktops, laptops, tablets e smartphones, além de ser compatível com diferentes navegadores, como Chrome, Firefox, Safari e Edge. Garantir a responsividade e adaptabilidade do sistema para diferentes tamanhos de tela e resoluções é essencial para proporcionar uma experiência consistente e satisfatória aos usuários.

Além das restrições técnicas, é importante considerar as limitações organizacionais e recursos disponíveis. O projeto deve ser desenvolvido dentro de um cronograma definido, levando em conta as demandas e prioridades de outras iniciativas e projetos da instituição. É fundamental avaliar os recursos humanos disponíveis, incluindo a equipe de desenvolvimento, a disponibilidade dos usuários para testes e *feedback*, bem como o suporte técnico e infraestrutura necessários para a implementação e manutenção do sistema.

Outra limitação a ser considerada é a possibilidade de mudanças nos requisitos durante o processo de desenvolvimento. À medida que o sistema é desenvolvido e os usuários interagem com ele, é natural que surjam novas necessidades e demandas. É importante estar preparado para lidar com essas mudanças, adotando uma abordagem ágil e flexível, que permita a adaptação do sistema de acordo com as necessidades identificadas.

Por fim, é fundamental considerar as restrições orçamentárias. O projeto deve ser desenvolvido dentro dos recursos financeiros disponíveis, levando em conta os custos de desenvolvimento, manutenção e suporte ao sistema. É importante priorizar os recursos e funcionalidades mais relevantes e estratégicos para atender às necessidades dos usuários e garantir a sustentabilidade do projeto a longo prazo.

Ao reconhecer e abordar essas restrições e limitações desde o início do projeto, é possível garantir uma implementação bem-sucedida e alinhada às expectativas dos usuários e da instituição. A conscientização dessas restrições que permitem tomar decisões informadas e realistas, maximizando o potencial de sucesso do sistema de gerenciamento de projetos de extensão universitária.

Agora serão detalhados alguns levantamentos foram feitos antes e durante o desenvolvimento do projeto. Entre eles, requisitos funcionais, não funcionais, requisitos de relatórios/métricas e requisitos de integração.

3.3 Requisitos funcionais

Requisito funcional é uma especificação detalhada de uma função ou capacidade que o sistema de software deve fornecer. Esses requisitos descrevem o que o sistema deve fazer, as funcionalidades que ele deve possuir e as operações que ele deve realizar para atender às necessidades e aos objetivos do negócio ou dos usuários. Eles são uma parte fundamental do processo de engenharia de requisitos e servem como base para o desenvolvimento e teste do sistema de software, garantindo que ele cumpra as expectativas e requisitos do cliente ou do usuário final. [17]

Um dos maiores questionamentos durante o planejamento da aplicação, é a respeito do cadastro de usuários, e o que eles poderiam acessar. Levando em consideração que o usuários alvo são tomadores de decisão no cenário de extensão (funcionários do DEX, por

exemplo), foi decidido que os usuários administradores poderiam adicionar novos usuários, e estes usuários administradores teriam controle sobre o acesso .

Outro requisito funcional é a exportação de dados em formato Excel ou PDF, de acordo com os filtros que o usuário selecionou na sua página de pesquisa. Este que foi feito utilizando uma biblioteca de terceiros, chamada "@react-pdf", que pode ser encontrada e instalada no projeto através do gerenciador de pacotes que foi instalado no *front-end*, *yarn*. Será explicada e demonstrada melhor na seção de desenvolvimento.

3.4 Requisitos de sistema

Requisitos de sistema são especificações que descrevem atributos e características do sistema de software, que não estão diretamente relacionados com suas funcionalidades específicas, mas são igualmente essenciais para sua eficácia, desempenho e qualidade geral. Esses requisitos são essenciais para garantir a qualidade e a eficiência do sistema, bem como a satisfação do usuário. Eles fornecem critérios e diretrizes para avaliar aspectos críticos além das funcionalidades, considerando aspectos técnicos, de desempenho e de usabilidade. [17]

Alguns exemplos de requisitos de sistema incluem:

- **Desempenho:** O sistema deve ser capaz de responder de forma rápida e eficiente, lidando com cargas de trabalho esperadas e tempos de resposta aceitáveis. Ao longo do desenvolvimento, diversas fatorações no *front-end* e *back-end* foram alteradas, afim de melhorar a organização de código, e velocidade de carregamento das páginas, então houve um grande esforço para que a aplicação obtivesse o melhor desempenho possível.
- **Usabilidade:** A interface do usuário deve ser intuitiva, de fácil aprendizado e uso, garantindo uma experiência agradável para os usuários. Para isso, a plataforma utiliza uma biblioteca de geração de gráficos que se adaptava em tempo real aos filtros que o usuário selecionasse, e também foram utilizados componentes do Material UI, que disponibilizam componentes responsivos e esteticamente agradáveis.
- **Confiabilidade:** O sistema deve ser confiável, evitando falhas ou interrupções inesperadas e garantindo a integridade dos dados.
- **Segurança:** A aplicação deve ser projetada e implementada com medidas de segurança adequadas para proteger os dados sensíveis e prevenir acesso não autorizado.
- **Manutenção:** O sistema deve ser facilmente mantido e atualizado, permitindo a correção de erros, implementação de melhorias e inclusão de novas funcionalidades.

Esses requisitos são fundamentais para garantir o sucesso do sistema e proporcionar uma experiência satisfatória aos usuários, atendendo às demandas técnicas e operacionais do projeto. Também é possível citar alguns requisitos de integração [18] que fazem parte do projeto em questão, entre eles:

- **Integração com o SIGAA:** A plataforma deve ser capaz de se integrar com o SIGAA, permitindo a coleta automatizada de dados relevantes para os projetos de extensão, como informações sobre pessoas matriculadas, bem como todas as informações relevantes aos projetos em si.
- **Rotinas de automatização:** Foi criada uma automatização para que os dados da plataforma sejam atualizados uma vez por dia, uma vez que o *script* que traz esses dados demora cerca de 40 minutos para ser atualizado.
- **Autenticação e Controle de Acesso:** O sistema deve implementar mecanismos de autenticação seguros, como login e senha, para garantir que apenas usuários autorizados tenham acesso à plataforma.

Nessa seção de Requisitos do Sistema, todos os requisitos que o *software* precisa atender estão descritos e explicados, todos os pontos a serem atendidos foram contextualizados. Foi apresentado também a estrutura a ser seguida em cada parte do projeto, bem como a sua causa de escolha pela equipe desenvolvedora, *front-end*, *back-end* e *script*, foram apresentados os requisitos de cada uma dessas partes do projeto. Na seguinte seção, o desenvolvimento do trabalho será descrito e cada passo apresentado.

Capítulo 4

Desenvolvimento

Nesta seção de introdução ao desenvolvimento, abordaremos a contextualização dos requisitos e a escolha de ferramentas que desempenharam um papel fundamental durante todo o processo de criação do *software*. Inicialmente, a demanda por um ambiente de edição de código tornou-se evidente, uma vez que todo o desenvolvimento do software dependia dessa etapa crucial. A escolha dessa ferramenta foi pautada pela sua praticidade e também pelo conhecimento prévio que a equipe de programadores possuía sobre ela. Assim, o Visual Studio Code foi selecionado como a ferramenta ideal para essa tarefa, proporcionando uma experiência de codificação mais eficiente.

Em seguida, voltando a atenção para o *back-end* do sistema, foi identificada a necessidade de realizar testes isolados, independentemente da execução do *front-end*. Para cumprir esse objetivo, precisávamos implementar uma lógica que permitisse chamar a rota disponibilizada pela API através de requisições. A solução encontrada foi a adoção da ferramenta *Insomnia*, que desempenhou um papel crucial ao possibilitar testes e interações com a API em tempo real, contribuindo significativamente para o progresso e a qualidade do software até o momento atual de desenvolvimento.

Além de atender aos requisitos específicos do projeto, buscamos seguir as boas práticas da Engenharia de *Software*. Com isso em mente, tornou-se fundamental aplicar a estrutura de versionamento nos códigos tanto do *back-end* quanto do *front-end*. Essa prática trouxe maior organização ao projeto e proporcionou aos programadores maior segurança no processo de produção do software. Assim, a ferramenta escolhida para esse propósito foi o *Git*, que permitiu realizar alterações de forma segura e controlada, registrando cada modificação realizada por cada desenvolvedor. Dessa forma, garantimos a rastreabilidade das alterações e a possibilidade de *rollback*, caso necessário.

Para tornar o processo de colaboração mais dinâmico e colaborativo, o *GitHub* foi adotado como a plataforma para armazenar o repositório central do projeto. Nele, foram registrados os *commits*, possibilitando que mais de uma pessoa pudesse trabalhar

simultaneamente no projeto. Essa abordagem colaborativa facilitou a troca de ideias, a revisão de código e a integração das alterações realizadas por diferentes membros da equipe, aprimorando consideravelmente a eficiência do desenvolvimento do projeto.

Para aqueles interessados em detalhar-se nas implementações e estruturas utilizadas, os repositórios oferecem uma visão completa do projeto. Especificamente, o *script* discutido neste trabalho está contido no repositório do *back-end*, localizado na pasta "*files*". Incentivamos a consulta a estes repositórios para uma compreensão aprofundada e para eventuais referências futuras.

- Repositório *back-end* (*script* + Node.js)
- Repositório *front-end* (React.js)

Durante o processo de elaboração e levantamento de requisitos, houve a identificação de uma demanda importante da gestão de extensão da Universidade de Brasília (DEX). Foi verificado, através de reuniões e da reprodução da rotina comum desses usuários, que os mesmos possuem dificuldades para acompanhar a situação do cenário de extensão na universidade, com a falta de informações simples e agrupadas, assim caso queiram alguma métrica mais informativa teriam que produzir um esforço grande para obter os dados.

Introduzindo um pouco da rotina do profissional do DEX, eles tinham que realizar um trabalho muito grande por informações simples e as vezes nem tão informativas, pois o SIGAA não oferece visualização de métricas gerais e personalizadas sobre o cenário e extensão na UnB. Desta forma seus usuários não possuem um controle sobre indicadores importantes para o acompanhamento da situação das atividades de extensão na universidade, e foram nesses indicadores, que serão apresentados nesta seção, cujo este trabalho foi concentrado, assim como apresentado nas seções anteriores. Com foi verificado que os profissionais do DEX necessitavam de uma ferramenta que pudesse coletar e exibir de forma mais clara e objetiva os dados do SIGAA.

A partir dessa demanda identificada, o projeto foi reestruturado e passou a ter como foco principal o desenvolvimento de uma plataforma que permitisse a coleta e a visualização de dados do SIGAA de forma simplificada e intuitiva. Nesse contexto, a plataforma passou a contar com recursos de coleta automatizada de dados, que foram implementadas por meio de *scripts* Python utilizando o *framework* Selenium, além de uma camada de *back-end* desenvolvida em Node.js [19] para processamento e transformação dos dados coletados.

Na primeira etapa do projeto, o objetivo era a construção de um *script* que fosse capaz de navegar no SIGAA, coletar dados individuais importantes sobre ações de extensão, tratar esses dados e calcular métricas para os usuários, como a simples noção da evolução do cenário de extensão na universidade, comparando a quantidade de docentes e discentes

que se envolveram com extensão, por exemplo. Tal métrica apesar de básica, não é gerada pelo sistema SIGAA, o que reforça ainda mais a necessidade dessa ferramenta como uma possível ferramenta.

Neste parágrafo a aplicação será contextualizada de uma maneira geral, afim de passar uma visão sobre as funcionalidades da ferramenta. Tendo em vista os levantamentos realizados sobre as necessidades dos usuários o projeto foi formulado, cada passo para a construção do projeto foi colocado em pauta e organizado, com o propósito de ter todas as fases da construção idealizadas e direcionadas. A ferramenta terá a capacidade de apresentar os dados aos usuários através de um *front-end* amigável, com boa visibilidade de dados e dispôr de funcionalidades personalizáveis, onde o usuário consiga alterar modos de visualização dos dados, bem como filtros interativos. Para preparar os dados, uma API foi construída, onde nela foi feito todos os cálculos e tratativas, assim o *front-end* recebe esses dados e fornece eles para o usuário através de requisições. O processo de construção do *front-end*, API, *script* e as demais necessidades para o funcionamento da ferramenta será descrito nos próximos parágrafos.

