Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Desenvolvimento e avaliação de uma aplicação Web para acompanhamento de estudantes com necessidades específicas

Henrique Lopes Curzio

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia de Computação

Orientador
Prof. Dr. Daniel de Paula Porto

Brasília
2023

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Desenvolvimento e avaliação de uma aplicação Web para acompanhamento de estudantes com necessidades específicas

Henrique Lopes Curzio

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia de Computação

Prof. Dr. Daniel de Paula Porto (Orientador)
CIC/UnB

Prof.a Dr.a Germana Menezes da Nóbrega
CIC/UnB

Prof.a Dr.a Fernanda Lima
CIC/UnB

Prof. Dr. João Luiz Azevedo de Carvalho
Coordenador do Curso de Engenharia de Computação

Brasília, 20 de julho de 2023

Dedicatória

Dedico este trabalho aos meus pais.

Agradecimentos

Agradeço primeiramente a Deus. Em segundo lugar, às pessoas que investiram caro na minha formação e tornaram este projeto possível; meus pais, avó, tia e tio. Em terceiro, agradeço aos que acreditaram em mim, me apoiaram e me incentivaram a continuar; irmãos, familiares e líderes. E, por fim, aos tutores que me capacitaram com conhecimento e habilidades necessárias para cumprir este trabalho e lidar com desafios futuros; meus pastores e professores (escolares, universitários e orientadores).

Resumo

Contexto – este trabalho descreve o processo de desenvolvimento de uma aplicação Web para acompanhamento de estudantes com necessidades específicas do Departamento de Ciência da Computação (CIC) da Universidade de Brasília (UnB), com foco na engenharia de usabilidade e na Interação Humano–Computador (IHC). **Objetivo** – disponibilizar a estes estudantes um ambiente de integração com professores e tutores de forma a fornecer orientações gerais, comunicação e compartilhamento de informações entre eles. **Métodos** – o processo de desenvolvimento foi feito utilizando tecnologias *Web Standards* (HTML, CSS e JavaScript) do lado do cliente, Node.js do lado do servidor e banco de dados SQLite, além de outras ferramentas auxiliares, protocolos, algoritmos e funções. Durante a criação do sistema é utilizado métodos de avaliação de usabilidade no ciclo de desenvolvimento de software como forma de examinar o design de interação e a experiência do usuário. **Resultados** – os requisitos iniciais da aplicação foram detalhados e apenas parte deles foram selecionados para este trabalho, em ordem de prioridade e aplicabilidade. O desenvolvimento teve como foco a engenharia de usabilidade e todas as páginas desenvolvidas foram avaliadas, inspecionadas e testadas segundo vários critérios de usabilidade como forma de identificar falhas e erros do sistema. **Conclusões** – este trabalho destacou a natureza cíclica e contínua do desenvolvimento de software, onde os testes e inspeções devem ser realizados recorrentemente buscando sempre melhorar a usabilidade. O prosseguimento de requisitos não cumpridos e a correção de falhas detectadas foram descritos como trabalhos futuros.

Palavras-chave: Interação Humano Computador, aplicação web, desenvolvimento web, avaliação de usabilidade, heurísticas de usabilidade, design de interação, acessibilidade

Abstract

Context – This work describes the process of developing a Web application to monitor students with specific needs at the Departamento de Ciência da Computação (CIC) at the Universidade de Brasília (UnB), focusing on usability engineering and human-computer interaction (HCI). **Purpose** – to provide these students with an integration environment with professors and tutors in order to provide general guidance, communication and information sharing between them. **Methods** – the development process was done using standard Web technologies (HTML, CSS and JavaScript) on the client side, Node.js on the server side and SQLite database, in addition to other auxiliary tools, protocols, algorithms and functions. During the creation of the system, usability evaluation methods are used in the software development cycle as a way to examine the interaction design and user experience. **Results** – the application's initial requirements were detailed and only part of them were selected for this work, in order of priority and applicability. The development focused on usability engineering and all pages developed were evaluated, inspected and tested according to various usability criteria as a way of identifying failures and system errors. **Conclusions** – This work highlighted the cyclical and continuous nature of software development, where tests and inspections must be performed recurrently, always seeking to improve usability. The continuation of unfulfilled requirements and the correction of detected failures were described as future work.

Keywords: Human Computer Interaction, web application, web development, usability evaluation, usability heuristics, interaction design, accessibility

Sumário

Lista de Figuras

Lista de Quadros

Lista de Abreviaturas e Siglas

ACID atomicity, consistency, isolation, durability.

API application programming interface.

CAD Conselho de Administração.

CEAD Centro de Educação a Distância.

CI/CD Continuous Integration/Continuous Deployment.

CIC Departamento de Ciência da Computação.

CRUD create, read, update, and delete.

CSS Cascading Style Sheets.

DACES Diretoria de Acessibilidade.

DACES/DAC Diretoria de Acessibilidade.

HCI human-computer interaction.

HTML HyperText Markup Language.

HTTP Hypertext Transfer Protocol.

IDE Integrated development environment.

IF-BrM Índice de Funcionalidade Brasileiro Modificado.

IHC Interação Humano–Computador.

JIT just-in-time.

JSON JavaScript Object Notation.

npm Node Package Manager.

PcD Pessoa com Deficiência.

PNAD Contínua Pesquisa Nacional por Amostra de Domicílios Contínua.

PTA Programa de Tutoria para Acessibilidade.

RDBMS Relational Database Management System.

SDLC software development life cycle.

SGBDR Sistema de Gerenciamento de Banco de Dados Relacional.

SIGAA Sistema Integrado de Gestão de Atividades Acadêmicas.

SPA Single Page Application.

TEA Transtorno do Espectro Autista.

TFE Transtornos Funcionais Específicos.

UI user interface.

UnB Universidade de Brasília.

UX user experience.

VCS Version Control System.

W3C World Wide Web Consortium.

WCAG Web Content Accessibility Guidelines.

WYSIWYG What You See Is What You Get.

Capítulo 1

Introdução

1.1 Motivação e contexto

Atualmente, na UnB, existem vários projetos para inclusão de estudantes com necessidades específicas. Esses serão detalhados posteriormente. No entanto, não existe um professor exclusivo que acompanhe o aluno em suas necessidades em relação ao curso de forma geral. Este ponto de falha é a principal motivação para o desenvolvimento deste trabalho. Portanto, este é um projeto aplicado que considerará um contexto onde já exista esse professor para acompanhar o aluno.

A ocasião deste trabalho está inserida em uma ideia para o Departamento de Ciência da Computação (CIC) da Universidade de Brasília (UnB) de desenvolver uma aplicação de acompanhamento e auxílio de alunos com necessidades específicas. Estes alunos são pessoas que se enquadram em critérios específicos de exigências e precisam de maior atenção e acompanhamento pelos professores. Os principais objetivos da aplicação são realizar integração, comunicação e compartilhamento de informações entre o aluno com a necessidade específica e um professor intitulado "professor-tutor". Este é um perfil hipotético em que cada estudante recebe o acompanhamento de um professor exclusivo para orientação durante todo o curso de forma a estabelecer uma conexão permanente entre o aluno e um auxiliador. Caso disponibilizado pela instituição, o sistema deve possibilitar também a integração com um aluno tutor intitulado apenas "tutor". Esta função é parte de um programa de acessibilidade na universidade.

Assim, considerando esta motivação e este contexto, é proposto um projeto de um sistema de integração dos alunos com esse professor-tutor e com o tutor já existente nos programas da UnB.

1.1.1 Acessibilidade

Este trabalho está inserido em um contexto de acessibilidade e, por isso, faz-se necessário um aprofundamento da sua atual situação no ambiente estudado. Isto é necessário como forma de identificar recursos já implantados e propostas de melhorias. Segundo a Lei Brasileira de Inclusão [6],

> "acessibilidade é a possibilidade e condição de alcance para utilização, com segurança e autonomia, de espaços, mobiliários, equipamentos urbanos, edificações, transportes, informação e comunicação, inclusive seus sistemas e tecnologias, bem como de outros serviços e instalações abertos ao público, de uso público ou privados de uso coletivo, tanto na zona urbana como na rural, por pessoa com deficiência ou com mobilidade reduzida."

O Índice de Funcionalidade Brasileiro Modificado (IF-BrM) e a Diferenciação e Acessibilidade Curricular [7] afirmam que a acessibilidade constitui-se de três níveis de competências [8] ilustradas na Figura 1.1:

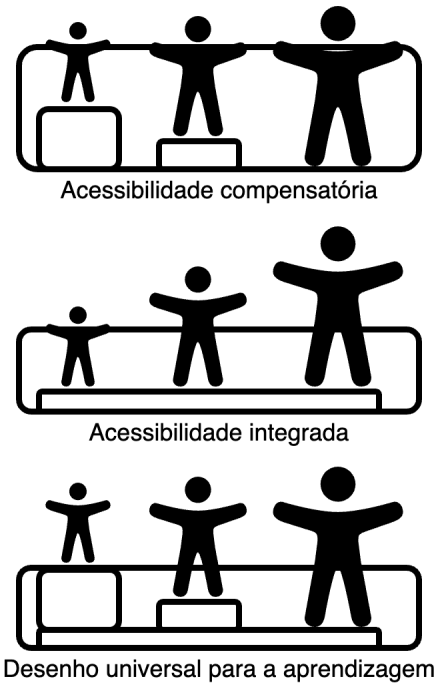

Figura 1.1: Níveis de competências da acessibilidade.

- Acessibilidade Compensatória: quanto mais complexos os comprometimentos sinalizados pelo sujeito e identificados pelos atores do contexto, maior a necessidade de planejamento cooperativo e apoio externo à instituição educacional ou profissional;

- Acessibilidade Integrada: quando o sujeito está em situação de risco de fracasso escolar, integram-se novas práticas, no próprio contexto, sem acionar apoios externos, individualizando algumas estratégias específicas;
- Desenho Universal para a Aprendizagem: integra a Acessibilidade Compensatória e a Acessibilidade Integrada, de maneira planejada, envolvendo todos os atores na mudança do cenário.

São consideradas pessoas com deficiência aquelas que se enquadrem no Estatuto da Pessoa com Deficiência, denominação da Lei Brasileira de Inclusão da Pessoa com Deficiência, Lei Nacional nº 13.146, de 6 de julho de 2015. Esta lei em vigor no Brasil garante os direitos das pessoas com deficiência e impõe as penalidades a quem infringir a lei [9].

- "Art. 1o É instituída a Lei Brasileira de Inclusão da Pessoa com Deficiência (Estatuto da Pessoa com Deficiência), destinada a assegurar e a promover, em condições de igualdade, o exercício dos direitos e das liberdades fundamentais por pessoa com deficiência, visando à sua inclusão social e cidadania."
- "Art. 2o Considera-se pessoa com deficiência aquela que tem impedimento de longo prazo de natureza física, mental, intelectual ou sensorial, o qual, em interação com uma ou mais barreiras, pode obstruir sua participação plena e efetiva na sociedade em igualdade de condições com as demais pessoas."

Acessibilidade na Universidade de Brasília

A "Política de Acessibilidade da UnB" [10] reafirma a acessibilidade como direito para a comunidade universitária. Esse trabalho fica aos cuidados da Diretoria de Acessibilidade (DACES/DAC). Ela é responsável por fazer toda a gestão de recursos humanos e materiais para atendimento e acompanhamento desses alunos e realizar programas de acessibilidade no campus. Além disso, ela faz parceria com o Centro de Educação a Distância (CEAD) de forma a promover recomendações de acessibilidade para conteúdos digitais da universidade. O público atendido pela Diretoria de Acessibilidade (DACES/DAC) vai além das pessoas com deficiências (PcD). Ela fornece cuidados a estudantes com quaisquer necessidades específicas, mesmo que não seja considerada uma deficiência.

Essa política ratifica a definição contida na Lei Brasileira de Inclusão e, em seu Art. 2°, reafirma a acessibilidade como direito para a comunidade universitária. Na política institucional da UnB, a acessibilidade é considerada em perspectiva multidimensional [11]. Para fins desta política, considera-se:

- Acessibilidade arquitetônica e urbanística: aquela existente nos edifícios, nas vias e espaços abertos ao público ou de uso coletivo;

- Acessibilidade nos transportes: aquela existente nos sistemas e meios de transportes;
- Acessibilidade na comunicação e na informação: aquela existente nos sistemas de comunicação e tecnologia da informação, e no acesso à informação e ao conhecimento;
- Acessibilidade atitudinal: aquela existente nas atitudes e comportamentos, com vistas a garantir a participação social da pessoa com deficiência, em igualdade de condições e oportunidades, com as demais pessoas; e
- Acessibilidade pedagógica: aquela compreendida nos processos de ensino e aprendizagem, bem como no acompanhamento acadêmico dos estudantes, com vistas a prevenir situações de retenção e evasão.

Nesse sentido, a promoção da acessibilidade no ensino se refere à eliminação de barreiras que impedem o acesso, a participação e a aprendizagem aos estudantes com deficiência e/ou necessidades educacionais específicas, em seus processos formativos, nos cursos de graduação e de pós-graduação [11].

O projeto especificado neste documento visa promover principalmente a acessibilidade na comunicação e na informação. Além disso, seus resultados devem refletir também na acessibilidade atitudinal e na acessibilidade pedagógica.

Diretoria de Acessibilidade (DACES/DAC)

O objetivo da DACES/DAC é garantir e promover a inclusão e a acessibilidade como uma política transversal na UnB, de forma a ampliar condições de acesso, acessibilidade, participação e aprendizagem aos estudantes que possuem deficiência, transtornos do espectro autista, altas habilidades/superdotação ou transtornos funcionais específicos. Entende-se que a construção de uma Universidade mais inclusiva se dá a partir da eliminação de barreiras e da articulação entre unidades acadêmicas e administrativas [12].

A Resolução do Conselho de Administração (CAD) n° 50/2019 [10][13], que institui a Política de Acessibilidade da UnB, em seu Art. 3°, define como público atendido o corpo discente, os servidores técnico-administrativos, os docentes e a comunidade, em geral, da Universidade, identificados como:

- Pessoa com Deficiência (PcD): as que têm impedimentos de longo prazo, de natureza sica, intelectual ou sensorial, os quais, em interação com diversas barreiras, podem obstruir sua participação plena e efetiva na sociedade, em igualdade de condições com as demais pessoas;
- Pessoas com transtornos globais do desenvolvimento ou com Transtorno do Espectro Autista (TEA): as que apresentam alterações qualitativas das interações sociais

recíprocas e da comunicação, um repertório de interesses e atividades restrito, estereotipado e repetitivo. Incluem-se nesse grupo pessoas com autismo, síndrome de Re, síndrome de Asperger e transtorno desintegrativo da infância;

- Pessoas com altas habilidades e superdotação: as que demonstram potencial elevado nas áreas intelectual, acadêmica, artística, de liderança e psicomotricidade, isoladas ou combinadas, além de apresentar grande criatividade, envolvimento na aprendizagem e realização de tarefas em áreas de seu interesse;
- Pessoas com Transtornos Funcionais Específicos (TFE): as que apresentam dislexia, disortografia, disgrafia, discalculia e transtorno de atenção e hiperatividade, entre outros. O público-alvo elencado neste argo poderá ser ampliado ou restringido de acordo com a legislação e demais normas vigentes, adotando-se a nomenclatura adequada.

Programa de Tutoria para Acessibilidade (PTA)

O Programa de Tutoria para Acessibilidade (PTA) integra o conjunto de ações da DACES/DAC, visando promover acessibilidade aos estudantes regularmente matriculados nos cursos de graduação cadastrados e acompanhados por esta Diretoria. O apoio de tutor é destinado a pessoas com deficiência, transtorno do espectro autista, altas habilidades ou superdotação e transtornos funcionais específicos. O tutor atua na promoção da acessibilidade e na mediação entre: o tutorado, a DACES, a coordenação do curso e os professores das disciplinas. Ele tem como objetivo eliminar barreiras que impedem o processo de aprendizagem e a plena participação na vida universitária dos estudantes que são público da Política de Acessibilidade da Universidade de Brasília. O tutor e o aluno tutorado devem estar matriculados no mesmo curso de graduação ou em cursos relacionados [14].

O sistema a ser desenvolvido visa ser um auxiliador deste programa de tutoria atuando na promoção da acessibilidade e na mediação do aluno. A aplicação busca fazer a mediação entre: o aluno tutorado, o professor intitulado "professor-tutor" e o tutor participante do PTA específico do aluno. Esta função de *professor-tutor* é uma proposta estabelecida neste projeto para utilização no Departamento de Ciência da Computação (CIC) para acompanhamento destes estudantes.

Centro de Educação a Distância (CEAD)

O Centro de Educação a Distância (CEAD) da Universidade de Brasília (CEAD/UnB) surgiu como órgão na estrutura da universidade, com a tarefa de desenvolver e viabilizar ações educativas a distância. Para tanto, desde o ano de 1979, promove acesso à educação,

à cultura e aos saberes em diversas áreas do conhecimento [15]. O CEAD é responsável pelos seguintes projetos relacionados a acessibilidade no ensino virtual:

- Parceria com a DACES/UnB: com o objetivo de disponibilizar plataformas virtuais para todos de forma acessível e inclusiva. As plataformas virtuais possuem um papel de destaque, ou mais que isso, é de vital importância para o ensino remoto e para a EaD;
- Constituição do Núcleo de Acessibilidade dentro do CEAD: uma equipe que tem como papel principal promover a acessibilidade nas atividades de educação a distância e de ensino remoto desenvolvidas pelo CEAD/UnB;

O trabalho tem como principal eixo as Diretrizes de Acessibilidade para o Conteúdo da Web por meio da Web Content Accessibility Guidelines (WCAG), que são as recomendações de acessibilidade para conteúdo da Web, ou seja, são diretrizes que explicam como tornar o conteúdo Web acessível, e fazem parte de uma série de recomendações para acessibilidade publicadas pela Web Accessibility Initiative do W3C, instituição mundialmente conhecida por pesquisar tecnologias que promovem padrões de uso e forma para a criação e a interpretação de conteúdos para a Web [15].

Guia de acessibilidade

O Guia de Orientações para a Promoção da Acessibilidade no Ensino Remoto é resultado da união de dois grandes grupos, o Centro de Educação a Distância (CEAD) e a Diretoria de Acessibilidade (DACES) e tem como propósito oferecer para discentes, docentes, técnicos, colaboradores e comunidade da Universidade de Brasília em geral ferramentas tecnológicas e orientações didático-pedagógicas na perspectiva de promover acessibilidade em suas atividades de ensino, pesquisa e extensão. As ações vinculadas ao guia têm como objetivo maior apoiar toda a comunidade acadêmica a planejar, ofertar e desenvolver melhor o ensino remoto de forma acessível [16].

O guia apresenta a definição das deficiências e de necessidades educacionais específicas dos estudantes atendidos pela DACES/DAC, bem como sugestões de recursos de acessibilidade para utilização no ensino remoto. Além disso, destaca a necessidade de os docentes sempre dialogarem com os estudantes, no sentido de verificar quais os recursos de acessibilidade que melhor atendem às suas demandas. Ele busca destacar que uma aula acessível amplia as oportunidades de aprendizagem e a participação para todos os estudantes, com ou sem deficiência. As principais necessidades são listadas abaixo:

- Com deficiência visual (cegueira): perda total da função visual ou pouquíssima capacidade de enxergar;

- Com deficiência visual (baixa visão): perda parcial da função visual. Nesse caso, o aluno possui resíduo visual, e seu potencial de utilização da visão para atividades escolares e de locomoção é prejudicado, mesmo após o melhor tratamento ou a máxima correção óptica específica;
- Surdos: consiste em impedimentos permanentes de natureza auditiva, ou seja, na perda total da audição que, em interação com barreiras comunicacionais e atitudinais, podem impedir a plena participação e aprendizagem do aluno;
- Com deficiência auditiva: consiste em impedimentos permanentes de natureza auditiva, ou seja, na perda parcial da audição que, em interação com barreiras comunicacionais e atitudinais, podem impedir a plena participação e aprendizagem do aluno;
- Surdocegos: trata-se de deficiência única, caracterizada pela associação da deficiência auditiva (com ou sem resíduo auditivo) e visual (com ou sem resíduo visual) concomitante. A surdocegueira pode ser classificada de duas formas: pré-linguística e pós-linguística. Na pré-linguística, a pessoa nasce surdocega ou adquire a surdocegueira muito precocemente, antes da aquisição de uma língua. Na forma pós-linguística, uma das deficiências (auditiva ou visual) ou ambas são adquiridas após a aquisição de uma língua (a Língua Portuguesa ou a Língua Brasileira de Sinais). Cabe destacar que essa condição apresenta outras particularidades, além daquelas causadas pela deficiência auditiva, surdez, baixa visão e cegueira;
- Com deficiência intelectual: caracteriza-se por alterações significativas, relacionadas a déficit tanto no desenvolvimento intelectual quanto na conduta adaptativa e na forma de expressar habilidades práticas, sociais e conceituais;
- Com deficiência múltipla: consiste na associação de duas ou mais deficiências;
- Com transtorno do espectro autista (TEA): quadro clínico caracterizado por deficiência persistente e clinicamente significativa que causa alterações qualitativas nas interações sociais recíprocas e na comunicação verbal e não verbal, ausência de reciprocidade social e dificuldade em desenvolver e manter relações apropriadas ao nível de desenvolvimento da pessoa. Além disso, a pessoa apresenta um repertório de interesses e atividades restrito e repetitivo, manifestados por comportamentos motores ou verbais estereotipados. Assim sendo, são comuns a excessiva adoção de rotinas e padrões de comportamento ritualizados, bem como interesses restritos e fixos;
- Com altas habilidades/superdotação: pessoas com altas habilidades/superdotação demonstram elevado potencial intelectual, acadêmico, de liderança, psicomotor e

artístico, de forma isolada ou combinada, além de apresentarem grande criatividade e envolvimento na aprendizagem e realização de tarefas em áreas de seu interesse;

- Com transtornos funcionais específicos: as que apresentam dislexia, disortografia, disgrafia, discalculia e transtorno de atenção e hiperatividade, entre outros;

Acessibilidade no Departamento de Ciência da Computação (CIC)

Em cada período de aulas da UnB, o Departamento de Ciência da Computação (CIC) e os outros departamentos da universidade recebem novos estudantes com necessidades específicas em seus cursos. Para identificá-los, os departamentos mantêm comunicação com a DACES/DAC, que também disponibiliza um recursos de identificação pela plataforma SIGAA. Nela, os próprios professores que ministram a disciplina cursada pelo estudante pode verificar se tem estudantes atendido pela DACES/DAC por meio de um símbolo especial no nome do aluno.

A partir deste ponto de identificação do aluno, cada professor é recomendado a seguir o Guia de Acessibilidade [16] em sua disciplina, podendo também trabalhar em colaboração com o tutor do Programa de Tutoria para Acessibilidade (PTA). Este acompanhamento é feito diretamente entre o professor da disciplina cursada e o aluno, ou seja, para cada disciplina que o aluno faz é necessário que este professor atenda às necessidades específicas do estudante. No entanto, não existe atualmente um professor externo a esta relação da disciplina que acompanhe o aluno em suas necessidades em relação ao curso de forma geral. Este ponto de falha é a principal motivação para o desenvolvimento deste trabalho.

Professor-tutor

Como forma de promover um melhor acompanhamento e auxílio dos estudantes com necessidades específicas do departamento, este trabalho propõe um perfil hipotético de "professor-tutor", em que cada estudante cadastrado no programa de tutoria recebe o acompanhamento de um professor exclusivo para orientação durante todo o curso. Esta proposta visa estabelecer uma conexão permanente entre o aluno e um auxiliador, diferente do programa de tutoria em que a conexão com outro aluno é feita provisoriamente. Desta forma, os alunos que possuem mais resistência a mudanças podem ser beneficiados. Além disso, no caso do estudante não se adaptar ao professor, ele pode solicitar outro professor-tutor ao coordenador do programa de tutoria do departamento.

1.1.2 Usabilidade

A usabilidade é um campo de estudo presente em várias áreas da engenharia e tem sido amplamente difundida nos sistemas de informação. Um aumento no uso de aplicações Web

tem demandado cada vez mais de um estudo da usabilidade desses sistemas buscando um design inclusivo por meio da união de acessibilidade, usabilidade e inclusão. A UnB tem vários sistemas Web em funcionamento, onde parte deles apresenta vários recursos de usabilidade, inclusive alguns de acessibilidade.

A temática detalhada como motivação e contexto tem entre os usuários finais, pessoas que podem ter maiores dificuldades de interação com o sistema. O desenvolvimento de sistemas focado neste público reflete uma melhoria na usabilidade até para usuários sem necessidades específicas e usuários especialistas. Basicamente o objetivo da usabilidade é atingir uma aplicação simples para que todos possam usá-la e entendê-la independentemente dos níveis de alfabetização técnica ou experiência.

O *design* centrado no ser humano e o *design* inclusivo permitem que todos os usuários dos sistemas sejam beneficiados quando envolve um estudo de sua usabilidade por meio de teorias consolidadas no campo de Interação Humano–Computador (IHC) [17]. Isto envolve um traçado de metas no design de interação, experiência de usuário e design centrado no ser humano. Porém, quando o sistema desenvolvido envolve diretamente usuários com necessidades específicas, mais do que nunca, este sistema deve ter uma boa usabilidade para facilitar o uso por parte desses usuários.

1.2 Objetivos

Objetivo geral

Este trabalho tem como objetivo geral criar uma aplicação de boa usabilidade que possibilite realizar a integração, comunicação e compartilhamento de informações entre alunos com necessidades específicas do Departamento de Ciência da Computação (CIC) e um professor intitulado "professor-tutor". Essa aplicação, chamada "Integrar", deve possibilitar também a integração com um aluno tutor intitulado "tutor", caso haja disponibilidade por parte da instituição.

Objetivos específicos

Como forma de atingir o objetivo geral, os seguintes objetivos específicos foram definidos:

- Identificar na literatura estudos sobre usabilidade e avaliação de usabilidade de Interação Humano–Computador (IHC)
- Elaborar os requisitos iniciais da aplicação e priorização dos que serão implementados nestre tabalho;
- Implementar a aplicação com base nas ferramentas e tecnologias mais apropriadas;

- Realizar avaliações de usabilidade concomitante com a implementação de forma de identificar falhas e erros do sistema;

Requisitos

A seguir é apresentada uma breve descrição das principais funcionalidades que o sistema deve atender:

- **Usuários**

  A aplicação deve ter uma seção dedicada à administração do sistema e gerenciamento de usuários acessível somente pelo usuário com perfil do tipo *coordenador*. Nela deve ser possível criar e deletar novos usuários e editar suas informações. Quatro perfis de usuários devem ser aceitos: *aluno*, *tutor*, *professor-tutor* e *coordenador*. Além disso, nesta seção deve ser possível estabelecer a relação entre cada aluno e seus tutores ou professores-tutor;

- **Orientações gerais**

  Esta seção deve fornecer aos usuários com perfil *professor-tutor* e *coordenador*, um editor WYSIWYG onde é possível fornecer informações gerais da aplicação e do funcionamento do programa de tutoria. As informações devem ser abertas para leitura por quaisquer usuários com acesso à página;

- **Acompanhamento**

  Esta seção deve disponibilizar aos usuários, um calendário para organização de tarefas estabelecidas por quaisquer usuários e atribuídas ou direcionadas a outro usuário com algum grau de relação com o criador da tarefa. Somente devem ser listadas as tarefas com algum grau de relação com o usuário atual logado no sistema;

- **Comunicação**

  Esta seção deve apresentar uma interface de chat onde cada usuário só pode se comunicar com outro usuário que tenha algum grau de relacionamento com ele na aplicação. As mensagens devem ser transmitidas em tempo-real para usuários ativos no sistema e cada mensagem seve ser datada.

1.3 Trabalhos relacionados

A Figura 1.1 mostra alguns trabalhos de conclusão de curso da Universidade de Brasília (UnB) com a área de conhecimento em comum com este trabalho. Eles englobam projetos dos cursos de engenharia de computação, ciência da computação e engenharia de software.

O assunto deste trabalho gira em torno de aplicação Web, desenvolvimento Web e Interação Humano–Computador (IHC)/usabilidade de software e como tema acessibilidade.

É importante destacar que este trabalho tem como tema a acessibilidade, porém tem não como foco desenvolver uma aplicação Web totalmente acessível, apesar de ser possível. Ou seja, diferentemente de alguns trabalhos destacados no Quadro 1.1, esta aplicação não terá por exemplo leitores de tela para cegos ou transcrição de áudio para surdos. O objetivo principal é a usabilidade independente do usuário final ser deficiente ou não. Estas funcionalidades estão destacadas como no último capítulo deste trabalho. Já a temática ficou definida como acessibilidade porque o usuário final da aplicação foi estabelecido que seriam alunos com necessidades específicas. Esta limitação deste trabalho se dá por motivos de requisitos de projeto, no entanto, como é detalhado a seguir no referencial teórico, quanto mais acessível é um sistema, mas ele se torna usável para todos.

- **AccessIT: uma aplicação para o mapeamento de recursos de acessibilidade da UnB** [18]

  Foi desenvolvida a aplicação Web em PHP que tem como objetivo fornecer informações sobre os recursos de acessibilidade disponíveis na UnB por meio da geolocalização dos pontos. Para tanto, são apresentadas as avaliações realizadas ao término do desenvolvimento, as quais foram executadas por meio da validação do código e avaliação de usabilidade;

- **Acessibilidade Web: uma análise sobre suas ferramentas e diretrizes** [19]

  Um estudo de ferramentas para acessibilidade Web, respectivamente: um Sistema de Gerenciamento de Conteúdo (Wordpress), um *framework* (Bootstrap) e uma biblioteca (jQuery);

- **Arquitetura da informação no desenvolvimento de aplicação Web** [20]

  Um estudo de caso é desenvolvido em Oracle APEX para colocar em prática processos de desenvolvimento que incluem atividades focadas em arquitetura da informação;

- **Interface Web para sistema de gestão de redes seguindo projeto centrado em usuário** [21]

  Desenvolvimento um sistema em *React* dedicado à gerência de redes de fibras óticas e seus elementos da infraestrutura da rede gerida pela Associação GigaCandanga. Visando a melhor usabilidade do sistema, bem como uma boa experiência de usuário, o desenvolvimento deste sistema contou com a utilização de diversas práticas adotadas no mercado de desenvolvimento de software em relação ao desenvolvimento de interfaces centrado no usuário.

Quadro 1.1: Trabalhos relacionados.

	Assunto				**Tema:** acessibilidade
	Aplicação web	Desenvolvimento web	IHC / usabilidade	Acessibilidade web	
Este trabalho					
Engenharia de usabilidade e desenvolvimento web para acompanhamento de estudantes com necessidades específicas	✓	✓	✓		✓
Trabalhos relacionados					
AccessIT: uma aplicação para o mapeamento de recursos de acessibilidade da UnB	✓	✓	✓	✓	
VISUMO: uma plataforma web para criação e realização de avaliação educacional em LIBRAS	✓	✓		✓	✓
Revista Contraste : do papel ao website multilíngue e acessível	✓			✓	✓
Catálogo de práticas de acessibilidade : um apoio online voltado à acessibilidade da Web	✓		✓	✓	
Interface web para sistema de gestão de redes seguindo projeto centrado em usuário	✓		✓		
Avaliação heurística em um contexto real	✓		✓		
Direcionamento das heurísticas de Nielsen no contexto de uso de landing pages	✓		✓		
iFrame: framework para o desenvolvimento de aplicações Web	✓	✓			
Arquitetura da informação no desenvolvimento de aplicação web	✓	✓			
Método de Avaliação de Comunicabilidade da engenharia semiótica: um estudo de caso em um sistema Web			✓	✓	
Acessibilidade web: uma análise sobre suas ferramentas e diretrizes	✓			✓	
Um estudo do relacionamento entre técnicas de usabilidade e testes automatizados em métodos empíricos de desenvolvimento de software			✓		
Tuhm: uma ferramenta de apoio para testes de usabilidade			✓		
Usability heuristics for mobile applications a systematic review			✓		

- **Interface Web para sistema de gestão de redes seguindo projeto centrado em usuário** [22]

  Desenvolvimento uma plataforma Web destinada à criação e realização de avaliação educacional bilíngue (Língua Brasileira de Sinais e Língua Portuguesa). A fim de comprovar a eficácia do sistema proposto, bem como para identificar os principais problemas que podem advir da sua implementação prática, foram realizados atividades na escola, além de aplicado formulários para professores de estudantes surdos de diversas instituições.

