

UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

**AVALIAÇÃO HEURÍSTICA DE UM AMBIENTE VIRTUAL DE
APRENDIZAGEM COLABORATIVA PARA O ENSINO DE
PROGRAMAÇÃO**

VINICIUS OLIVEIRA AMARAL

VITÓRIA DA CONQUISTA
DEZEMBRO/ 2021

UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

**AVALIAÇÃO HEURÍSTICA DE UM AMBIENTE VIRTUAL DE
APRENDIZAGEM COLABORATIVA PARA O ENSINO DE
PROGRAMAÇÃO**

Projeto entregue à disciplina Trabalho Supervisionado II como requisito para aprovação na disciplina, ministrada pela Prof. Dr. Maísa Soares dos Santos Lopes, no pl 2020.2.

Discente: Vinicius Oliveira Amaral
Prof. Orientadora: Maísa Soares dos Santos Lopes
Prof. Coorientador: Gidevaldo Novais

VITÓRIA DA CONQUISTA
DEZEMBRO / 2021

AGRADECIMENTOS

A Deus, pela minha vida, e por me permitir ultrapassar todas as dificuldades encontradas durante a realização deste trabalho.

A minha família por todo apoio dado a mim ao longo de todo o processo de formação.

Agradeço à minha orientadora, a Professora Dra. Máisa Soares dos Santos Lopes, por me guiar durante todo este processo com muito respeito, dedicação, empatia e paciência. Toda minha gratidão por tudo que a senhora fez por mim durante todos esses anos.

Agradeço ao meu coorientador, o Professor Me. Gidevaldo Novais, por me auxiliar durante o desenvolvimento deste trabalho, contribuindo para definir o escopo e a organização do mesmo.

Um agradecimento especial aos meus amigos Jenifer, Natalia, Celina e Tharcio. Sem vocês confesso que não teria chegado até aqui. Obrigado pelos bons momentos, conselhos e por estarem comigo durante todo este período.

Por fim, agradeço aos professores e todos os colegas que me ajudaram e me inspiraram durante esses anos.

RESUMO

A usabilidade de uma aplicação busca oferecer ao usuário uma interação eficiente, eficaz, fácil e agradável ao usuário, de modo a tornar simples e natural a execução de tarefas e ações. Entretanto, muitas vezes essa questão é deixada de lado, e sistemas acabam apresentando problemas que dificultam a utilização do mesmo. Dessa forma, surge a necessidade de realizar algum processo que possa identificar e tratar estes problemas. Se tratando de interfaces virtuais voltadas para o ensino de programação de maneira colaborativa alguns requisitos adicionais devem ser considerados para fornecer uma boa experiência ao usuário, tanto na parte de interação em si, quanto na de colaboração. Com isso, este trabalho tem como objetivo aplicar uma avaliação heurística que engloba essas duas questões. Para isso, foram utilizados 2 conjuntos de heurísticas, as de Nielsen, que trata de questões de usabilidade em sistemas WEB, e as de Baker, Greenberg e Gutwin que cuidam da parte colaborativa. Para validar a importância desses princípios, foi analisado um protótipo de interface. Como resultado foi identificado que o protótipo atende 8 heurísticas, contudo, foi detectado a violação de 10 heurísticas. Nessas últimas foi sugerido possíveis soluções para estes problemas.

Palavras-chave: Sistemas colaborativos, ensino de programação, usabilidade, experiência do usuário, avaliação heurística.

ABSTRACT

The usability of an application seeks to offer the user an efficient, effective, easy and user-friendly interaction, in order to make the execution of tasks and actions simple and natural. However, this issue is often left out, and systems end up presenting problems that make it difficult to use. Thus, there is a need to carry out some process that can identify and treat these problems. When dealing with virtual interfaces aimed at teaching programming in a collaborative way, some additional requirements must be considered to provide a good user experience, both in terms of interaction itself, and in terms of collaboration. Thus, this work aims to apply a heuristic evaluation that encompasses these two issues. For this, 2 sets of heuristics will be used: Nielsen, which deals with usability issues in WEB systems, and Baker, Greenberg and Gutwin, which take care of the collaborative part. To validate the importance of these principles, an interface prototype was analyzed. As a result, the violation of 10 heuristics was identified, as well as the suggestion of possible solutions for these problems.