Para criação do software, também foi essencial a utilização de metodologias ágeis, como o *pair programming*, que pode trazer benefícios significativos para o desenvolvimento de softwares. Segundo Begel e Nagappan [20] (2008), o *pair programming* é uma técnica em que dois desenvolvedores trabalham juntos em um mesmo código, com o objetivo de aumentar a qualidade e a eficiência do código produzido. Essa técnica pode ajudar a identificar e corrigir erros de forma mais rápida, além de proporcionar uma melhor comunicação e colaboração entre os membros da equipe, e foi assim que durante todo o semestre o projeto foi sendo construído.

Como apresentado na seção de Metodologia Ágil, esse projeto foi construído seguindo uma aproximação da metodologia ágil, onde em reuniões semanais eram definidas as demandas da semana seguinte, e também a validação das demandas implementadas na semana anterior, a validação ocorria com o auxílio da orientadora deste projeto. Seguindo esse padrão de desenvolvimento as demandas eram organizadas e sempre acompanhadas, desta forma a equipe conseguiu ter um bom fluxo de entrega, visto que as entregas semanais realmente eram feitas.

4.1 Script

Nas seções anteriores, foram destacadas todas as decisões tomadas para definir a escolha do *script*, em Python, para coletar os dados necessários para a aplicação. Um dos benefícios promovidos pelo *script* é a possibilidade de coletar dados de uma aplicação sem ter em posse acessos privilegiados, como o banco de dados de um software.

A ideia principal do *script* é desenvolver um mecanismo que funciona como uma espécie de robô virtual, que toma ações predefinidas pela equipe e assim vai executando instruções desejadas. Com isso é possível de especificar ações em uma página web, como no caso o SIGAA, e assim o *script* vai navegando pela aplicação e coletando os dados disponíveis, e assim armazenando em um tipo de coleção de dados, no caso do projeto, chamado JSON, que será explicado na próxima seção.

4.1.1 Caminho para os dados pelo SIGAA

Nessa seção, será demonstrado onde os dados estão localizados, e como a coleta foi feita através do *script*, inicialmente, após fazer o login, o usuário verá a página inicial do SIGAA, e logo acima, haverá um menu geral, e para acessar as informações relativas à extensão universidade, basta selecionar a opção "Extensão", e depois "Consultar Ações", do seguinte modo:

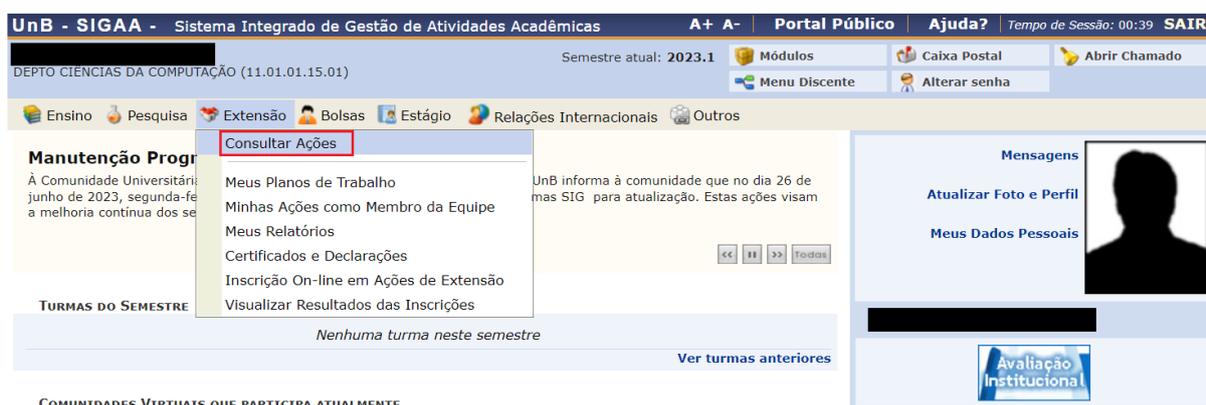


Figura 4.1: Menu inicial SIGAA.

Após selecionar "Consultar Ações", o usuário é redirecionado para uma tela a qual deve filtrar quais projetos/eventos de extensão ele deseja visualizar, do seguinte modo:

Feito os passos anteriores, clicando então no ícone da lupa, os detalhes com toda informação necessárias serão abertos. Uma informação, a equipe de desenvolvimento do software não tinha posse de acessos mais privilegiados ao sistema SIGAA, como por exemplo uma conta de um professor, durante a produção do *script* foi utilizado acesso de aluno, e isso faz com que filtros mais detalhados não sejam exibidos para esse tipo de acesso.

Sabendo sobre como e onde foram coletados os dados necessários no SIGAA, começou assim a implementação do *script*, visando fazer esse processo de forma automatizada. Uma biblioteca que auxiliou bastante na implementação, por promover diversas facilidades no quesito de usabilidade da funcionalidade *script*, foi a biblioteca *selenium*. Com ela, funcionalidades como executar ações de preenchimento de campos (inputs), cliques em botões, são bem mais simples de serem executadas.

Vale ressaltar que todos os dados a serem coletados pelo *script* estão detalhados em Indicadores do Projeto, assim o *script* foi construído para coletar esses dados. O caminho para esses dados foi apresentado nessa seção, na próxima seção será apresentada a forma de capturar esses dados através da automação.

4.1.2 Coletando os dados

Por esse projeto se preocupar com a LGPD (Lei Geral de Proteção de Dados Pessoais, Lei nº 13.709/2018) foi escolhido não explicar de maneira mais detalhada como foi feita a coleta dos dados passo a passo na plataforma. Como em vários sistemas, o SIGAA contém páginas diferentes onde é possível, através do *script*, entrar nessas páginas da seguinte forma:

```
from selenium import webdriver
driver = webdriver.Chrome(ChromeDriverManager().install())
driver.get("https://sigaa.*PAGINA_DESEJADA_PARA_O_PROJETO*")
```

Com essas três linhas de código é possível chegar em uma tela desejada, e nela efetuar outras ações utilizando das funcionalidades do *webdriver*, da biblioteca *selenium*, a seguir um exemplo de como executar uma ação em um botão:

```
driver.find_element(By.XPATH, '//[@id="idDoBotaoDesejado:element"]')
.click()
```

Dessa forma, navegando pela plataforma, executando ações, preenchendo campos, se chega a tela desejada onde os dados a serem coletados estão localizados. Para coletar

esses dados, um exemplo, um dos dados desejados estavam contidos dentro de uma tabela HTML, para localizar esta, foi necessário utilizar a ferramenta inspecionar que é disponibilizada por qualquer navegador padrão atualmente, no caso do *Google Chrome* se chama *DevTools* por exemplo, e assim observar elementos que identifiquem essa tabela, passando essa identificação ao *script*, é possível localizar essa tabela e pegar dados de interesse contidos na mesma, no caso desse projeto da seguinte forma:

```
element_financiamento = driver.find_element_by_xpath
('//th[contains(text(), "Fonte de Financiamento:")]
/following-sibling::td')
```

No exemplo acima, foi apresentado como pegar um dos dados para a métrica de financiamento, alguns detalhes foram mascarados para não comprometer o SIGAA mas chega perto da realidade.

Desta forma a implementação progrediu, pegando todos os dados desejados para formulação das métricas (mostrados na próxima seção, Objeto Coletado) e armazenando estes em um arquivo no formato JSON, tal formato foi utilizado por conta de motivos específicos e importantes para a aplicação, esses motivos serão explicados na seção de Back-end. Um ponto bastante importante a ser tratado nesse relatório, foi um dos lados negativos de utilizar o *script* como ferramenta de coleta de dados, a grande demora de execução é um peso a ser considerado de maneira bem severa, visto que nas execuções feitas o *script* demorou cerca de 40 minutos para coletar os dados, como mostra a seção Tempo de execução do *script*, diante da grande quantidade de dados a serem coletados e de muitas ações feitas pelo *Script* no sistema SIGAA.

4.1.3 Objeto Coletado

Nessa seção, será mostrado um exemplo dos dados coletados pela execução do *script* construído. Vale ressaltar que cada informação contida no JSON resultante são necessárias para o cálculo dos indicadores solicitados pelo DEX. Algumas das informações foram mascaradas a fim de agir conforme a confidencialidade de dados (segundo a LGPD), pois como observado pela equipe, o SIGAA contém diversas informações sensíveis e que devem ser tratadas conforme a legislação vigente.

Analisando a próxima figura apresentada, primeiro, atenção para três informações contidas nesse objeto resultante, a propriedade *estimado_interno*, *estimado_externo* e *publico_attingido*, como explicado na seção de Indicadores esses dados são necessários para o cálculo da massa de pessoas atingidas por projetos de extensão, sejam elas externas ou internas à Universidade de Brasília. De início, esses dados foram pegos de cada projeto,

```

{
  "codigo": "XXXX-2020",
  "projeto": "Projeto de extensão na Universidade de Brasília",
  "tipo_financiamento": "AÇÃO AUTO-FINANCIADA",
  "situacao": "CONCLUÍDA",
  "estimado_interno": "50 pessoas",
  "estimado_externo": "70 pessoas",
  "publico_atingido": 150,
  "qntd_externo": 1,
  "qntd_discente": 1,
  "qntd_docente": 1,
  "docentes": [
    {
      "nome": "DOCENTE FICTÍCIO",
      "Funcao": "COORDENADOR(A) GERAL",
      "Departamento": "FAC"
    }
  ],
  "discente": [
    {
      "nome": "DISCENTE FICTÍCIO",
      "Funcao": "ALUNO(A) VOLUNTARIO(A)",
      "Departamento": "ADM"
    }
  ],
  "externo": [
    {
      "nome": "EXTERNO FICTÍCIO",
      "Funcao": "ALUNO(A) VOLUNTARIO(A)",
      "Departamento": "OUTROS"
    }
  ],
  "ano": "2020"
}

```

Figura 4.4: Parte do objeto coletado pelo script.

assim é obtido a informação do alcance dos projetos de extensão da Universidade de maneira individual, com isso é possível gerar o cálculo da média geral da UnB sobre estimativas de público, esses cálculos bem como seus resultados serão apresentados e explicados na seção de Resultados.

A próxima informação contida nesse objeto que é analisada para a criação de um indicador está contida no atributo "tipo_financiamento", essa métrica é destinada para saber de onde vem a fonte de verba para a realização do projeto, assim coletando a fonte de cada projeto é possível ter uma noção geral da Universidade sobre quais são as origens de financiamento procuradas pelos líderes dos projetos. Tais informações e resultados são de interesse aos funcionários do DEX, bem como já levantados nas seções anteriores, e

serão apresentadas e discutidas na seção de Resultados.

Continuando a análise dos dados coletados, se chega na informação de "docentes", essa foi a primeira meta de coleta pois proporcionava para a equipe um indicador bastante interessante. Primeiro, o atributo "docentes" contém o nome do mesmo, informação que exige confidencialidade, em seguida seu departamento e sua função. Com essas informações é possível calcular a quantidade total de docentes que já se envolveram em projetos de extensão na Universidade de Brasília, bem como ter a estimativa correta do crescimento dessa participação com o passar dos anos. Outra informação valiosa tem relação com os departamentos desses docentes, através dessa informação é possível saber quais são os departamentos que mais participam de projetos de extensão, essas informações são importantes para o gerenciamento do cenário na UnB, e reforçando, esses dados não eram disponibilizados para esses funcionários do DEX de maneira clara e intuitiva. Os resultados da coleta desses dados e formulação de seus indicadores serão apresentados na seção de Resultados.

Em "discentes" é contido todos os alunos da Universidade de Brasília que participaram do projeto de extensão, trazendo também informações do seu departamento e função, com isso é possível criar a métrica que se assemelha a de docentes, calculando a quantidade total de discentes que já participaram de projetos extensionistas na Universidade, bem como a evolução da participação desses alunos com o passar dos anos. Outra informação possível na métrica de discentes é informar a participação de alunos de cada departamento em projetos, assim é possível ter o dado de como cada departamento participa dessas atividades. Os resultados calculados a partir desses dados serão apresentados e discutidos na seção de Resultados.