- **iFrame : framework para o desenvolvimento de aplicações Web** [23]

  Desenvolvimento de um *framework Web* PHP que é utilizado em um estudo de caso para o desenvolvimento de uma aplicação Web, denominada "intranet", o que permitirá a coleta e análise de dados sobre seu funcionamento e eficácia;

- **Método de Avaliação de Comunicabilidade da engenharia semiótica: um estudo de caso em um sistema Web** [24]

  Realização de avaliações em um sistema real, que não fez uso de um método científico durante o seu desenvolvimento, para identificar falhas de comunicabilidade, utilizando o Método de Avaliação de Comunicabilidade (MAC) da Engenharia Semiótica;

- **Um estudo do relacionamento entre técnicas de usabilidade e testes automatizados em métodos empíricos de desenvolvimento de software** [25]

  Estudo sobre aplicabilidade de técnicas de usabilidade, além de técnicas de BDD (desenvolvimento de software dirigido por comportamento) em projetos de software, além de compreender como inserir as práticas de usabilidade estudadas no desenvolvimento empírico de software;

- **Catálogo de práticas de acessibilidade: um apoio online voltado à acessibilidade da Web** [26]

  Busca contribuir para a criação de interfaces mais acessíveis, ao oferecer um apoio online que reúne e resume aspectos relevantes da literatura especializada sobre o tema, com ênfase na acessibilidade da Web. Esse apoio, livre e de código aberto, apresenta recursos informativos sobre práticas e tecnologias relacionadas à acessibilidade, bem como provê exemplos de interfaces acessíveis e não-acessíveis;

- **Revista Contraste: do papel ao *website* multilíngue e acessível** [27]

  Discussão da importância do multilinguismo no ciberespaço e a necessidade dos recursos de acessibilidade na Web e detalhamento do processo utilizado para a in-

serção destas características no código HTML do *website*, de acordo com a Cartilha de Acessibilidade na Web e da iniciativa WAI-ARIA, ambas da W3C;

- **Avaliação heurística em um contexto real** [28]

  Apresentação de avaliação de usabilidade em um sistema de busca de documentos oficiais do Tribunal de Contas da União (Pesquisa textual);

- **Usability heuristics for mobile applications a systematic review** [29]

  Revisão sistemática da literatura, complementada por uma busca manual e em bola de neve para obtenção de heurísticas de usabilidade e avaliações heurísticas para aplicativos móveis;

- **Direcionamento das heurísticas de Nielsen no contexto de uso de landing pages** [30]

  Entendimento de como as dez heurísticas podem ser otimizadas no design de Landing Pages, fornecendo uma priorização do impacto de cada uma delas; e um direcionamento prático de como designers, desenvolvedores e gestores podem utilizá-las;

- **Tuhm: uma ferramenta de apoio para testes de usabilidade** [31]

  Foi realizada uma revisão da literatura com a finalidade de encontrar estudos relacionados às heurísticas de usabilidade e as ferramentas desenvolvidas para realizar avaliação heurística. A partir das heurísticas identificadas na literatura, foi desenvolvida uma ferramenta para auxiliar o especialista na avaliação de usabilidade.

Outros artigos relacionados, além da Universidade de Brasília (UnB), são listados abaixo:

- **Avaliação da usabilidade e acessibilidade em interação humano-computador em portais: um estudo de caso sobre três portais da transparência municipais da Paraíba.** [32]

  Este trabalho realizou uma avaliação de Acessibilidade e Usabilidade nos portais da transparência das cidades de Queimadas, Sapé e Catolé do Rocha, na Paraíba, escolhidos devido às características inerentes às interfaces dessas páginas;

- **Importância das técnicas de interação humano computador para desenvolvimento de sites** [33]

  Este trabalho visa demonstrar que o uso das técnicas de interação pode surtir um diferencial na usabilidade de *websites*, fazendo com que a navegação se torne algo prazeroso e não cansativo, o que ocorre com interfaces desorganizadas e sem critérios de ergonomia;

- **Desenvolvimento Web: a usabilidade como foco no processo de desenvolvimento** [34]

  Este trabalho se refere a garantia da qualidade no que se refere a usabilidade de software, no qual é utilizada como foco central o estudo do padrão NBR ISO/IEC 9126, que foca a qualidade do software;

- **Usabilidade e acessibilidade no design para a Web** [35]

  Este trabalho propõe o desenvolvimento de um processo de design em que o utilizador seja o ponto central e procurando deduzir e sistematizar um conjunto de boas práticas que possam ser empregadas na concepção, implementação e manutenção eficaz de projetos Web.

1.4 Organização do trabalho

O Capítulo 1 apresentou a *introdução* e descreveu a motivação do trabalho focada na acessibilidade e seu contexto baseado no desenvolvimento de uma aplicação para acompanhamento de estudantes com necessidades específicas do Departamento de Ciência da Computação (CIC) da Universidade de Brasília (UnB), ressaltando critérios de usabilidade.

O Capítulo 2 mostra o *referencial teórico* e expõe acerca das duas principais áreas de conhecimento que abrange este trabalho: Interação Humano–Computador (IHC) e aplicação Web. A primeira descreve as avaliações de usabilidade no desenvolvimento de software com explicação de diferentes critérios que são utilizados no desenvolvimento da aplicação. A segunda área, de aplicação Web, descreve os padrões de linguagens e arquitetura para desenvolvimento Web.

O Capítulo 3 exibe a lista de *tecnologias* detalhando o processo de escolha dos instrumentos utilizados na aplicação a ser desenvolvida e no processo de desenvolvimento. Esta escolha se dá por meio de critérios que facilitem a evolução e correção dos métodos de usabilidade e interação. Outras ferramentas auxiliares, protocolos, algoritmos e funções também são descritos com suas respectivas justificativas de escolha.

O Capítulo 4 mostra uma visão geral da *ferramenta Integrar*, onde o sistema desenvolvido é apresentado a partir de um momento de parada no fim do ciclo de desenvolvimento para avaliação do sistema. É feito um detalhamento de cada página da aplicação por meio de capturas de tela e dados criados para simulação de uso real do sistema.

O Capítulo 5 mostra a *avaliação de usabilidade*, onde são realizado testes, inspeções e consulta de usabilidade do sistema. Isto é efetuado segundo diferentes modos conforme citado na seção de referencial teórico. Para cada critério de avaliação, suas falhas são

listadas e justificadas. Por fim, é dado destaque em uma seção exclusiva para a avaliação heurística da aplicação.

O Capítulo 6 expõe as *conclusões* mostrando considerações finais acerca da natureza cíclica e contínua do desenvolvimento de software, onde os testes e inspeções devem ser realizados recorrentemente buscando sempre melhorar a usabilidade do sistema. Além disso, é descrito com trabalhos futuros o prosseguimento de requisitos não cumpridos e a correção de falhas detectadas no processo de avaliação.

Capítulo 2

Referencial teórico

2.1 Interação Humano–Computador (IHC)

A Interação Humano–Computador (IHC) é um campo de estudo multidisciplinar com foco no design de tecnologia de computador e, em particular, na interação entre humanos (os usuários) e computadores. Embora inicialmente preocupada com computadores, a IHC desde então se expandiu para cobrir quase todas as formas de design de tecnologia da informação [36]. IHC é o estudo de como as pessoas interagem com computadores, especialmente no que se refere ao design de tecnologia. Design centrado no usuário, UI e UX são combinados com IHC para fornecer tecnologia e produtos intuitivos.

De acordo com John Carroll,

> "...não faz mais sentido considerar a IHC como uma especialidade da ciência da computação. A IHC tornou-se mais ampla, maior e muito mais diversificada do que a própria ciência da computação. A IHC expandiu de seu foco inicial no comportamento do usuário individual e genérico para incluir computação social e organizacional, acessibilidade para idosos, deficientes físicos e cognitivos e para todas as pessoas, e para o espectro mais amplo possível de experiências e atividades humanas. Ele se expandiu de aplicativos de *desktop* para incluir jogos, aprendizado e educação, comércio, saúde e aplicativos médicos, planejamento e resposta a emergências e sistemas para apoiar a colaboração e a comunidade. Ele se expandiu das primeiras interfaces gráficas do usuário para incluir inúmeras técnicas e dispositivos de interação, interações multimodais, suporte a ferramentas para especificação de interface do usuário baseada em modelo." [36]

A Interação Humano–Computador (IHC) também é chamada de interação humano-máquina, interação homem-máquina. Aplicativos de *desktop*, navegadores de internet e computadores portáteis fazem uso das interfaces gráficas de usuário (GUI) predominantes de hoje. As interfaces de usuário de voz (VUI) são usadas para sistemas de reconhecimento e síntese de fala, e as emergentes interfaces de usuário multimodais e gráficas (GUI)

permitem que os humanos se envolvam com agentes de personagens incorporados de uma forma que não pode ser alcançada com outros paradigmas de interface.

2.1.1 Design de interação

Segundo Rogers, Sharp e Preece em *Interaction Design: Beyond Human-Computer Interaction*, design de iteração é o projeto de produtos interativos para apoiar a maneira como as pessoas se comunicam e interagem em suas vidas cotidianas e profissionais. Dito de outra forma, trata-se de criar experiências de usuário que aprimoram e aumentam a maneira como as pessoas trabalham, se comunicam e interagem [1].

Historicamente, a Interação Humano–Computador (IHC) tinha um foco estreito no design e usabilidade de sistemas de computação, enquanto o *interaction design* (design de interação) era visto como mais amplo, preocupado com a teoria, pesquisa e prática de projetar experiências de usuário para todos os tipos de tecnologias, sistemas e produtos. No entanto, hoje em dia, a IHC expandiu muito seu escopo, tanto que se sobrepõe muito mais ao design de interação [1].

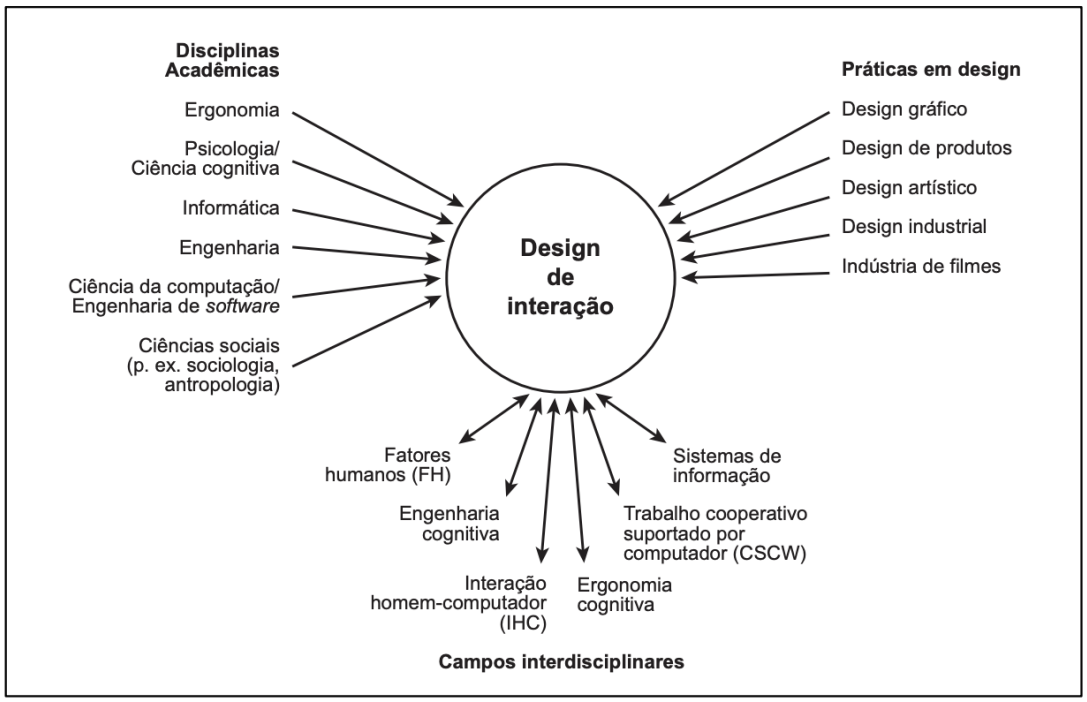

Figura 2.1: Relação entre disciplinas acadêmicas contribuintes, práticas de design e campos interdisciplinares relacionados ao design de interação (setas de duas pontas significam sobreposição) (Fonte: adaptado de [1] por [2]).

O design de interação é fundamental para muitas disciplinas, campos e abordagens que se preocupam com a pesquisa e o design de sistemas baseados em computador para

pessoas. A Figura 2.1 apresenta os principais, juntamente com os campos interdisciplinares que compreendem um ou mais deles. O design de interação é idealmente realizado por equipes multidisciplinares, onde são utilizados os conjuntos de habilidades de engenheiros, designers, programadores, psicólogos, antropólogos, sociólogos, profissionais de marketing, artistas, fabricantes de brinquedos, gerentes de produto e outros. Raramente é o caso,no entanto, que uma equipe de design teria todos esses profissionais trabalhando juntos. Quem incluir em uma equipe dependerá de vários fatores, incluindo a filosofia de design da empresa, tamanho, propósito e linha de produtos [1].

Princípios do design de interação

Os princípios de design são usados por designers de interação para auxiliar seu pensamento ao projetar para a experiência do usuário. Estas são abstrações generalizáveis destinadas a orientar os designers a pensar sobre diferentes aspectos de seus projetos [1]. Os mais conhecidos preocupam-se em determinar o que os usuários devem ver e fazer ao realizar suas tarefas usando um produto interativo:

- Visibilidade: quanto mais visíveis forem as funções, maior a probabilidade de os usuários saberem o que fazer a seguir;
- Retorno (*feedback*): envolve o envio de informações sobre qual ação foi realizada e o que foi realizado, permitindo que a pessoa continue com a atividade;
- Restrições: refere-se a determinar formas de restringir os tipos de interação do usuário que podem ocorrer em um determinado momento;
- Mapeamento: relação entre os controles e os seus efeitos no mundo;
- Consistência: design de interfaces para ter operações semelhantes e usar elementos semelhantes para realizar tarefas semelhantes;
- Fornecimento (*affordance*): termo usado para se referir a um atributo de um objeto que permite que as pessoas saibam como usá-lo.

Metas do design de interação

As metas do design de interação são classificadas de acordo com o foco na usabilidade e na experiência de usuário: as metas de usabilidade se preocupam em atender a critérios específicos de usabilidade, como eficiência, enquanto as metas de experiência do usuário se preocupam em explicar a natureza da experiência do usuário, por exemplo, para ser esteticamente agradável [1]. A Figura 2.2 apresenta internamente ao círculo as metas de usabilidade e no círculo externo as metas decorrentes da experiência do usuário.

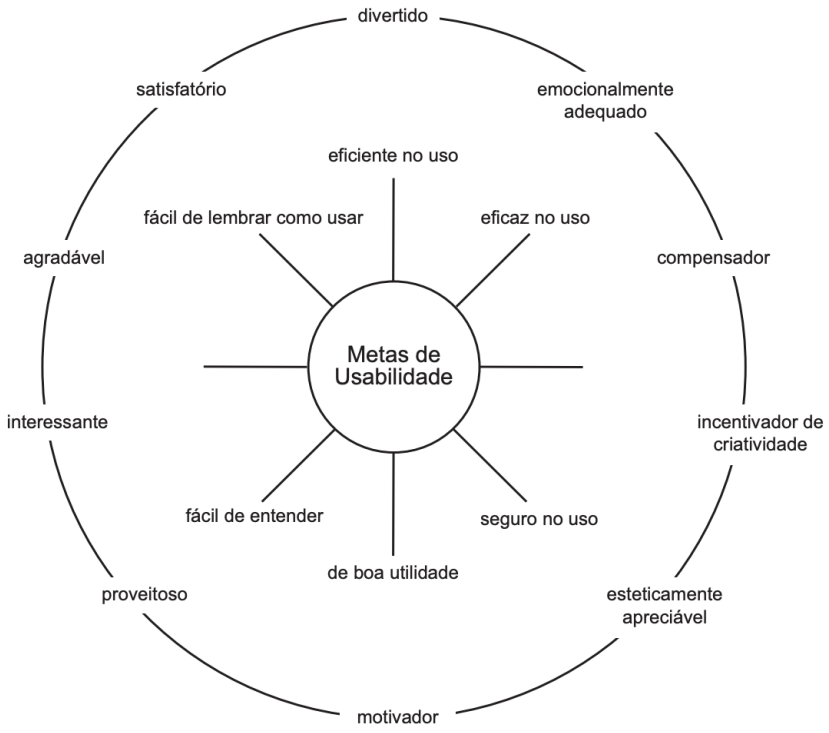

Figura 2.2: Metas de usabilidade e experiência do usuário (Fonte: adaptado de [1] por [2]).

Usabilidade refere-se a garantir que os produtos interativos sejam fáceis de aprender, eficazes de usar e agradáveis do ponto de vista do usuário. Trata-se de otimizar as interações das pessoas com produtos interativos para que possam realizar suas atividades no trabalho, na escola e no dia a dia. Mais especificamente, a usabilidade é dividida nas seguintes seis metas [1]:

- Eficácia: é um objetivo geral e refere-se a quão bom um produto é em fazer o que deve fazer;
- Eficiência: refere-se à forma como um produto oferece suporte aos usuários na realização de suas tarefas;
- Segurança: envolve proteger o usuário de condições perigosas e situações indesejáveis;
- Utilidade: refere-se à medida em que o produto fornece o tipo certo de funcionalidade para que os usuários possam fazer o que precisam ou desejam fazer;
- Entendimento: refere-se à facilidade com que um sistema é aprendido a ser usado;

- Memorabilidade: refere-se à facilidade com que um produto é lembrado como usá-lo, uma vez aprendido.

Uma diversidade de metas de experiência do usuário foi articulada no design de interação, que abrange uma gama de emoções e experiências sentidas. Estas metas incluem aspectos desejáveis (ilustrados no círculo externo da Figura 2.2) e indesejáveis:

- Aspectos desejáveis: satisfatório, divertido, agradável, emocionalmente agradável, compensador, incentivador de criatividade, esteticamente apreciável, motivador, proveitoso, interessante, etc;
- Aspectos indesejáveis: tedioso, desagradável, frustrante, arrogante, que faz se sentir culpado, que faz se sentir estúpido, chato, bonitinho, infantil, enigmático, etc.

Muitas dessas metas são qualidades subjetivas e estão relacionadas com a sensação de um sistema para o usuário. O processo de seleção de termos que melhor transmitem os sentimentos, estado de ser, emoções, sensações e assim por diante de um usuário ao usar ou interagir com um produto em um determinado momento e local pode ajudar os designers a entender a natureza multifacetada e mutável da experiência do usuário [1].

2.1.2 Experiência do usuário

De acordo com Yablonski em *Leis da Psicologia Aplicadas a UX: Usando psicologia para projetar produtos e serviços melhores*, existem 20 regras que todos os designers e desenvolvedores deveriam consultar e aplicar em seus projetos. A interseção da psicologia e do design de UX tem se tornado um tópico cada vez mais relevante em uma época em que as funções do design têm um impacto cada vez mais forte nas organizações. Junto com um foco crescente no design, houve um aumento no debate em torno de quais habilidades adicionais os designers devem aprender, se houver, para aumentar seu valor e contribuição [37].

Os seres humanos têm um "diagrama" subjacente de como percebem e processam o mundo ao seu redor, e o estudo da psicologia ajuda a decifrar esse diagrama. Os designers podem usar esse conhecimento para criar produtos e experiências mais intuitivos e centrados no ser humano, em vez de forçar os usuários a se adaptarem ao design de um produto ou experiência. Alguns princípios chave da psicologia podem ser usados como um guia para projetar de uma maneira adaptada às pessoas [37]. Abaixo está uma listagem com um detalhamento resumido das 20 regras citadas no livro:

1. Efeito da usabilidade estética: os usuários geralmente percebem o design esteticamente agradável como um design mais utilizável;

2. Limiar de Doherty: a produtividade aumenta quando um computador e os usuários interagem em um ritmo (<400ms) que garante que nenhum deles precise esperar pelo outro;
3. Lei de Fitts: o tempo para interagir com um alvo é uma função da distância e tamanho do alvo;
4. Lei de Hick: O tempo necessário para tomar uma decisão aumenta com o número e a complexidade das escolhas;
5. Lei de Jakob Nielsen: os usuários passam a maior parte do tempo em outros sites. Isso significa que os usuários preferem que seu site funcione da mesma maneira que todos os outros sites que eles já conhecem;
6. Lei da Região Comum: os elementos tendem a ser percebidos em grupos se eles estiverem compartilhando uma área com um limite claramente definido;
7. Lei de Prägnanz: as pessoas perceberão e interpretarão imagens ambíguas ou complexas como a forma mais simples possível, porque é a interpretação que requer o menor esforço cognitivo de nós;
8. Lei da Proximidade: objetos próximos ou próximos um do outro tendem a ser agrupados;
9. Lei da Similaridade: o olho humano tende a perceber elementos semelhantes em um design como uma imagem, forma ou grupo completo, mesmo que esses elementos estejam separados;
10. Lei da Conexão Uniforme: elementos visualmente conectados são percebidos como mais relacionados do que elementos sem conexão;
11. Lei de Miller: a pessoa média só pode manter 7 (mais ou menos 2) itens em sua memória de trabalho;
12. Navalha de Occam: entre as hipóteses concorrentes que preveem igualmente bem, deve-se selecionar aquela com o menor número de suposições;
13. Princípio de Pareto: afirma que, para muitos eventos, aproximadamente 80% dos efeitos vêm de 20% das causas;
14. Lei de Parkinson: qualquer tarefa aumentará até que todo o tempo disponível seja gasto;
15. Regra do Ponto Final: as pessoas julgam uma experiência amplamente baseadas em como se sentiram no auge e no final, em vez da soma total ou média de cada momento da experiência;

16. Lei de Postel: seja liberal no que você aceita e conservador no que você envia;
17. Efeito de Posição Serial: os usuários têm uma propensão a se lembrar melhor do primeiro e do último item de uma série;
18. Lei de Tesler: também conhecida como Lei da Conservação da Complexidade, afirma que, para qualquer sistema, existe uma certa quantidade de complexidade que não pode ser reduzida;
19. Efeito Von Restorff: também conhecido como Efeito Isolamento, prevê que, quando vários objetos semelhantes estiverem presentes, é mais provável que aquele que difere do resto seja lembrado;
20. Efeito Zeigarnik: as pessoas se lembram melhor de tarefas incompletas ou interrompidas do que de tarefas concluídas.

2.1.3 Avaliação de usabilidade

Os métodos de avaliação de usabilidade são usados para aumentar a usabilidade avaliando a interação humana com o sistema. As interfaces do sistema são avaliadas em diferentes etapas para avaliar sua qualidade. Essas técnicas de avaliação de usabilidade podem ser aplicadas em diferentes níveis do ciclo de vida de desenvolvimento de software para alcançar os melhores resultados de usabilidade. Métodos diferentes revelam problemas diferentes em interfaces quando aplicados de maneiras diferentes e para um conjunto diferente de requisitos [38].

O Quadro 2.1 mostra diferentes classes dos métodos de avaliação de usabilidade. Muitos outros métodos são usados para avaliação de usabilidade, mas os métodos listados são os mais comuns. As três classes são consideradas apropriadas para avaliação somativa e formativa de usabilidade. Essas classes de métodos de avaliação de usabilidade têm sido utilizadas principalmente no campo da engenharia de software. Além disso o Quadro 2.2 apresenta várias características de cada um desses métodos de forma detalhada com relação ao tamanho recomendado da equipe necessária para avaliação e as questões de usabilidade abordadas [38].

A. Teste de usabilidade *(usability testing)*

Na abordagem de Teste de Usabilidade, os usuários representativos trabalham em tarefas típicas usando o sistema (ou o protótipo) e os avaliadores usam os resultados para ver como a interface do usuário suporta os usuários para realizar suas tarefas [39]. Os métodos de teste incluem o seguinte:

Quadro 2.1: Classes dos métodos de avaliação de usabilidade (Fonte: adaptado de [38] e [39]).

A. Teste de usabilidade (*usability testing*)	**B. Inspeção de usabilidade** (*usability inspection*)	**C. Consulta de usabilidade** (*usability inquiry*)
Método de coaching (*Coaching Method*)		
Aprendizagem de co-descoberta (Co-discovery Learning)		
Medição de desempenho (*Performance Measurement*)	Passo a passo cognitivo (*Cognitive Walkthroughs*)	Observação de campo (*Field Observation*)
Protocolo de perguntas (*Question-asking Protocol*)	Inspeção de recursos (*Feature Inspection*)	Grupos de foco (*Focus Groups*)
Teste remoto (*Remote Testing*)	Avaliação heurística (*Heuristic Evaluation*)	Entrevistas (*Interviews*)
Teste retrospectivo (*Retrospective Testing*)	Passo a passo pluralístico (*Pluralistic Walkthroughs*)	Registro de uso real (*Logging Actual Use*)
Método de sombreamento (*Shadowing Method*)	Inspeção baseada em perspectiva (*Perspective-based Inspection*)	Estudo de campo proativo (*Proactive Field Study*)
Método de ensino (*Teaching Method*)		Questionários (*Questionnaires*)
Protocolo pensando em voz alta (*Thinking Aloud Protocol*)		

- Método de coaching (*Coaching Method*): nesta técnica os participantes podem fazer quaisquer perguntas relacionadas ao sistema a um especialista que responderá da melhor maneira possível. O objetivo desta técnica é descobrir as necessidades de informação dos usuários, a fim de fornecer melhor treinamento e documentação, bem como possivelmente redesenhar a interface para evitar a necessidade de perguntas. Quando um usuário especialista é usado como treinador, o modelo mental do usuário especialista do sistema também pode ser analisado pelo testador [40];
- Aprendizagem de co-descoberta (*Co-discovery Learning*): Durante um teste de usabilidade, dois usuários de teste tentam executar tarefas juntos enquanto são observados. Eles devem ajudar uns aos outros da mesma maneira que fariam se estivessem trabalhando juntos para atingir um objetivo comum usando o produto. Eles são encorajados a explicar o que estão pensando enquanto trabalham nas tarefas. Em comparação com o protocolo de pensamento em voz alta, essa técnica torna mais natural para os usuários do teste verbalizarem seus pensamentos durante o teste [40][41][42];
- Medição de desempenho (*Performance Measurement*): esta técnica é utilizada para obter dados quantitativos sobre o desempenho dos participantes do teste quando

Quadro 2.2: Características dos métodos de avaliação de usabilidade (Fonte: adaptado de [39]).

Características	Equipe necessária para a avaliação			Questões de usabilidade abordadas				
	Especialistas em usabilidade	Desenvolvedores de software	Usuários	Eficácia: fazer as coisas certas	Eficiência: fazer certo as coisas	Satisfação	Pode ser realizado remotamente	Pode obter dados quantitativos
A. Teste de usabilidade (*usability testing*)								
Método de coaching (*Coaching Method*)	1	0	4	✓	✗	✓	✗	✗
Aprendizagem de co-descoberta (Co-discovery Learning)	1	0	6	✓	✗	✓	✗	✗
Medição de desempenho (*Performance Measurement*)	1	0	6	✓	✗	✓	✗	✓
Protocolo de perguntas (*Question-asking Protocol*)	1	0	4	✓	✗	✓	✗	✗
Teste remoto (*Remote Testing*)	1	0	5	✓	✓	✓	✓	✓
Teste retrospectivo (*Retrospective Testing*)	1	0	4	✓	✓	✓	✗	✓
Método de sombreamento (*Shadowing Method*)	1	0	4	✓	✓	✗	✗	✓
Método de ensino (*Teaching Method*)	1	0	4	✓	✗	✓	✗	✗
Protocolo pensando em voz alta (*Thinking Aloud Protocol*)	1	0	4	✓	✗	✓	✗	✗
B. Inspeção de usabilidade (*usability inspection*)								
Passo a passo cognitivo (*Cognitive Walkthroughs*)	1-4	0-2	0	✓	✗	✗	✗	✗
Inspeção de recursos (*Feature Inspection*)	1	0	0	✓	✗	✗	✓	✗
Avaliação heurística (*Heuristic Evaluation*)	4	0	0	✓	✓	✗	✓	✗
Passo a passo pluralístico (*Pluralistic Walkthroughs*)	1	1	2	✓	✗	✓	✗	✗
Inspeção baseada em perspectiva (*Perspective-based Inspection*)	1	0	0	✓	✓	✓	✗	✗
C. Consulta de usabilidade (*usability inquiry*)								
Observação de campo (*Field Observation*)	1	0	2	✓	✗	✓	✗	✗
Grupos de foco (*Focus Groups*)	1	0	6	✓	✗	✓	✗	✗
Entrevistas (*Interviews*)	1	0	2	✓	✗	✓	✗	✗
Registro de uso real (*Logging Actual Use*)	1	0	6	✓	✓	✗	✓	✗
Estudo de campo proativo (*Proactive Field Study*)	1	0	2	✗	✗	✗	✗	✗
Questionários (*Questionnaires*)	1	0	6	✓	✗	✓	✓	✗

eles executam as tarefas durante o teste de usabilidade. Isso geralmente proibirá uma interação entre o participante e o testador durante o teste que afetará os dados quantitativos de desempenho. Deve ser conduzido em um laboratório de usabilidade formal para que os dados possam ser coletados com precisão e possíveis interferências inesperadas sejam minimizadas. Os dados quantitativos são mais úteis para fazer testes comparativos ou testes em relação a referências predefinidas;

O procedimento consiste em definir metas para o teste de usabilidade em termos de atributo de usabilidade (por exemplo, fácil de aprender, eficiente de usar, fácil de lembrar, poucos erros, subjetivamente agradável). Em seguida, equilibrar os vários componentes das metas, decidir sobre sua importância relativa e quantificar esses problemas de usabilidade por diferentes medidas. E, finalmente, realizar o teste evitando interrupções inesperadas e, sempre que possível, deve ser gravado em vídeo para apoiar a coleta de dados após o teste [40][43];

- Protocolo de perguntas (*Question-asking Protocol*): durante um teste de usabilidade, além de permitir que os usuários de teste verbalizem seus pensamentos como no protocolo de pensamento em voz alta, os testadores os estimulam fazendo perguntas diretas sobre o produto, a fim de entender seu modelo mental do sistema e das tarefas, e onde eles têm dificuldade em entender e usar o sistema. Esta é uma maneira mais natural do que o método de pensar em voz alta, permitindo que o usuário do teste verbalize seus pensamentos;

- Teste remoto (*Remote Testing*): é usado quando o testador está separado no espaço e/ou tempo dos participantes. Isso significa que o testador não pode observar o processo de teste diretamente e que os participantes geralmente não estão em um laboratório formal de usabilidade. A sessão de teste do usuário pode ser guiada e registrada por meio de um software especial, bem como código adicional adicionado ao sistema que está sendo testado [44];

- Teste retrospectivo (*Retrospective Testing*): consiste em gravar um vídeo de uma sessão de teste de usabilidade e, em seguida, o testador pode coletar mais informações revisando o vídeo junto com os usuários participantes e fazendo perguntas sobre seu comportamento durante o teste. Esta técnica deve ser utilizada em conjunto com outras técnicas, principalmente aquelas em que a interação entre os testadores e os participantes é restrita. Mas usar essa técnica significa que cada teste leva pelo menos o dobro do tempo. Outro requisito óbvio para o uso dessa técnica é que a interação do usuário com o computador precisa ser gravada e reproduzida [40];

- Método de sombreamento (*Shadowing Method*): durante um teste de usabilidade, o testador tem um usuário especialista (no domínio da tarefa) sentado próximo a ele e

explicando o comportamento do usuário de teste ao testador. Essa técnica é usada quando não é apropriado que o usuário do teste pense em voz alta ou fale com o testador enquanto trabalha nas tarefas;

- Método de ensino (*Teaching Method*): durante um teste de usabilidade, deixe os usuários de teste interagirem com o sistema primeiro, para que eles se familiarizem com ele e adquiram alguma experiência na realização de tarefas usando o sistema. Em seguida, apresente um usuário ingênuo para cada usuário de teste. Os usuários novatos são instruídos pelo testador a limitar sua participação ativa e não se tornar um solucionador de problemas ativo. Cada usuário de teste é solicitado a explicar ao iniciante como o sistema funciona e demonstrar a ele/ela um conjunto de tarefas pré-determinadas [45][46];
- Protocolo pensando em voz alta (*Thinking Aloud Protocol*): durante um teste de usabilidade, deixe os usuários de teste interagirem com o sistema primeiro, para que eles se familiarizem com ele e adquiram alguma experiência na realização de tarefas usando o sistema. Em seguida, apresente um usuário ingênuo para cada usuário de teste. Os usuários novatos são instruídos pelo testador a limitar sua participação ativa e não se tornar um solucionador de problemas ativo. Durante o curso de um teste de usabilidade, os usuários de teste são solicitados a verbalizar seus pensamentos, sentimentos e opiniões enquanto interagem com o sistema [40].