Keywords: Collaborative systems, programming teaching, usability, user experience, heuristic evaluation.

SUMÁRIO

1 - INTRODUÇÃO	8
1.1 - PROBLEMA DE PESQUISA	10
1.2 - OBJETIVOS	10
1.2.1 Geral	10
1.2.2 Específicos	10
1.3 - JUSTIFICATIVA	11
2 - FUNDAMENTAÇÃO TEÓRICA	12
2.1 ENSINO DE PROGRAMAÇÃO	12
2.2 APRENDIZAGEM COLABORATIVA SUPORTADA POR COMPUTADOR	12
2.3 PROGRAMAÇÃO COLABORATIVA	13
2.4 INTERAÇÃO HUMANO-COMPUTADOR	14
2.4.1 Usabilidade ISO/IEC 9241-11	16
2.4.2 Interface dos Ambientes Virtuais de Aprendizagem	16
2.4.3 Avaliação Heurística	17
2.4.3.1 Heurísticas de Nielsen	17
2.4.3.2 Heurísticas de Baker, Greenberg e Gutwin	18
2.4.3.3 Grau de severidade	20
2.5 TRABALHOS RELACIONADOS	20
3 - MATERIAIS E MÉTODOS	21
3.1 METODOLOGIA	21
3.2 TECNOLOGIAS E MATERIAIS UTILIZADOS	21
3.2.1 Interface analisada	22
3.2.2 Guia para realização da avaliação	23
3.3 APLICAÇÃO DA AVALIAÇÃO HEURÍSTICA	25
4 - RESULTADOS E DISCUSSÃO	27
4.1 RESULTADOS DA AVALIAÇÃO HEURÍSTICA - NIELSEN	27
4.1.1 Visibilidade do estado do sistema	28
4.1.2 Controle e liberdade do usuário	29
4.1.3 Consistência e padronização	30
4.1.4 Prevenção de erros	30
4.1.4 Estética e design minimalista	31
4.1.5 Ajude os usuários a reconhecer, diagnosticar e se recuperar de erros	33
4.1.6 Ajuda e documentação	34
4.2 RESULTADOS DA AVALIAÇÃO HEURÍSTICA - BAKER, GREENBERG E GUTWIN	35
4.2.1 Oferecer comunicação não intencional através do uso de artefatos compartilhados (feedthrough)	37
4.2.2 Apoiar os usuários na coordenação de suas ações	37
4.2.3 Facilitar a identificação de colaboradores e ao contato entre eles	37

4.3 CONSIDERAÇÕES FINAIS	38
5 - CONCLUSÃO	39
REFERÊNCIAS	40
APÊNDICE A - GUIA PARA REALIZAÇÃO DA AVALIAÇÃO	44

1 - INTRODUÇÃO

O desenvolvimento dos recursos tecnológicos no decorrer dos últimos anos tem gerado diversas alterações nas formas tradicionais de ensino. As Tecnologias de Informação e Comunicação (TIC's) têm proporcionado novas possibilidades de utilização de tais artifícios como ferramentas de apoio ao processo de ensino e aprendizagem. Sendo assim, com o objetivo de facilitar o aprendizado, ambientes virtuais de aprendizagem vêm sendo cada vez mais utilizados pelas instituições de ensino (FONSECA et al., 2016), estando mais evidente no momento devido ao cenário de pandemia em que o mundo vive, no qual é recomendado o isolamento social.

Entretanto, diversos desafios vêm sendo encontrados pelos educadores e pesquisadores da área para garantir um bom aproveitamento dessas ferramentas, principalmente para assegurar a interação e colaboração. A primeira está relacionada à presença social, que se refere a mostrar a presença dos alunos nesses ambientes, mesmo que eles estejam distantes fisicamente (NUNES et al., 2016), e permitir que eles expressem suas características pessoais. Já a segunda, está ligada ao trabalho em conjunto para a criação de algum artefato, seja ele uma solução algorítmica, uma troca de perspectivas entre outras coisas.