Por fim, no atributo de "externos", são armazenados todos os participantes dos projetos de extensão que são externos à Universidade, Como o colaborador externo não é filiado a departamentos da UnB, o departamento do coordenador de cada projeto foi colocado como o departamento do participante externo. A partir desse dado é possível ter a métrica de participação de pessoas de fora da Universidade em projetos de extensão, o que foi bastante interessante. Os resultados calculados serão disponibilizados na seção de Resultados.

4.1.4 Testes Unitários

No processo de garantir a eficácia contínua do *script* de coleta de dados, a implementação de testes unitários usando o *framework* Selenium se mostrou fundamental. A primeira camada desses testes se concentra na presença de elementos essenciais na página da web. Isso significa que, a cada execução, o *script* verifica se campos de entrada, botões e outros elementos-chave ainda estão presentes e atuam conforme o esperado.

Para a confecção dos testes unitários, foi-se utilizada a biblioteca Unittest, que é um *framework* de testes unitários incorporado ao Python, projetado para validar se partes individuais de código estão funcionando corretamente. Ela fornece uma estrutura para criar e gerenciar casos de teste, bem como definir ações que confirmam o comportamento esperado de um código.

A primeira linha de defesa desses testes está no teste de login. Dada a importância do acesso seguro e confiável à plataforma, desenvolvemos testes que verificam se a autenticação no SIGAA ocorre conforme esperado. A cada execução, o *script* tenta acessar a plataforma usando credenciais e verifica se o login foi bem-sucedido, servindo como uma confirmação de que a interface de login da plataforma continua consistente e acessível.

Uma vez que o acesso é garantido, a próxima etapa crítica do *script* é a seleção e extração de informações das ações de extensão. Portanto, implementamos testes que simulam a navegação até a seção de projetos de extensão, seguindo pelo processo de seleção de um projeto específico. O objetivo aqui é duplo: primeiro, garantir que a estrutura da plataforma e a localização das informações permaneçam consistentes; segundo, confirmar que o *script* pode, de fato, capturar os detalhes do projeto de maneira precisa e completa.

Estas medidas de teste, embora pareçam simples, constituem a base fundamental da eficácia do *script*. Elas não apenas garantem que estamos coletando os dados corretos, mas também servem como um alerta precoce para qualquer mudança na plataforma SIGAA que possam afetar as operações de coleta e atualização de dados. Assim, é garantida a confiabilidade e precisão dos dados coletados, permitindo manutenções eficientes sempre que necessário.

4.2 Back-end

O *back-end* é responsável por lidar com a lógica de negócio, o processamento de dados e a comunicação com bancos de dados e outros serviços. Ele é responsável por fornecer os dados e as funcionalidades necessárias para que o *front-end* possa interagir de forma eficiente e segura. No desenvolvimento de aplicações *back-end* em Node.js, são utilizados *framework* e bibliotecas que simplificam a criação e a organização do código.

O *framework* utilizado no projeto foi o Express. O Express oferece uma estrutura minimalista e flexível para a construção de APIs. Ele fornece um conjunto de métodos e ferramentas que facilitam o roteamento, o tratamento de requisições e respostas, o gerenciamento de sessões, entre outras funcionalidades. Com o Express, é possível criar rapidamente *endpoints* de API com diferentes verbos HTTP (get, post, put, delete, etc), definir *middlewares* para processar requisições e implementar autenticação de forma simples. Esses *endpoints* servem para compartilhar informações entre o cliente e a API.

4.2.1 Node.js

O Node.js é construído sobre o mecanismo de tempo de execução V8, desenvolvido pelo Google, que também é utilizado no navegador Google Chrome. Ele permite que o JavaScript seja executado no lado do servidor, fornecendo um ambiente de execução eficiente e orientado a eventos.

Quando um código Node.js é executado, ele é processado por meio de uma abordagem baseada em eventos e ações de I/O (entrada e saída). Isso significa que o Node.js é capaz de lidar com múltiplas operações de entrada e saída de forma assíncrona, sem bloquear o fluxo de execução. Isso é fundamental para lidar com operações de I/O intensivas, como leitura de arquivos, acesso a bancos de dados e várias requisições, de forma simultânea.

O Node.js utiliza uma única *thread* para processar eventos e operações de I/O, mas é capaz de lidar com várias conexões simultaneamente devido ao modelo de I/O não bloqueante. Ele utiliza uma estrutura de *loop* de eventos para receber, processar e despachar eventos de forma eficiente. Isso permite que o Node.js seja altamente escalável e capaz de lidar com uma grande quantidade de requisições concorrentes.

Como dito, no trabalho da equipe também utilizou do *framework Express*, seguindo o padrão de arquitetura MVC (*Model-View-Controller*), explicado na seção Modelo MVC, porém sem a camada de visualização (*view*), e também foi feito o uso da biblioteca Knex para facilitar a realização de consultas ao banco de dados, que será posteriormente explicado.

Quando uma aplicação Express é iniciada, é criado um servidor HTTP. O servidor começa a ouvir por requisições em uma determinada porta especificada. Quando uma requisição é recebida, o Express processa essa requisição através de uma série de *middlewares*, que são funções que podem ser adicionadas ao pipeline de processamento de uma requisição, ou seja, atuam como intermediários entre a requisição recebida, e a resposta enviada. Cada *middleware* tem acesso ao objeto de requisição (*request*) e ao objeto de resposta (*response*), permitindo a realização de operações como validação, autenticação, manipulação de dados e outros.

4.2.2 Modelo MVC

No projeto o padrão MVC (Model View Controller) foi utilizado para o desenvolvimento da API. Esse modelo é amplamente utilizado no desenvolvimento de software, mesmo quando a camada de visualização (View) é tratada por uma tecnologia separada, no caso deste projeto, o React.

No modelo MVC, a responsabilidade de cada componente é definida da seguinte forma:

- Model: O Model no projeto é responsável por toda a comunicação com o banco de dados. Ele representa as entidades e regras de negócio da aplicação. No contexto de um projeto Node.js, o Model é responsável por definir os esquemas de dados, realizar operações de leitura e escrita no banco de dados, implementar validações e realizar cálculos ou transformações nos dados quando necessário.
- View: Responsável por criar a interface de interação com o usuário, nela os dados passados pela Modal são disponibilizados.
- Controller: O Controller é responsável por processar as ações solicitadas e interagir com o Model apropriado. Ele atua como o intermediário entre a camada de visualização (React) e o Model, definindo quais dados são necessários para atender às requisições e como eles devem ser manipulados. O Controller também é responsável por enviar os dados para o *front-end*, onde no projeto, sempre foi utilizado o formato JSON.

4.2.3 Organização/Estrutura de código

O projeto adotou uma organização simples e intuitiva, de fácil manutenção e escalável, buscando separar as responsabilidades da melhor forma possível. A imagem abaixo mostra a estrutura da pasta principal do projeto *back-end*, a pasta *src*:

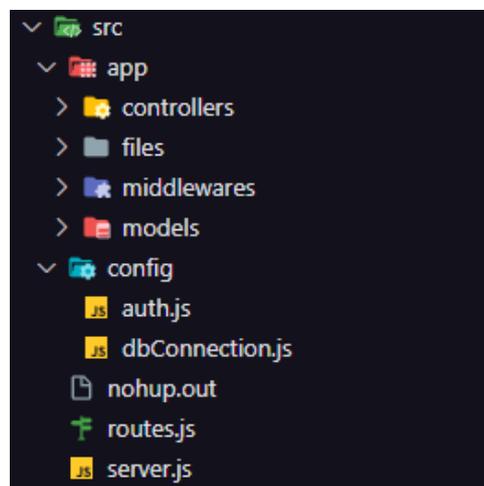


Figura 4.5: Estrutura principal do código de *back-end*.

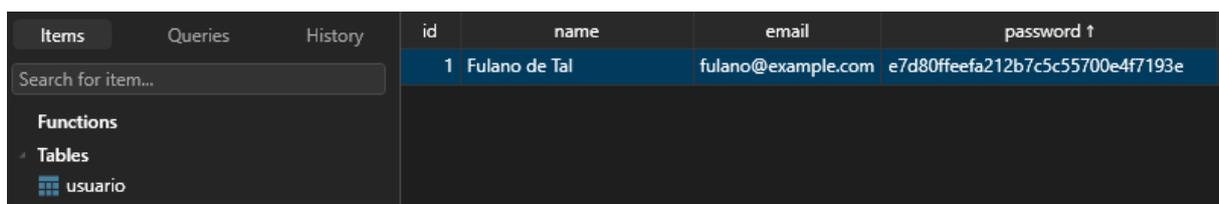
A pasta "app" é a principal pasta do código, contendo as pastas "*controllers*", "*files*", "*middlewares*" e "*models*". A *Controller* possui a lógica/cálculos necessários, "*models*" possui arquivos que fazem as *queries* ao banco de dados, "*files*" contém os arquivos com os dados da aplicação, em formato JSON, e por fim, a pasta "*middlewares*" contém

apenas um *middleware* de autenticação, onde no sistema de roteamento do *back-end*, as rotas protegidas são envolvidas por esse *middleware*, que contém a lógica para validar o *token* do usuário, enviado pelo *front-end*, e se estiver tudo certo, a requisição irá seguir seu fluxo normalmente.

A pasta *config* possui os arquivos *auth.js* e *dbConnection.js*. O arquivo "auth" possui o segredo para decodificar o *token* JWT, e o tempo de expiração do *token* (ou seja, quanto tempo o usuário poderá ter sessão até que logue novamente).

4.2.4 Banco de dados

Foi feito também um simples banco de dados para que fosse possível a autenticação de usuários, uma vez que os dados importantes da aplicação são armazenados no arquivo JSON gerado pelo *script python*. O motivo de implementar autenticação nesse sistema é a preocupação em não compartilhar os dados, presentes no SIGAA, para usuários não desejados, e como já mencionado a LGPD. O banco utilizado foi um simples banco relacional MySQL, pois a equipe já tinha familiaridade com a ferramenta, onde foi utilizado o Knex para fazer as consultas necessárias ao banco, para pegar informações do usuário, autenticação de email e senha, entre outros. O Knex é uma biblioteca de consulta SQL para Node.js que permite a construção de consultas ao banco de dados de forma simples e segura. Ele suporta diferentes bancos de dados, como MySQL, PostgreSQL, SQLite e outros. Com o Knex, é possível criar consultas complexas e interagir com o banco de dados de maneira padronizada e eficiente, evitando erros de queries.



Items	Queries	History	id	name	email	password ↑
Search for item...			1	Fulano de Tal	fulano@example.com	e7d80ffefa212b7c5c55700e4f7193e

Functions
Tables
usuario

Figura 4.6: Tabela de usuários simples, para que o usuário possa realizar o login.

A senha do usuário é codificada no padrão MD5 (Message Digest Algorithm 5) no momento da criação de usuário. O MD5 é um algoritmo de hash de função de resumo criptográfico, o que significa que ele converte uma entrada (mensagem ou arquivo) em uma sequência de caracteres de tamanho fixo, conhecida como hash. O tamanho do hash gerado pelo MD5 é de 128 bits (ou 16 bytes). O objetivo principal do MD5 é garantir a integridade dos dados, ou seja, detectar qualquer alteração acidental ou intencional nos dados.

4.2.5 JSON

JSON (JavaScript Object Notation) é um formato leve de troca de dados que se tornou amplamente utilizado na comunicação entre sistemas. Ele é baseado na sintaxe de objeto JavaScript, que é a stack tecnológica de todo o projeto. O JSON utiliza uma estrutura de pares chave-valor para representar os dados, permitindo a organização e o armazenamento de informações de forma hierárquica. Os dados são formatados em texto simples, o que torna o JSON fácil de ler e escrever para humanos, além de ser facilmente interpretado e processado por máquinas.