B. Inspeção de usabilidade *(usability inspection)*

As avaliações que ocorrem sem envolver os usuários são conduzidas em ambientes onde o pesquisador tem que imaginar ou modelar como uma interface provavelmente é usada. Os métodos de inspeção são comumente empregados para prever o comportamento do usuário e identificar problemas de usabilidade com base no conhecimento da usabilidade, comportamento dos usuários, contextos nos quais o sistema é usado e tipos de atividades que os usuários realizam. Os exemplos incluem avaliação heurística que aplica o conhecimento de usuários típicos guiados por regras práticas e orientações que envolvem percorrer um cenário ou responder a um conjunto de perguntas para um protótipo detalhado. Outras técnicas incluem análises e modelos [1].

Na abordagem de Inspeção de Usabilidade, especialistas em usabilidade, e às vezes desenvolvedores de software, usuários e outros profissionais, examinam aspectos relacionados à usabilidade de uma interface de usuário [39]. Os métodos de inspeção comumente usados são:

- Passo a passo cognitivo (*Cognitive Walkthroughs*): envolve um ou um grupo de avaliadores inspecionando uma interface de usuário passando por um conjunto de

tarefas e avaliando sua compreensibilidade e facilidade de aprendizado. A interface do usuário geralmente é apresentada na forma de uma maquete em papel ou um protótipo funcional, mas também pode ser uma interface totalmente desenvolvida. A entrada para o passo a passo também inclui o perfil do usuário, especialmente o conhecimento do usuário sobre o domínio da tarefa e a interface, e os casos da tarefa. Os avaliadores podem incluir engenheiros de fatores humanos, desenvolvedores de software ou pessoas de marketing, documentação etc. Essa técnica é melhor usada no estágio de design do desenvolvimento. Mas também pode ser aplicado durante os estágios de código, teste e implantação [47][48];

- Inspeção de recursos (*Feature Inspection*): Essa técnica de inspeção se concentra no conjunto de recursos de um produto. Os inspetores geralmente recebem casos de uso com o resultado final a ser obtido com o uso do produto. Cada recurso é analisado por sua disponibilidade, compreensão e outros aspectos de usabilidade. Por exemplo, um cenário de usuário comum para o uso de um processador de texto é produzir uma carta. Os recursos que seriam usados incluem inserir texto, formatar texto, verificação ortográfica, salvar o texto em um arquivo e imprimir a carta. Cada conjunto de recursos usados para produzir a saída necessária (uma carta) é analisado quanto à sua disponibilidade, compreensão e utilidade geral;

- Avaliação heurística (*Heuristic Evaluation*): uma heurística é uma diretriz ou princípio geral ou regra prática que pode guiar uma decisão de design ou ser usada para criticar uma decisão que já foi tomada. A avaliação heurística, desenvolvida por Jakob Nielsen e Rolf Molich, é um método para estruturar a crítica de um sistema usando um conjunto de heurísticas relativamente simples e gerais.

  A ideia geral por trás da avaliação heurística é que vários avaliadores avaliam independentemente um sistema para encontrar possíveis problemas de usabilidade. É importante que haja vários desses avaliadores e que as avaliações sejam feitas de forma independente. A experiência da Nielsen indica que cerca de 5 avaliadores geralmente resultam na descoberta de cerca de 75% dos problemas gerais de usabilidade.

  A avaliação heurística é melhor usada como uma técnica de avaliação de tempo de design, porque é mais fácil corrigir muitos dos problemas de usabilidade que surgem. Mas tudo o que é realmente necessário para fazer a avaliação é algum tipo de artefato que descreva o sistema, e isso pode variar de um conjunto de *storyboards* dando uma visão geral rápida do sistema até um sistema totalmente funcional que está em uso no campo [48];

- Passo a passo pluralístico (*Pluralistic Walkthroughs*): na fase de projeto, quando o protótipo em papel está disponível, um grupo de usuários, desenvolvedores e engenheiros de fatores humanos se reúnem para passar por um conjunto de tarefas, discutindo e avaliando a usabilidade de um sistema. As orientações em grupo têm a vantagem de fornecer uma gama diversificada de habilidades e perspectivas para lidar com problemas de usabilidade. Como em qualquer inspeção, quanto mais pessoas procurarem por problemas, maior a probabilidade de encontrar problemas. Além disso, a interação entre a equipe durante o passo a passo ajuda a resolver problemas de usabilidade mais rapidamente [48];
- Inspeção baseada em perspectiva (*Perspective-based Inspection*): as inspeções baseadas em perspectiva são úteis para ampliar a capacidade de encontrar problemas de avaliadores experientes e inexperientes. Por exemplo, se a equipe é composta principalmente por jovens, pode-se pedir a um ou mais membros da equipe que adotem a perspectiva de uma pessoa idosa. Adotar uma perspectiva pode envolver ler sobre problemas com usuários mais velhos, simular os efeitos do envelhecimento usando óculos que embaçam a visão ou passar um tempo com alguns usuários mais velhos antes da inspeção para entender alguns dos problemas que eles podem enfrentar [49].

B.1 Avaliação heurística

Entre os métodos de inspeção, a avaliação heurística recebe um maior enfoque neste trabalho, pois seus critérios objetivos tornam rápida a análise do software durante os ciclos de desenvolvimento nos pontos de pausa para avaliação do sistema. Por isso, em *Resultados* no Capítulo 4, uma seção exclusiva foi dedicada a este tipo de avaliação da aplicação desenvolvida. Entre os métodos heurísticos difundidos e disponibilizados abertamente as heurísticas de Jakob Nielsen são provavelmente as heurísticas de usabilidade mais usadas para o design da interface do usuário [50][51]. Por este motivo, e por questões de simplificação, apenas elas são listadas e detalhadas neste trabalho.

Na avaliação heurística, os pesquisadores, guiados por um conjunto de princípios de usabilidade conhecidos como heurísticas, avaliam se os elementos da interface do usuário, como caixas de diálogo, menus, estrutura de navegação e assim por diante, estão em conformidade com os princípios testados e comprovados. Essas heurísticas se assemelham muito aos princípios de design de alto nível (como tornar os designs consistentes, reduzir a carga de memória e usar termos que os usuários entendam). A avaliação heurística foi desenvolvida por Jakob Nielsen e posteriormente modificada por outros pesquisadores para avaliar a Web e outros tipos de sistemas. Além disso, muitos pesquisadores e profissionais converteram diretrizes de design em heurísticas que são então aplicadas na avaliação heu-

rística. O conjunto original de heurísticas para avaliação de interação humano-computador foi derivado empiricamente da análise de 249 problemas de usabilidade; segue uma versão revisada dessas heurísticas [1]:

Figura 2.3: Heurísticas de Nielsen (Fonte: [52]).

1. Visibilidade do status do sistema: o sistema deve sempre manter os usuários informados sobre o que está acontecendo, por meio de *feedback* apropriado e dentro de um prazo razoável;
2. Correspondência entre o sistema e o mundo real: o sistema deve falar a linguagem do usuário, com palavras, frases e conceitos familiares ao usuário, em vez de termos orientados ao sistema. Deve seguir as convenções do mundo real, fazendo com que as informações apareçam em uma ordem natural e lógica;
3. Controle e liberdade do usuário: os usuários geralmente escolhem as funções do sistema por engano e precisarão de uma saída de emergência claramente marcada para sair do estado indesejado sem ter que passar por um diálogo extenso. O sistema deve suportar desfazer e refazer;
4. Consistência e padronização: os usuários não devem se perguntar se diferentes palavras, situações ou ações significam a mesma coisa. O sistema deve seguir as convenções da plataforma;

5. Prevenção de erros: em vez de apenas boas mensagens de erro, o sistema deve incorporar um design cuidadoso que evite a ocorrência de um problema em primeiro lugar. Elimine as condições propensas a erros ou verifique-as e apresente aos usuários uma opção de confirmação antes que eles se comprometam com a ação;

6. Reconhecimento em vez de recordação: minimize a carga de memória do usuário tornando objetos, ações e opções visíveis. O usuário não deve ter que se lembrar de informações de uma parte da caixa de diálogo para outra. As instruções de uso do sistema devem estar visíveis ou facilmente recuperáveis sempre que apropriado.

7. Eficiência e flexibilidade de uso: aceleradores, invisíveis para o usuário novato, podem muitas vezes acelerar a interação para o usuário experiente de forma que o sistema possa atender usuários inexperientes e experientes. Permita que os usuários personalizem ações frequentes;

8. Estética e design minimalista: os diálogos não devem conter informações irrelevantes ou raramente necessárias. Cada unidade extra de informação em um diálogo compete com as unidades relevantes de informação e diminui sua visibilidade relativa;

9. Ajude os usuários a reconhecer, diagnosticar e recuperar-se de erros: as mensagens de erro devem ser expressas em linguagem simples (não códigos), indicar com precisão o problema e sugerir uma solução de forma construtiva;

10. Ajuda e documentação: embora seja melhor que o sistema possa ser usado sem documentação, pode ser necessário fornecer ajuda e documentação. Qualquer informação desse tipo deve ser fácil de pesquisar, focada na tarefa do usuário, listar etapas concretas a serem executadas e não ser muito grande.

A Figura 2.3 representa de forma didática e visual cada uma das 10 heurísticas listadas acima.

C. Consulta de usabilidade *(usability inquiry)*

Aqui, os avaliadores de usabilidade obtêm informações sobre o que os usuários gostam, não gostam, precisam e entendem o sistema conversando com eles, observando-os usando o sistema no trabalho real (não para fins de teste de usabilidade) ou deixando-os responder a perguntas verbalmente ou em forma escrita [39]. Como seus métodos incluem se envolver no meio físico real e ter um relacionamento pessoal com os usuários, eles costumam estar presentes no momento inicial e nos momentos finais do processo de desenvolvimento de software. Os métodos de consulta incluem:

- Observação de campo (*Field Observation*): os engenheiros de fatores humanos vão ao local de trabalho dos usuários representativos e os observam trabalhar, para entender como os usuários estão usando o sistema para realizar suas tarefas e que tipo de modelo mental os usuários têm sobre o sistema. Este método pode ser utilizado nas fases de teste e implantação do desenvolvimento do produto [40];
- Grupos de foco (*Focus Groups*): esta é uma técnica de coleta de dados onde cerca de 6 a 9 usuários são reunidos para discutir questões relacionadas ao sistema. Um engenheiro de fatores humanos desempenha o papel de moderador, que precisa preparar previamente a lista de assuntos a serem discutidos e buscar colher as informações necessárias da discussão. Isso pode capturar reações e ideias espontâneas do usuário que evoluem no processo de grupo dinâmico [40];
- Entrevistas (*Interviews*): nesta técnica, os engenheiros de fatores humanos formulam perguntas sobre o produto com base no tipo de questões de interesse. Em seguida, eles entrevistam usuários representativos para fazer essas perguntas a fim de coletar as informações desejadas. É bom para obter informações detalhadas, bem como informações que só podem ser obtidas a partir do processo interativo entre o entrevistador e o usuário. Em uma entrevista de avaliação, um entrevistador lê as perguntas para o usuário, o usuário responde verbalmente e o entrevistador registra essas respostas [40];
- Registro de uso real (*Logging Actual Use*): o registro envolve fazer com que o computador colete automaticamente estatísticas sobre o uso detalhado do sistema. É útil porque mostra como os usuários executam seu trabalho real e porque é fácil coletar dados automaticamente de um grande número de usuários trabalhando em diferentes circunstâncias. Normalmente, um log de interface conterá estatísticas sobre a frequência com que cada usuário usou cada recurso do programa e a frequência com que vários eventos de interesse (como mensagens de erro) ocorreram. As estatísticas que mostram a frequência de uso de comandos e outros recursos do sistema podem ser usadas para otimizar os recursos usados com frequência e para identificar os recursos que raramente são usados ou não usados. As estatísticas que mostram a frequência de várias situações de erro e o uso da ajuda on-line podem ser usadas para melhorar a usabilidade de versões futuras do sistema, reprojetando os recursos que causam mais erros e mais acesso à ajuda on-line. Essa técnica pode ser usada nos estágios de teste ou implantação do desenvolvimento de software [40];
- Estudo de campo proativo (*Proactive Field Study*): antes de projetar um sistema, a fim de entender os usuários, suas tarefas e seu ambiente de trabalho, os engenheiros de fatores humanos vão ao local de trabalho dos usuários representativos e conversam

com eles, observam-nos trabalhar e fazem perguntas, para entender as características do usuário, o fluxo de trabalho, os recursos do sistema de que precisam, etc. Essa técnica deve ser usada durante o requisito ou estágio inicial do desenvolvimento de software. Este deve ser o primeiro passo do trabalho de usabilidade para um projeto [40];

- Questionários (*Questionnaires*): são uma técnica bem estabelecida para coletar dados demográficos e opiniões dos usuários. Eles são semelhantes às entrevistas, pois podem ter perguntas fechadas ou abertas, mas uma vez que o questionário é produzido, ele pode ser distribuído a um grande número de participantes sem a necessidade de recursos adicionais de coleta de dados. Assim, mais dados podem ser coletados do que normalmente seria possível em um estudo de entrevista. Além disso, os participantes que estão localizados em locais remotos ou aqueles que não podem comparecer a uma entrevista em um determinado momento podem ser envolvidos com mais facilidade. Frequentemente, uma mensagem é enviada eletronicamente aos participantes em potencial, direcionando-os para um questionário online [1].

2.1.4 Engenharia de usabilidade

O processo de anotar critérios de usabilidade formais, verificáveis, e portanto mensuráveis, é uma característica chave de uma abordagem de design de interação chamada engenharia de usabilidade. Ela envolve a especificação de medidas quantificáveis de desempenho do produto, documentando-as em uma especificação de usabilidade e avaliando o produto em relação a elas [1].

A engenharia de usabilidade é uma abordagem na qual medidas específicas para os objetivos de usabilidade do produto são acordadas no início do processo de desenvolvimento e são usadas para rastrear o progresso à medida que o desenvolvimento avança. Isso garante que a usabilidade receba a devida prioridade e facilita o acompanhamento do progresso. O mesmo vale para as metas de experiência do usuário. Embora seja mais difícil identificar medidas quantificáveis que rastreiam essas qualidades, é necessário um entendimento de sua importância durante a atividade de requisitos [1].

Ciclo de vida de desenvolvimento de software

Um ciclo de vida de desenvolvimento de software, ou do inglês *software development life cycle (SDLC)*, é composto de fases de trabalho distintas que são usadas por engenheiros de sistemas e desenvolvedores de sistemas para entregar sistemas de informação. Como qualquer coisa que é fabricada em uma linha de montagem, um SDLC visa produzir sistemas de alta qualidade que atendam ou excedam as expectativas, com base nos re-

quisitos, entregando sistemas dentro de prazos programados e estimativas de custo. Os sistemas de computador são complexos e frequentemente vinculam componentes com origens variadas. O conceito SDLC se aplica a uma variedade de configurações de hardware e software, pois um sistema pode ser composto apenas de hardware, apenas software ou uma combinação de ambos. Geralmente há cinco estágios neste ciclo, mas que podem váriar: requisitos (*requirement*), *design*, desenvolvimento (*code*), teste (*test*), implantação (*deployment*) [53].

2.1.5 Design centrado no ser humano

Humane by Design [54] é um recurso que fornece orientação para projetar produtos e serviços digitais eticamente humanos por meio de padrões focados no bem-estar do usuário. Segundo Jon Yablonski, "em vez de a tecnologia aprimorar nossas habilidades como humanos, nós a vimos se tornar um veículo para atrair nossa atenção, monetizar nossas informações pessoais e explorar nossas vulnerabilidades psicológicas. Como designers, desempenhamos um papel fundamental na criação de tal tecnologia, e é hora de assumirmos a responsabilidade pelo impacto que esses produtos e serviços que construímos estão tendo nas pessoas que deveriam servir".

Jon Yablonski estabeleceu sete princípios em *Humane by Design* [54] para projetar produtos e serviços digitais eticamente humanos por meio de padrões focados no bem-estar do usuário: empoderamento, finitude, inclusão, resiliência, respeito, atenção e transparência.

- Resiliência: o design resiliente se concentra no bem-estar dos mais vulneráveis e antecipa o potencial de abuso;
- Capacitação: o design capacitador garante que os produtos se concentrem no valor que fornecem às pessoas em detrimento da receita que podem gerar;
- Finitude: o design finito maximiza a qualidade geral do tempo gasto, limitando a experiência e priorizando o conteúdo significativo e relevante;
- Inclusão: o design inclusivo é uma metodologia que se baseia em toda a diversidade humana;
- Intenção: o design intencional usa redutores de velocidade para evitar abusos, protege a privacidade, orienta as pessoas para hábitos digitais mais saudáveis e considera as consequências de longo prazo em detrimento do ganho de curto prazo;
- Respeito: o design respeitoso prioriza o tempo, a atenção e o bem-estar digital geral das pessoas;

- Transparência: o design transparente é claro sobre as intenções, honesto nas ações e livre de padrões obscuros.

Segundo Rachel Powers [17],

> "o design inclusivo é onde o design encontra a diversidade. É onde acessibilidade e usabilidade se unem, usando uma lente centrada no ser humano para construir a inclusão. A acessibilidade traz a tecnologia para pessoas com diversas habilidades sem modificações especiais. Práticas como reduzir ações repetitivas, diminuir a frustração do usuário e fornecer bons recursos e restrições ajudam a permitir que uma tecnologia seja muito mais operável por todos. O objetivo é mantê-lo simples, removendo a complexidade desnecessária para que todos possam usá-lo e entendê-lo, independentemente dos níveis de alfabetização técnica ou experiência [17]."

conforme ilustrado na Figura 2.4.

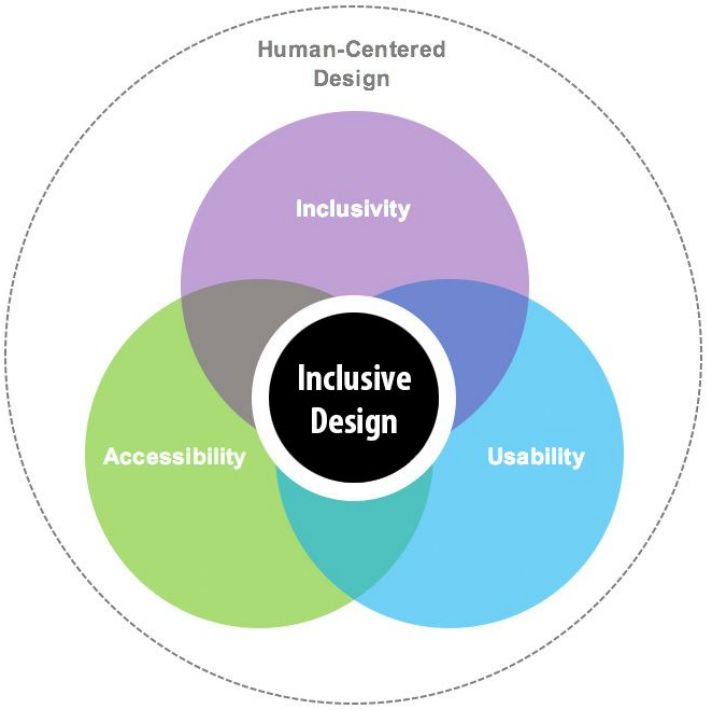

Figura 2.4: Design centrado no ser humano (Fonte: [17]).

Design universal ou inclusivo

O design inclusivo (ou universal) [55] é uma metodologia que se baseia em toda a diversidade humana. De acordo com Lillian Xiao, "O design inclusivo é vantajoso para clientes e empresas. Amplia o alcance do seu produto, estimula a inovação e ajuda a sua empresa a

assumir uma postura de responsabilidade social" [56]. Segundo o *Centre for Excellence in Universal Design*, "design universal é o design e a composição de um ambiente de modo que possa ser acessado, compreendido e utilizado na maior extensão possível por todas as pessoas, independentemente de sua idade, tamanho, habilidade ou deficiência" [57].

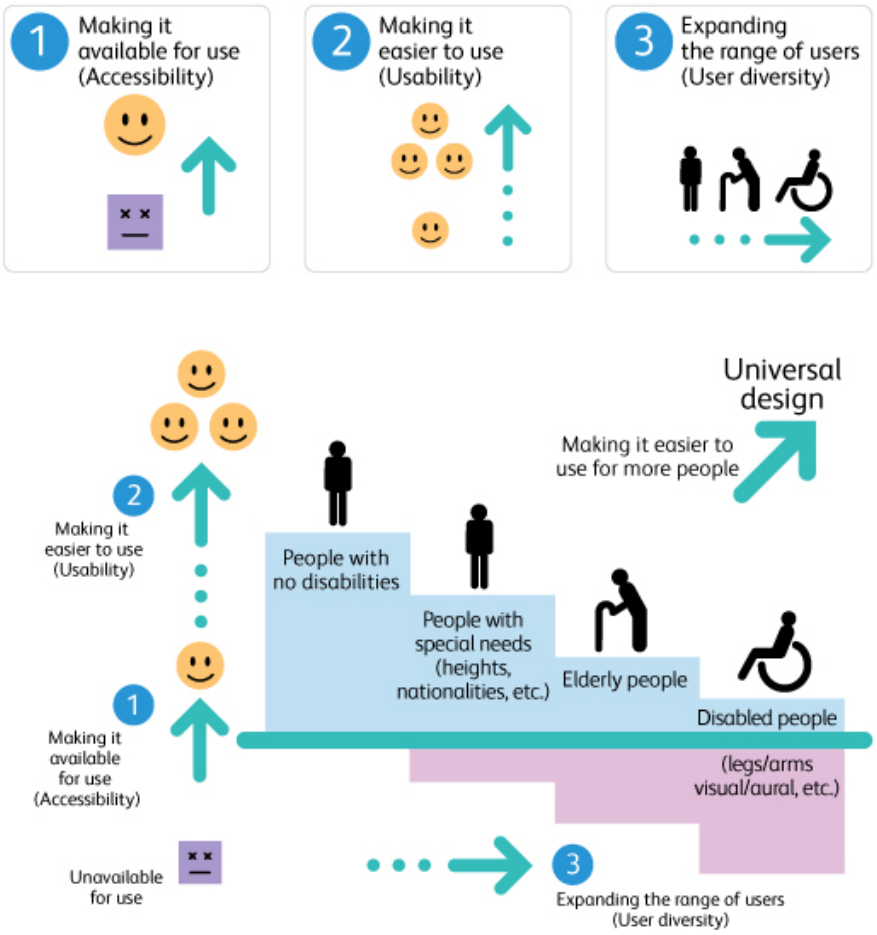

Figura 2.5: Design inclusivo/universal (Fonte: [17]).

A Figura 2.5 apresenta um *banner* desenvolvido pela *Fuji Xerox* que mostra de forma didática o relacionamento entre acessibilidade, usabilidade e diversidade. Ela destaca a importância do design universal para a facilitação do uso de recursos por mais pessoas.

Existem inúmeros benefícios a serem obtidos ao fazer com que projetos inovadores tenham apelo universal e sejam o mais inclusivos possível. O *Centre for Excellence in Universal Design* [58] os divide nas seguintes categorias:

- Benefícios para o indivíduo: fácil de usar e conveniente, mas também respeita a dignidade, os direitos e a privacidade do usuário;
- Benefícios para a sociedade: maior expectativa de vida e desenvolvimentos médicos, mais assistência baseada em tecnologia necessária com mudanças demográficas;
- Benefícios de negócios: aumentos em mercados potenciais para maior satisfação do cliente;
- Legislação e normas: promove a conformidade, mas tem um potencial muito mais amplo para melhorar a acessibilidade e usabilidade, além dos requisitos mínimos impostos por lei.

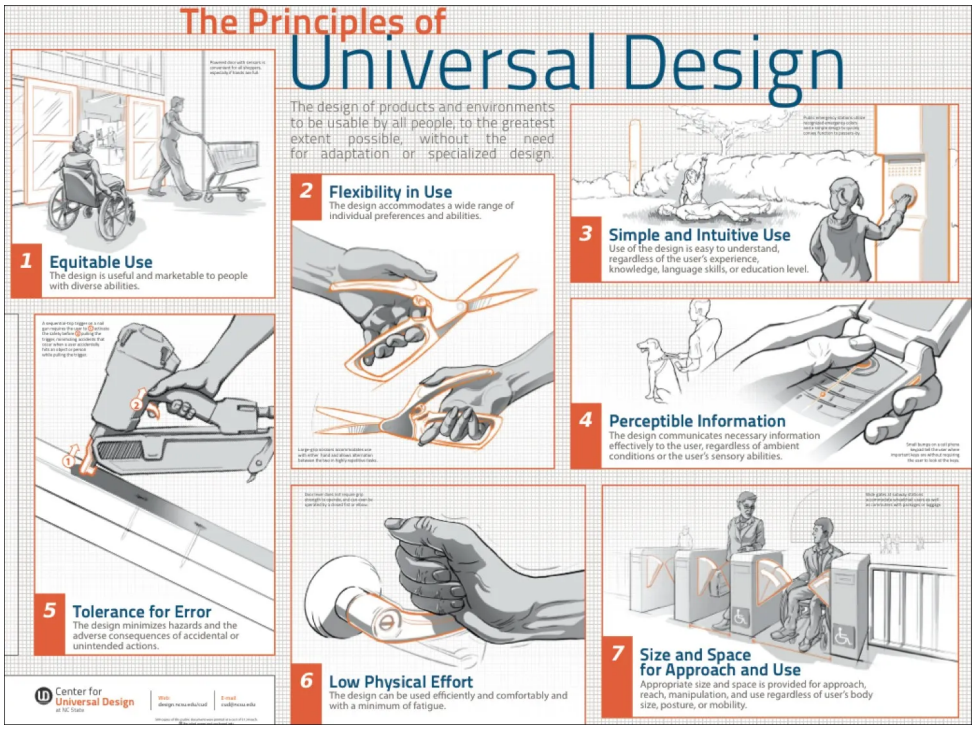

Figura 2.6: Princípios do design universal (Fonte: [59]).

A Figura 2.6 mostra os princípios do design universal, criados por Ron Mace e um grupo de pesquisadores e profissionais de design nos Estados Unidos. Foi publicado em 1997 pela *NC State University, The Center for Visual Design*. Os Princípios do design universal são um recurso inestimável usado para planejar e orientar o processo de design de forma inteligente. Ele visa orientar o design de produtos e ambientes para serem usados por todas as pessoas, na medida do possível, sem a necessidade de adaptação ou design especializado [60]. A seguir uma descrição resumida dos 7 princípios:

- Uso equitativo: o design é útil e comercializável para pessoas com diversas habilidades;
- Flexibilidade no uso: o design acomoda uma ampla gama de preferências e habilidades individuais;
- Uso simples e intuitivo: o uso do design é fácil de entender, independentemente da experiência, conhecimento, habilidades linguísticas ou nível de concentração atual do usuário;
- Informação perceptível: o design comunica as informações necessárias de forma eficaz ao usuário, independentemente das condições do ambiente ou das habilidades sensoriais do usuário;
- Tolerância ao erro: o design minimiza os perigos e as consequências adversas de ações acidentais ou não intencionais;
- Baixo esforço físico: o design pode ser usado de forma eficiente e confortável e com um mínimo de fadiga;
- Tamanho e espaço para abordagem e uso: tamanho e espaço apropriados são fornecidos para abordagem, alcance, manipulação e uso, independentemente do tamanho do corpo, postura ou mobilidade do usuário.

2.2 Aplicação Web

Um aplicativo Web (ou *web app*) é um software de aplicativo acessado por meio de um navegador da Web [61]. Enquanto aplicativos nativos precisam ser desenvolvidos individualmente para cada plataforma, os aplicativos Web estão no extremo oposto. O código não é escrito visando a plataforma, mas sim qualquer navegador rodando sobre ela [62]. Existem também algumas ferramentas intermediárias como compilador cruzado (*cross compiled*), por exemplo o Flutter[1] e *Web native apps*, por exemplo o React Native[2]. No entanto, algumas estatísticas mostradas a seguir sugerem que a escolha do aplicativo Web é mais adequada para este projeto.

De acordo com o *The 2017 U.S. Mobile App Report*, [63] nos Estados Unidos, os usuários de *smartphones* gastam quase 80% do uso total de aplicativos em seus três aplicativos favoritos e 96% em seus 10 principais aplicativos pessoais conforme ilustrado na Figura 2.7. Este cenário, mesmo que não represente uma amostra de estudantes da universidade e nem mesmo da população brasileira, é um indicativo de uma tendência

[1] `https://flutter.dev/`
[2] `https://reactnative.dev/`

dos usuários de utilizarem apenas poucos aplicativos nos celulares. Esta mesma pesquisa afirma que a fidelidade de um usuário a um aplicativo móvel é muito mais relevante do que a sites móveis.

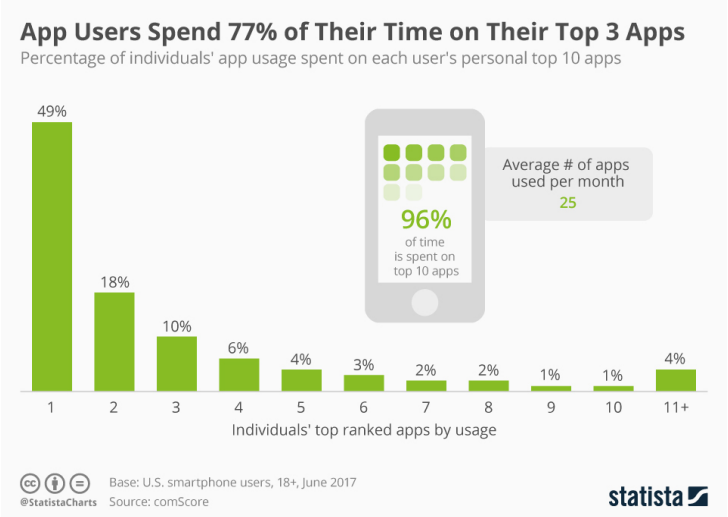

Figura 2.7: Os usuários de aplicativos gastam 77% de seu tempo em seus 3 principais aplicativos (Fonte: [64]).

No entanto, a mesma pesquisa afirma que, apesar dos usuários gastarem 16 vezes mais tempo nos principais aplicativos do que nos principais sites, os sites tendem a atrair públicos maiores, ou um maior número de visitantes por mês conforme ilustrado na Figura 2.8.

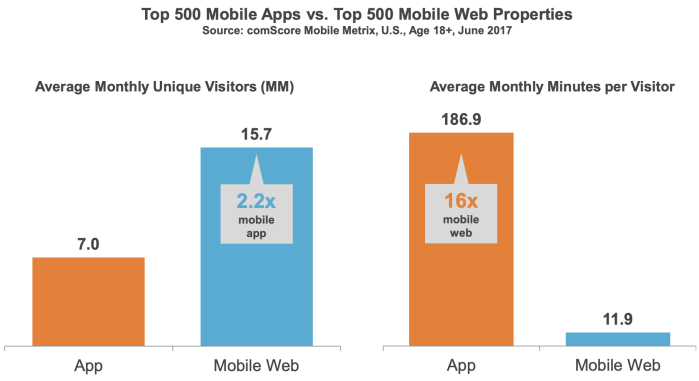

Figura 2.8: *The 2017 U.S. Mobile App Report* (Fonte: [63]).

Considerando estas informações, foi tomada a decisão pelo desenvolvimento de um aplicativo Web já que a aplicação para integração de estudantes não visa ser um aplicativo de uso recorrente como os principais aplicativos utilizados pelo usuário mostrado na pesquisa. Além disso, considerando a resistência existente por parte dos usuários para

instalação de novas aplicações nos seus dispositivos, a solução proposta visa não causar resistência de uso para os usuários. Por fim, o desenvolvimento para uma única plataforma, o navegador, facilita a manutenção do software e provê mais agilidade na entrega do produto.

2.2.1 *Web standards*: HTML, CSS e JavaScript

Considerando a escolha pela aplicação Web, consequentemente é definida a escolha pelas ferramentas base para desenvolvimento Web: HTML, CSS e JavaScript.