Segundo Murphy (2004), para que haja a colaboração é necessário que exista uma interação preliminar entre os participantes do ambiente, ou seja, que eles possuam uma percepção da presença dos outros e comecem a se relacionar como um grupo. Sendo assim, o ambiente deve fornecer mecanismos para que os usuários possam se comunicar e se apresentar para os demais. Dessa forma, os sistemas de aprendizagem colaborativa devem fornecer diversos meios de comunicação (chat, canal de áudio, fóruns etc), além de garantir um suporte pedagógico (STAHL; KOSCHMANN; SUTHERS, 2006).

Em se tratando do ensino de programação de maneira colaborativa, alguns requisitos adicionais são necessários para garantir o desenvolvimento de software e de revisão da solução (SERRANO-CÁMARA et al., 2014).

Para Baranauskas (2006), ao projetar um sistema computacional, a preocupação com a qualidade da interação do usuário com a interface deve ser aspecto fundamental a ser analisado. Pois, a forma como o usuário visualiza e interage com ela precisa possibilitar sua comunicação, a execução de suas ações e o recebimento de informações. Já a interpretação que o usuário faz da interface pode definir que ele atinja ou não os objetivos educacionais propostos com as atividades planejadas pelo professor no ambiente.

Em Sistemas Colaborativos, os usuários devem interagir não somente com o software, mas também devem usá-lo para se comunicar com os outros usuários. Sendo assim, lidar com as particularidades dos indivíduos de um mesmo grupo, promover a diversidade na composição dele, bem como na divisão das tarefas podem ser considerados desafios a mais nesse tipo de aplicação (PRATES; DE SOUZA, 2002).

Com esse intuito, o Laboratório Acadêmico em Redes de Aprendizagem (LARA), um projeto da Universidade Estadual do Sudoeste da Bahia (UESB), disponibiliza experimentos de robótica para auxiliar o ensino-aprendizagem de programação contextualizada e colaborativa (LOPES et al, 2016). Uma vez que, citado por (LOISELLE, 2002), "a presença de atividades de aprendizagem colaborativa é uma condição necessária ao estabelecimento de ambientes de aprendizagem ricos", o LARA busca promover um ambiente colaborativo, no qual estudantes podem trabalhar em equipe, tanto na formulação de uma solução quanto em sua codificação.

Dessa forma, o laboratório possui um experimento remoto de robótica móvel e atualmente está desenvolvendo um experimento de domótica e um de robótica móvel virtual. Sendo que os dois primeiros contam com elementos reais (hardware) controlados por uma interface virtual online. Já o último, é uma aplicação WEB que simula um experimento físico.

1.1 - PROBLEMA DE PESQUISA

Gerenciar o esforço colaborativo dos membros de um grupo, bem como monitorar o processo colaborativo são desafios importantes na aprendizagem colaborativa. É necessário que os membros do grupo identifiquem e trabalhem suas habilidades para que de fato todos contribuam de forma justa para a criação de um artefato (LAND; ZEMBAL-SAUL, 2003).

Outro ponto importante é que os esforços individuais de cada membro devem ser comunicados e sincronizados, pois é necessário que todos trabalhem na mesma direção e façam contribuições realmente relevantes (HÄMÄLÄINEN, 2008). Com isso, estratégias de design devem ser incorporadas à solução para que essas características sejam respeitadas.

Entretanto, na literatura existem poucos materiais voltados para a avaliação da interface desse tipo de aplicação. Desse modo, tem-se como pergunta de pesquisa “Como avaliar se a interface de um ambiente de aprendizagem colaborativa de programação possui uma boa usabilidade em relação a experiência do usuário?”

1.2 - OBJETIVOS

1.2.1 Geral

- Propor melhorias na interface do protótipo do Ambiente Virtual de Aprendizagem Colaborativa para o ensino de programação, por meio de uma avaliação heurística.

1.2.2 Específicos

- Realizar uma revisão da literatura para fundamentação dos métodos de avaliação de interface de sistemas colaborativos suportados por computador;
- Validar a interface por meio de avaliação heurística;
- Analisar os resultados obtidos e propor soluções com base nos problemas encontrados.

1.3 - JUSTIFICATIVA

A programação de computadores é uma habilidade muito requisitada hoje no mercado de trabalho. Porém, o estudo de programação é visto como algo difícil, lento e pouco atrativo para muitos estudantes (SOUZA; BATISTA; BARBOSA, 2016). Além disso, são requisitadas outras habilidades como criatividade, lógica e trabalho em equipe. Embora existam diversos estudos sendo realizados para melhorar este quadro, ainda há uma falta de ferramentas educacionais que motivem os alunos a trabalhar tais competências.