Nesse projeto o JSON foi adotado como forma de armazenamento dos dados coletados pelo *script* Python, e esse JSON é tratado por diversas camadas da aplicação para ser entregue ao *front-end* da melhor forma possível. Através de um grande JSON foi feito o armazenamento de todos os dados que são importantes para a aplicação, os dados de docentes, discentes, externos, já tratados previamente pelo *script*. Esse grande arquivo tem quase duzentas mil linhas, porém, seu tratamento (laços de iteração e comparações) são feitos de forma praticamente instantânea.

As bibliotecas de processamento de JSON, como o `JSON.parse()` em JavaScript, são geralmente projetadas com algoritmos eficientes que permitem uma análise rápida dos dados. Essas bibliotecas são otimizadas para lidar com grandes volumes de dados e são continuamente aprimoradas para melhorar o desempenho. Foi utilizado o uso de abordagens otimizadas, como a utilização de índices e mapas, para reduzir a complexidade da iteração e melhorar o desempenho.

4.2.6 Autenticação

A autenticação é um aspecto crítico em muitas aplicações web, pois permite verificar a identidade dos usuários e controlar o acesso aos recursos protegidos. Uma abordagem comum para a autenticação no *back-end* é utilizar tokens JWT (JSON Web Tokens) em conjunto com Node.js e o *framework* Express.

O JWT é um padrão aberto (RFC 7519) que define um formato compacto e seguro para representar informações como um objeto JSON. Ele consiste em três partes principais: um cabeçalho, um *payload* e uma assinatura. O cabeçalho contém informações sobre o tipo do token e o algoritmo de assinatura utilizado. O *payload* contém os dados do usuário ou outras informações relevantes, como permissões ou papéis. A assinatura é uma combinação do cabeçalho, *payload*, e uma chave secreta, usada para verificar a integridade do token.

Quando o cliente deseja acessar uma rota protegida, ele deve enviar o token JWT no cabeçalho da requisição (utilizando o cabeçalho "Authorization" com o valor "Bearer

<token>"). Um *middleware* do Express é utilizado para verificar a autenticidade e integridade do token. Isso envolve a validação da assinatura e a decodificação do *payload* do token para então obter informações do usuário que está fazendo a requisição.

Boas práticas de segurança são adotadas para toda a parte de autenticação, como o armazenamento adequado das chaves secretas e definição de expiração dos tokens, tudo de acordo com a documentação oficial da biblioteca "jsonwebtoken".

4.3 Organizando demandas (KanBan)

Como citado na seção de Referencial Teórico, nesse projeto padrões da Engenharia de Software foram aplicados na intenção de tornar o processo de desenvolvimento mais organizado e padronizado, também chegando em um produto final com uma boa qualidade e que atenda as necessidades dos usuários.

Durante as reuniões semanais de levantamento das demandas, a equipe organizava as necessidades mais urgentes e separava entre os programadores do projeto. Para melhor organizar e acompanhar o desenvolvimento dessas demandas, foi utilizado um quadro KanBan do próprio Teams, ferramenta essa utilizada para as reuniões da equipe. No KanBan, as demandas eram organizadas por grau de importância, assim dando visão maior para implementações mais urgentes. Segue uma imagem de umas das etapas do quadro da equipe.

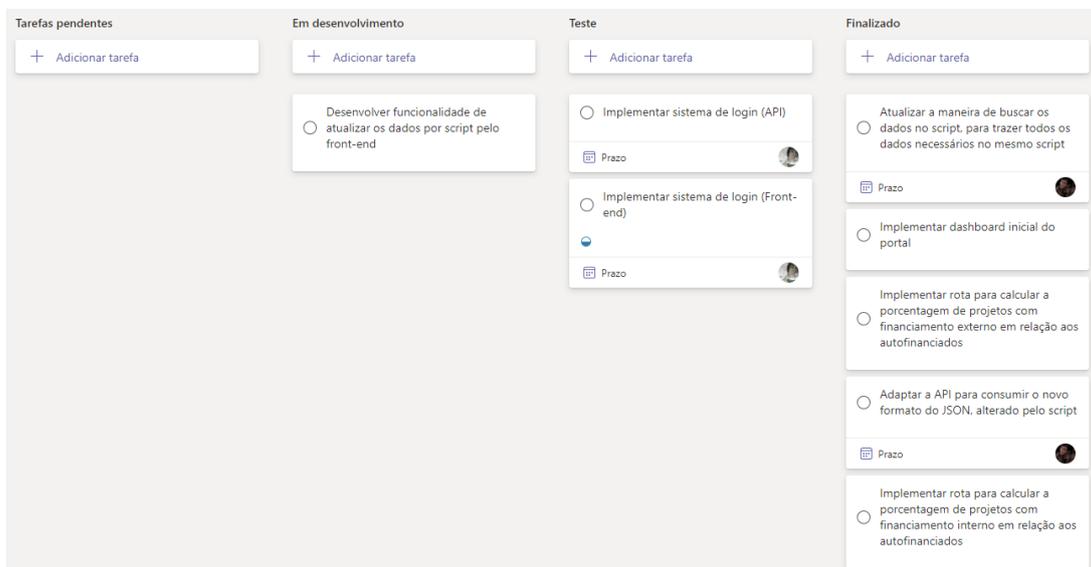


Figura 4.7: KanBan do projeto.

4.4 Interface Front-End

O *front-end* é a parte de uma aplicação ou site com a qual os usuários interagem diretamente. É a interface interativa que engloba elementos como botões, menus, formulários e qualquer outro conteúdo que seja visível. O desenvolvimento *front-end* consiste em criar e implementar esses elementos, buscando uma experiência de usuário atraente e funcional.

Para um desenvolvimento mais ágil e organizado, são utilizados os *framework front-end*, pode-se entender como *framework* uma estrutura que oferece ferramentas, bibliotecas e padrões para facilitar o desenvolvimento de software. O *framework* fornece uma base sólida e estruturada para os desenvolvedores construírem suas aplicações, automatizando tarefas repetitivas e oferecendo soluções pré-construídas para problemas comuns. Geralmente, um *framework* define uma arquitetura e um fluxo de trabalho específicos, orientando os desenvolvedores sobre como organizar o código, os arquivos e implementar determinadas funcionalidades. Isso pode incluir componentes reutilizáveis, bibliotecas de terceiros, sistemas de roteamento, gerenciadores de estado e muito mais.

Para suprir as necessidades discutidas no levantamento de requisitos, foi escolhido o *framework* React. O React é uma biblioteca JavaScript amplamente adotada, que oferece recursos avançados para a construção de interfaces. Com o React, é possível criar componentes reutilizáveis, que encapsulam tanto a lógica quanto a interface, permitindo um desenvolvimento mais modular e escalável. Além disso, o React utiliza o conceito de Virtual DOM, otimizando a atualização e renderização dos elementos da página.

4.4.1 Utilização do Next.js

A construção do *front-end* também envolveu o *framework* Next JS. O NextJS é um *framework* React de código aberto utilizado para a construção de aplicativos web do lado do servidor (SSR) e do lado do cliente (CSR). Com base no ReactJS, o NextJS adiciona recursos adicionais e abstrai complexidades, oferecendo algumas vantagens para o desenvolvimento. Foi utilizado o Next por conta de algumas funcionalidades que realmente fazem a diferença para o desempenho da aplicação, tais como roteamento e gerenciamento de estado mais simplificado, o qual será discutido mais tarde, e também por conta da renderização do lado do servidor (SSR), o qual foi utilizado na aplicação.

Neste exemplo acima, é destacada página de docentes, onde há uma busca dos dados na API através da rota `'/docentes'` pela função `getServerSideProps`, que é assíncrona e retorna um objeto com os dados que serão passados como propriedades para o componente principal da página de docentes, este que utilizará os dados que recebeu como parâmetro.

Uma abordagem tradicional de React, sem a utilização dos recursos do Next JS, seria utilizar um *hook* chamado `useEffect`, este usado para lidar com lógica relacionada a ciclos

```

import Docentes from '@/screens/docentes';
import DefaultLayout from '@/layouts/Default';
import { getAPIClient } from '@/services/axios';

export default function DocentesPage({ data }) {
  return (
    <DefaultLayout>
      <Docentes data={data} />
    </DefaultLayout>
  );
}

export const getServerSideProps = async ctx => {
  const api = getAPIClient(ctx);

  const { data } = await api.get('docentes');

  if (!data) {
    return {
      redirect: {
        destination: '/404',
        permanent: false
      }
    }
  }

  return {
    props: {
      data: data
    }
  };
}

```

Figura 4.8: Página de docentes utilizando-se de SSR.

de vidas, assincronia, eventos e integração com APIs externas. Em relação à diferença entre o uso do *getServerSideProps* e um *useEffect* com uma chamada a API no React, o *getServerSideProps* traz alguns benefícios adicionais, por exemplo, ao utilizar o *getServerSideProps*, a chamada à API é feita no servidor, permitindo que os dados sejam pré-renderizados antes de enviar a página para o cliente. Isso garante que os usuários recebam uma página com dados atualizados a cada solicitação. Em contraste, o uso de um *useEffect* com uma chamada a API no lado do cliente resultaria em uma solicitação adicional ao servidor após o carregamento inicial da página. Isso pode levar a um atraso perceptível no carregamento e na exibição dos dados, pois a chamada à API é feita somente após a renderização inicial da própria página e seus componentes. Além disso, ao usar o *getServerSideProps*, os dados obtidos são passados como propriedades para a página durante a renderização, o que simplifica o acesso e a manipulação desses dados nos componentes das páginas filhas. No caso do *useEffect*, os dados seriam obtidos no lado do cliente e, em seguida, gerenciados no estado do componente, exigindo uma lógica

adicional para lidar com atualizações, erros e sincronização.

4.4.2 UX Design

O UX (*User Experience*) design, refere-se à uma área do design que se concentra na criação de experiências satisfatórias para os usuários durante a interação com um produto ou serviço, no caso deste projeto, a aplicação web SIAPE. O objetivo do UX design é tornar a jornada do usuário intuitiva, eficiente e agradável, levando em consideração seus objetivos, e é de extrema importância, pois influencia diretamente a forma como os usuários interagem com o sistema.

Ao projetar a página inicial informativa, o objetivo é fornecer ao usuário informações relevantes e claras sobre a aplicação, apresentando de forma concisa o propósito, os recursos e os benefícios que ela oferece. Foram utilizados princípios de UX design (como estética agradável ao usuário, simplicidade na apresentação das informações) para organizar o conteúdo de maneira estruturada, priorizando a legibilidade e a facilidade de compreensão.



Figura 4.9: Página inicial.

Na tela de login, foi aplicado princípios de usabilidade para garantir que o processo de autenticação seja simples, direto e livre de obstáculos. Foi construído um formulário de login amigável, com feedback em tempo real, validação de campos e orientações visuais para auxiliar o usuário a fornecer as informações necessárias de maneira correta.

Ao projetar as telas com gráficos e filtros, o fornecimento de uma visualização clara e interativa dos dados foi o foco, permitindo que os usuários explorem e compreendam as informações de maneira eficiente. Foram utilizadas técnicas de design visual, como o uso

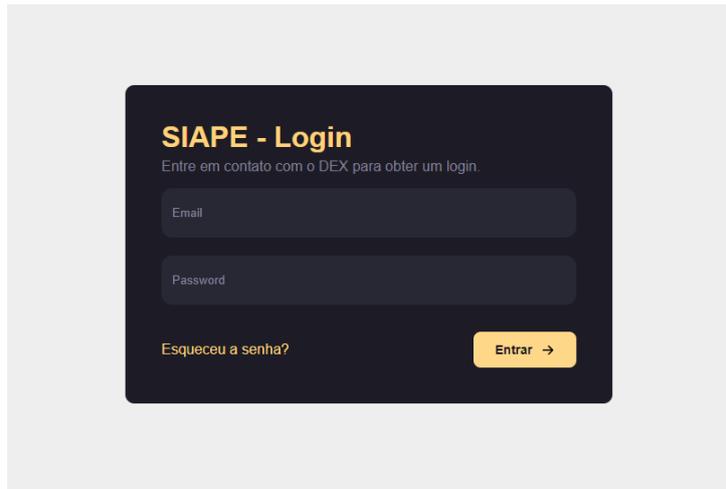


Figura 4.10: Tela de login.

de cores, ícones e gráficos intuitivos, para facilitar a compreensão dos dados e tornar a experiência de análise e filtragem mais intuitiva.