Web Standards, traduzido como "Normas para Web", são padrões formais e não proprietários e outras especificações técnicas que definem e descrevem aspectos da *World Wide Web* definidos pela World Wide Web Consortium (W3C). O termo tem sido associado com mais frequência à tendência de endossar um conjunto de práticas recomendadas padronizadas para a construção de sites da Web. Quando um site ou página da Web é descrito como em conformidade com os padrões da Web, isso geralmente significa que o site ou a página tem HTML , CSS e JavaScript.[3]

HTML

A especificação oficial do HyperText Markup Language (HTML)[4] o define como "linguagem de marcação central da *World Wide Web*. Originalmente, o HTML foi projetado principalmente como uma linguagem para descrever semanticamente documentos científicos. A sua concepção geral, no entanto, permitiu que fosse adaptada, ao longo dos anos seguintes, para descrever uma série de outros tipos de documentos e mesmo aplicações" [65]. Além disso, a especificação define sua posição na pilha de especificações da plataforma Web em relação a outras especificações conforme mostrado na Figura 2.9.

CSS

A especificação oficial do Cascading Style Sheets (CSS)[5] o define como "uma linguagem para escrever *style sheet* (folhas de estilo)". Este termo é usado para expressar a apresentação de documentos estruturados, ou seja, em que algum método de marcação é usado para identificar o todo e as partes do documento como tendo vários significados além de sua formatação. A especificação também afirma que o CSS é "projetado para descrever a renderização de documentos estruturados (como HTML e XML) em uma variedade de mídias. CSS é usado para descrever a apresentação de um documento de origem e ge-

[3] `https://www.webstandards.org/`
[4] `https://html.spec.whatwg.org/`
[5] `https://www.w3.org/TR/CSS/`

Figura 2.9: Especificação HTML na pilha de especificações da plataforma Web (Fonte: [65]).

ralmente não altera a semântica subjacente expressa por sua linguagem de documento" [66].

JavaScript

JavaScript é uma linguagem interpretada de alto nível, geralmente compilação *just-in-time (JIT)* também conhecida como tradução dinâmica ou em tempo de execução, que está em conformidade com o padrão ECMAScript [67]. Este é um padrão destinado a garantir a interoperabilidade de páginas da Web em diferentes navegadores. É padronizado pela Ecma International no documento ECMA-262.[6] O ECMAScript é comumente usado para scripts do lado do cliente na World Wide Web e está sendo cada vez mais usado para escrever aplicativos e serviços do lado do servidor usando Node.js e outros ambientes de tempo de execução. O JavaScript é a linguagem de script dominante do lado do cliente da Web, com 98% de todos os sites usando-o para essa finalidade [68].

2.2.2 Single Page Application (SPA)

As *Single Page Application (SPA)* ou aplicativos de página única são aplicativos ou sites que interagem com o usuário reescrevendo a página dinamicamente com novos dados do servidor, em vez do método padrão de um navegador da Web que carrega novas páginas inteiras. Este recurso tem o objetivo de fazer transições mais rápidas. As SPAs têm como vantagem a performance após o carregamento inicial e a experiência do usuário intuitiva e de baixa fricção com sensação de coesão e imersão, construindo uma familiaridade com a

[6] `https://www.ecma-international.org/publications-and-standards/standards/ecma-262/`

interface rapidamente. Além disso, a página pode ser armazenada no *cache* do navegador e pode funcionar *offline* a partir dos dados já recebidos. Também promove agilidade de desenvolvimento já que, por serem baseadas em *frameworks* muito populares, as SPAs não apresentam grandes desafios de *debugging* [69]. Todas essas vantagens são essenciais para o projeto em questão, já que os estudantes precisam construir uma familiaridade com a aplicação para se sentirem participantes ativos, ou seja, criadores de conteúdo e fornecedores de informações da universidade, não apenas leitores.

Conforme mostrado na Figura 2.10, uma página inicial pode fazer parte de um site de várias páginas ou de uma única página. Desta forma, é possível perceber que as SPAs estimulam o reaproveitamento de código e mantêm a consistência de itens repetidos entre páginas, por exemplo, o logo e o menu.

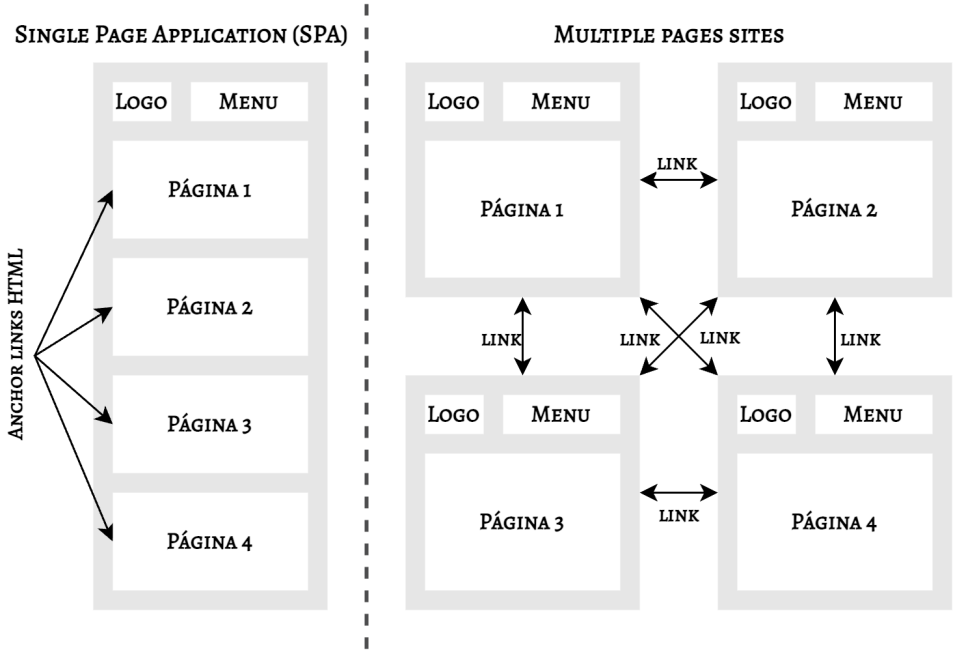

Figura 2.10: Single Page Application (SPA) vs multiple pages sites (Fonte: adaptado de [3]).

Os mesmos fatores que tornam as SPAs uma escolha inteligente para muitas aplicações, também podem representar obstáculos [69]. Uma das desvantagens é com relação a mecanismos de busca que têm maiores dificuldades em conseguir detectar ou indexar este conteúdo, pois atualizam a mesma página para exibir diversas informações. Já existem tecnologias para gerenciamento de rotas que resolvem este problema, mas para o projeto em questão este problema não se aplica já que é uma ferramenta interna, ou seja, todo o conteúdo produzido pelos alunos só está disponível para usuários autenticados. Então as SPAs continuam sendo vantajosas.

Outra desvantagem é com relação ao tempo de carregamento. As SPAs costumam ter um carregamento inicial ligeiramente mais lento do que páginas estáticas. Além da diferença no tamanho do arquivo, a quantidade de conteúdo a ser renderizado pode significar um tempo de carregamento maior, mesmo quando a página está armazenada no *cache*. Como alternativa para este problema, o projeto deve ser focado na recuperação de conteúdo por meio de paginação [70]. Assim, a diminuição na recuperação de dados do banco de dados pode compensar na rapidez do carregamento dos arquivos estáticos.

2.2.3 Arquitetura

A arquitetura de uma aplicação Web é composta dos elementos mostrados na Figura 2.11. O capítulo a seguir detalha a escolha das ferramentas e recursos utilizados no projeto para cada uma dos elementos da arquitetura Web [4].

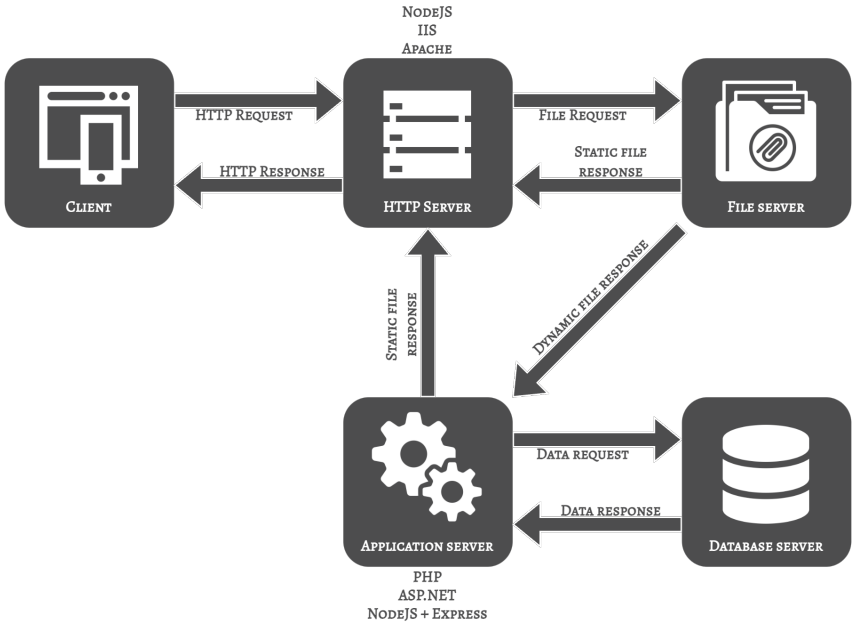

Figura 2.11: Arquitetura de aplicações Web (Fonte: adaptado de [4]).

Capítulo 3

Tecnologias

3.1 Do lado do cliente

Bootstrap

O Bootstrap é uma estrutura CSS gratuita e de código aberto voltada para o desenvolvimento Web front-end responsivo e móvel. Ele contém HTML, CSS e modelos de design baseados em JavaScript para tipografia, formulários, botões, navegação e outros componentes de interface [71].

Em março de 2023, o Bootstrap é o 16º projeto com mais estrelas no GitHub, possuindo mais de 162.000 estrelas [72]. De acordo com a W3Techs, o Bootstrap é usado por 79,0% de todos os sites cujo framework CSS é conhecido. Isso representa 19,1% de todos os sites [73].

Esta grande relevância do Bootstrap representa uma abundância de recursos da comunidade desenvolvido sobre a ferramenta, uma grande quantidade de modelos de *design* de páginas para inspiração e uma evolução ativa e contínua da tecnologia. Além disso, a ferramenta facilita o desenvolvimento de aplicações responsivas, o que aumenta a acessibilidade ao software desenvolvido, já que telefone celular segue como principal equipamento para acesso à internet no Brasil. Segundo a Pesquisa Nacional por Amostra de Domicílios Contínua (PNAD Contínua) ano de 2021, o aparelho é utilizado em 99,5% dos domicílios com acesso à internet [74].

O Bootstrap é usado para fazer o layout de sites usando itens como contêineres, linhas e colunas. Ele também permite que você adicione facilmente coisas como cartões e *accordions*. Um tema com cores e fontes globais pode ser configurado facilmente. Ele também possui muitas classes CSS predefinidas para estilizar facilmente. Dois dos principais recursos do Bootstrap são os *grid systems* e os *breakpoints*.

Quadro 3.1: Bootstrap *breakpoints* (Fonte: [75]).

xs	Extra small <576px	portrait mobile
sm	Small ≥576px	landscape mobile
md	Medium ≥768px	portrait tablets *navbar collapse*
lg	Large ≥992px	landscape tablets
xl	Extra large ≥1200px	laptops, desktops, TVs

O *grid system* é uma grade *flexbox* móvel (um modelo de layout da Web CSS3) para criar layouts de todas as formas e tamanhos, graças a um sistema de doze colunas, seis níveis responsivos padrão e dezenas de classes predefinidas. O sistema de grade do Bootstrap usa uma série de contêineres, linhas e colunas para fazer o layout e alinhar o conteúdo. É construído com *flexbox* e é totalmente responsivo [76]. Já os *breakpoints* são larguras personalizáveis que determinam como seu layout responsivo se comporta em dispositivos ou tamanhos de viewport no Bootstrap [77]. O Quadro 3.1 apresenta classes de grade predefinidas para criar rapidamente layouts de grade para diferentes tipos de dispositivos [75].

Outras bibliotecas

Uma biblioteca JavaScript é uma biblioteca de código pré-escrito que permite um desenvolvimento mais fácil de aplicativos baseados em JavaScript. Algumas das bibliotecas utilizadas neste projeto são listadas abaixo:

- **Bootbox.js:** é uma pequena biblioteca JavaScript que permite criar caixas de diálogo modais personalizadas usando modais Bootstrap, sem ter que se preocupar em criar, gerenciar ou remover qualquer um dos elementos DOM necessários ou manipuladores de eventos JavaScript. Um modal Bootstrap permite adicionar caixas

de diálogo ao site para *lightboxes*, notificações do usuário ou conteúdo totalmente personalizado;

- **FullCalendar:** biblioteca JavaScript que fornece uma interface de calendário com recursos de arrastar e soltar, linha do tempo, criação de eventos, exibições de ano, mês e dia, datas selecionáveis, eventos de segundo plano, tematização, localidades e fusos horários;
- **Summernote:** biblioteca JavaScript que ajuda a criar editores Web do tipo What You See Is What You Get (WYSIWYG), cuja tradução significa "O que você vê é o que você obtém". O Summernote tem alguns recursos especiais: colar imagens da área de transferência, UI simples, funciona com o framework Bootstrap e tem muitos *plugins* e conectores fornecidos juntos.

3.2 Do lado do servidor

Node.js

O Node.js foi criado como uma alternativa ao Apache HTTP Server buscando resolver algumas limitações desse, como o suporte relativamente baixo a múltiplas conexões. Este problema é resolvido com uma abordagem focada em eventos e execução assíncrona, sendo uma opção com menos consumo de recursos do que os servidores "tradicionais" [78].

O Node.js é um ambiente de tempo de execução de JavaScript de back-end, executado no V8 *JavaScript Engine* e executa o código fora de um navegador da Web [79]. Casciaro e Mammino destaca em *Node.js Design Patterns* os principais componentes da arquitetura do Node.js conforme mostrado na Figura 3.1 [79].

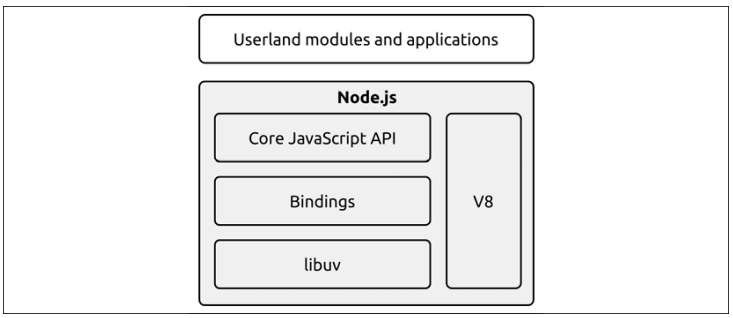

Figura 3.1: Arquitetura Node.js (Fonte: [79]).

- V8: é um interpretador JavaScript, originalmente desenvolvido pelo Google para o navegador Chrome.

- *libuv*: tem como objetivo de tornar o Node.js compatível com todos os principais sistemas operacionais e normalizar o comportamento não bloqueador dos diferentes tipos de recursos. *libuv* representa o mecanismo de E/S de baixo nível do Node.js e, segundo Casciaro e Mammino [79], é provavelmente o componente mais importante no qual o Node.js é construído.
- *bindings*: são um conjunto de ligações responsáveis por agrupar e expor o *libuv* e outras funcionalidades de baixo nível para JavaScript.
- *Core JavaScript API*: é uma biblioteca JavaScript básica que implementa a API Node.js de alto nível.

Node Package Manager (npm) é o gerenciador de pacotes padrão do Node.js. Ele dá acesso a um enorme repositório de bibliotecas e módulos que podem ser adicionados ao projeto, contendo ferramentas e soluções para os mais diversos problemas e requisitos [78]. O npm também pode instalar todas as dependências de um projeto definidas no arquivo package.json, o que facilita a distribuição do software.

O site *Stack Overflow*, um portal focado em perguntas e respostas para profissionais e entusiastas na área de programação de computadores, realizou em 2022 uma Pesquisa do Desenvolvedor com 73.268 desenvolvedores de software de 180 países ao redor do mundo [80]. Na seção *Web frameworks and technologies* foi feita uma pergunta que pode ser traduzida como: *Em quais frameworks e tecnologias da Web você fez um trabalho de desenvolvimento em grande quantidade no ano passado e em quais você deseja trabalhar no próximo ano?*

Esta questão obteve 49.625 respostas. A partir dos resultados, conforme mostrado na Figura 3.2, é perceptível uma relevância notável do Node.js na comunidade de desenvolvedores. Esta grande participação de mercado representa, consequentemente, a disponibilidade de um grande número de ferramentas e recursos desenvolvidos sobre a plataforma e uma evolução ativa e contínua da tecnologia.

Considerando os fatores acima de arquitetura e relevância do Node.js, esta aplicação foi adotada no projeto também por adotar uma mesma linguagem de desenvolvimento, tanto no *front-end* como no *back-end*, o JavaScript. Desta forma, existe a possibilidade de manter o ecossistema de aplicações e toda a base de código em uma só linguagem de programação.

SQLite

O site oficial do SQLite o define como "uma biblioteca de linguagem C que implementa um mecanismo de banco de dados SQL leve, rápido, independente, de alta confiabilidade e completo. O SQLite está embutido em todos os telefones celulares e na maioria dos

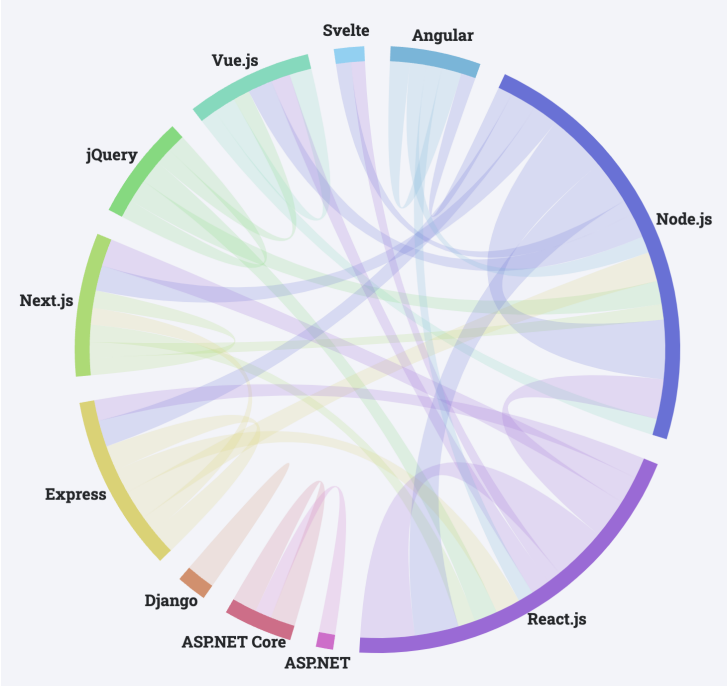

Figura 3.2: *Stack Overflow Developer Survey 2022 - Web frameworks and technologies* (Fonte: [80]).

computadores e vem embutido em inúmeros outros aplicativos que as pessoas usam todos os dias. O formato de arquivo SQLite é estável, multiplataforma e compatível com versões anteriores e os desenvolvedores se comprometem a mantê-lo assim até o ano de 2050. Arquivos de banco de dados SQLite são comumente usados como contêineres para transferir conteúdo entre sistemas e como um formato de arquivamento de longo prazo para dados. Existem mais de 1 trilhão de bancos de dados SQLite em uso ativo [81]".

Normalmente, um *Relational Database Management System (RDBMS)*, em português Sistema de Gerenciamento de Banco de Dados Relacional (SGBDR), como MySQL, PostgreSQL, etc., requer um servidor separado para operar. As aplicações que desejam acessar o servidor de banco de dados utilizam o protocolo TCP/IP para enviar e receber solicitações [5]. Já o SQLite é um bancos de dados independente, contido em arquivos. Como ele está contidos em arquivos, não há servidor necessário para manter os dados. Isso também significa que o banco de dados é específico do projeto, pois o arquivo é acessado no diretório do projeto. A Figura 3.3 ilustra a arquitetura cliente/servidor de um RDBMS (o mesmo se aplica para bancos não-relacionais em geral) em comparação à arquitetura sem servidor do SQLite.

Outras características do SQLite incluem ser independente de plataforma, requer suporte mínimo do sistema operacional, configuração zero por causa da arquitetura sem servidor, transacional, ou seja, todas as consultas e alterações são atômicas, consistentes,

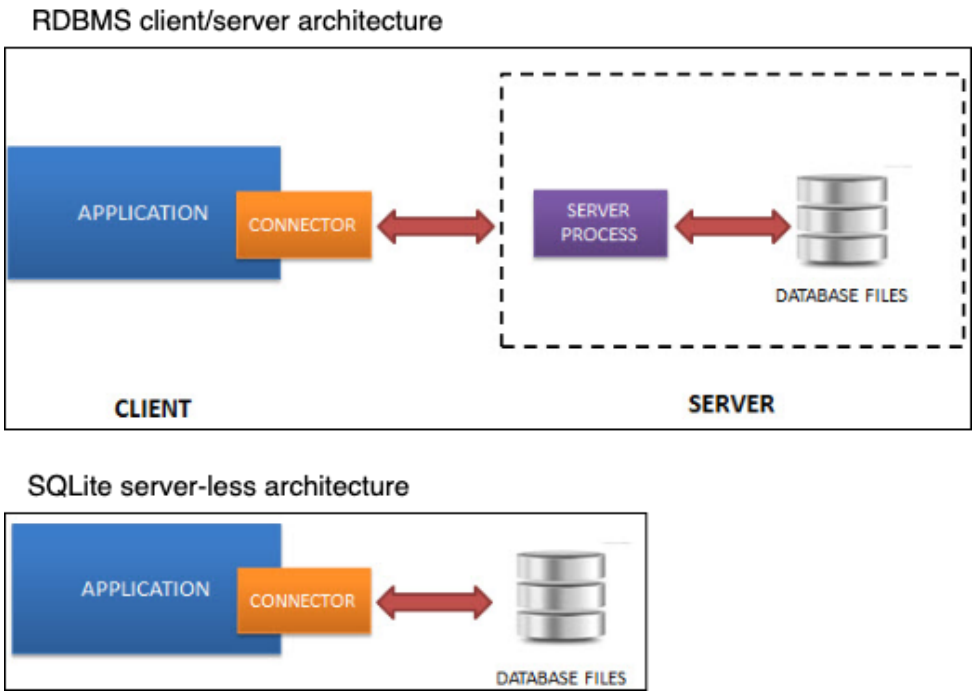

Figura 3.3: *RDBMS vs SQLite* (Fonte: adaptado de [5]).

isoladas e duráveis (ACID) [5].

Desta forma, o SQLite foi a escolha feita para o desenvolvimento da aplicação de integração de estudantes por sua simplicidade e maneira compacta de armazenamento. Esses critérios são adequados para aplicativos que não exigem grandes sistemas de banco de dados como é o caso.

Knex.js

O knex.js é um *query builder* por meio do javascript que unifica a forma de fazer *querys* para os bancos de dados SQL, não dependendo de um banco específico e podendo trocar sempre que necessário. Sendo assim, a aplicação ainda funcionará da mesma forma sem alterar o código. O ambiente em que ele é mais utilizado é o Node.js. O knex.js tem suporte completo ao Postgres, Sql Server, Mysql, MariaDB, Sqlite3, Oracle e Amazon Redshift. Além disso ele disponibiliza *migrations* (migrações) e *seeders* (propagadores) que referem-se, respectivamente, ao gerenciamento de alterações controladas por versão, incrementais e reversíveis em esquemas de banco de dados relacionais; e ao preenchimento de um banco de dados com um conjunto inicial de dados [82].

Estas funcionalidades providas pelo Knex.js são essenciais para facilitar uma possível mudança futura no banco de dados utilizado pelo projeto de integração de estudantes.

Esta alteração não é o propósito deste trabalho em específico, mas o projeto se torna, assim, mais escalável e manutenível.

3.3 De desenvolvimento

Sistema de controle de versões

O *Version Control System (VCS)*, ou no português sistema de controle de versões, é uma categoria de ferramentas de software que ajudam a registrar as alterações feitas nos arquivos, mantendo um registro das modificações feitas no código [83]. Ele acompanha cada modificação no código em um tipo especial de banco de dados. Se um erro for cometido, os desenvolvedores podem voltar no tempo e comparar as versões anteriores do código para ajudar a corrigir o erro, minimizando a interrupção de todos os membros da equipe [84].

As três ferramentas de controle de versão mais conhecidas são Git, Subversion (SVN) e Mercurial [85]. A pesquisa *Stack Overflow Developer Survey 2022* realizou, na seção *Version control systems*, a seguinte pergunta, que pode ser traduzida como: *Quais são os principais sistemas de controle de versão que você usa? Selecione tudo que se aplica* [80].

Esta questão obteve 71.379 respostas. A partir dos resultados, conforme mostrado na Figura 3.4, pode-se perceber que a maioria das pessoas atualmente usam, no mercado de trabalho, a ferramenta Git como sistema de controle de versões. Em decorrência desta ampla utilização e de seus recursos que são abordados a seguir, esta ferramenta foi escolhida para utilização no projeto.

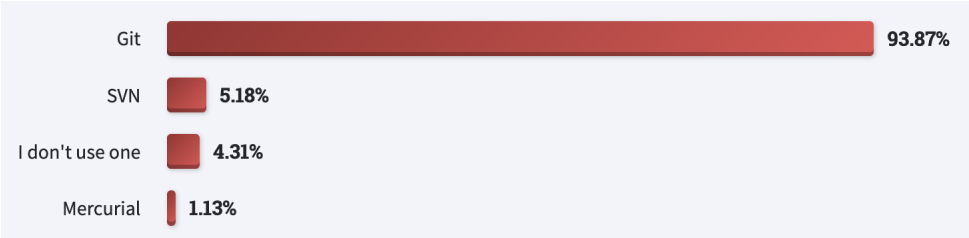

Figura 3.4: *Stack Overflow Developer Survey 2022 - Version control systems* (Fonte: [80]).

Git

O Git é um sistema de controle de versão distribuído que rastreia alterações em qualquer conjunto de arquivos de computador, geralmente usado para coordenar o trabalho entre programadores que desenvolvem código-fonte de forma colaborativa durante o desenvol-

vimento de software. Seus objetivos incluem velocidade, integridade de dados e suporte para fluxos de trabalho não lineares distribuídos [86].

A Figura 3.5 mostra as principais partes que são importantes ao usar o Git. Como ele é um sistema distribuído, tem-se um repositório local (*local repository*) que é onde todas as alterações são registradas. Todas as suas alterações são primeiro confirmadas em seu repositório local e devem, então, ser explicitamente enviadas para um repositório remoto (*remote repository*). Os arquivos com seu código estão em um diretório de trabalho (*working directory*). Entre seu diretório de trabalho e o repositório local há uma área de preparação (*staging area*) que reúne todas as alterações antes de serem confirmadas no repositório local [87].

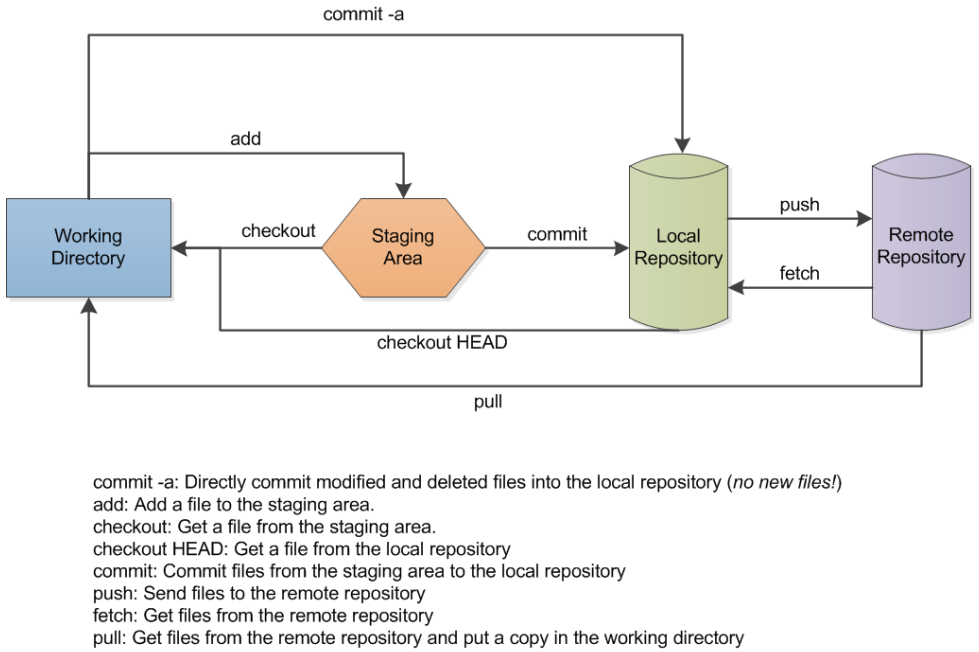

Figura 3.5: *Fluxo de dados Git* (Fonte: [88]).

Ainda na Figura 3.5, é possível ver os principais fluxos de dados Git e suas nomenclaturas. As alterações são adicionadas (*add*) à *staging area*. O *commit* descreve o processo de adicionar arquivos ao repositório local a partir da área de teste, enquanto um *push* envia todas as alterações para o repositório remoto. O *fetch* significa obter todas as alterações do repositório remoto para o repositório local. Um *pull* copia diretamente o repositório remoto para o repositório local. O *checkout* reverte as alterações em seu repositório local e restaura o estado dos arquivos da *staging area* ou do repositório local [87].

GitHub

O GitHub é um serviço de hospedagem de projetos de desenvolvimento de software na internet. Ele fornece o controle de versão distribuída do Git, controle de acesso, rastreamento de bugs, solicitações de recursos de software, gerenciamento de tarefas, integração contínua e wikis para cada projeto [89].

Para o projeto de integração de estudantes da universidade, o GitHub é usado como uma hospedagem para o projeto, já que ele oferece hospedagem gratuita principalmente para projetos *open-source* tornados público. Este é o caso da aplicação deste trabalho.[1] Além de hospedar o código fonte, ele permite expor todo o processo de desenvolvimento por meio dos *commits* permitindo novos desenvolvedores entenderem o processo em que foi desenvolvido. Como o projeto proposto neste documento é apenas um modelo inicial que pode ter seu desenvolvimento continuado, o GitHub auxilia o desenvolvimento em equipe por meio de recursos como *forks* e *branches* do Git

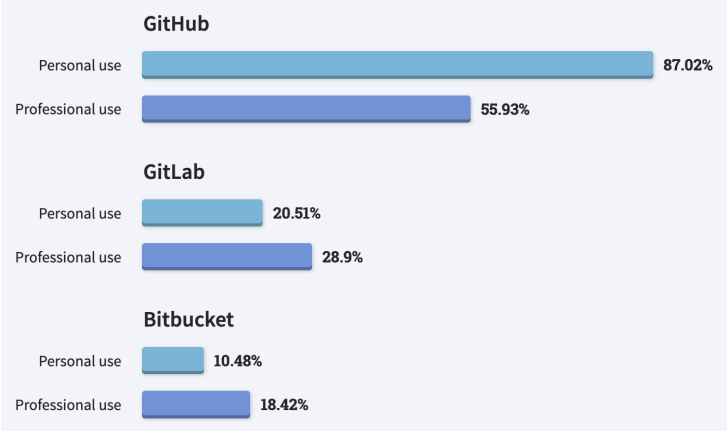

Figura 3.6: *Stack Overflow Developer Survey 2022 - Version control platforms* (Fonte: [80]).

A pesquisa *Stack Overflow Developer Survey 2022* realizou, na seção *Version control platforms*, a seguinte pergunta, que pode ser traduzida como [80]: *Qual serviço de hospedagem de controle de versão você está usando?*

Esta questão obteve 67.035 respostas. A partir dos resultados, conforme mostrado na Figura 3.6, é possível notar uma utilização maior do GitHub em relação às outras alternativas. Em decorrência desta maior utilização e de seus recursos apresentados acima, esta ferramenta foi escolhida para fazer o controle de versões do projeto de integração de estudantes.