Outro fator, é que muitas ferramentas já existentes não dispõem de soluções para garantir a interação de maneira centralizada nos usuários, ou seja, não disponibilizam formas de comunicação direta entre os integrantes da plataforma, histórico de alterações etc, prejudicando assim analisar se elas de fato garantem a colaboração.

Tendo em vista tais dificuldades, este trabalho tem como o objetivo compreender quais elementos um ambiente colaborativo para o desenvolvimento de software deve possuir para que promova a colaboração entre seus usuários.

Uma vez que avaliar sistemas colaborativos é um grande desafio, algumas metodologias de sistemas monousuários serão utilizadas de maneira adaptada, sendo tudo isso feito de maneira remota. Dessa forma, serão feitos testes com especialistas, para validar ou realizar alterações no ambiente.

2 - FUNDAMENTAÇÃO TEÓRICA

2.1 ENSINO DE PROGRAMAÇÃO

O ensino e aprendizagem de programação é conhecido como algo bastante complexo e trabalhoso, por isso, muitos cursos superiores que possuem tais componentes curriculares na sua grade curricular, apresentam altas taxas de reprovação e desistência (PITEIRA; HADDAD, 2011). De modo geral, devido à falta de uma base matemática sólida e pelo alto nível de abstração exigido, muitos estudantes possuem dificuldades em assimilar determinados conteúdos e conceitos, sendo alguns deles: declaração de variáveis, condicionais, recursão etc (HELMINEN; MALMI, 2010) (MEERBAUM-SALANT; ARMONI; BENARI, 2010) (PITEIRA; COSTA, 2013).

Contudo, existem diversos trabalhos que apontam que grande parte desses estudantes conseguem entender tais conceitos, porém possuem dificuldades em utilizá-los durante o desenvolvimento de alguma aplicação, compreensão e análise de programas (HU; TSENG; LEE, 2013) (JANTAN; ALJUNID, 2012) (PANWONG; KEMAVUTHANON, 2014). Além disso, outra questão bastante discutida é a motivação dos estudantes, visto que muitos deles encaram a programação em si como algo chato, cansativo e entediante (CORNEY; TEAGUE; THOMAS, 2010) (IBRAHIM; JAAFAR, 2010) (SHI; WHITE, 2013).

Desta forma, alguns estudos e propostas estão sendo realizadas a fim de reduzir tais dificuldades. Dentre eles está a programação colaborativa, na qual duas ou mais pessoas desenvolvem de maneira conjunta algum algoritmo. Essa abordagem possui diversos benefícios, sendo eles: compartilhamento de conhecimento, diferentes visões sobre a solução de problemas e maior possibilidade de validação e correção de código.

2.2 APRENDIZAGEM COLABORATIVA SUPOSTADA POR COMPUTADOR

Existem diversas abordagens de ensino que buscam aprimorar o processo de aprendizado e incentivam os estudantes a terem uma participação ativa neste percurso, tornando-os assim protagonistas na construção do conhecimento. Uma

delas é a aprendizagem colaborativa, que está baseada na interação e participação ativa dos alunos. Ela tem como principais pilares: o cooperativismo, troca de experiências e participação ativa.

Este tipo de abordagem tem sido utilizado por educadores e praticada nos mais diversos níveis escolares, desde o ensino fundamental à pós-graduação. Ela não é algo novo, contudo, devido a uma maior disponibilidade das tecnologias de comunicação e de interação social, o seu modelo vem sendo cada vez mais buscado. Dessa forma, Ambientes Virtuais Colaborativos de Aprendizagem estão cada vez mais sendo utilizados nos mais diversos níveis de ensino. Tendo o contexto de isolamento social incentivado uma maior adesão.

O uso recorrente desse tipo de prática pedagógica traz consigo diversos benefícios, dentre eles estão: desenvolvimento de um olhar crítico, habilidade para resolver problemas, trabalho em equipe. Para (CASTRO; MENEZES, 2019), “O exercício da colaboração requer sistemas apropriados para o registro das produções individuais, para a socialização das produções, para a coordenação das ações, para a recuperação inteligente das informações produzidas e a respectiva reflexão sobre o produto final”.