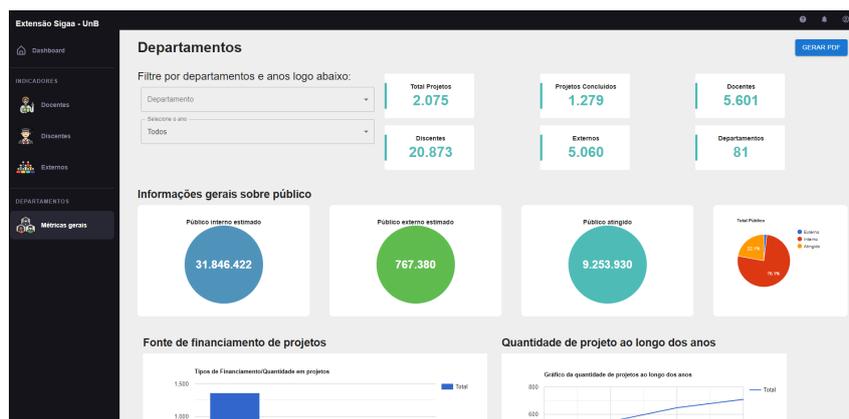


Figura 4.11: Tela de métricas gerais.

Vale ressaltar que em todo o processo de design e criação, foi levado em consideração aspectos de acessibilidade e responsividade, para garantir que a aplicação possa ser utilizada por um público amplo, independentemente do dispositivo utilizado.

4.4.3 Organização/Estrutura de código

Foi tomado um cuidado muito grande com boas práticas de programação durante todo o desenvolvimento, pensando não só na organização estrutural do código, como também em manutenibilidade e desempenho. Nesse projeto as boas práticas de programação podem ser apresentadas dessas maneiras, código limpo e facilmente entendível, boa organização

estrutural do projeto, utilização de técnicas de programação performáticas, para chegar em um projeto com essas características vale ressaltar a importância de realizar um Requisitos do Sistema bem feito e compreender o Referencial Teórico. Nesse contexto, os *Design Patterns*, ou padrões de projeto, surgem como uma abordagem que encapsula as melhores práticas estabelecidas ao longo do tempo, proporcionando soluções eficientes para problemas recorrentes.

Os *Design Patterns* são soluções pré-definidas que abstraem e encapsulam o conhecimento sobre como resolver problemas específicos no desenvolvimento de software. Eles são o resultado de anos de experiência acumulada por desenvolvedores e especialistas na área, que identificaram e documentaram as abordagens mais eficazes para lidar com desafios comuns em projetos de software.

Ao adotar *Design Patterns*, é possível estruturar o código de forma mais organizada, modular e principalmente, reutilizável. Uma abordagem que foi utilizada em todo projeto, é o uso de React Hooks junto com componentes funcionais React, que oferecem uma maneira simples e direta de acessar recursos comuns do React, como *props*, estado, contexto e ciclo de vida.

Nesse projeto o código foi organizado mantendo a seguinte estrutura, buscando separar cada funcionalidade e responsabilidade em pastas distintas, para melhor organização do projeto:

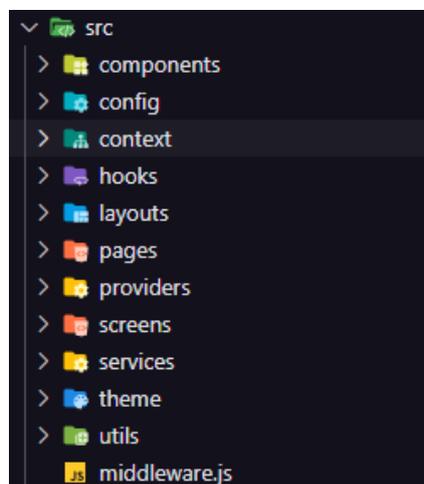


Figura 4.12: Estrutura principal de código.

A pasta "src" contém todas as pastas que são necessárias para o desenvolvimento organizado, como a pasta *components* que contém todos os componentes necessários que são reutilizáveis na aplicação, como por exemplo, o componente "Selectors" que é utilizado em quase todas as páginas, este recebe os dados da API para que os usuários possam filtrar os dados por departamento, ano, se querem considerar pessoas não

repetidas ou não, e outros. Outra pasta muito importante no projeto é a pasta *context*, essa pasta possui todos os *contexts* da aplicação, entenda o *Context* no React como um recurso permite compartilhar dados entre componentes sem a necessidade de passá-los explicitamente por meio de *props* (informações passadas de um componente para outro específico), oferecendo uma maneira conveniente de criar um canal de comunicação global, onde os dados podem ser acessados por qualquer componente descendente dentro de uma árvore de componentes.

O *context* é especialmente útil em situações em que vários componentes precisam ter acesso aos mesmos dados, como temas, preferências e dados do usuário (como nome e email) e autenticação, que foram utilizados na aplicação. Em vez de passar manualmente esses dados por várias camadas de componentes, o *context* permite que você defina um provedor de contexto e consuma-o onde necessário.

4.4.4 Autenticação

Foi utilizado uma implementação comum de autenticação que é presente e consolidada em diversas aplicações web, que envolve o uso de tokens JWT (JSON Web Tokens) e o armazenamento desses tokens em *cookies*, que são basicamente pequenos arquivos de texto que são armazenados no navegador de um usuário quando ele visita um site. Essa abordagem proporciona uma maneira segura e eficiente de autenticar usuários e gerenciar sessões.

O JWT é um padrão aberto que define uma forma compacta e autocontida de transmitir informações entre partes de forma segura. Ele consiste em três partes: um cabeçalho, um *payload* e uma assinatura. O cabeçalho contém informações sobre o tipo de token e o algoritmo de assinatura usado. O *payload* contém dados adicionais, como o ID do usuário e outras informações relevantes para a autenticação. A assinatura é usada para verificar a integridade do token.

Ao implementar a autenticação com JWT, o processo geralmente envolve as seguintes etapas:

- O usuário envia suas credenciais (no caso do sistema, email e senha) para o servidor.
- O servidor Node.js autentica as credenciais e gera um token JWT.
- O token JWT é enviado de volta ao cliente e armazenado em um cookie.
- Em solicitações subsequentes, o cliente envia o token JWT através do cabeçalho de autorização em cada requisição.
- O servidor verifica a validade e integridade do token JWT e concede acesso às rotas ou recursos protegidos, se o token for válido.

O uso de *cookies* para armazenar o token JWT traz algumas vantagens. Primeiro, os *cookies* são automaticamente enviados pelo navegador em cada solicitação, facilitando a persistência e o envio do token. Em resumo, a implementação de autenticação com token JWT armazenado em *cookies* é uma abordagem segura e eficiente para autenticar os usuários. Ela oferece a capacidade de transmitir informações de forma segura e persistente, facilitando o gerenciamento de sessões e o acesso a recursos protegidos. No entanto, é fundamental adotar boas práticas de segurança para proteger o token e garantir a integridade dos dados.

4.4.5 Google Charts

Foi usado neste projeto o Google Charts, que é uma biblioteca JavaScript desenvolvida pelo Google, onde permite a criação de gráficos dinâmicos e interativos para a exibição de dados, e oferece uma ampla variedade de tipos de gráficos, como gráficos de linhas, barras, pizza, entre outros.

Como dito anteriormente, as entradas são centralizadas em componente chamado *Selectors*, onde dentro dele estão todos os seletores possíveis para os dados. Graças ao Google Charts, é possível enviar esses dados para o componente de gráfico, e o usuário pode filtrar os dados e as mudanças são feitas em tempo real em toda a página. Funciona da seguinte forma: a página recebe os dados da *api* através de uma requisição do Next.js, só após esses dados estarem disponíveis, a página é carregada, e esses dados são mandados de forma "pura" para o componente *Selectors*, que filtra os dados e atualiza um estado chamado *contextData*, que é utilizado pelos gráficos da página, que podem ser alterados da forma que o usuário quiser.

A capacidade de alterar o tipo ou a configuração de um gráfico pode trazer novas ideias e *insights* valiosos sobre os dados em análise. Ao modificar o tipo de gráfico utilizado, como por exemplo um gráfico em tortas por um em barras, é possível visualizar os dados a partir de diferentes perspectivas e identificar padrões ou tendências que podem passar despercebidos em outros formatos. Por exemplo, ao mudar de um gráfico de barras para um gráfico de linhas, é possível identificar variações ao longo do tempo de maneira mais clara, permitindo a compreensão de padrões sazonais ou flutuações significativas.

Essa flexibilidade na visualização oferecida pelo *Google Charts* auxilia na tomada de decisões embasadas e na identificação de oportunidades de melhorias dependendo do contexto a qual está sendo analisado.

4.4.6 Exportação de Dados

É disponibilizado em todas as telas a opção de gerar os dados vistos nos gráficos em um arquivo PDF, para isso foi utilizada uma biblioteca Javascript pública, chamada *@react-pdf/renderer* junto com o Material UI.

Um componente chamado *PDFGenerator* é responsável por criar o conteúdo do PDF. Ele recebe os dados do contexto (*contextData*) e uma função de fechamento (*onClose*). O modal é exibido através do componente Modal do Material UI.

Dentro do modal, há um botão "Fechar" que aciona a função *onClose* para fechar o modal quando clicado.

Dentro desse grande componente, há outro componente *PDFViewer* do *@react-pdf/renderer*, que é responsável por renderizar o conteúdo do documento PDF. Dentro do *PDFViewer*, é localizado o componente *Document* que representa o documento PDF em si.

Dentro do *Document*, se encontra um único Page que contém todo o conteúdo do PDF. O estilo do *Page* é definido pelo objeto *styles.page*, que especifica a família de fontes, tamanho da fonte e margens. Finalmente então, um código HTML faz toda a estrutura do PDF. Por fim, um rodapé do documento exibe a mensagem "Gerado pelo software SIAPE".

O documento pode ser descarregado no computador do usuário normalmente, através das ferramentas do próprio navegador.

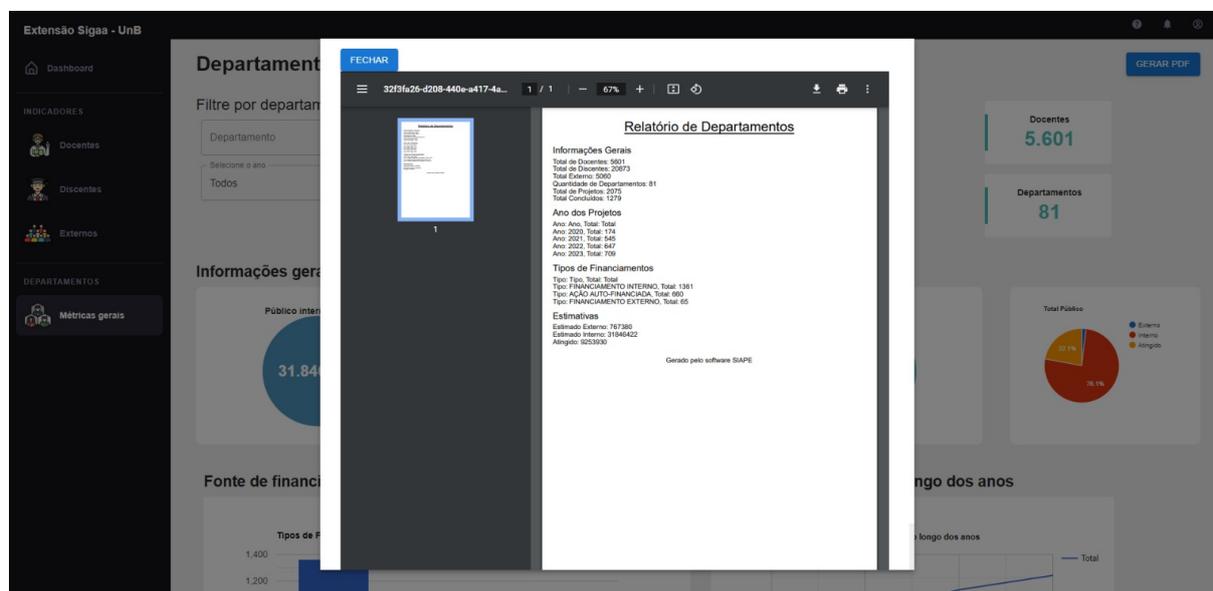


Figura 4.13: Conteúdo que aparece ao clicar em "Gerar PDF".