[1] `https://github.com/hlcurzio/projeto-final`

Visual Studio Code

Visual Studio Code, também conhecido como VS Code, é um ambiente de desenvolvimento integrado, do inglês *Integrated development environment (IDE)* desenvolvido pela Microsoft. A Microsoft lançou a maior parte do código-fonte no GitHub sob a licença permissiva MIT, enquanto os lançamentos da Microsoft são *freeware* proprietário [90].

A pesquisa *Stack Overflow Developer Survey 2022* realizou, na seção *Integrated development environment*, a seguinte pergunta que pode ser traduzida como [80]: *Quais ambientes de desenvolvimento você usou regularmente no ano passado e com quais deseja trabalhar no próximo ano? Por favor, verifique tudo o que se aplica.*

Esta questão obteve 71.010 respostas. A partir dos resultados, conforme mostrado na Figura 3.7, é possível notar um maior número de utilização do Visual Studio Code em relação às outras IDEs.

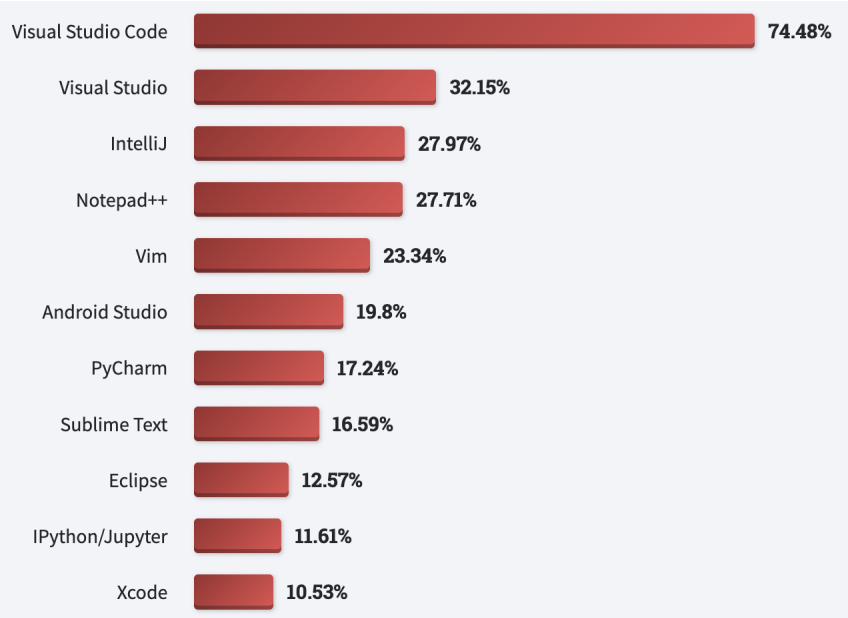

Figura 3.7: *Stack Overflow Developer Survey 2022 - Integrated development environment (IDE)* (Fonte: [80]).

O Visual Studio Code pode ser estendido por meio de extensões, disponíveis por meio de um repositório central. Isso inclui adições ao editor e suporte a idiomas. Um recurso notável é a capacidade de criar extensões que adicionam suporte para novos idiomas, temas, depuradores, depuradores de *time traveling*, realizar análise de código estático e adicionar *linters* de código usando o *Language Server Protocol* [91].

Ao contrário de muitos outros editores de código, o Visual Studio Code possui um depurador embutido, tornando o fluxo de desenvolvimento mais fluido e mantém uma visão única com código e depurador. Isso torna o processo de rastreamento de *bugs* e

execução de código muito mais fácil e rápido. Não é necessário ter várias telas para executar os diferentes consoles e reorganizá-los cada vez que precisar minimizar algo. Ele está embutido no design e na configuração do espaço de trabalho desejado [92].

Wappler

O Wappler é um ambiente de desenvolvimento proprietário (com avaliação gratuita) multiplataforma (Windows, macOS e Linux), executado em cima do Visual Studio Code e com várias ferramentas visuais para auxiliar no desenvolvimento. O código-fonte gerado visualmente pode ser modificado através da visualização de código, dentro da própria plataforma e em outras IDEs ou editores de código [93]. A aplicação criada pode ser executada em *localhost* ou publicada em qualquer provedor de hospedagem. Ele é um produto da empresa DMXzone, que desenvolve extensões para a IDE proprietária Adobe Dreamweaver e passou a desenvolver também para o VS Code, combinando todas as extensões em um só produto comercial e focando os esforços da empresa nele, o Wappler [94]. Para todo o sistema a ser desenvolvido é utilizado apenas a versão de avaliação gratuita do Wappler, com limite de 14 dias e sem perda de recursos. Para isto, múltiplas contas são criadas reaproveitando o mesmo código-fonte.

Os recursos suportados pelas ferramentas visuais incluem:

- *Application Server*: NodeJS, PHP, ASP.NET e Classic ASP;
- Frameworks: Bootstrap, Framework7, jQuery;
- Bancos de dados: SQLite, MySQL, MariaDB, MS SQL Server, Postgres e CouchDB;
- Ambientes de desenvolvimento: Servidor local, Firebase Hosting, Heroku e Docker;
- *HTTP Server*: NodeJS, Apache e IIS;
- *Deploys*: FTP, SFTP e FTPS;
- Git: GitHub, Gitlab e BitBucket.

Para o projeto em questão, o processo de escolha para cada ferramenta foi descrito nas outras seções ao longo deste documento. As mesmas ferramentas se encontram sublinhadas na listagem acima.

O Quadro 3.2 apresenta a comparação de algumas plataformas low-code líderes de participação de mercado e listadas em *Characteristics and Challenges of Low-Code Development: The Practitioners' Perspective* [96]. Entre elas, apenas o Wappler permite desenvolver de forma local, em várias linguagens de programação e *frameworks* diferentes e com código fonte totalmente disponível para desenvolvimento em paralelo com outras IDEs e bibliotecas não suportadas pela ferramenta.

Quadro 3.2: Plataformas low-code (Fonte: inspirado em [62] e [95]).

Plataformas low-code		Small business / Mid-market	Enterprise
Client-side (Web / Mobile)	Web / PWA (HTML/CSS/JS)	Bubble Glide (PWA) Wordpress	Oracle APEX
	Hybrid (HTML/CSS/JS)	Wappler (Capacitor/Cordova)	OutSystems (Cordova)
	Web native (JavaScript)	AppGyver (React Native) Thunkable (React Native)	SAP Build Apps (React Native)
	Cross-compiled	Flutterflow (Flutter)	
	Native	Kodular (java) Kodika (Swift/Kotlin)	
Server-side		Wappler (PHP/Node.js/ASP.NET) Xano (PHP) Bubble (Node.js) Backendless (Node.js)	Outsystems (ASP.NET)
Database		Airtable (MySQL) Firestore / Realtime DB	Firestore / Realtime DB Caspio (MS SQL Server)

Web development tools

As *Web development tools*, em português ferramentas de desenvolvimento da Web, também chamadas de *devtools* ou inspecionador de elemento, permitem que os desenvolvedores da Web testem e depurem seus códigos-fonte. São ferramentas usadas para testar a interface do usuário de um site ou aplicativo da Web [97].

As ferramentas de desenvolvimento da Web vêm como recursos integrados nos navegadores da Web. Os navegadores mais populares, como Google Chrome, Firefox, Internet Explorer, Safari, Microsoft Edge e Opera possuem estas ferramentas integradas. Em todos os navegadores, a funcionalidade básica das ferramentas do desenvolvedor permanece a mesma, apenas as convenções de nomenclatura e as terminologias mudam [97].

Algumas funcionalidades, baseadas no Google Chrome, são:

- Elementos: permite ver o HTML e CSS da página da Web inspecionada no momento;
- Console: usado para depurar o JavaScript presente no código-fonte. Atua como a janela de depuração que permite lidar com o JavaScript que não está funcionando conforme o esperado;
- Fontes: permite definir pontos de interrupção e observar o valor das variáveis;
- Rede: usado para garantir que todos os recursos que estão sendo baixados ou carregados estão sendo executados conforme o esperado;
- Desempenho: é registrado em tempo de execução e informa como a página funciona quando está em execução, em vez de quando é carregada

- Aplicativo: utilizado para acompanhar e gerenciar *cookies*, sessões, cache e armazenamento local.

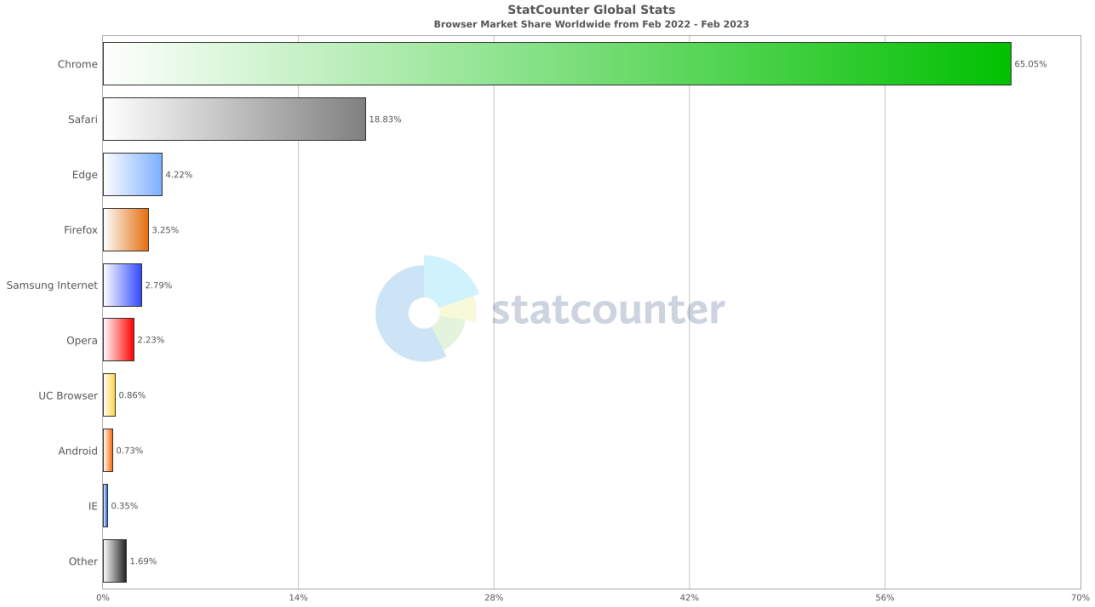

Figura 3.8: *Browser Market Share Worldwide* (Fonte: [98]).

Todas essas funcionalidades são essenciais durante o processo de desenvolvimento do projeto. Considerando que os usuários finais da aplicação podem acessá-la de diferentes navegadores Web, é adequado fazer testes em mais de um deles durante a criação do software. Para este projeto, dois deles são escolhidos, O Google Chrome e o Safari. Esta decisão é tomada com base na liderança deles na participação de mercado. A Figura 3.8 destaca uma pesquisa feita pela statCounter com a participação de mercado dos principais navegadores Web. Nela é possível notar uma disparidade da primeira e segunda colocações para as demais.

3.4 Auxiliares

MailDev

MailDev é um Servidor SMTP local e interface Web para visualização e teste de e-mails durante o desenvolvimento. Ele dispõe de uma imagem Docker para utilização [99]. Esta ferramenta é usada para testar o envio de e-mails no projeto durante, por exemplo, o cadastro de usuários. Desta forma, evita o envio desnecessário de mensagens para caixas de e-mail reais. Além disso, permite o teste de e-mail de forma local, sem a necessidade

de conexão com um provedor de hospedagem SMTP. Este recurso é também conhecido como "*Fake SMTP*". A Figura 3.9 mostra a interface da ferramenta.

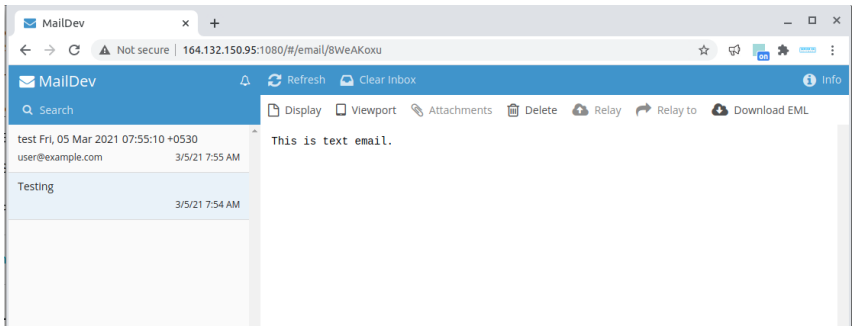

Figura 3.9: Interface de usuário do MailDev.

JSON Viewer

JavaScript Object Notation (JSON) é um formato de arquivo padrão aberto e formato de intercâmbio de dados que usa texto legível para armazenar e transmitir objetos de dados que consistem em pares atributo-valor e vetores (ou outros valores serializáveis). É um formato de dados comum com diversos usos no intercâmbio eletrônico de dados, incluindo o de aplicações Web com servidores. Ele é um formato de dados independente de linguagem. Foi derivado do JavaScript, mas muitas linguagens de programação modernas incluem código para gerar e analisar dados no formato JSON. Os nomes de arquivos usam a extensão *.json* [100].

O JSON Viewer é uma extensão de navegador para imprimir JSON. Esta ferramenta facilita a visualização por meio de realce, tematização e recursos de interação, como URLs clicáveis.

Postman

O Postman é uma plataforma para desenvolvedores utilizarem no projeto, construção e teste de API. Ela permite que os usuários armazenem, cataloguem e colaborem com artefatos de API em uma plataforma central em redes públicas, privadas ou de parceiros. Além disso, ela provê ferramentas de cliente de API, design de API, documentação de API, teste de API, servidores fictícios e detecção de API [101].

Ao contrário de uma user interface (UI), que conecta um computador a uma pessoa, uma application programming interface (API) conecta computadores ou softwares uns aos outros. Não se destina a ser usado diretamente por uma pessoa (o usuário final) que não seja um programador de computador que o esteja incorporando ao software. Uma

API geralmente é composta de diferentes partes que atuam como ferramentas ou serviços disponíveis para o programador. Diz-se que um programa ou programador que usa uma dessas partes chama essa parte da API. As chamadas que compõem a API também são conhecidas como sub-rotinas, métodos, solicitações ou *endpoints*. Uma especificação de API define essas chamadas, o que significa que explica como usá-las ou implementá-las [102].

3.5 Algoritmos e funções

Argon2

O Argon2 [103] é uma função de derivação de chave criptográfica projetado por uma equipe da Universidade de Luxemburgo, liderada por Dmitry Khovratovich, anteriormente conhecido por descobrir os melhores ataques criptanalíticos ao AES. Para derrotar ataques de força bruta e de dicionário, o Argon2 pode ser ajustado para usar o máximo de computação e memória que o aplicativo tolerar e suportar multi-threading. Ele usa o algoritmo de *hash* BLAKE2 para embaralhar com segurança os dados de entrada (senha e *salt*) [104]. O Argon2 foi selecionado como vencedor do *Password Hashing Competition* de 2015 [105].

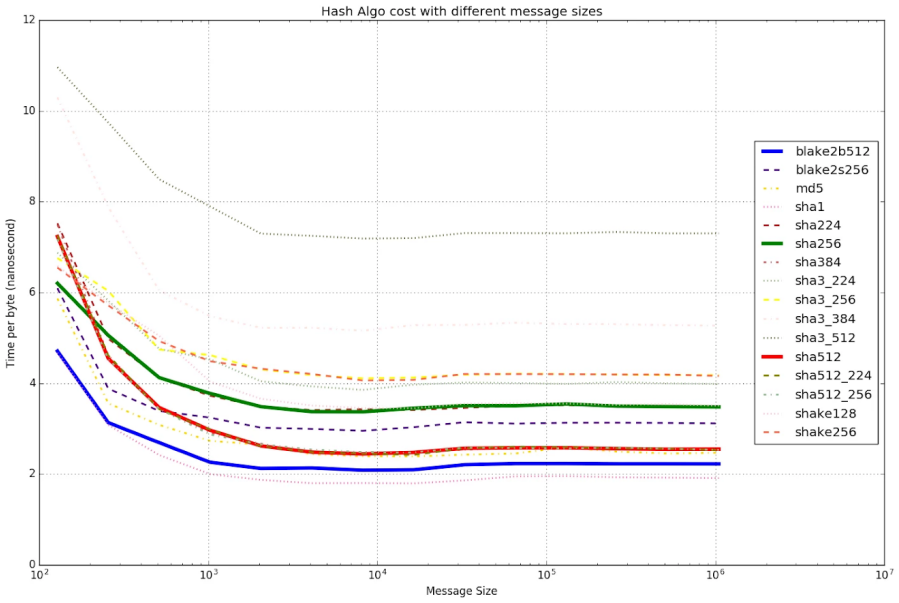

Figura 3.10: Custo de algoritmos de *hash* com diferentes tamanhos de mensagens (Fonte: [106]).

BLAKE2 [107] é uma função *hash* criptográfica baseada em BLAKE, criada por Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O'Hearn e Christian Winnerlein. O objetivo do projeto era substituir os algoritmos MD5 e SHA-1 amplamente usados, mas com problemas de segurança, em aplicativos que exigem alto desempenho em software. BLAKE2 foi anunciado em 21 de dezembro de 2012 [108]. A Figura 3.10 mostra uma comparação de algoritmos de *hash*. Nela é possível perceber que o sha256 e o sha512 têm bom desempenho e o blake2b512 é ainda mais rápido [106].

Expressão regular

As expressões regulares fornecem um método poderoso, flexível e eficiente para processar texto. A extensa notação de correspondência de padrões de expressões regulares permite analisar rapidamente grandes quantidades de texto para: encontrar padrões de caracteres específicos; validar o texto para garantir que ele corresponda a um padrão predefinido (como um endereço de e-mail); extrair, editar, substituir ou excluir substrings de texto; adicionar strings extraídas a uma coleção para gerar um relatório [109]. No projeto em questão, este recurso foi utilizado na validação de e-mail para cadastro do usuário.

3.6 Metodologias

Engenharia de requisitos

A engenharia de requisitos refere-se ao processo de definir, documentar e manter requisitos no processo de projeto de engenharia. A engenharia de requisitos fornece o mecanismo adequado para entender o que o cliente deseja, analisando a necessidade e avaliando a viabilidade, negociando uma solução razoável, especificando a solução com clareza, validando as especificações e gerenciando os requisitos à medida que são transformados em um sistema funcional. Assim, a engenharia de requisitos é a aplicação disciplinada de princípios, métodos, ferramentas e notação comprovados para descrever o comportamento pretendido de um sistema proposto e suas restrições associadas [110].

Continuous Integration/Continuous Deployment (CI/CD)

Continuous Integration/Continuous Deployment (CI/CD) são as práticas combinadas de integração contínua e entrega contínua. A primeira é a prática de mesclar todas as cópias de trabalho dos desenvolvedores em uma linha principal compartilhada várias vezes ao dia. A segunda é uma abordagem de engenharia de software na qual as equipes produzem software em ciclos curtos, garantindo que o software possa ser lançado de forma confiável

a qualquer momento e, seguindo um *pipeline* através de um "ambiente de produção", sem fazê-lo manualmente [111].

O GitHub permite desenvolver *workflows* CI/CD utilizando o recurso "Actions" [112]. Além disso, o Railway permite configurar *triggers* do GitHub para que o aplicativo possa ser implantado automaticamente quando enviado para uma ramificação selecionada no repositório conectado [113]. Isso permite automatizar várias partes do fluxo de trabalho de desenvolvimento.

CRUD

O termo create, read, update, and delete (CRUD) refere-se às quatro operações básicas que um aplicativo de software deve ser capaz de executar: criar, ler, atualizar e excluir. Nesses aplicativos, os usuários devem ser capazes de criar dados, ter acesso aos dados na interface de usuário (UI) por meio da leitura, atualizar ou editar os dados e excluir os dados [114]. A Figura 3.11 mostra uma representação, na interface de usuário (UI) destas operações.

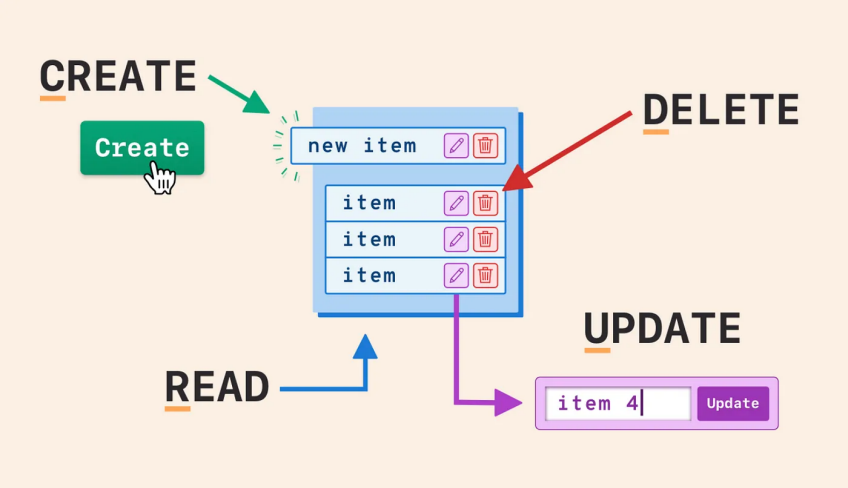

Figura 3.11: CRUD na interface de usuário (UI) (Fonte: [115]).

Em aplicativos completos, os aplicativos CRUD consistem em 3 partes: uma application programming interface (API) ou servidor, um banco de dados e uma interface de usuário (UI). A API contém o código e os métodos, o banco de dados armazena e ajuda o usuário a recuperar as informações, enquanto a interface do usuário ajuda os usuários a interagir com o aplicativo [114].

Essas quatro funções principais são usadas para interagir com aplicativos de banco de dados e são um lembrete de quais funções de manipulação de dados são necessárias para que um aplicativo pareça completo. Ao trabalhar com serviços da Web, o CRUD

corresponde aos métodos HTTP, que comunicam a um servidor da Web como você deseja interagir com um site [115]. A Figura 3.12 mostra um mapeamento comum entre as operações de CRUD e HTTP.

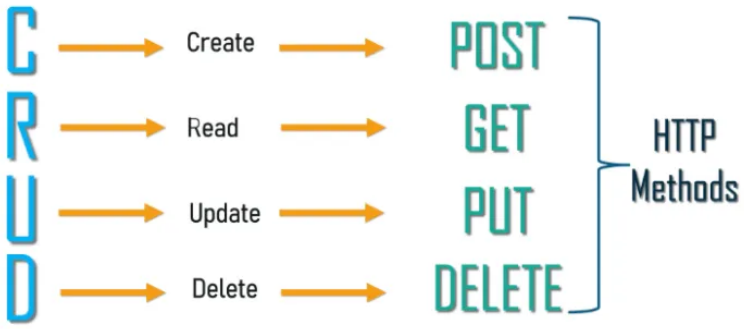

Figura 3.12: Mapeamento comum entre as operações de CRUD e HTTP (Fonte: [115]).

Capítulo 4

A aplicação: Integrar

4.1 Visão geral

Neste capítulo é apresentado como ficou o sistema desenvolvido e no capítulo seguinte os resultados da análise de usabilidade realizada. A interface da aplicação "Integrar" é mostrada por meio de capturas de tela com casos de uso simulados e dados criados para teste, detalhados na seção adiante. A visão geral do sistema é apresentada na Figura 4.1. Ele foi desenvolvido utilizando técnicas de engenharia de usabilidade com acompanhamento do professor orientador.

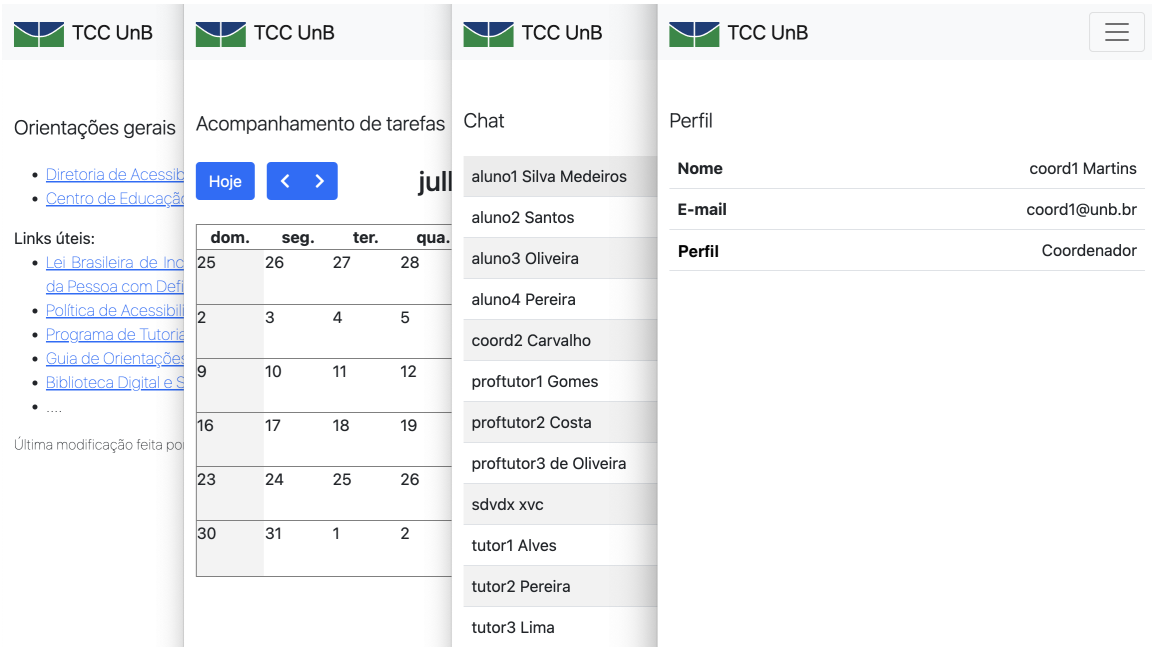

Figura 4.1: Visão geral.

Considerando que o desenvolvimento de software demanda de um processo contínuo de melhorias, foi determinado um momento de parada no ciclo de desenvolvimento para avaliações positivas e negativas do sistema em relação aos vários critérios descritos anteriormente. Eles incluem: design de interação, experiência de usuário e avaliação de usabilidade com foco no design centrado no ser humano. A Figura 4.1 apresenta algumas telas do sistema desenvolvido do ponto de vista de um usuário com perfil *coordenador*, que possui privilégios de acesso como por exemplo conversar com qualquer usuário do sistema pelo chat.

Aqui consideramos que esta visão geral do sistema, detalhada a seguir, já funciona como uma etapa de inspeção de recursos *(Feature Inspection)* dentro dos métodos de avaliação de usabilidade do tipo "inspeção" *(usability inspection)*. Posteriormente outros métodos são descritos neste documento.

Usuários

A página de usuários mostrada na Figura 4.2 lista todos os usuários castrados no banco de dados e, ao clicar em algum item da lista, um modal é aberto com o detalhamento de cada um destes usuários. Somente o usuário com perfil *coordenador* tem acesso a essa página, pois é uma seção de administração do sistema. Ela também permite editar informações de qualquer usuário e deletar ele do sistema conforme mostrado no centro e à direita da Figura 4.2.

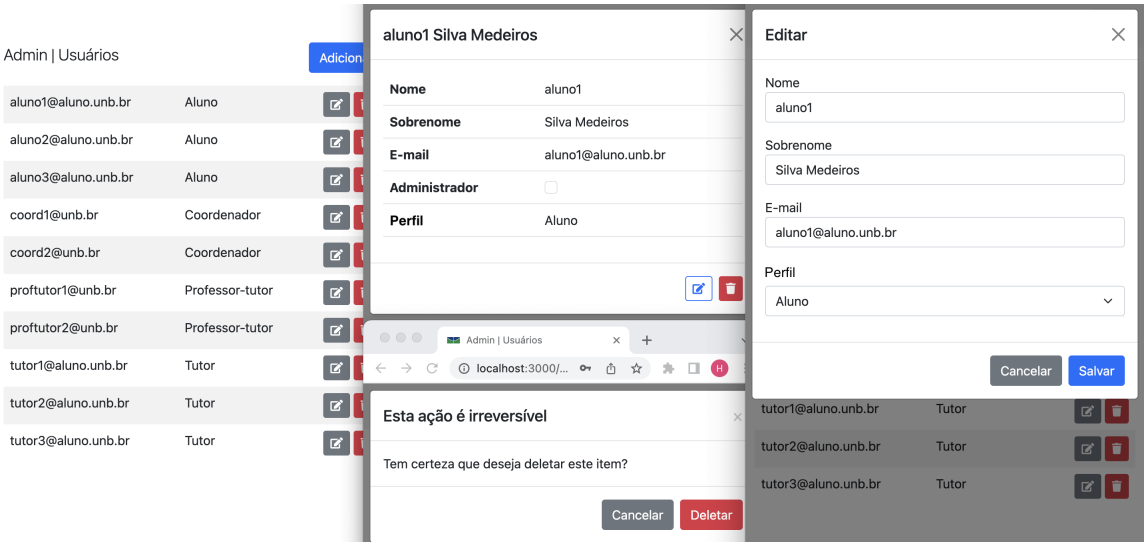

Figura 4.2: Página de administração de usuários.

O usuário com perfil *coordenador* também pode inserir um novo usuário no sistema ao clicar em adicionar e preencher o formulário do modal que é aberto. Neste caso, no

entanto, a senha deverá ser passada pessoalmente para a pessoa que utilizará a conta. Caso queira, ela poderá fazer a recuperação de senha em seguida ou posteriormente.

Para fins de teste do sistema, 10 usuários fictícios são criados para analisar os recursos conforme as permissões de acesso para cada tipo de usuário. Além disso, os relacionamentos entre usuários também são testados com esses casos de uso. Por exemplo, descrever quem é o *tutor* e quem é o *professor-tutor* de um determinado aluno. Veja os usuários na Figura 4.3.

Nome	E-mail	Perfil	Tutor	Professor-tutor	Ações
aluno1 Silva	aluno1@aluno.unb.br	Aluno	tutor1 Alves	proftutor1 Gomes	
aluno2 Santos	aluno2@aluno.unb.br	Aluno	tutor2 Pereira	proftutor2 Costa	
aluno3 Oliveira	aluno3@aluno.unb.br	Aluno	tutor3 Lima		
coord1 Martins	coord1@unb.br	Coordenador			
coord2 Carvalho	coord2@unb.br	Coordenador			
proftutor1 Gomes	proftutor1@unb.br	Professor-tutor			
proftutor2 Costa	proftutor2@unb.br	Professor-tutor			
tutor1 Alves	tutor1@aluno.unb.br	Tutor			
tutor2 Pereira	tutor2@aluno.unb.br	Tutor			
tutor3 Lima	tutor3@aluno.unb.br	Tutor			

Figura 4.3: Usuários fictícios para teste do sistema.

Para facilitar o entendimento, o nome de cada usuário foi cadastrado como sendo o seu tipo de perfil com um número em seguida. Já o sobrenome foi cadastrado com algum sobrenome dos mais comuns no país para tornar mais real em contraponto com o nome que não sugere uma situação realística. Além disso, o número colocado em seguida do nome está de acordo com a relação do aluno com o tutor e o professor. Por exemplo o *aluno* "aluno1" tem como *tutor* o "tutor1" e como *professor-tutor* o "proftutor1". *aluno* "aluno2" tem como *tutor* o "tutor2" e como *professor-tutor* o "proftutor2".

Para fazer o relacionamento de usuário *aluno* com *tutor* e *professor-tutor*, um usuário *coordenador* deve acessar o modal de edição de usuários e selecionar a opção desejada no menu suspenso conforme mostrado na Figura 4.4. Observe que o menu suspenso só aparece no caso do usuário a ser modificado ser um *aluno*. Além disso, o *tutor* e/ou o *professor-tutor* para aparecer no menu suspenso já devem estar matriculados na aplicação e cadastrados corretamente com seus devidos perfis no sistema.

A Figura 4.5 apresenta a página de perfil do usuário para usuários com diferentes perfis. Cada um dos quatro quadrantes da figura representa uma sessão de usuário aberta em um navegador diferente ou no mesmo navegador em modo anônimo. Veja que diferentes tipos de perfil mostram diferentes registros. O *coordenador* mostra apenas os campos "nome",

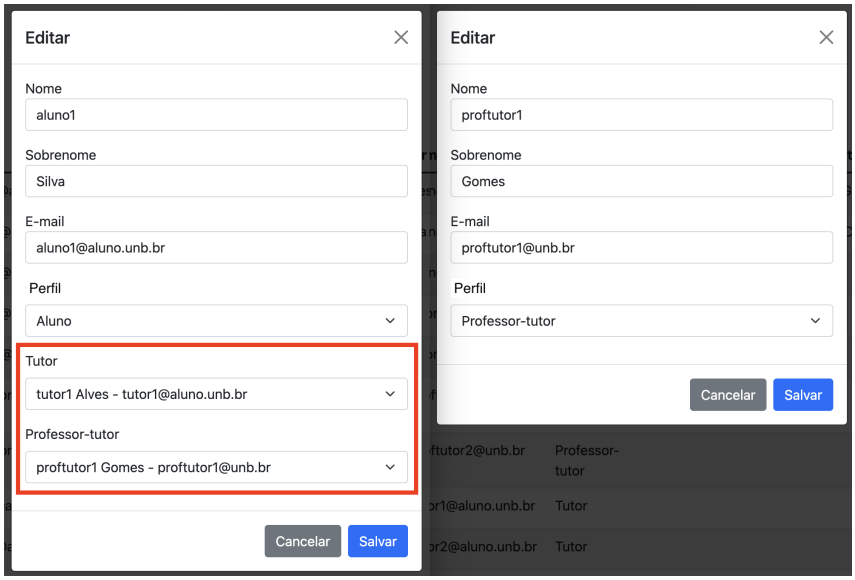

Figura 4.4: Modal de edição de usuários.

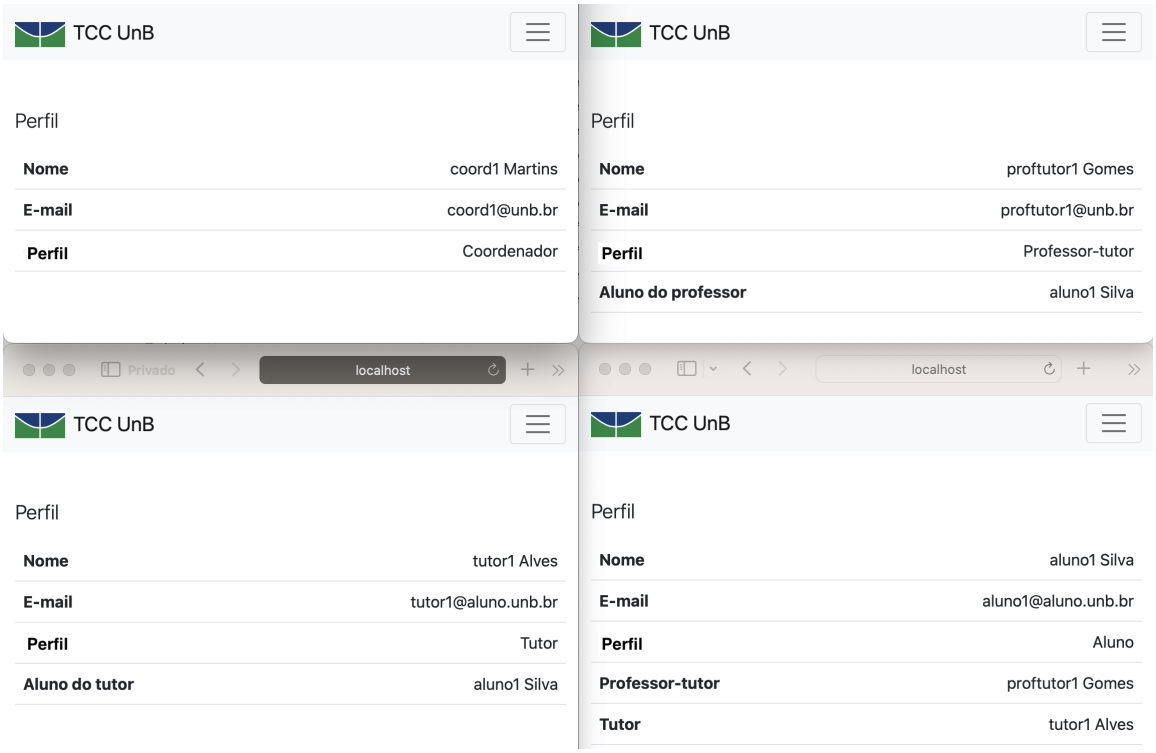

Figura 4.5: Página de perfil para diferentes usuários.

"e-mail" e "perfil", que são os campos comuns entre todos os usuários. O *professor-tutor*, além destes três campos mostra o campo "Aluno do professor", no qual ele deve auxiliar. Para o *tutor* a mesma coisa se aplica com o campo "Aluno do tutor". Por fim, o usuário *aluno* apresenta dois campos a mais, que são o "Professor-tutor" e o "Tutor".

Orientações gerais

A página de orientações gerais mostrada na Figura 4.6 permite que todos os usuários obtenham informações de ajuda da aplicação e dos recursos oferecidos pela instituição. As informações inseridas neste campo são de escolha dos usuários que podem editar e é a mesma informação que aparece para todos. A página não fornece tratamento de disputa no caso de dois usuários entrarem em desacordo quanto ao conteúdo da página.

Esta é uma das três páginas que ficam visíveis para usuários não logados no sistema. Então toda informação disponibilizada nela deve ser pública e seus editores devem tomar cuidado com as informações que inserir, pois qualquer pessoa que acessar o site poderá acessar esta página.

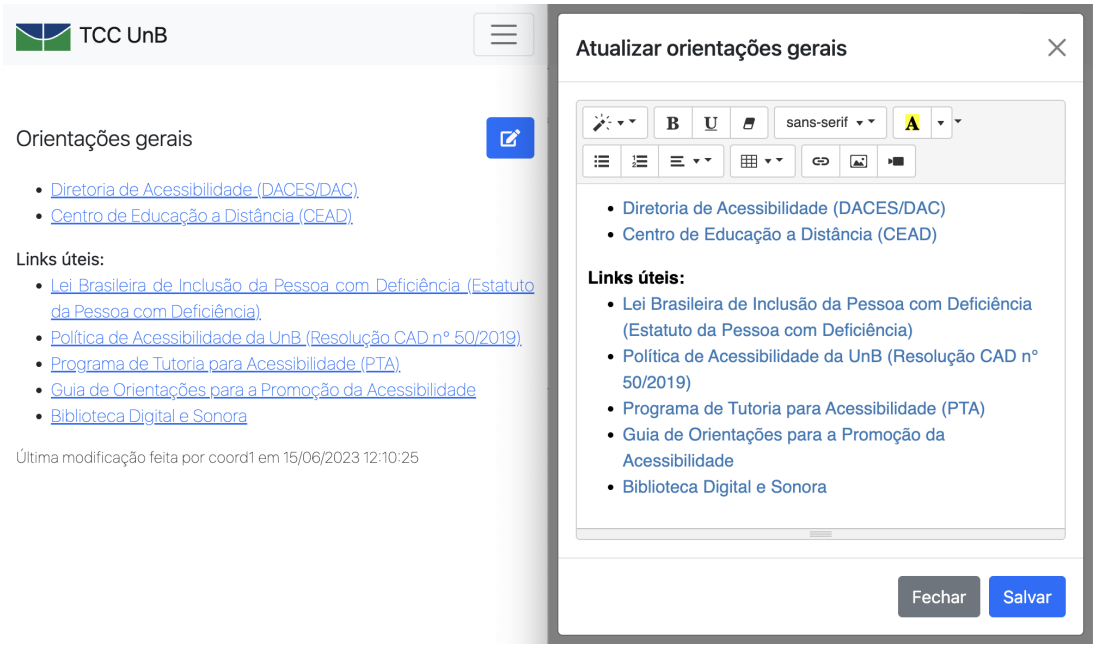

Figura 4.6: Página de orientações gerais.

Os usuários públicos, o *aluno* e o *tutor* apenas podem visualizar a página, mas não editar seu conteúdo. Por isso o ícone de edição mostrado na Figura 4.7 (botão azul com lápis) não aparece para eles. Já os usuários *professor-tutor* e *coordenador*, além de visualizar eles podem editar o conteúdo utilizando um editor de texto *Summernote*. Isto permite que sejam inseridos recursos como imagens, links, diferentes tipos de fontes de forma a fornecer uma gama de recursos para facilitar a transmissão de informação entre usuários.

Outro recurso disponibilizado na página de orientações gerais é apresentado na parte inferior da página conforme mostrado na Figura 4.6 ou na Figura 4.7. Ele mostra um texto com informações da última modificação feita no conteúdo da página. Primeiro o

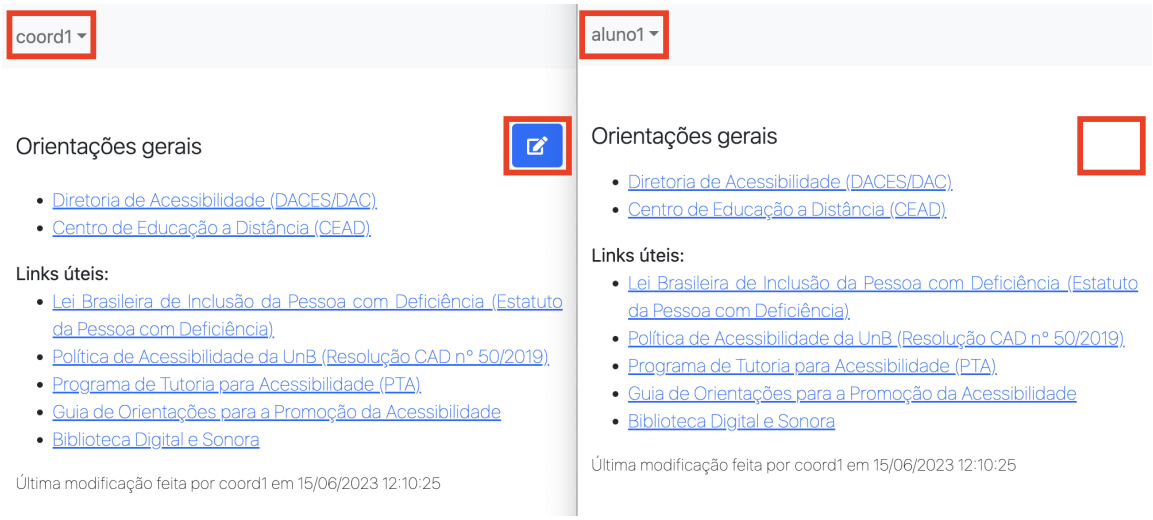

Figura 4.7: Permissões da página de orientações gerais.

nome do usuário que fez a alteração e, em seguida a data e hora da última modificação. Estas informações são atualizadas automaticamente ao enviar o formulário de edição.

Acompanhamento

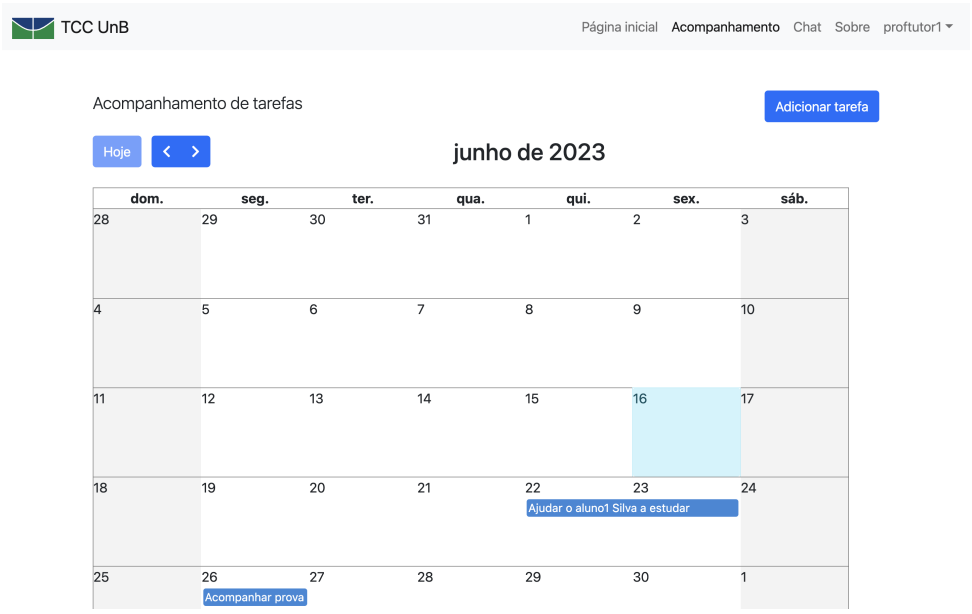

Figura 4.8: Visão geral da página de acompanhamento.

A página de acompanhamento apresenta um calendário de eventos ou tarefas (componente JavaScript *FullCalendar*) na qual cada usuário pode criar novas tarefas e atribuir ou direcionar a outros usuários ou a ele próprio. A Figura 4.8 mostra uma visão geral da pá-

gina de acompanhamento em que é mostrada tarefas relacionadas ao usuário “proftutor1” (usuário atual da sessão no navegador).

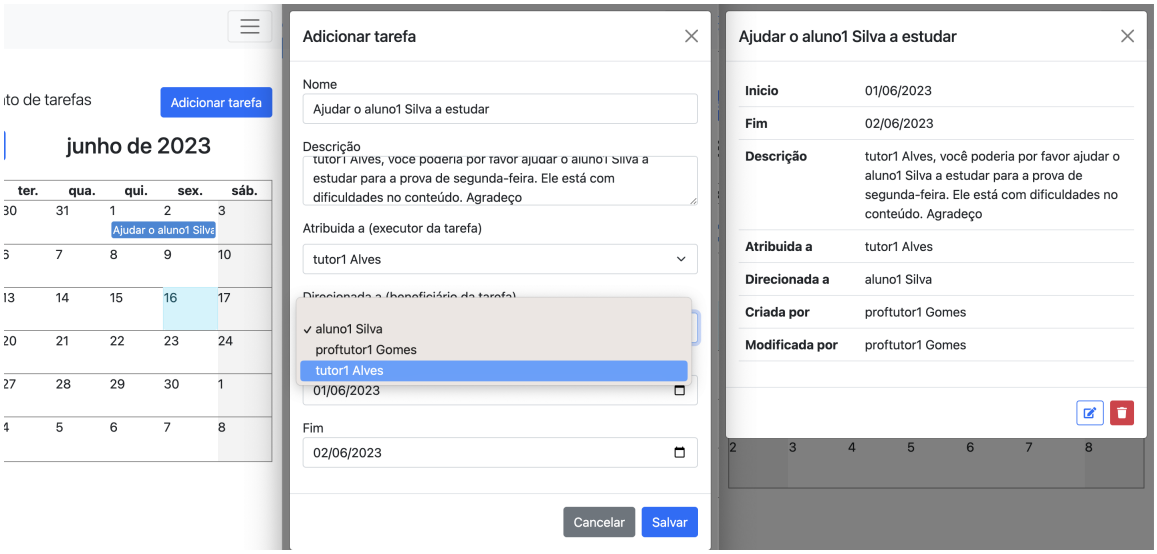

Figura 4.9: Criação de uma nova tarefa.

Cada tarefa tem um campo “Atribuída a” e outro campo “Direcionada a”. O primeiro se refere ao executor da tarefa, ou seja, o usuário que deve realizar alguma ação. O segundo se refere ao beneficiário da tarefa, ou seja, o usuário que deve receber o auxílio; normalmente um *aluno*, mas pode ser também qualquer outro como *tutor* ou *professor-tutor*. Ao centro da Figura 4.9 mostra um formulário de criação de uma nova tarefa. Veja que o criador da tarefa só pode atribuir e direcionar a tarefa a ele próprio ou a quem tem algum grau de relacionamento consigo no sistema. Ou seja, a seu *tutor* ou seu *professor-tutor* no caso do *aluno*; ou ao seu *aluno* ou ao *tutor* do seu *aluno* no caso do *professor-tutor*; ou ao seu *aluno* ou ao *professor-tutor* do seu *aluno* no caso do *tutor*. Uma exceção ocorre quando o usuário tem perfil *coordenador*, em que ele pode direcionar ou atribuir a tarefa a qualquer usuário do sistema. Ao enviar o formulário de criação de uma nova tarefa, o campo “Criada por” é atribuído ao usuário atual.

À direita da Figura 4.9 mostra um modal de detalhamento da tarefa em que os quatro últimos atributos são relacionamentos da tarefa com diferente usuários do sistema. A Figura 4.10 mostra o ponto de vista do calendário para três usuários diferentes. As tarefas listadas no calendário são apenas tarefas com algum grau de relacionamento com o usuário atual. Ou seja, são apresentadas somente se a tarefa for “Atribuída a”, ou “Direcionada a”, ou “Criada por” ou “Modificada por” o usuário da sessão. Novamente uma exceção ocorre quando o usuário tem perfil *coordenador*, em que ele tem a visualização de todas as tarefas do sistema. À esquerda da Figura 4.10 mostra a perspectiva do *coordenador*

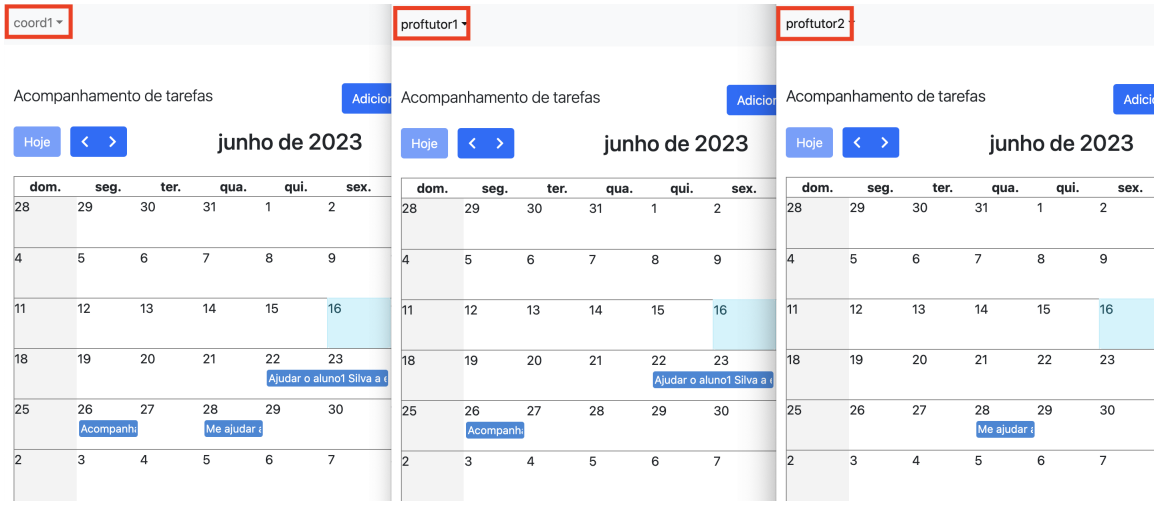

Figura 4.10: Filtro de tarefas pelo usuário.

em que as tarefas, tanto do "proftutor1" (ao centro) quanto do "proftutor2" (à direita) são listadas.

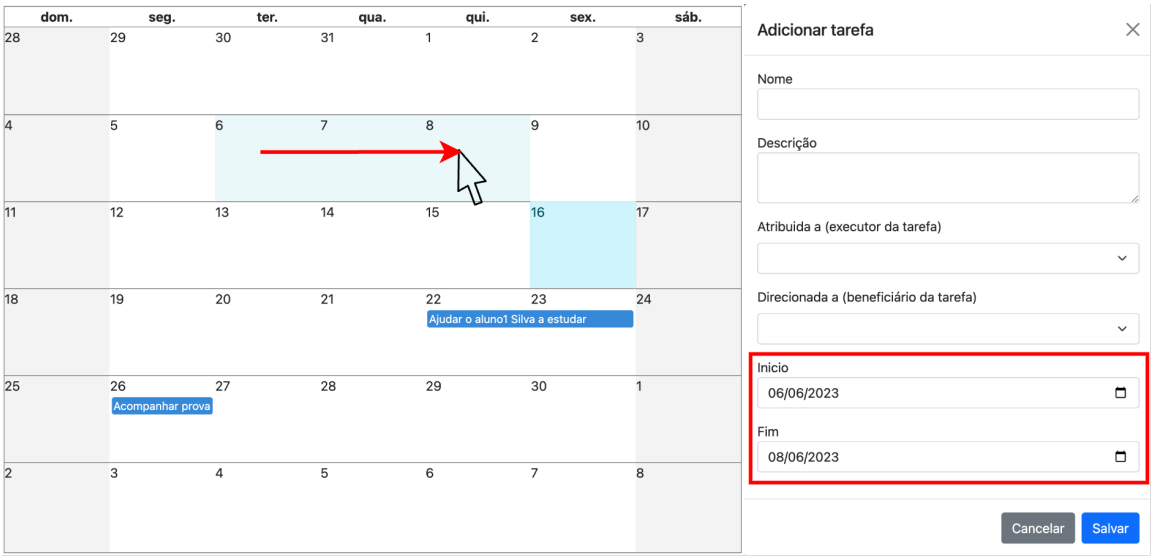

Figura 4.11: Selecionar datas pelo calendário para adicionar nova tarefa.

O calendário apresenta ainda o recurso de selecionar datas pelo calendário para adicionar uma nova tarefa conforme ilustrado à esquerda da Figura 4.11. Para isto basta clicar em uma data para criar um evento de um dia ou clicar em uma data inicial e arrastar para uma data final para criar um evento de mais de um dia. A data selecionada fica com um fundo destacado da cor do tema da aplicação (azul, neste caso). Em seguida, ao clicar no botão "Adicionar tarefa", o modal se abre com as datas pré-selecionadas, facilitando a usabilidade do usuário.

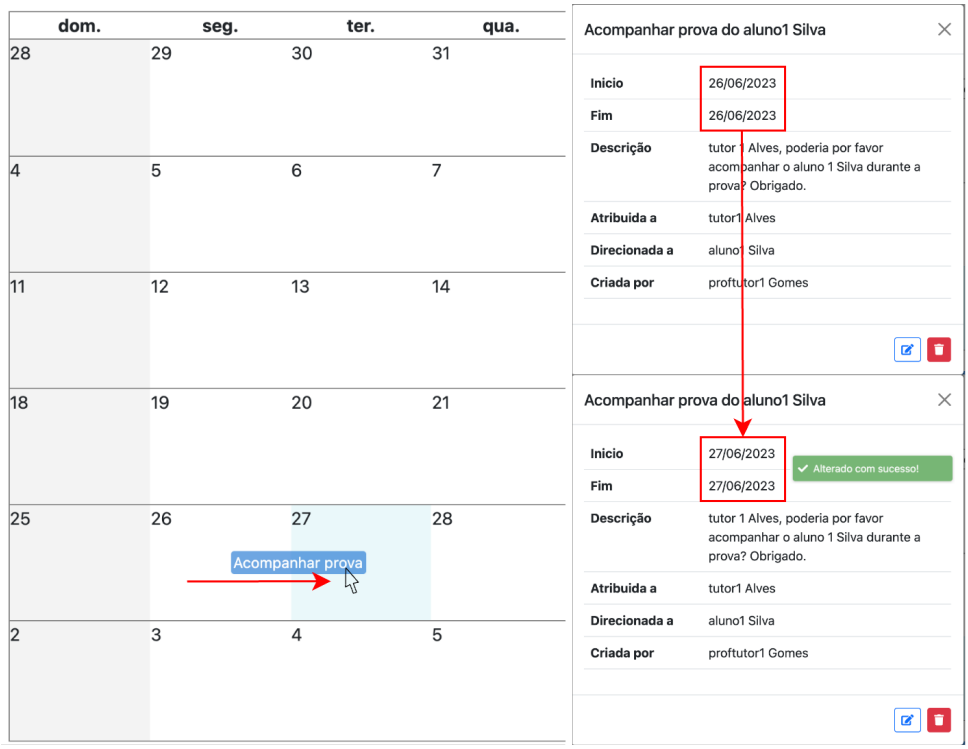

Figura 4.12: Editar data da tarefa arrastando pelo calendário.

Depois que a tarefa já foi criada, sua data também pode ser modificada diretamente pelo calendário clicando na tarefa e arrastando para a nova data desejada conforme ilustrado à esquerda da Figura 4.12. Isto é válido tanto para tarefas de um dia quanto para tarefas de múltiplos dias. Ao fazer a modificação um alerta de sucesso é apresentado, assim como ocorre no envio de formulários. À direita da Figura 4.12 mostra uma ilustração deste alerta e a mudança de data no modal de detalhamento da tarefa. Este recurso agiliza a ação do usuário de forma a melhorar a usabilidade da aplicação economizando trabalho por meio de atalhos. Esta mesma ação também pode ser executada por meio do modal de edição, clicando no ícone de lápis abaixo do modal de detalhamento.

A Figura 4.13 mostra o exemplo de uma edição e exclusão de uma tarefa a partir da pagina de detalhamento. No canto inferior direito mostra dois botões, um com o ícone de lápis que ao clicar abre o modal mostrado à esquerda da figura, e o outro com um ícone de lixeira que abre o modal mostrado no canto superior direito da figura. O modal para edição apresenta os mesmos campos do modal de criação da tarefa apresentado anteriormente inclusive a limitação dos usuários nos menus suspensos “Atribuída a” e “Direcionada a”. Porém, ao enviar o formulário, o campo “Modificada por” é alterado para o usuário atual, enquanto no formulário de criação é modificado apenas o campo “Criada por”. Já o modal para exclusão é uma forma de evitar que usuário cometa erros

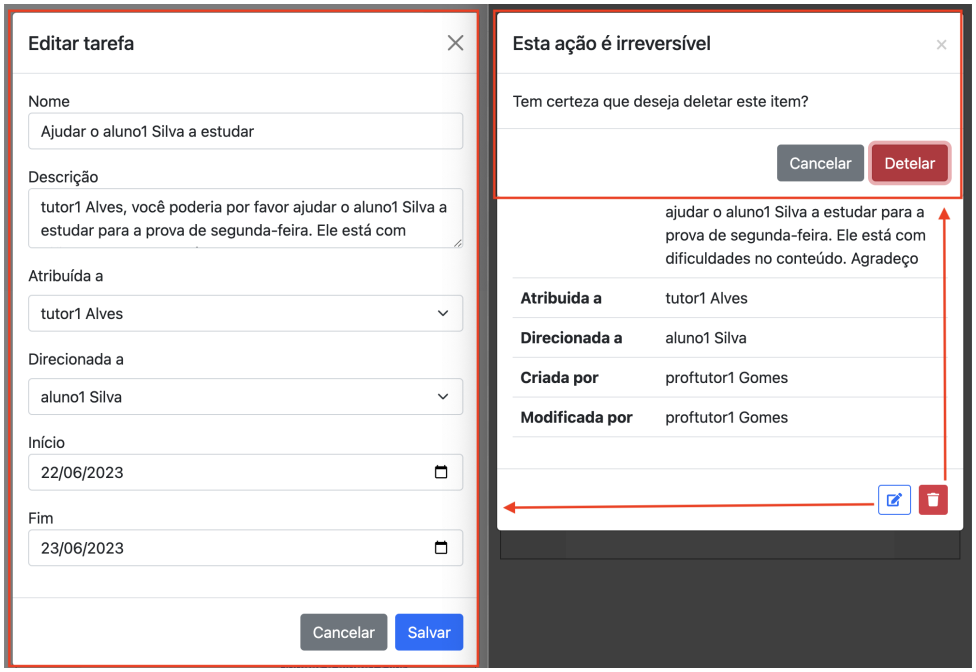

Figura 4.13: Detalhamento, edição e exclusão de tarefa.

na utilização do sistema e possa voltar atrás na ação de exclusão.

Comunicação

A página de comunicação visa ser um acesso rápido para esclarecer e tirar dúvidas sobre as tarefas criadas no calendário da página de acompanhamento. A Figura 4.14 mostra uma visualização geral da página do ponto de vista do usuário "proftutor1". Ao fundo é possível ver uma lista de usuários na qual o usuário atual pode conversar. Ao selecionar um item da lista, o modal com as mensagens é mostrado. A Figura 4.14 mostra um exemplo de mensagem com usuário "tutor Alves". No canto superior direito aparece um alerta quando o usuário se conectou no chat. Ele representa a conexão do usuário no *websocket*, recurso que permite a atualização da página assincronamente.

A Figura 4.15 apresenta a mesma conversa da Figura 4.14, porém mostrando as diferentes visualizações para cada usuário participante da conversa. Nela, duas conversas paralelas entre três usuários são representadas. A primeira entre o "tutor1 Alves" e o "proftutor1 Gomes" e a segunda entre o "tutor1 Alves" e o "aluno1 Silva". Este exemplo ilustra uma simulação de um possível caso real de uso do sistema. É uma conversa decorrente da tarefa criada na Figura 4.9 onde o "proftutor1 Gomes" atribui uma tarefa ao "tutor1 Alves", que teve dúvidas e foi tirar com o professor e ele o recomendou que conversasse com o aluno diretamente.

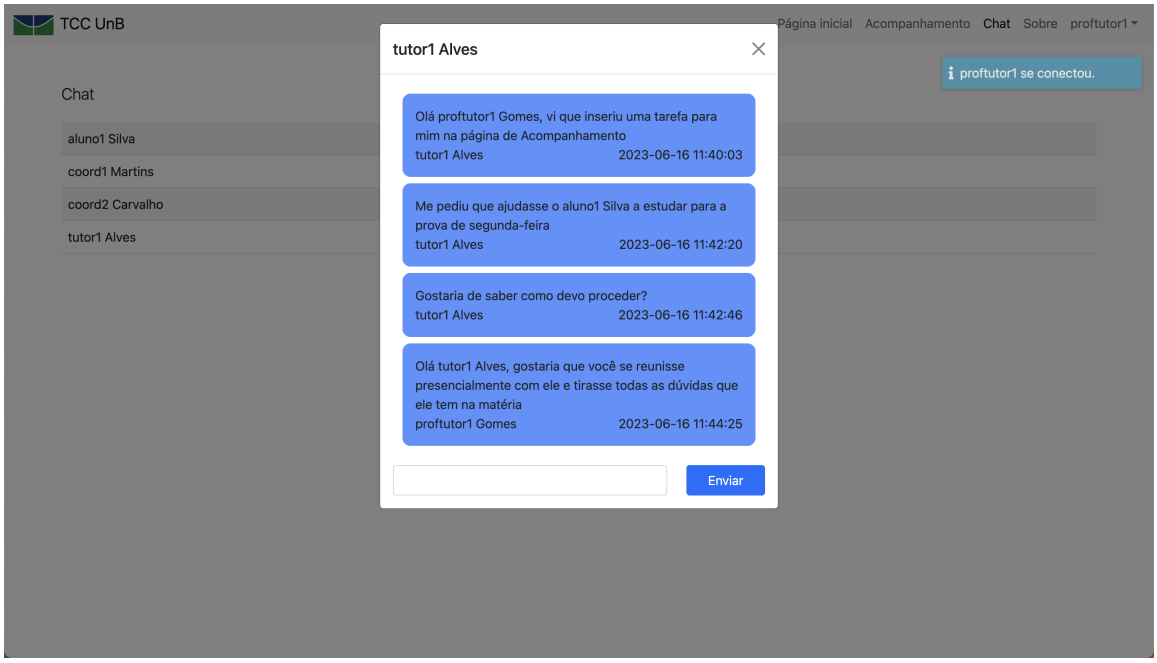

Figura 4.14: Visão geral da página de comunicação.