Nessa seção de Desenvolvimento todo o processo de trabalho e evolução, dentre as necessidades estabelecidas pelas seções anteriores a esta, foi explicada. Ao fim dessa

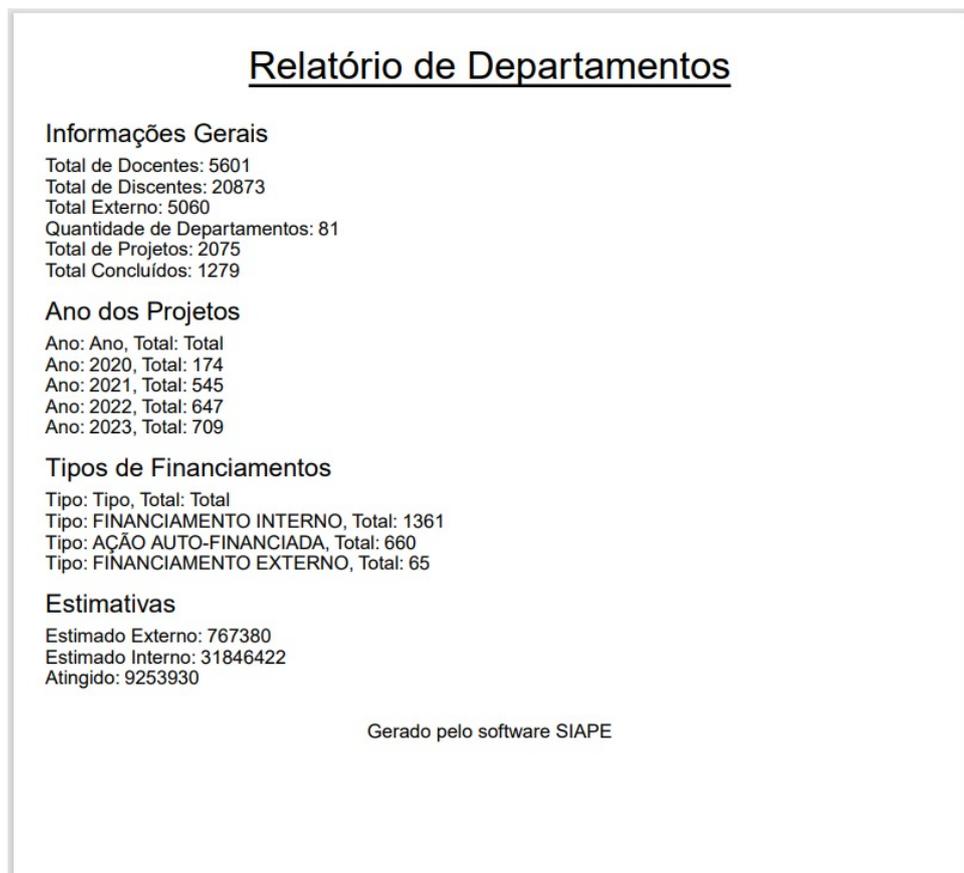


Figura 4.14: PDF exemplo gerado.

seção é possível entender o processo de desenvolvimento utilizado, e também o fluxo de trabalho aplicado ao projeto. A plataforma desenvolvida foi planejada seguindo os padrões da Engenharia de Software, buscando uma interface limpa, com um bom desempenho e dados claros. O sistema desenvolvido atingiu as ideias iniciais do projeto, indicadores importantes foram construídos para melhorar a experiência dos usuários, possibilitando análises mais precisas acerca do cenário de extensão na UnB. Na seção seguinte, serão apresentados os resultados provenientes dos indicadores e de todo o software construído.

Capítulo 5

Resultados

Nessa seção de resultados todo o desenvolvimento feito até o momento será apresentado e sustentado, cada métrica criada será discutida e comprovada. O projeto envolveu muitas áreas diferentes a serem implementadas, *Python* para coleta de dados, armazenamento e tratativa dos dados, API para calcular e disponibilizar os dados e o *front-end*, onde os dados são apresentados e a interação do usuário com o sistema é feita. Vale ressaltar que todos os indicadores foram sugeridos por clientes, o que deixa a equipe em um ponto confortável, já que a finalidade desses indicadores já são necessárias para o cliente, e assim o projeto de criação do software tem uma base forte, pois fornece ferramentas que atendem as expectativas dos usuários do DEX.

Como é possível mostrar todos os resultados pelo *front-end* da plataforma criada, a seguir serão apresentadas imagens do software e a explicação de cada uma de forma detalhada.



Figura 5.1: Tela de apresentação.

Na figura acima uma tela é apresentada, onde a mesma foi criada com o propósito de ser o primeiro contato do usuário com o sistema, nela é destacado o que é o software SIAPE e sua finalidade. Nesta tela o usuário que possuir acesso consegue efetuar o *login* clicando no botão "Entrar", localizado no canto superior direito da tela. A criação dessa primeira tela desenvolvida para criar uma noção ao usuário do que se trata o sistema, pensando no caso do um usuário que não conhece as funcionalidades e o propósito da plataforma. Informações que imitam um cenário real são colocadas nos gráficos apresentados para a maior segurança dos dados e também possibilitam um entendimento maior de como funciona o sistema por dentro.

5.1 Tela inicial

A seguir é apresentado o primeiro contato com o usuário dentro do sistema, com foco em apresentar métricas gerais de maneira bem clara e intuitiva assim como no restante do sistema.

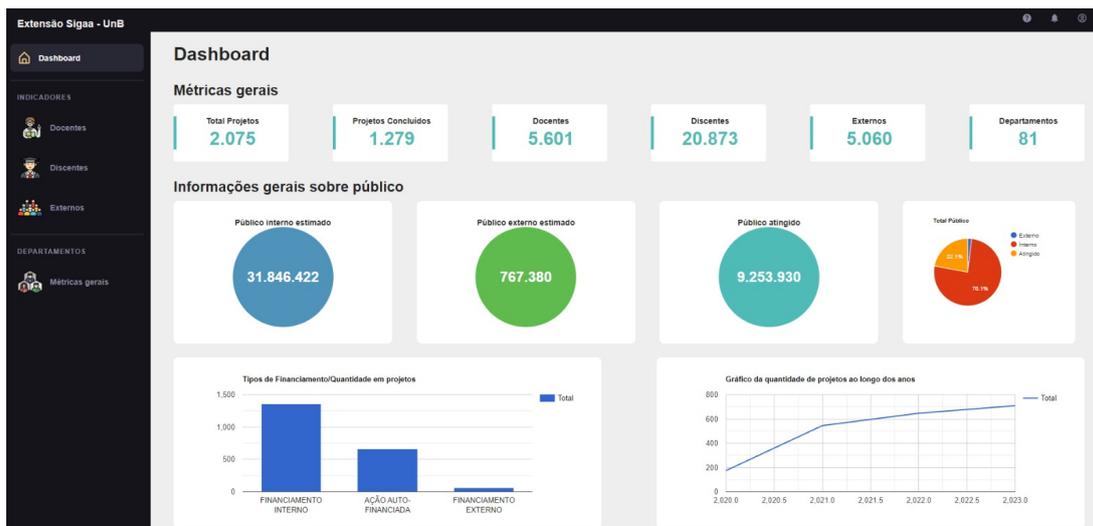


Figura 5.2: Dashboard inicial.

Um *dashboard* é uma representação visual e interativa de informações, dados e métricas relevantes para uma organização ou processo específico. Geralmente, é apresentado em uma única tela ou painel, permitindo aos usuários ter uma visão consolidada e em tempo real do desempenho, progresso ou status de diferentes indicadores-chave de desempenho (KPIs) ou elementos de interesse.

Essa representação visual pode incluir gráficos, tabelas, mapas, medidores, indicadores de progresso e outros elementos visuais que facilitam a compreensão e a análise dos dados.

O objetivo do *dashboard* é fornecer informações claras, rápidas e facilmente compreensíveis, permitindo que os usuários tomem decisões informadas e aprimorem a eficiência e a eficácia em suas atividades, projetos ou negócios. Os *dashboards* são comumente utilizados em diversas áreas, como negócios, marketing, gestão de projetos, análise de dados, monitoramento de processos e muito mais.

De início, vale ressaltar que todos os resultados obtidos foram coletados dos anos de 2020, 2021, 2022 e 2023, que foram escolhidos por serem os anos mais recentes, como prova de conceito. Desta forma é possível apresentar de maneira mais clara a evolução, e também a possibilidade de comparação dos resultados entre os anos coletados. Abaixo do texto "Métricas gerais" são apresentados os resultados coletados que indicam um geral do cenário de extensão na Universidade de Brasília, foi observado que desde 2020 até 2023 foram criados 2075 projetos de extensão e deles 1279 já foram concluídos, essas disparidades de conclusão ocorrem devido ao tempo de duração dos projetos, por exemplo um projeto criado em 2022 pode estar em duração no ano de 2023. Ao analisar as métricas gerais, observa-se que há 5601 participações de docentes, 20873 de discentes e 5060 de indivíduos externos em projetos de extensão. É notável que esses números são significativamente elevados, atribuídos à existência de duplicidades nas participações dos docentes, discentes e externos em diferentes projetos. Por exemplo, um docente pode estar envolvido em mais de um projeto, e o sistema registra essas ocorrências sem eliminar as duplicidades. No entanto, para acessar informações sem duplicações, o usuário pode encontrá-las na tela dedicada aos "Docentes". Por fim na informação de Departamentos, é apresentado somatório de todos os departamentos da Universidade de Brasília que já participaram de projetos de extensão, essa métrica é calculada observando o departamento do coordenador geral de cada projeto, o resultado obtido foi de 81 departamentos envolvidos no cenário de extensão da UnB. Vale ressaltar, que a Universidade de Brasília possui 51 departamentos no total, porém o SIGAA trata alguns departamentos de pós-graduação e colaborações entre departamentos (por exemplo, CIC/EST) como departamentos gerais. As próximas informações compartilhadas nessa tela de inicial serão tratadas mais a frente.

5.2 Resultados indicador de docentes

Agora serão apresentados os resultados obtidos nos cálculos do indicador de Docentes.

No indicador de docentes foi coletado informações interessantes, de primeiro as informações gerais coletadas foram o total de 5601 docentes existentes em projetos de extensão (considerando que existem docentes repetidos nesse número, pois docentes podem ter participado de mais de um projeto), 81 departamentos com docentes envolvidos nesses projetos. Para a coleta do dado total de docentes envolvidos, é necessário percorrer todos

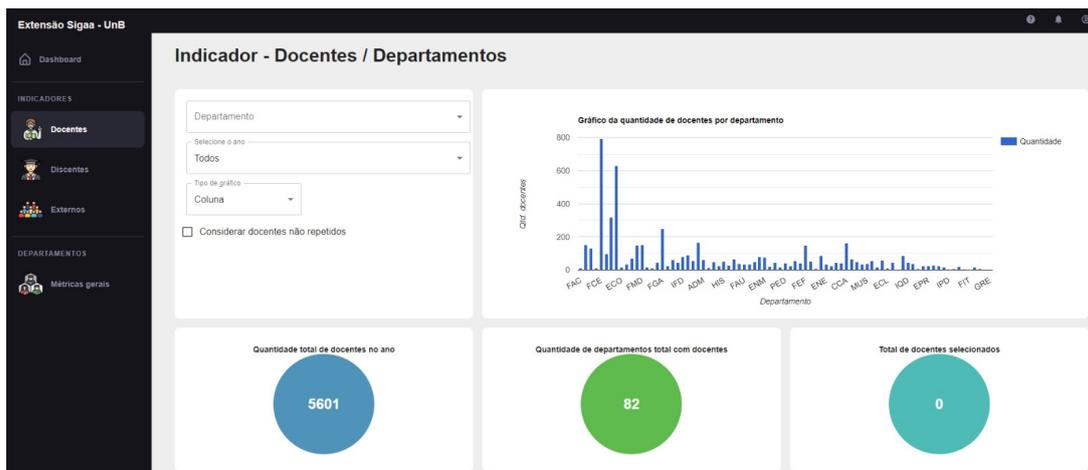


Figura 5.3: Tela de docentes.

os projetos de extensão na Universidade de Brasília até então e observar o cargo de cada participante, se for docente o totalizador do mesmo é incrementado.