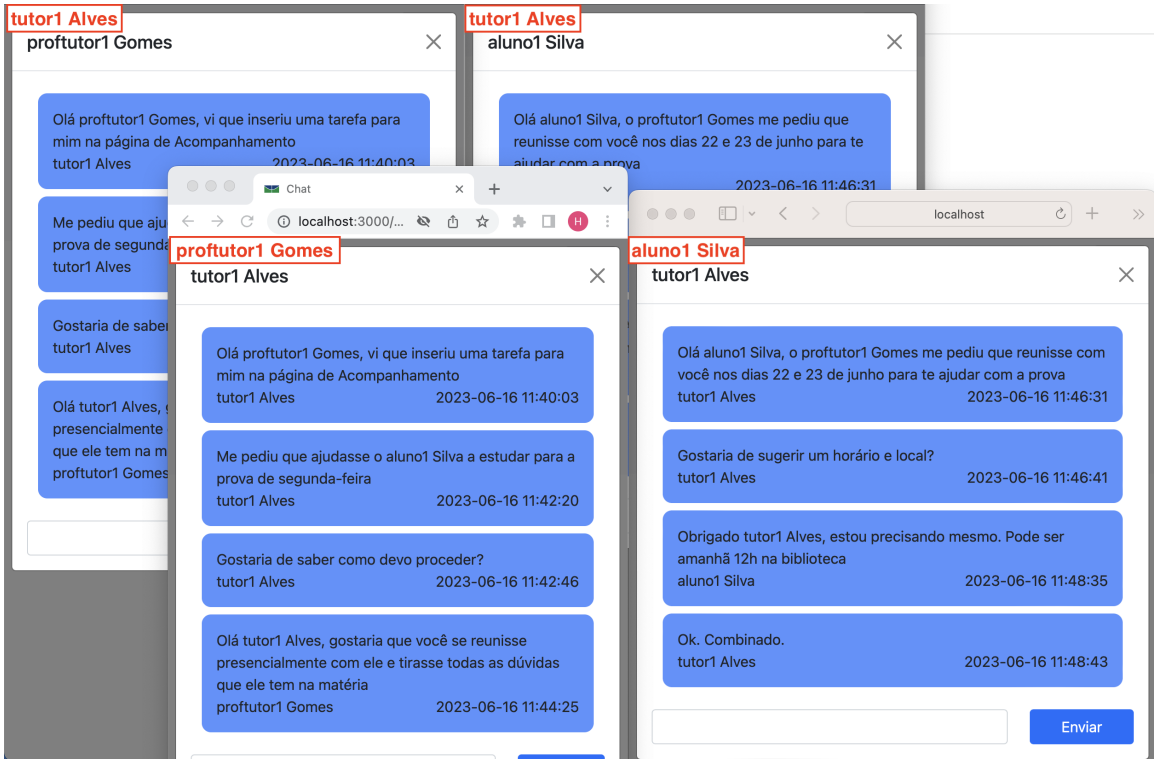

Figura 4.15: Exemplo de conversa no chat.

Da mesma forma que as tarefas são listadas apenas se tem alguma relação com o usuário, o chat só pode ser estabelecido entre usuários relacionados. Veja na Figura 4.16 a

listagem de usuários que cada usuário pode se comunicar. Observe que o perfil *coordenador* pode se comunicar com todos. Além disso, o *aluno* pode se comunicar com seu *tutor* ou seu *professor-tutor*; o *professor-tutor* pode se comunicar com o seu *aluno* ou o *tutor* do seu *aluno*; e o *tutor* pode se comunicar com o seu *aluno* ou com o *professor-tutor* do seu *aluno*.

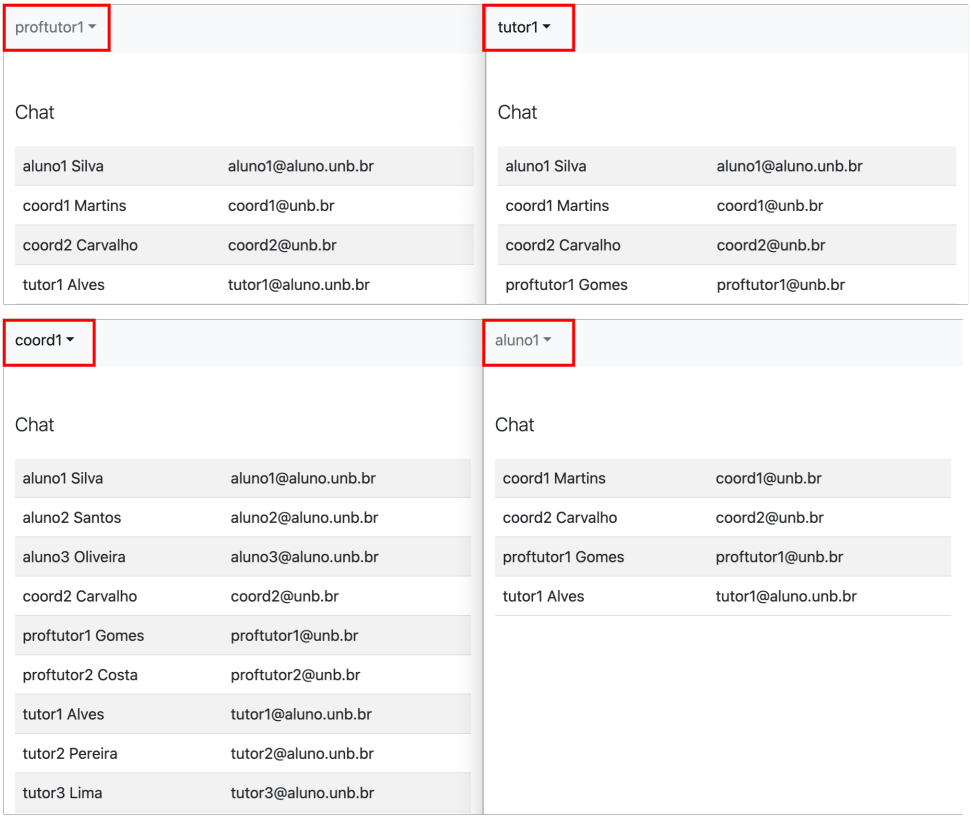

Figura 4.16: Listagem de usuários para comunicação pelo chat.

Um recurso de usabilidade importante que foi utilizado no desenvolvimento do chat é a atualização em tempo real das mensagens se ambos os usuários estiverem conectados na sessão do navegador e com a aba de comunicação aberta. Sempre que um primeiro usuário manda mensagem para um segundo, a mensagem aparece para este segundo sem que ele tenha que recarregar a página e fazer uma nova solicitação de todas a lista de conversas para o servidor.

Outras páginas

- **Sobre**

  A página "sobre" mostrada na Figura 4.17 é uma das três páginas disponibilizadas publicamente, para usuários não logados na plataforma. Ela é uma área de meta-

informações do site que descreve seus objetivos e detalha sua origem em um trabalho de conclusão de curso. Dois botões estão disponíveis na página: o primeiro com um link para o projeto hospedado no sistema de controle de versões GitHub e o segundo com um link para o arquivo da monografia referente ao projeto.

Figura 4.17: Página sobre.

- **Entrar**

  A página "entrar" mostrada na Figura 4.18 é uma das três páginas disponibilizadas publicamente é é responsável por fornecer uma interface para ações de autenticação dos usuários. Nela três abas são apresentadas: "Entrar", "Cadastrar" e "Redefinir senha", sendo a primeira aberta como padrão no carregamento da página.

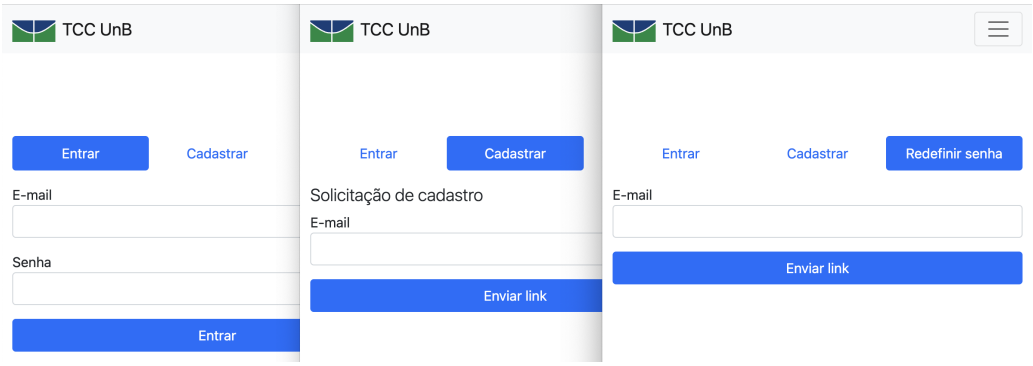

Figura 4.18: Página entrar.

A aba “Entrar” mostra um formulário com os campos de e-mail e senha com as devidas validações de envio. Entre elas a verificação de e-mail válido, a já existência do e-mail no banco de dados, a verificação de domínios de e-mail permitidos (somente @unb.br e @aluno.unb.br) e o preenchimento obrigatório dos campos. A aba “Cadastrar” apresenta um campo de e-mail para preencher e segue o processo detalhado na Figura 5.2 para o cadastramento do usuário no sistema. E por último a “Redefinir senha” também apresenta um campo de e-mail e para preencher e segue o processo detalhado na Figura 5.3.

- **Cadastrar**

  A página “cadastrar” mostrada na Figura 4.19 é disponibilizada publicamente, porém apresenta um formulário que é inutilizável se não tiver um código de token válido disponibilizado como parâmetro na URL (o *code*). Esta página deve ser acessada por meio de um link enviado por e-mail ao solicitar o cadastro no sistema pelo formulário de cadastro da Figura 4.18. Ela é responsável por um dos processos detalhados na Figura 5.2. A Figura 4.19 destaca em vermelho os dois parâmetros da URL que devem estar preenchidos. Além disso, todos os campos seguem as devidas regras de validação de e-mail e senha.

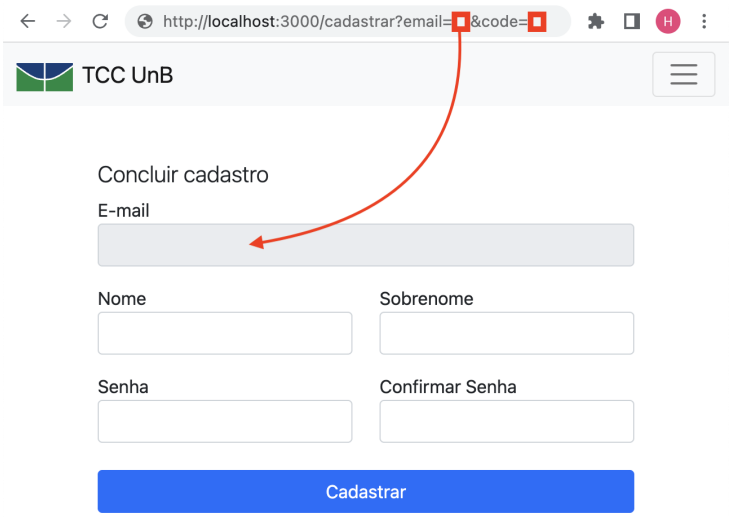

Figura 4.19: Página cadastrar.

- **Redefinir senha**

  A página “Redefinir senha” mostrada na Figura 4.20 é disponibilizada publicamente, porém apresenta um formulário que é inutilizável se não tiver um código de token válido disponibilizado como parâmetro na URL (o *id*). Esta página deve ser acessada por meio de um link enviado por e-mail ao solicitar a redefinição de senha no

sistema pelo formulário da última aba na Figura 4.18. Ela é responsável por um dos processos detalhados na Figura 5.3. A Figura 4.20 destaca em vermelho os dois parâmetros da URL que devem estar preenchidos. Além disso, todos os campos de senha seguem as devidas regras de validação, como as senhas que devem ser iguais.

Figura 4.20: Página redefinir senha.

Capítulo 5

Avaliação de usabilidade

Neste capítulo, a análise de usabilidade é feita utilizando diferentes métodos de avaliação, separados nas seções de *teste*, *inspeção* e *consulta*. Nesta etapa, a análise é realizada para cada uma das páginas do sistema separadamente. Por fim, é dado destaque em uma seção exclusiva para a *avaliação heurística* (tipo de inspeção), onde todo o sistema é analisado em conjunto para cada uma das heurísticas de Nielsen, uma das mais difundidas conforme detalhado no referencial teórico.

5.1 Teste de usabilidade

O método de *coaching* foi utilizado durante as reuniões onde o usuário testador (professor orientador) tinha acesso diretamente a um usuário conhecedor do sistema (o desenvolvedor). Portanto o testador tinha acesso a um especialista para ensinar durante a utilização.

Por ser uma aplicação pouco orientada a dados, os métodos quantitativos como medição de desempenho, teste remoto, teste retrospectivo e método de sombreamento foram ignorados. Além disso, também foram ignorados os métodos que dependem de mais de um usuário desconhecedor da aplicação. São eles o método de ensino, a aprendizagem de co-descoberta e o protocolo pensando em voz alta.

5.2 Inspeção de usabilidade

Os vários métodos de inspeção de usabilidade são utilizados em conjunto. Desta forma, não é possível identificar uma demarcação clara entre um método e outro. Como algumas análises dos métodos se sobrepõem, esta abordagem evitará repetição e redundância. Por exemplo, durante o passo a passo cognitivo *(Cognitive Walkthroughs)*, já são analisadas as heurísticas de Nielsen. Além disso, foi considerada que a visão geral do sistema detalhada

anteriormente no capítulo 4, já funciona como uma etapa de inspeção de recursos *(Feature Inspection)*.

A ideia geral por trás da avaliação heurística é que vários avaliadores avaliam independentemente um sistema para encontrar possíveis problemas de usabilidade. É importante que haja vários desses avaliadores e que as avaliações sejam feitas de forma independente. A experiência de Nielsen indica que cerca de 5 avaliadores geralmente resultam na descoberta de cerca de 75% dos problemas gerais de usabilidade. Esta avaliação, no entanto, é feita apenas por um avaliador. Portanto está sujeita à presença de viés do próprio desenvolvedor. Sendo assim, o método de avaliação passo a passo pluralístico *(Pluralistic Walkthroughs)*, não é executado neste trabalho, pois requer um grupo de usuários, desenvolvedores e engenheiros de fatores humanos reunidos, o que não é o propósito atual deste projeto.

Entrar

A Figura 5.1 ilustra o passo a passo de um caso de sucesso no processo de entrar no sistema. Na avaliação ilustrada na imagem um e-mail já cadastrado é digitado com sua senha correta o usuário é redirecionado para a página inicial, de orientações gerais. Além do caminho de sucesso percorrido, outros caminhos de falha também foram avaliados. Entre eles o de e-mail não cadastrado no sistema ou senha incorreta.

Falhas de usabilidade detectadas:

- Ao digitar um e-mail não cadastrado o login não é realizado, porém nenhum alerta é mostrado para o usuário;
- Ao digitar um e-mail válido com uma senha incorreta o login não é realizado, porém nenhum alerta é mostrado para o usuário.

Cadastrar

A Figura 5.2 ilustra o passo a passo de um caso de sucesso no processo de entrar no sistema. Na avaliação ilustrada na imagem um e-mail válido é inserido no primeiro formulário da página "Entrar" (aba "Cadastrar") e, desta forma, um e-mail é enviado. O processo de validação do e-mail com uma *honeypot*[1] evita que sejam enviados *spam* que sobrecarregue o servidor de email.

Ao verificar o e-mail (centro da Figura 5.2) um *link* envia o usuário para a página de cadastro e o campo de e-mail já é preenchido com o valor carregado no formulário anterior. Este processo foi detalhado anteriormente na Figura 4.19. Em seguida, o cadastro só é

[1] *Honeypot* é um mecanismo de segurança de computador definido para detectar, desviar ou, de alguma forma, neutralizar tentativas de uso não autorizado de sistemas de informação.[116]

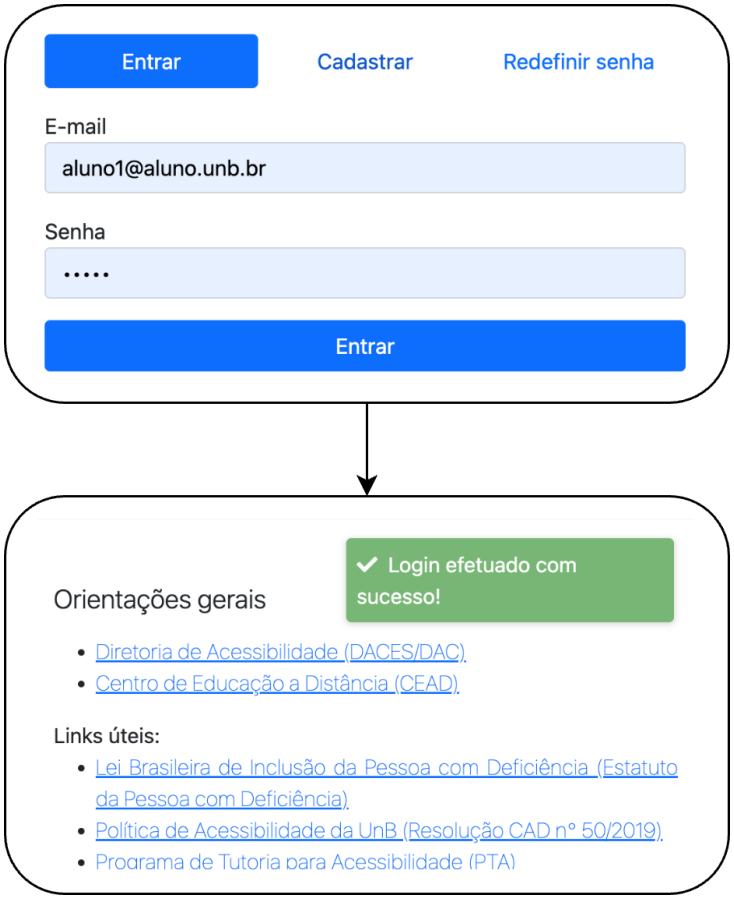

Figura 5.1: Entrar.

inserido no banco de dados caso o e-mail o o token disponibilizado no parâmetro da URL sejam válidos.

Falhas de usabilidade detectadas:

- Alguns serviços de e-mail apresentam falha ao clicar no *link*. Desta forma é adequado disponibilizar também no corpo do e-mail um *link* para copiar;
- Quando as senhas não correspondem a verificação acontece corretamente e o alerta no campo também. Porém aparece um alerta de erro sem qualquer descrição na parte superior da página;
- Qualquer senha pode ser utilizada desde que não seja vazia. Isto não impede que o usuário cadastre uma senha insegura.

Recuperação de senha

A Figura 5.3 ilustra o passo a passo de um caso de sucesso no processo de recuperação de senha. Primeiramente um formulário com um único campo de e-mail deve ser preenchido.

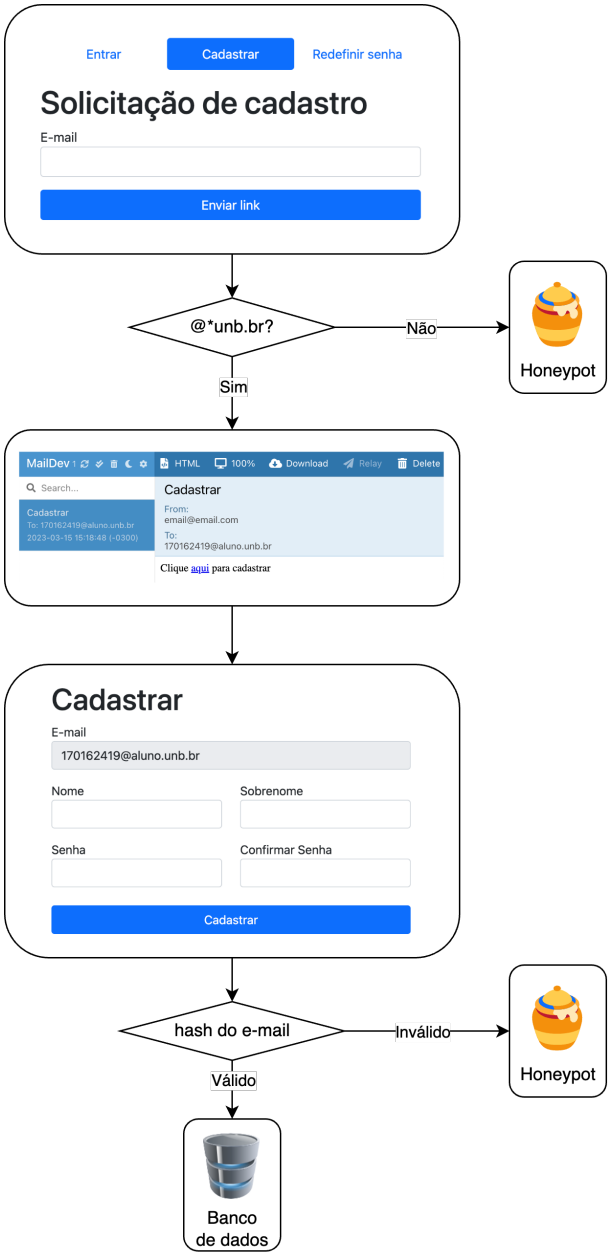

Figura 5.2: Cadastro.

Ele verifica se o e-mail está cadastrado e se não tiver é mostrado um alerta no campo. Caso esteja cadastrado, um *link* é enviado para o e-mail. Em seguida, ao clicar no *link* a página de redefinir senha é aberta. Caso o usuário tente modificar qualquer valor do token, como mostrado na Figura 4.20 um alerta é mostrado e a mudança não é feita no banco de dados. Caso contrário a alteração é feita.

Falhas de usabilidade detectadas:

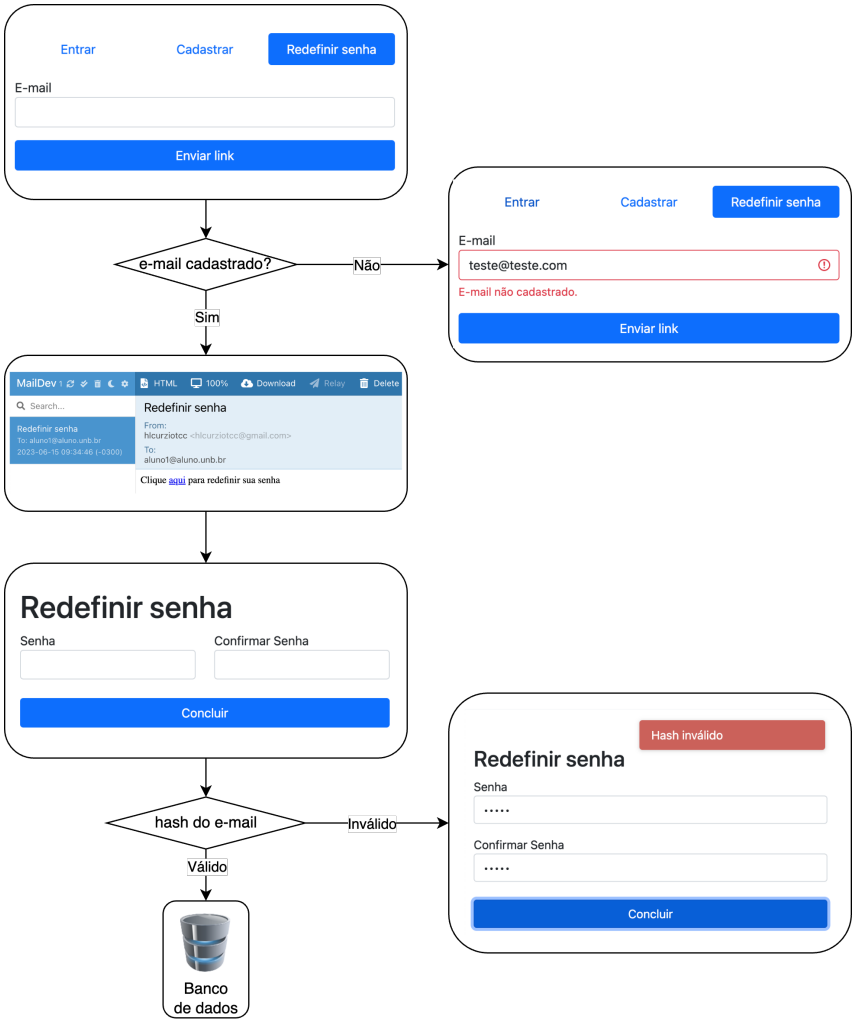

Figura 5.3: Recuperação de senha.

- Alguns serviços de e-mail apresentam falha ao clicar no *link*. Desta forma é adequado disponibilizar também no corpo do e-mail um *link* para copiar;
- Quando as senhas não correspondem a verificação acontece corretamente e o alerta no campo também. Porém aparece um alerta de erro sem qualquer descrição na parte superior da página;
- Qualquer senha pode ser utilizada desde que não seja vazia. Isto não impede que o usuário cadastre uma senha insegura;
- Ao concluir a redefinição o usuário é redirecionado para a página "entrar". Teria uma melhor usabilidade redirecionar o usuário para a página inicial como ocorre no cadastro do usuário.

Orientações gerais

Na página de orientações gerais mostrada na Figura 4.6, por ser extremamente simples, não foi detectado problemas de usabilidade, porém a própria simplicidade pode acabar aparentando uma falta de funcionalidade e propósito bem definido. Portanto seria adequado encontrar outras finalidades e novos recursos para a página de forma a promover maior importância e notoriedade por parte do usuário com o sistema.

Acompanhamento

A página de acompanhamento mostrada na Figura 4.8 com suas funcionalidades descritas anteriormente apresentou algumas falhas de funcionalidade que foram detectadas também durante a entrevista da consulta de usabilidade *(usability inquiry)* que é apresentada posteriormente.

Falhas de usabilidade detectadas:

- O aluno pode criar novas tarefas, mas há dúvidas se realmente seria adequado ele poder fazer isso;
- É possível criar novas tarefas anteriores à data atual, o que não faz sentido se a tarefa é uma atribuição futura.

Comunicação

A página de comunicação mostrada na Figura 4.14 com suas funcionalidades descritas anteriormente apresentou algumas falhas de funcionalidade que foram detectadas também durante a inspeção baseada em perspectiva *(Perspective-based Inspection)* mostrada na seção seguir e na entrevista da consulta de usabilidade *(usability inquiry)* que é apresentada posteriormente.

Falhas de usabilidade detectadas:

- Há uma baixa visibilidade e legibilidade das mensagens no chat devido à falta de contraste entre as cores do texto e do fundo escuro;
- Existe uma dificuldade em diferenciar as próprias mensagens e as mensagens recebidas, pois possuem as mesmas cores e interface.

Inspeção baseada em perspectiva *(Perspective-based Inspection)*

Adotando uma inspeção baseada em perspectiva com principal foco nas metas de usabilidade e experiência do usuário foi detectada uma falha de usabilidade na página de comunicação ao assumir o ponto de vista de uma pessoa com dificuldades de visão. Existe uma

dificuldade em diferenciar as próprias mensagens e as mensagens recebidas, pois possuem as mesmas cores e interface. Esta falha foi descrita durante o processo de inspeção da avaliação de usabilidade.

Responsividade

A responsividade do sistema foi desenvolvida utilizando os padrões de design do *framework* Bootstrap que fornece uma barra de navegação ajustável conforme o tamanho da tela como mostrado na Figura 5.4. Ao navegar pelos itens do menu, as páginas são acessadas por meio de Single Page Application (SPA) e o nome da páginas são atualizadas na aba dos navegadores.

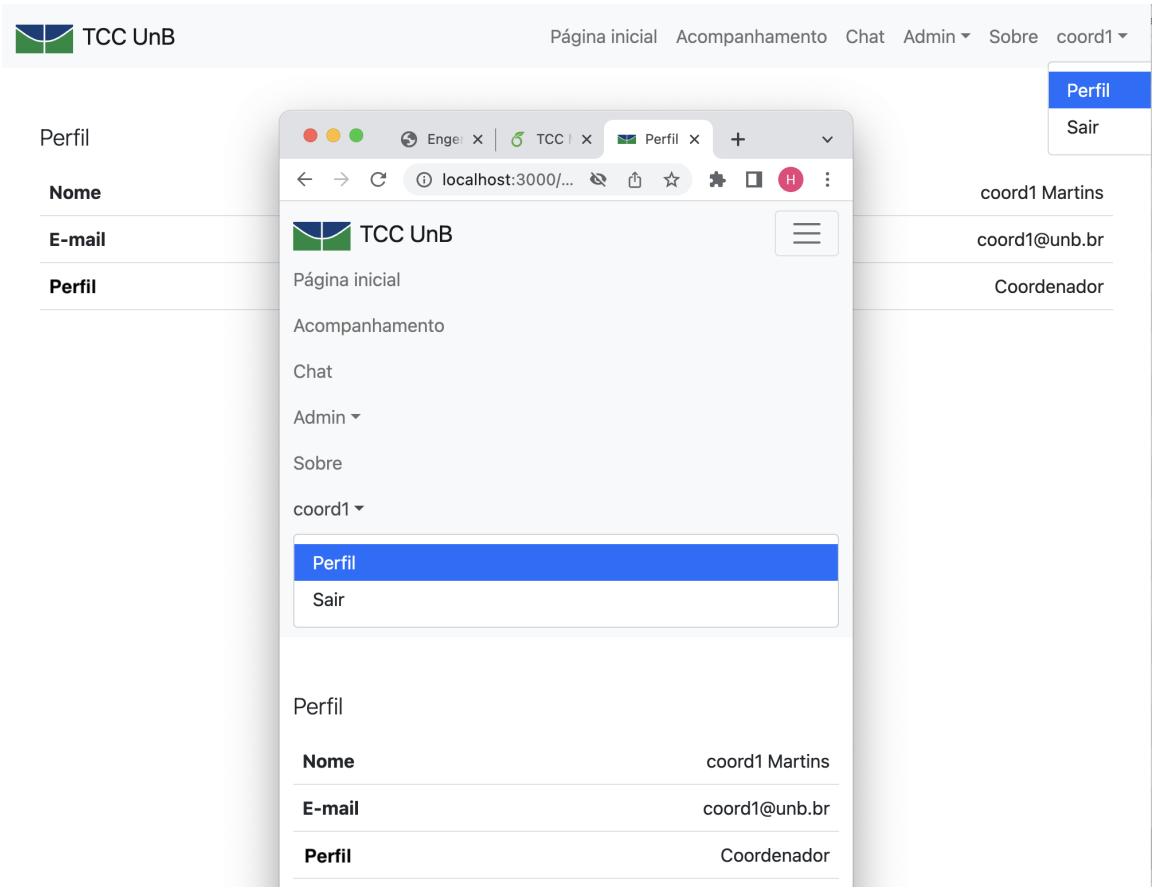

Figura 5.4: Responsividade.

Além da barra de navegação, outros recursos como tabelas, calendário e editores de texto também são responsivos permitindo o funcionamento tanto em dispositivos móveis como em aparelhos com telas maiores.

5.3 Consulta de usabilidade

No processo de consulta de usabilidade *(usability inquiry)* o avaliador de usabilidade deve obter informações diretamente no local físico e ter um relacionamento pessoal com os usuários. Assim seus métodos costumam estar presentes no momento inicial e nos momentos finais do processo de desenvolvimento de software.

Para realização desta etapa, foi identificado como os principais usuários do sistema e seus ambientes: o *aluno* e o *tutor* (fornecidos pela DACES); o *professor-tutor* e o *coordenador* do sistema (fornecidos pelo Departamento de Ciência da Computação (CIC)). Portanto a DACES e o CIC foram definidos como os ambientes de consulta de usabilidade para entender as características do usuário, o fluxo de trabalho, os recursos do sistema de que precisam, etc.

Para obter as principais necessidades dos usuários do CIC (com perfil *professor-tutor* e *coordenador* do sistema) foi considerado a comunicação com o próprio professor orientador do projeto, que faz parte deste departamento e é responsável por parte da gestão dos alunos com cadastro no programa de tutoria. Já para obter as principais necessidades dos usuários da DACES (*aluno* e *tutor*) este trabalho não detalha o contato direto com os usuários e considera um material já realizado anteriormente por uma equipe da DACES e do CEAD (o Guia de Acessibilidade) [16]. Este material lista as principais ações a serem realizadas para cada exigência do aluno conforme suas necessidades.

Estudo de campo proativo *(Proactive Field Study)*

Inicialmente, antes de começar o projeto do sistema, foi desenvolvido um estudo de campo proativo *(Proactive Field Study)* a partir do Guia de Acessibilidade [16]. Este guia disponibiliza orientações para 10 diferentes necessidades específicas. Neste projeto foi definido como foco as principais necessidades dos estudantes com Transtorno do Espectro Autista (TEA). Esta escolha foi feita para facilitar o desenvolvimento do sistema evitando inicialmente os requisitos, por exemplo, de leitores de tela para estudantes com deficiência visual e transcrição de áudio para estudantes com deficiência auditiva.

Ao final, as orientações foram classificadas em três categorias de requisitos. Para cada requisito foi considerado como solução uma página diferente no sistema. São elas, página de: orientações gerais, comunicação, e acompanhamento. Desta forma somente é necessário três perfis de usuários do sistema: o *aluno*, o *tutor* e o *professor-tutor*. Além dos requisitos definidos no estudo de campo proativo, existem os requisitos da própria aplicação que foram descritos nas seções anteriores. Entre eles está principalmente o gerenciamento de usuários e administração do sistema. Para isto foi definido também o usuário com perfil do tipo *coordenador*.