Ainda no indicador de docentes, apresentando de maneira mais detalhada, foi feita a construção de uma visualização em gráfico da quantidade de docentes presentes em cada departamento, assim é possível que o usuário visualize de maneira individual a participação dos docentes, vale ressaltar uma implementação de filtro que funciona bem para a visualização individual, caso o usuário deseje visualizar informações de departamentos específicos, basta o mesmo selecionar os departamentos desejados como sugere a imagem a seguir:

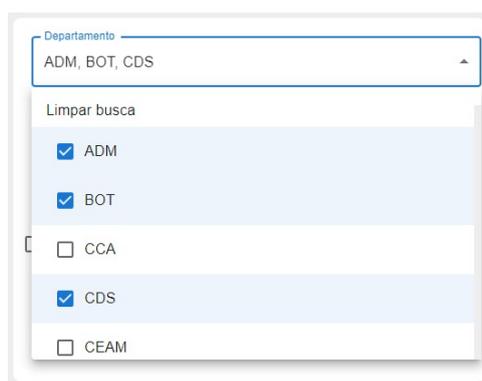


Figura 5.4: Seleção de departamentos.

Outra funcionalidade interessante é a possibilidade de alterar os anos das informações, de início as informações apresentadas envolvem todos os anos da coleta dos dados pelo *script*, no caso 2020, 2021, 2022 e 2023, caso o usuário queira visualizar separadamente cada ano basta o mesmo selecionar o ano desejado no filtro. Outra possibilidade é a

mudança do tipo de gráfico para apresentar os dados, é disponibilizada a visualização em gráfico torta, barra e de colunas.



Figura 5.5: Evolução de docentes em projetos de extensão.

No gráfico acima, foi criada a visualização em linha de tempo da evolução no número de docentes participantes em projetos de extensão, foram coletadas informações que mostram um crescimento constante no número de docentes, visto que em 2020 um total de 430 docentes participavam em projetos de extensão e em 2023 ao todo 1876 docentes estavam envolvidos, o que resulta em um crescimento 336% de docentes envolvidos em projetos de extensão, comparando o primeiro e o último ano. Vale lembrar que esses dados não excluem a duplicidade de participação de docentes em mais de um projeto. Segue uma tabela contendo os mesmos dados da evolução de docentes nos anos destacados, como será explicado nas próximas métricas, o mesmo vale pra essa, no resultado obtido no ano de 2023, vale lembrar que a coleta desses dados foram feitas antes mesmo do fim do primeiro semestre, e ainda assim o resultado obtido foi maior que o ano de 2022, o que mostra um crescimento do número de docentes no ano.

5.3 Resultados - Indicador de Discentes

Nos resultados focados no indicador de alunos participantes de projetos de extensão, foram encontrados números bem interessantes, visto a grande evolução na participação desses alunos com o passar dos anos. Primeiramente foi coletado e calculado uma métrica geral, no resultado dessa métrica observa-se que a UnB acumulou 20873 ocorrências de alunos em projetos de extensão nos anos de 2020, 2021, 2022 e 2023, vale lembrar a participação possível de um aluno em mais de um projeto do extensão, foi decidido manter a duplicidade de dados para dar uma noção da quantidade de participação de alunos nesses projetos. Outra métrica calculada bem interessante na universidade é o total de departamentos que possuem discentes participantes desses projetos, ao todo um total de 130 departamentos foram resultantes nessa métrica, explicando melhor o porque, ao chegar nesse resultado foi observado que departamentos de PPG, Programa de Pós Graduação, também são

incluídos pelo SIGAA, logo há um número grande de participação desses departamentos. Estes resultados são apresentados aos usuários em uma tela do software, como indica a imagem a seguir.

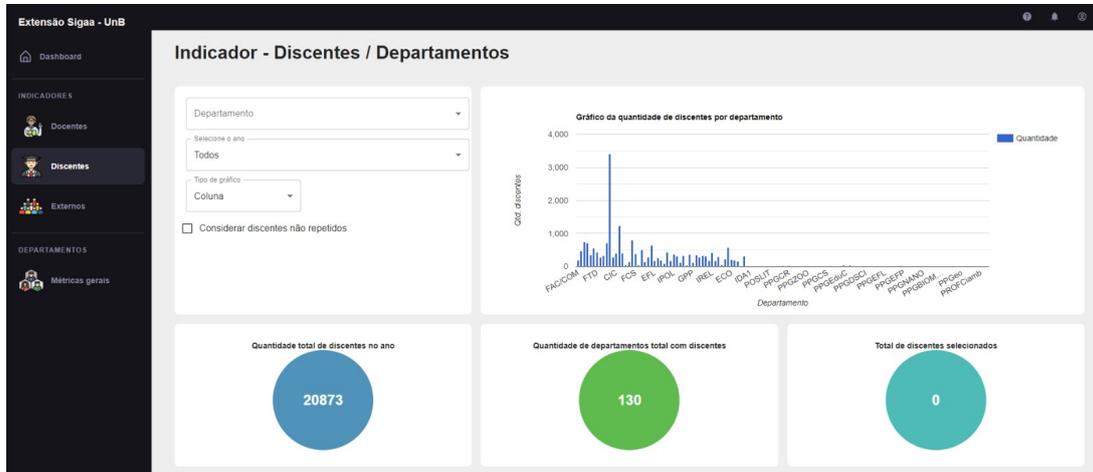


Figura 5.6: Indicador de Discentes.

Segue e evolução da participação de alunos nos projetos da Universidade de Brasília com o passar dos anos, a amostra contém dados baseados nos anos de 2020, 2021, 2022 e 2023. Vale ressaltar que em todas as métricas o resultados são observados conforme a data da execução do *script*, logo se ele for executado uma semana após ou até mesmo um dia ou horas, o resultado pode ser diferente, o que já explica a quantidade menor de discentes no ano de 2023 em comparação ao ano de 2022, visto que a maioria dos projetos são finalizados no fim do ano. Logo a ideia de disponibilizar esses dados no software é promover um controle diário sobre as métricas ao usuário, visto que na plataforma construída os dados serão atualizados diariamente.



Figura 5.7: Gráfico evolução de Discentes.

5.4 Resultados - Indicador de Externos

Os resultados do indicador de externos em projetos de extensão na UnB surpreendem positivamente, esse indicador é responsável por mostrar a quantidade de participantes das equipes do projetos que não são da Universidade de Brasília, como é possível ver na figura 5.8, foi calculado um total de 5060 participantes, para chegar nesse total foi necessário percorrer todos os projetos existentes na UnB, desde 2020 até o ano de 2023, e assim cada ocorrência de participantes externos incrementa o totalizador desse indicador. Caso o usuário deseje visualizar a quantidade de externos em cada ano, ou até mesmo em cada departamento basta ele preencher os filtros.

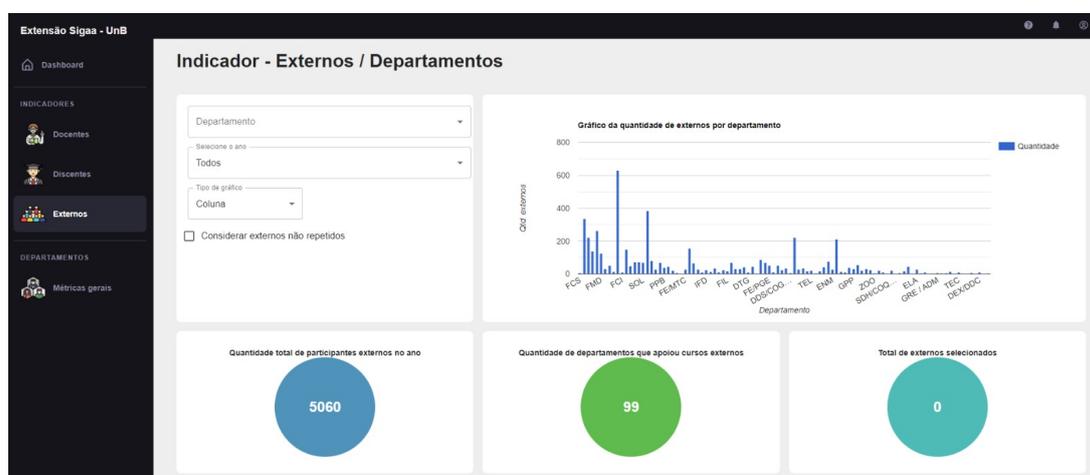


Figura 5.8: Indicador de Externos.

Foi coletado também a quantidade total de externos em cada ano, desde 2020 até 2023, os dados estão organizados na seguinte tabela para visualização, vale ressaltar a constante evolução dos números, o que mostra a expansão do cenário de extensão na Universidade.

Esclarecendo o resultado de 1208 ocorrências de externos no ano de 2023, como a coleta desses dados foi feita antes mesmo do fim do primeiro semestre na Universidade, o resultado acabou sendo menor do que o coletado no ano de 2022, e o mesmo serve para docentes e discentes. A seguir, para ter ideia melhor do resultado dessa métrica informada, os dados foram organizados em um gráfico do tipo linha, e fica nítido a evolução observada entre 2020 e 2022, e a tendência dessa evolução continuar no ano de 2023.

Agora tratando do resultado obtido do total de departamentos com participantes externos, como explicado na seção de Desenvolvimento, para calcular essa métrica o departamento considerando como sendo o pertencente ao externo é o do coordenador geral do projeto, tal decisão foi tomada em reuniões em conjunto com a orientadora desse trabalho. Outro ponto é o grande número observado, ao todo 99 departamentos, tal número de



Figura 5.9: Gráfico evolução de Externos.

expressividade se dá por conta de uniões de departamentos para a realização de um projeto, por exemplo, um projeto de extensão foi criado pelos departamentos GRE e ADM em conjunto e esses são contabilizado como departamentos distintos dos demais, como informa o SIGAA.

5.5 Resultados - Indicador de Público

Para a criação dos indicadores referentes a público interno e externo, foi necessário percorrer cada projeto de extensão no SIGAA que já foi concluído ou ainda não, observando três dados, público estimado interno, público estimado externo e o público atingido. Uma observação, o valor contido em público atingido só é entregue pelos coordenadores dos projetos quando o mesmo é concluído, logo esse dado pode ter uma divergência, já que os dados de estimados não consideram essa variável. A seguir é apresentada uma visão de como esses dados estão visíveis na plataforma criada para os funcionários do DEX, optando utilizar padrões já existentes no sistema para maior familiaridade dos usuários com o projeto.



Figura 5.10: Métricas de público.

O público estimado interno refere-se a todas as pessoas que, de alguma maneira, participaram desses projetos. Essas informações são previstas pelos próprios organizadores do projeto, o que significa que o controle sobre os dados não reside na equipe, mas sim

em suas fontes. O resultado é um total de 31.846.422 participantes, um número que impressiona pelo alto nível de engajamento previsto pelos membros da Universidade de Brasília. Deve-se notar que estes não são exclusivamente discentes, portanto, os participantes podem ter diferentes funções, como o Sistema Integrado de Gestão de Atividades Acadêmicas (SIGAA) não especifica a função do público.

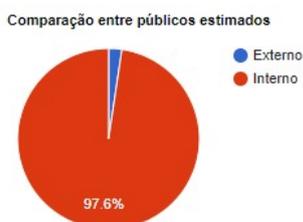


Figura 5.11: Gráfico público.

A figura 5.11 fornece uma visão comparativa ao usuário, permitindo visualizar a porcentagem que cada dado representa quando combinados. Observou-se que 76.1% do público geral são estimados como internos da universidade e, posteriormente, 2% destes são previstos como externos. Este dado é apreciado porque mostra que os projetos buscam, na medida do possível, engajar pessoas de diferentes áreas, aumentando o impacto da extensão da UnB. No entanto, a ideia de que essa estimativa pode ser aprimorada não foi descartada. Após o cálculo do público interno e externo estimado, foi observado que 22.1% dessas projeções foram atingidas, o que reflete a perspectiva otimista dos coordenadores dos projetos. Importante ressaltar que este dado se baseia em projetos já concluídos, portanto, os números podem mudar significativamente com o passar do tempo.

5.6 Resultados - Indicador de Financiamento

Nesse último indicador informações interessantes foram obtidas. Esses dados foram coletados pela busca em cada projeto e observando o tipo de financiamento de cada projeto, assim em cada ocorrência de um tipo seu valor era incrementado, desta forma chegou-se nos seguintes dados:

Foi observado que de todos os projetos existentes na Universidade de Brasília até o momento da coleta dos dados, 1361 projetos tem como fonte de verba o financiamento interno, logo os custos com os projetos são de responsabilidade de própria Universidade e seus meios.

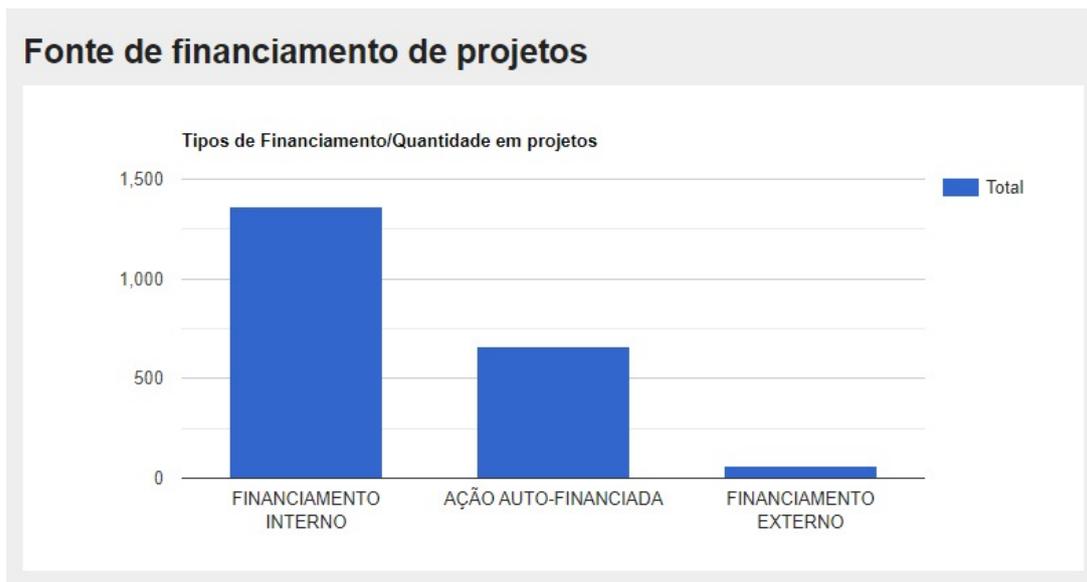


Figura 5.12: Gráfico de tipos de financiamentos (projetos de extensão).

Em seguida, como apresentado no gráfico destacado, projetos com fonte de verba buscando o autofinanciamento chegaram ao total de 660 em todos os anos de 2020 até 2023. Nesse tipo de financiamento os próprios participantes arcam com os gastos ou criam meios de arcar com as despesas necessárias para a criação e manutenção dos projetos, por isso tal resultado surpreendeu a equipe pois havia uma crença de que os projetos seriam financiados pelos meios universitários.

Por último, os projetos com fonte de financiamento externo somam um total de 65. Essa fonte de financiamento não tem relação com a Universidade, logo os organizadores dos projetos buscam meios externos para suprir os gastos necessários para criação e manutenção das atividades propostas.

5.7 Evolução dos projetos

Nessa seção é apresentada uma métrica coletada que mostra bem a realidade e expectativa que envolve o cenário de extensão na Universidade de Brasília. Foram coletados dados que mostram a quantidade de projetos existentes na UnB até o momento, desde 2020 até o ano atual 2023, o resultado alcançado é mostrado no gráfico a seguir:

Analisando este resultado é observado que em 2020 foram criados 174 projetos na Universidade, em 2021 surgiram 545, 647 em 2022 e por fim até o momento 709 no ano de 2023, para organizar melhor segue uma tabela com os dados.

Vale ressaltar que esses dados podem sofrer alteração nos resultados de 2022, pois como verificado em reuniões com a equipe, ainda existem projetos ativos que surgiram em



Figura 5.13: Gráfico evolução de projetos.

2022 até o momento, outro ponto é a grande quantidade de projetos no ano de 2023, no atual momento, comparado com o ano anterior, tal diferença é positiva pois mostra que mesmo estando no primeiro semestre do ano o resultado já ultrapassa o ano anterior.

5.8 Tempo de execução do *script*

Os resultados até agora demonstram que o *script* Python, responsável pela coleta de dados, possui um tempo de execução considerável, ainda que dentro dos limites esperados e definidos pelos critérios de aceitação da equipe. Tais critérios incluem parâmetros como tempo de execução aceitável, eficiência de uso dos recursos computacionais, confiabilidade na coleta de dados e gestão apropriada de possíveis erros. A coleta de projetos de extensão pelo sistema SIGAA, por exemplo, tem uma duração aproximada de 59 minutos. Para outras métricas associadas a diferentes ações, o tempo de execução pode variar ligeiramente, mas em geral, permanece próximo a este valor.

Para coletar essa métrica foi utilizada a biblioteca *time* do Python, assim, no início do código para a execução do *script* a hora atual é armazenada em uma variável, e ao final do código outra variável é criada para armazenar a hora no final da execução, assim basta realizar a diferença entre a hora final pela hora inicial e se obtém, em segundos, o tempo de execução.

Capítulo 6

Conclusão

O projeto para esse Trabalho de Graduação em Ciência da Computação na Universidade de Brasília (UnB) teve como meta a produção de um software para sanar diversas dores de usuários do DEX, estes possuem muita dificuldade de analisar e acompanhar o cenário que envolve o contexto de extensão na Universidade, com isso esse software foi feito com base em dicas e vontades colhidas pela equipe que suprissem as necessidades desses usuários. Ao fim desse projeto foi obtido um produto que contém informações valiosas para o correto acompanhamento das métricas que envolvem projetos de extensão.

Por conta da falta de ferramentas que disponibilizassem um acompanhamento mais produtivo por conta das limitações do SIGAA, muitos usuários geravam relatórios massivos e superficiais para o acompanhamento dos dados de extensão. A ferramenta focou nesses pontos fracos, colhendo todas as informações necessárias do SIGAA, por meio de automação *script*, na linguagem *Python*, e assim esses dados foram tratados e calculados no *back-end* para a apresentação dos indicadores no *front-end*. O trabalho focou em uma relação amigável e agradável ao usuário, utilizando padrões de design focados no UX, assim chegando em uma interface de fácil interação com esses funcionários.

Os resultados informados pelos indicadores foram satisfatórios, pois a base de dados contava com o próprio sistema da Universidade de Brasília, o SIGAA, dessa forma foi preciso apenas coletar esses dados para chegar em resultados positivos após os cálculos. Foi notado a evolução constante do cenário de extensão na UnB e vale destacar a importância dessa evolução, pois esses projetos fornecem aos alunos e pessoas afetadas uma vivência focada na realidade, saindo muitas vezes do meio apenas teóricos de aulas e levando o conhecimento adquirido para a prática.

O *script* para coleta de dados mostrou alternativas interessantes que, após a ideia de utilização, outro trabalho na Universidade de Brasília foi iniciado utilizando esse mesmo mecanismo. As vantagens pela utilização desse meio tem um peso maior do que as suas limitações, na percepção da equipe, pois oferece uma maior liberdade em relação aos dados

disponíveis, visto que não foi preciso passar por processos burocráticos na UnB para ter a autorização de acessar uma base de dados, em contrapartida a demora para a execução do *script* apresenta um ponto de atenção, mas para as finalidades deste trabalho, essa demora não compromete o projeto.

Referências

- [1] Gadotti, Moacir: *Extensão universitária: para quê*. Instituto Paulo Freire, 15:1–18, 2017. 1
- [2] Deus, Sandra de Fatima Batista de: *A valorização da extensão universitária no brasil*. Revista Brasileira de Extensão Universitária, 8(2):121–124, 2017. 1
- [3] SUPERIOR, MINISTÉRIO DA EDUCAÇÃO. CONSELHO NACIONAL DE EDUCAÇÃO. CÂMARA DE EDUCAÇÃO: *Resolução nº 7, de 18 de dezembro de 2018 (*) (**) estabelece as diretrizes para a extensão na educação superior brasileira e regimenta o disposto na meta 12.7 da lei nº 13.005/2014*. Diário Oficial da União, Brasília, DF, 18 dez. 2018. Seção 1, p. 34, 2018. Disponível em: https://normativasconselhos.mec.gov.br/normativa/pdf/CNE_RES_CNECESN72018.pdf. Acesso em: 05/08/2023. 1
- [4] Dex/UnB: *Extensão será obrigatória no currículo da graduação em 2023*, 2023. <https://dex.unb.br/noticias/931-extensao-sera-obrigatoria-no-curriculo-da-graduacao-em-2023>, acesso em 2023-07-26. 1
- [5] Dex/UnB: *Página de editais da divisão de extensão da universidade de Brasília*, 2023. <https://dex.unb.br/editais>, acesso em 2023-07-26. 2
- [6] BORTOLINI, MARISTELA HELENA ZIMMER, MAURÍCIO RISSI e JULIANA DE BONA GARCIA VENDRÚSCOLO: *Indicadores de extensão universitária*. XIX Colóquio Internacional de Gestão Universitária, 25, 26, 27 de dezembro de 2019, INPEAU/UFSC, 2019. 4
- [7] MAXIMINIANO JÚNIOR, Manoel, Ana Inês SOUSA, Dalva Maria de Oliveira SILVA, Etevaldo Almeida SILVA, Maristela Helena Zimmer BORTOLINI, Nadege da Silva DANTAS e Regina Lúcia Monteiro HENRIQUES: *Indicadores brasileiros de extensão universitária (ibeu)*. 2017. 5, 12, 13
- [8] Tufte, Edward: *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT, 2001. 5
- [9] Healy, Kieran: *Data Visualization: A Practical Introduction*. Princeton University Press, Princeton, NJ, 2018. 5
- [10] Krug, Steve: *Don't Make Me Think: A Common Sense Approach to Web Usability*. New Riders, Berkeley, CA, 2000. 6

- [11] Doglio, Fernando, Doglio e Corrigan: *REST API Development with Node.js*, volume 331. Springer, 2018. 6
- [12] Cantelon, Mike, Marc Harter e TJ Rajlich: *Node.js in Action*. Manning Publications, Shelter Island, NY, 2017. 6
- [13] Sweigart, Al: *Automate the Boring Stuff with Python*. No Starch Press, San Francisco, CA, 2015. 7
- [14] Banks, Alex e Eve Porcello: *Learning React: Functional Web Development with React and Redux*. O'Reilly Media, Sebastopol, CA, 2017. 7
- [15] Li, Qian: *Next.js Quick Start Guide*. Packt Publishing, Birmingham, UK, 2020. 7
- [16] Schwaber, Ken e Jeff Sutherland: *Agile Project Management with Scrum*. Microsoft Press, 2002. 8
- [17] Vazquez, Carlos Eduardo e Guilherme Siqueira Simões: *Engenharia de Requisitos: software orientado ao negócio*. Brasport, 2016. 16, 17
- [18] Silva, Rodrigo Eduardo: *Requisitos para integração de ferramentas de engenharia de software*. Relatório interno. Departamento de Ciências da Computação. UFMG. Belo Horizonte-MG, 2007. 18
- [19] Bangare, SL, S Gupta, M Dalal e A Inamdar: *Using node.js to build high speed and scalable backend database server*. International Journal of Research in Advent Technology, 4:19, 2016. 20
- [20] Amershi, Saleema, Andrew Begel, Christian Bird, Robert DeLine, Harald Gall, Ece Kamar, Nachiappan Nagappan, Besmira Nushi e Thomas Zimmermann: *Software engineering for machine learning: A case study*. Em *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, páginas 291–300. IEEE, 2019. 21