Entrevista *(Interview)*

Como o desenvolvimento deste projeto se deu apenas em ambiente de desenvolvimento e de teste, mas não se deu em ambiente de produção, a entrevista foi realizada por meio de várias reuniões com o professor orientador já que ele é um provável usuário com perfil *coordenador* ou *professor-tutor* do sistema. Durante as entrevistas, várias sugestões, conselhos e críticas foram abordados. As informações coletadas vão desde a escolha do tema, sugestões de documentação e estrutura do projeto, até o conteúdo presente em cada seção e página da aplicação desenvolvida.

Com relação à consulta de usabilidade, na entrevista que ocorreu após a definição do momento de parada no ciclo de desenvolvimento para análise do software foram destacados os seguintes aspectos, já discutidos na seção de inspeção:

- Na página de acompanhamento, é possível criar novas tarefas anteriores à data atual, o que não faz sentido se a tarefa é uma atribuição futura;
- Na página de comunicação, há uma baixa visibilidade e legibilidade das mensagens no chat devido à falta de contraste entre as cores do texto e do fundo escuro;
- Na página de comunicação, existe uma dificuldade em diferenciar as próprias mensagens e as mensagens recebidas, pois possuem as mesmas cores e interface.

Esta etapa de entrevista poderá também ser realizada em conjunto com outros métodos após uma possível implantação do software em produção com a presença de todos os tipos de usuários. Além disso, novas demandas podem surgir durante entrevistas com os mesmos usuários que já realizaram a entrevista antes, pois o desenvolvimento de *software* é um processo contínuo e os objetivos dos usuários mudam de acordo com o contexto e o momento.

5.4 Avaliação heurística

Na análise por inspeção feita anteriormente, em cada um dos critérios a seguir foram detectadas algumas falhas de usabilidade. O objetivo desta seção é listar casos de sucesso no desenvolvimento para cada uma das heurísticas de Nielsen. Considerando o desenvolvimento com foco na engenharia de usabilidade, estas heurísticas foram consideradas no processo de criação do sistema.

1. Visibilidade do status do sistema

   A Figura 5.5 ilustra vários alertas e *feedbacks* que mantêm os usuários informados do que está acontecendo.

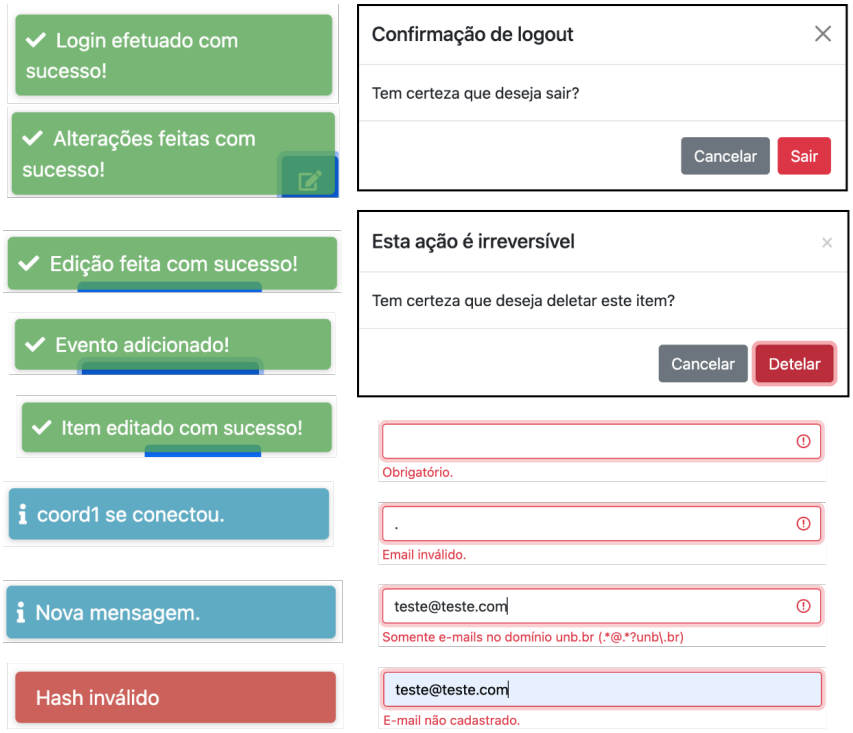

Figura 5.5: Visibilidade do status do sistema.

2. Correspondência entre o sistema e o mundo real:

   O sistema busca atingir esta correspondência em alguns locais como por exemplo nos ícones de lápis para edição e de lixeira para exclusão. Isto pode ser visualizado na Figura 4.7, na Figura 4.9 e na Figura 4.13.

3. Controle e liberdade do usuário:

   No canto superior direito da Figura 5.5 mostra alguns modais que permitem o usuário reverter a ação, caso tenha sido por engano. Outros casos são por exemplo nos botões de "x" acima dos modais e dos botões "cancelar" abaixo dos modais. Uma ilustração pode ser vista na Figura 4.13.

4. Consistência e padronização:

   Toda a aplicação foi desenvolvida utilizando o mesmo *framework* de *frontend* de forma a construir uma interface padronizada. Além disso, existe uma consistência entre a aparência de cada uma das diferentes páginas.

5. Prevenção de erros:

   A Figura 5.5 também ilustra alguns casos de alerta de erro na parte inferior. Esses alertas são decorrentes da detecção desses erros que ocorre anteriormente e utiliza o alerta para mostrar ao usuário.

6. Reconhecimento em vez de recordação:

   Todos os caminhos do sistema são acessíveis pela barra de menus sempre que este recurso está disponível para este usuário. Veja na parte superior da Figura 4.8.

7. Eficiência e flexibilidade de uso:

   Todas as páginas são navegáveis utilizando o recurso de Single Page Application (SPA) o que torna a navegação mais fluida. Além disso, no chat mostrado na Figura 4.14 e Figura 4.15 o recurso de *websocket* é utilizado para recarregar a página mais rapidamente.

8. Estética e design minimalista:

   Cada requisito da aplicação tem seu recurso alocado em uma única página exclusiva para ele. Isto torna a estética e navegação simples e com o mínimo de contaminação na página.

9. Ajude os usuários a reconhecer, diagnosticar e recuperar-se de erros:

   A aplicação oferece recurso de edição dos dados ao clicar em ícones de lápis como mostrado na Figura 4.7, na Figura 4.9 e na Figura 4.13. Isto permite que sejam recuperados os erros.

10. Ajuda e documentação:

    Logo na página inicial mostrada na Figura 4.7 o principal recurso apresentado é uma ferramenta de ajuda e descrição. Outra página "sobre" também serve para descrever o projeto. Estas duas páginas não requerem autenticação.

5.5 Ameaças a validade

Abaixo são listadas algumas ameaças à validade da avaliação de usabilidade. Estas ameaças incluem tanto viés do analisador quanto fragilidade a erros de avaliação devido à pequena amostra de analisadores.

- **Teste de usabilidade:** Durante o teste de usabilidade, vários métodos que o compõem dependem do desconhecimento inicial do sistema por parte do usuário testador. Cada um dos métodos listados na Figura 2.2 dependem de no mínimo quatro usuários avaliadores para um teste de usabilidade. Neste trabalho, no entanto, a avaliação é feita por apenas um avaliador, portanto é uma pesquisa de baixa precisão estatística. Além disso, todos os testes foram realizados diretamente entre o

desenvolvedor e o professor orientador, que é um provável usuário com perfil *coordenador* ou *professor-tutor* do sistema, portanto ainda existe o viés do próprio desenvolvedor.

- **Inspeção de usabilidade:** o passo a passo cognitivo *(Cognitive Walkthroughs)* envolve um ou um grupo de avaliadores inspecionando uma interface de usuário passando por um conjunto de tarefas e avaliando sua compreensibilidade e facilidade de aprendizado. Neste projeto, a avaliação é feita por apenas um avaliador, portanto é uma pesquisa de baixa precisão estatística. Além disso, o avaliador do sistema é o mesmo desenvolvedor, assim a análise é passível de viés. Algumas das principais tarefas que descrevem as funcionalidades mais essenciais foram selecionadas. Para melhor representar o passo a passo das tarefas, elas foram organizadas por meio de fluxogramas com capturas da tela em cada uma das páginas do sistema.
- **Consulta de usabilidade:** como o desenvolvimento deste projeto se deu apenas em ambiente de desenvolvimento e de teste, mas não se deu em ambiente de produção, os métodos observação de campo *(Field Observation)*, grupos de foco *(Focus Groups)* e registro de uso real *(Logging Actual Use)* não foram realizados. Além disso, como os métodos de entrevistas *(Interviews)* e questionários *(Questionnaires)* ambos se destinam coletar opiniões dos usuários, foi optado por realizar somente a entrevista com professor orientador deste projeto já que ele é um provável usuário com perfil *coordenador* ou *professor-tutor* do sistema.

Capítulo 6

Conclusões

6.1 Considerações finais

Este trabalho resultou em uma aplicação prática e funcional que foi desenvolvida em paralelo com o foco na Interação Humano–Computador (IHC). Este paralelismo, conhecido como engenharia de usabilidade, se deu desde a escolha das tecnologias utilizadas até a avaliação de usabilidade feita após a conclusão do desenvolvimento, ou após um momento parada no fim de ciclo de desenvolvimento definido para avaliação. A escolha das tecnologias se deu de forma a favorecer a experiência do usuário e buscando atingir cada uma das metas e dos princípios de usabilidade e do design de interação. Além disso, todo o trabalho foi orientado em materiais de referência na área que possibilitaram um desenvolvimento efetivo da aplicação e uma avaliação do sistema final.

O design centrado no ser humano desde o princípio do desenvolvimento *(humane by design)* possibilitou a criação e avaliação de um sistema que buscou a acessibilidade, inclusão e usabilidade a tornando uma aplicação de design universal (ou inclusivo). Isto se deu desde a origem do projeto com a concepção buscando identificar os benefícios para o indivíduo e para a sociedade. Além disso, o progresso do sistema buscou estar vigente com as legislações e normas relacionadas a acessibilidade nos diferentes ambientes relacionados aos usuários utilizadores da aplicação.

A avaliação do sistema definida a partir de um momento de parada no fim de ciclo permitiu realçar a natureza cíclica e contínua do desenvolvimento de software, onde os testes e inspeções devem ser realizados recorrentemente buscando sempre melhorar a usabilidade mesmo que não seja possível atingir uma meta de perfeição do sistema. Desta forma, o conserto de cada uma das falhas de usabilidade detectadas durante o processo de avaliação são o ponto de partida para trabalhos futuros baseados neste sistema.

6.2 Trabalhos futuros

O desenvolvimento do sistema a partir de tecnologias *open-source* possibilita uma continuação deste projeto a partir do código-fonte já existente. Como ponto de partida pode ser considerado os requisitos eliminados neste projeto por questão de simplificação e a correção dos pontos de falha identificados durante o processo de avaliação. Os pontos de falha de usabilidade a serem corrigidos são encontrados listados na seção de inspeção de usabilidade. Outro possível trabalho é a avaliação por meio de testes e consulta de usabilidade com os estudantes com necessidades específicas.

Ainda em trabalhos futuros, a realização de novos estudos de campo proativo com utilização de engenharia de requisitos podem detectar novas demandas de continuidade no desenvolvimento da aplicação com novos recursos e categorias de usuário. A disponibilização de ferramentas de acessibilidade digital, que não foi o foco deste trabalho, é uma solução de grande importância como, por exemplo, leitores de tela para cegos ou transcrição de áudio para surdos. E, por fim, à medida que o desenvolvimento avança, outras áreas de conhecimento podem colaborar com diferentes partes do projeto como o tratamento de informações com ciência de dados e inteligência artificial ou mesmo integração com sistemas embarcados destinados a acessibilidade.

Referências

[1] Helen Sharp, Jennifer Preece, Yvonne Rogers: *Interaction Design: Beyond Human-Computer Interaction*. Wiley, 2019. ix, 18, 19, 20, 21, 27, 30, 33

[2] Porto Alegre Muniz, Maria Isabella de: *Usabilidade pedagógica e design de interação: processos de comunicação e colaboração em ambientes virtuais de aprendizagem*. Pontifícia Universidade Católica do Rio de Janeiro - PUC-Rio, 2016. ix, 18, 20

[3] Inseritore: *Single vs multi site*. Disponível em: `https://commons.wikimedia.org/wiki/File:Single-vs-multi-site.png`, acesso em 2023-03-14. ix, 42

[4] Wappler: *Web architecture with node.js*. Disponível em: `https://www.youtube.com/watch?v=6dmX2qCAeP0`, acesso em 2023-03-15. ix, 43

[5] SQLiteTutorial: *What is sqlite*. Disponível em: `https://www.sqlitetutorial.net/what-is-sqlite/`, acesso em 2023-03-14. ix, 48, 49

[6] Brasil: *Lei brasileira de inclusão*. Disponível em: `http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm`, acesso em 2023-05-08. 2

[7] Cabral, Leonardo Santos Amâncio: *Índice de funcionalidade brasileiro modificado (if-brm) e diferenciação e acessibilidade curricular (dac)*. Disponível em: `https://www.scielo.br/j/ccedes/a/c5RwSRJ5F9VKpBLgYtgh7Df/?lang=pt`, acesso em 2023-05-08. 2

[8] UFSCar: *Você é ou não uma pessoa com deficiência?* Disponível em: `https://www.acessibilidade.ufscar.br/apresentacao/conceitos-especificos`, acesso em 2023-05-08. 2

[9] Brasil: *Estatuto da pessoa com deficiência*. Disponível em: `https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm`, acesso em 2023-05-08. 3

[10] Moura, Márcia Abrahão: *Resolução do cad n° 50/2019*. Disponível em: `http://ppne.unb.br/index.php?option=com_content&view=article&id=64:resolucao-cad-n-5-2019&catid=2&Itemid=674`, acesso em 2023-05-08. 3, 4

[11] UnB, CEAD: *O que é acessibilidade?* Disponível em: `https://cead.unb.br/saladeaulavirtual/guia-de-acessibilidade/2-publicacoes/51-o-que-e-acessibilidade`, acesso em 2023-05-10. 3, 4

[12] DACES/DAC: *Sobre a daces*. Disponível em: http://www.acessibilidade.unb.br/index.php?option=com_content&view=article&id=22&Itemid=684, acesso em 2023-05-08. 4

[13] CEAD/UnB: *Público atendido pela daces*. Disponível em: https://www.cead.unb.br/index.php?option=com_content&view=article&id=65:teatro&catid=176:cursos-uab, acesso em 2023-05-08. 4

[14] DACES/DAC: *Programa de tutoria para acessibilidade (pta)*. Disponível em: http://www.acessibilidade.unb.br/index.php?option=com_content&view=article&id=108&Itemid=378, acesso em 2023-05-08. 5

[15] CEAD/UnB: *Centro de educação a distância*. Disponível em: https://cead.unb.br/saladeaulavirtual/guia-de-acessibilidade/2-publicacoes/66-centro-de-educacao-a-distancia, acesso em 2023-05-10. 6

[16] CEAD/UnB: *Guia de acessibilidade*. Disponível em: https://cead.unb.br/saladeaulavirtual/guia-de-acessibilidade, acesso em 2023-05-08. 6, 8, 84

[17] Consulting.us: *Inclusive design is integral to building better digital experiences*. Disponível em: https://www.consulting.us/news/6742/inclusive-design-is-integral-to-building-better-digital-experiences, acesso em 2023-05-20. 9, 35, 36

[18] Souza, Livia Soares Jardim e Morgana Dourado de: *Accessit : uma aplicação para o mapeamento de recursos de acessibilidade da unb*. Monografia (graduação) - Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação, 2017. 11

[19] Rachadel, Rodrigo Augusto: *Acessibilidade web : uma análise sobre suas ferramentas e diretrizes*. Monografia(graduação) - Universidade de Brasília, Instituto de Letras, Departamento de Línguas Estrangeiras e Tradução, 2017. 11

[20] Ferreira, Guilherme Fischmann: *Arquitetura da informação no desenvolvimento de aplicação web*. Monografia (graduação) - Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação, 2015. 11

[21] Chianca, Rafael Martins Pereira: *Interface web para sistema de gestão de redes seguindo projeto centrado em usuário*. Monografia (graduação) - Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação, 2020. 11

[22] Viana, Ciro Luis Trindade: *Visumo : uma plataforma web para criação e realização de avaliação educacional em libras*. Monografia (graduação) — Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação, 2017. 13

[23] Silva, Luiz Antônio da: *iframe : framework para o desenvolvimento de aplicações web*. Monografia (graduação) - Universidade de Brasília, Instituto de Exatas, Departamento de Ciência da Computação, 2014. 13

[24] Herlanio Leite Gonçalves, Arthur Thiago Barbosa Nobrega e: *Método de avaliação de comunicabilidade da engenharia semiótica: um estudo de caso em um sistema web.* Monografia (graduação) - Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação, 2013. 13

[25] Mendonça, Rodrigo Medeiros Soares da Silva e Jônatas Medeiros de: *Um estudo do relacionamento entre técnicas de usabilidade e testes automatizados em métodos empíricos de desenvolvimento de software.* Monografia (graduação) - Universidade de Brasília, Faculdade UnB Gama, Curso de Engenharia de Software, 2014. 13

[26] Oliveira Hargreaves, Felipe de: *Catálogo de práticas de acessibilidade: um apoio online voltado à acessibilidade da web.* Monografia (graduação) - Universidade de Brasília, Faculdade UnB Gama, Engenharia de Software, 2021. 13

[27] Catunda, Louise Ferraz: *Revista contraste: do papel ao website multilíngue e acessível.* Monografia (graduação) - Universidade de Brasília, Instituto de Letras, Departamento de Línguas Estrangeiras e Tradução, 2017. 13

[28] Santos Pergentino, Ana Carolina dos: *Avaliação heurística em um contexto real.* Monografia (graduação) - Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação, 2019. 14

[29] Dourado, Marcos Antonio Durães: *Usability heuristics for mobile applications a systematic review.* Monografia (graduação)—Universidade de Brasília, Faculdade UnB Gama, 2018. 14

[30] Higor Gabriel Azevedo Santos, Davi Cunha Farias Dupin e: *Direcionamento das heurísticas de nielsen no contexto de uso de landing pages.* Monografia (graduação) — Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação, 2022. 14

[31] Caldas, Luan Pignata: *Tuhm: uma ferramenta de apoio para testes de usabilidade.* Monografia (graduação) - Universidade de Brasília, Instituto de Ciências Exatas, Departamento de Ciência da Computação, 2021. 14

[32] Gomes, Paula Priscilla Fagundes Araújo Barros: *Avaliação da usabilidade e acessibilidade em interação humano-computador em portais: um estudo de caso sobre três portais da transparência municipais da paraíba.* Dissertação (Mestrado em Design) – Programa de Pós-Graduação em Design, Centro de Ciências e Tecnologias - Universidade Federal de Campina Grande, 2022. 14

[33] Ademir Moreno Junior, Edson Aparecido Martins: *Importância das técnicas de interação humano computador para desenvolvimento de sites.* Jornada Científica e Tecnológica - Fatec de Botucatu, 2018. 14

[34] Silva, Tiago Rodrigues da Silva Thiago Soares da: *Desenvolvimento web: a usabilidade como foco no processo de desenvolvimento.* Universidade Estadual de Goiás – Câmpus Porangatu – Sistemas de Informação - Porangatu, 2017. 15

[35] Sousa Martins Ferreira, Ana Cristina de: *Usabilidade e acessibilidade no design para a web.* Dissertação - Universidade do Porto - Faculdade de Belas Artes, 2018. 15

[36] Foundation, Interaction Design: *Human-computer interaction (hci).* Disponível em: `https://www.interaction-design.org/literature/topics/human-computer-interaction`, acesso em 2023-05-24. 17

[37] Yablonski, Jon: *Leis da Psicologia Aplicadas a UX: Usando psicologia para projetar produtos e serviços melhores.* Novatec Editora, 2020. 21

[38] Ahmed, Ali: *A systematic literature review of usability inspection methods.* Institutionen för datavetenskap - Department of Computer and Information Science, 2013. 23, 24

[39] Zhang, Zhijun William: *Usability evaluation.* Disponível em: `http://www.usabilityhome.com/`, acesso em 2023-06-01. 23, 24, 25, 27, 31

[40] Nielsen, Jakob: *Usability Engineering.* Academic Press, 1993. 24, 26, 27, 32, 33

[41] Dumas e Redish: *A Practical Guide to Usability Testing,.* Ablex Publishing, 1999. 24

[42] Rubin, Jeffrey: *Handbook of Usability Testing.* John Wiley and Sons, 2008. 24

[43] Soken, Reinhart, Vora e Metz: *Methods for Evaluating Usability.* Honeywell, 1992. 26

[44] Hartson, Rex: *Remote evaluation: The network as an extension of the usability laboratory.* Association for Computing Machinery, 1996. 26

[45] Vora e Helander: *A teaching method as an alternative to the concurrent think-aloud method for usablity testing.* Advances in Human Factors/Ergonomics, 1995. 27

[46] Anzai, Yuichiro e Hirohiko Mori: *Symbiosis of Human and Artifact.* North-Holland, 1995. 27

[47] Wharton, Cathleen: *The cognitive walkthrough method: a practitioner's guide.* John Wiley and Sons, 1994. 28

[48] Nielsen, Jakob e Robert Mack: *Usability Inspection Methods.* Wiley, 1994. 28, 29

[49] Wilson, Chauncey: *Method 10 of 100: Perspective-based inspection.* Disponível em: `https://shorturl.at/cuwB8`, acesso em 2023-06-02. 29

[50] Liyanage, Eranga: *10 usability heuristics explained.* Disponível em: `https://medium.com/@erangatl/10-usability-heuristics-explained-caa5903faba2`, acesso em 2023-07-03. 29

[51] Sonika: *How to improve ux with 10 usability heuristics by neilsen?* Disponível em: `https://shorturl.at/xBDWX`, acesso em 2023-07-03. 29

[52] Rossetti, Micaela: *Heurísticas de nielsen*. Disponível em: https://softdesign.com.br/blog/heuristicas-de-nielsen/, acesso em 2023-05-24. 30

[53] Alexandra: *Systems development life cycle*. Disponível em: https://stackify.com/what-is-sdlc/, acesso em 2023-06-02. 34

[54] Yablonski, Jon: *Humane by design*. Disponível em: https://humanebydesign.com/, acesso em 2023-05-20. 34

[55] Yablonski, Jon: *Inclusive design*. Disponível em: https://humanebydesign.com/principles/inclusive/, acesso em 2023-05-20. 35

[56] Xiao, Lillian: *6 principles for inclusive design*. Disponível em: https://uxplanet.org/6-principles-for-inclusive-design-3e9867f7f63e, acesso em 2023-05-20. 36

[57] Universal Design, Centre for Excellence in: *What is universal design*. Disponível em: https://universaldesign.ie/what-is-universal-design/, acesso em 2023-05-22. 36

[58] Universal Design, Centre for Excellence in: *Benefits and drivers*. Disponível em: https://www.universaldesign.ie/What-is-Universal-Design/Benefits-and-drivers, acesso em 2023-05-22. 36

[59] Team, ECS Accessibility: *Inclusive / universal design*. Disponível em: https://access.ecs.soton.ac.uk/blog/training/universal-design/, acesso em 2023-05-20. 37

[60] NC State University, The Center for Universal Design: *The principles of universal design*. Disponível em: https://design.ncsu.edu/research/center-for-universal-design/, acesso em 2023-05-22. 37

[61] Academy, Edge: *What is a web application?* Disponível em: https://www.stackpath.com/edge-academy/what-is-a-web-application/, acesso em 2023-03-13. 38

[62] Cunha, Jose Berardo: *A deeply detailed but never definitive guide to mobile development architecture*. Disponível em: https://shorturl.at/gzHPZ, acesso em 2023-03-13. 38, 55

[63] Comscore: *The 2017 u.s. mobile app report*. Disponível em: https://www.comscore.com/Insights/Presentations-and-Whitepapers/2017/The-2017-US-Mobile-App-Report, acesso em 2023-03-13. 38, 39

[64] Richter, Felix: *Os usuários de aplicativos gastam 77% de seu tempo em seus 3 principais aplicativos*. Disponível em: https://www.statista.com/chart/3835/top-10-app-usage/, acesso em 2023-03-13. 39

[65] WHATWG: *Html living standard*. Disponível em: https://html.spec.whatwg.org, acesso em 2023-03-13. 40, 41

[66] W3C: *Css snapshot 2023*. Disponível em: https://www.w3.org/TR/CSS/, acesso em 2023-03-13. 41

[67] ECMA: *Ecma-262 ecmascript® 2022 language specification*. Disponível em: https://www.ecma-international.org/publications-and-standards/standards/ecma-262/, acesso em 2023-03-13. 41

[68] W3Techs: *Usage statistics of javascript as client-side programming language on websites*. Disponível em: https://w3techs.com/technologies/details/cp-javascript/, acesso em 2023-03-13. 41

[69] iugu: *Spa (single page application): O que é, vantagens e desvantagens*. Disponível em: https://www.iugu.com/iugu4devs/blog/single-page-applications, acesso em 2023-03-14. 42

[70] Bootstrap: *Bootstrap pagination*. Disponível em: https://getbootstrap.com/docs/4.0/components/pagination/, acesso em 2023-03-14. 43

[71] Bootstrap: *Bootstrap (framework front-end)*. Disponível em: https://getbootstrap.com, acesso em 2023-03-14. 44

[72] GitHub: *Most starred github projecs*. Disponível em: https://github.com/search?q=stars:%3E100000&type=repositories, acesso em 2023-03-14. 44

[73] W3Techs: *Usage statistics and market share of bootstrap for websites*. Disponível em: https://w3techs.com/technologies/details/cs-bootstrap, acesso em 2023-03-14. 44

[74] ASCOM: *Celular segue como aparelho mais utilizado para acesso à internet no brasil*. Disponível em: https://shorturl.at/mARU9, acesso em 2023-03-14. 44

[75] Beta-Labs: *Bootstrap grid system*. Disponível em: https://www.beta-labs.in/2020/11/bootstrap-grid-system.html, acesso em 2023-03-15. 45

[76] Bootstrap: *Bootstrap grid system*. Disponível em: https://getbootstrap.com/docs/5.0/layout/grid/, acesso em 2023-03-15. 45

[77] Bootstrap: *Bootstrap breakpoints*. Disponível em: https://getbootstrap.com/docs/5.0/layout/breakpoints/, acesso em 2023-03-15. 45

[78] Melo, Diego: *O que é node.js? [guia para iniciantes]*. Disponível em: https://tecnoblog.net/responde/o-que-e-node-js-guia-para-iniciantes/, acesso em 2023-03-14. 46, 47

[79] Mario Casciaro, Luciano Mammino: *Node.js Design Patterns*. Packt Publishing, 2020. 46, 47

[80] StackOverflow: *Stack overflow developer survey 2022*. Disponível em: https://survey.stackoverflow.co/2022/, acesso em 2023-03-14. 47, 48, 50, 52, 53

[81] SQLite: *Most widely deployed and used database engine*. Disponível em: https://sqlite.org/mostdeployed.html, acesso em 2023-03-14. 48

[82] *Knex.js: Sql query builder for javascript.* Disponível em: https://knexjs.org/, acesso em 2023-03-14. 49

[83] soumya08: *Version control systems.* Disponível em: https://www.geeksforgeeks.org/version-control-systems/, acesso em 2023-03-15. 50

[84] Atlassian: *What is version control?* Disponível em: https://www.atlassian.com/git/tutorials/what-is-version-control, acesso em 2023-03-15. 50

[85] GitLab: *What is version control?* Disponível em: https://about.gitlab.com/topics/version-control/, acesso em 2023-03-15. 50

[86] Git: *Git scm.* Disponível em: https://git-scm.com/, acesso em 2023-03-14. 51

[87] Wikibooks: *Git and subversion.* Disponível em: https://en.wikibooks.org/wiki/Game_Creation_with_XNA/Programming/Git_and_Subversion, acesso em 2023-03-14. 51

[88] Lbhtw: *Git data flow.* Disponível em: https://commons.wikimedia.org/wiki/File:Git_data_flow.png, acesso em 2023-03-14. 51

[89] *Github.* Disponível em: https://github.com/, acesso em 2023-03-14. 52

[90] *microsoft/vscode.* Disponível em: https://github.com/microsoft/vscode, acesso em 2023-03-14. 53

[91] *Visual studio code.* Disponível em: https://code.visualstudio.com, acesso em 2023-03-14. 53

[92] Tabnine: *Is visual studio code really the best code editor?* Disponível em: https://www.tabnine.com/blog/visual-studio-code-really-the-best-code-editor/, acesso em. 54

[93] Pleysier, Ben: *Wappler - the alternative.* Disponível em: https://www.youtube.com/watch?v=cXdzCHFgZvw, acesso em 2023-03-14. 54

[94] *About wappler.* Disponível em: https://wappler.io/about/, acesso em 2023-03-14. 54

[95] Hiram: *The no-code landscape.* Disponível em: https://baserow.io/blog/no-code-landscape, acesso em 2023-07-04. 55

[96] Luo, Yajing: *Characteristics and challenges of low-code development: The practitioners' perspective.* ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), 2021. 54

[97] piyushagg: *Browser developer tools.* Disponível em: https://www.geeksforgeeks.org/browser-developer-tools/, acesso em 2023-03-15. 55

[98] GlobalStats: *Browser market share worldwide.* Disponível em: https://gs.statcounter.com/browser-market-share, acesso em 2023-03-15. 56

[99] *Dockerhub maildev/maildev*. Disponível em: https://hub.docker.com/r/maildev/maildev, acesso em 2023-03-15. 56

[100] *Json*. Disponível em: https://www.json.org/json-pt.html, acesso em 2023-06-12. 57

[101] *Postman (software)*. Disponível em: https://www.postman.com/, acesso em 2023-06-12. 57

[102] Amazon: *Api*. Disponível em: https://aws.amazon.com/pt/what-is/api/, acesso em 2023-06-12. 58

[103] Alex Biryukov, Daniel Dinu e Dmitry Khovratovich: *Argon2 specification*. Disponível em: https://www.cryptolux.org/images/0/0d/Argon2.pdf, acesso em 2023-07-27. 58

[104] Aumasson, JP: *Closing the password hashing competition, releasing argon2*. Disponível em: https://research.kudelskisecurity.com/2015/11/04/closing-the-password-hashing-competition-releasing-argon2/, acesso em 2023-04-17. 58

[105] Khovratovich, Dmitry: *Password hashing competition*. Disponível em: https://www.password-hashing.net/, acesso em 2023-04-17. 58

[106] Zochowski, Michael: *Benchmarking hash and signature algorithms*. Disponível em: https://medium.com/logos-network/benchmarking-hash-and-signature-algorithms-6079735ce05, acesso em 2023-04-17. 58, 59

[107] *Blake2 specification*. Disponível em: https://www.blake2.net/, acesso em 2023-07-27. 59

[108] O'Whielacronx, Zooko: *introducing blake2: an alternative to sha-3, sha-2, sha-1, and md5*. Disponível em: https://tahoe-dev.tahoe-lafs.narkive.com/fov06aDD/introducing-blake2-an-alternative-to-sha-3-sha-2-sha-1-and-md5, acesso em 2023-04-17. 59

[109] Microsoft: *Regular expression*. Disponível em: https://learn.microsoft.com/en-us/dotnet/standard/base-types/regular-expression-language-quick-reference, acesso em 2023-06-12. 59

[110] javaTpoint: *Requirement engineering*. Disponível em: https://www.javatpoint.com/software-engineering-requirement-engineering, acesso em 2023-06-21. 59

[111] Sacolick, Isaac: *What is ci/cd? continuous integration and continuous delivery explained*. Disponível em: https://shorturl.at/rwFKV, acesso em 2023-03-23. 60

[112] Lo, Victoria: *Github actions 101: Develop a ci/cd workflow.* Disponível em: https://lo-victoria.com/github-actions-101-develop-a-cicd-workflow, acesso em 2023-03-24. 60

[113] Patankar, Faraz: *Using github actions with railway.* Disponível em: https://blog.railway.app/p/github-actions, acesso em 2023-03-24. 60

[114] Chris, Kolade: *Crud operations – what is crud?* Disponível em: https://www.freecodecamp.org/news/crud-operations-explained/, acesso em 2023-03-27. 60

[115] Pang, Avelon: *Crud operations explained.* Disponível em: https://medium.com/geekculture/crud-operations-explained-2a44096e9c88, acesso em 2023-03-27. 60, 61

[116] Lutkevich, Ben: *honeypot (computing).* Disponível em: https://www.techtarget.com/searchsecurity/definition/honey-pot, acesso em 2023-07-07. 78