Os ambientes virtuais colaborativos de aprendizagem são caracterizados como espaços compartilhados de interação que promovem a construção, inserção e troca de informações pelos participantes, visando a construção social do conhecimento.

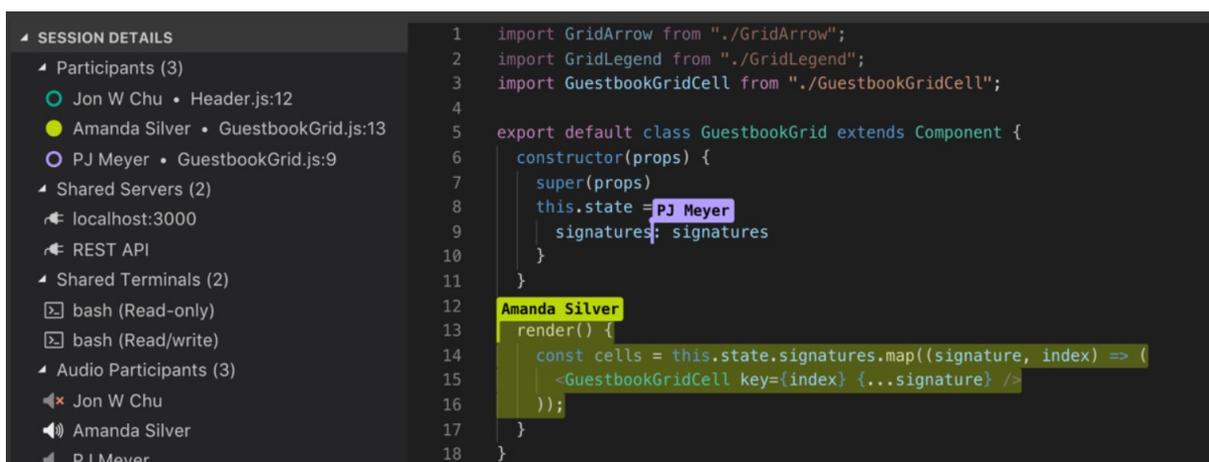
Sendo assim, ao desenvolver uma aplicação que faça uso dessa abordagem ela deve levar em consideração todas essas informações citadas acima. Para que tais benefícios sejam realmente alcançados.

2.3 PROGRAMAÇÃO COLABORATIVA

A programação colaborativa é uma abordagem na qual duas ou mais pessoas trabalham de maneira conjunta em um mesmo ambiente de trabalho ou editor de código. Neste típico de prática é extremamente importante que os participantes visualizem as ações dos colegas, tanto de maneira síncrona quanto assíncrona, possibilitando assim que as ações sejam validadas ou até mesmo questionadas

pelos demais integrantes. Uma extensão do editor de texto Visual Studio Code¹ bastante utilizada para este fim é o Live Share² (figura), por meio dela os utilizadores conseguem acompanhar as atitudes dos outros usuários que estão no mesmo ambiente de codificação.

Figura 1 - Exemplo de uso do Live Share



The screenshot shows the Visual Studio Code interface during a Live Share session. On the left, the 'SESSION DETAILS' sidebar is open, showing a list of participants: Jon W Chu (Header.js:12), Amanda Silver (GuestbookGrid.js:13), and PJ Meyer (GuestbookGrid.js:9). Below this, it lists shared servers (localhost:3000, REST API) and shared terminals (two bash shells). At the bottom, audio participants are listed: Jon W Chu, Amanda Silver, and PJ Meyer. The main editor area displays JavaScript code for a class named 'GuestbookGrid'. The code includes imports for 'GridArrow', 'GridLegend', and 'GuestbookGridCell'. The class has a constructor that takes 'props' and sets 'this.state' to 'PJ Meyer' and 'signatures' to 'signatures'. The 'render()' method uses 'this.state.signatures.map()' to render 'GuestbookGridCell' components. The name 'Amanda Silver' is highlighted in yellow in the code editor, indicating she is the current viewer.

Fonte: LiveShare (2021)

Assim como outras práticas colaborativas, essa abordagem possui inúmeros benefícios, tais como: diferentes pontos de vista na solução de algum problema ou estruturas a serem utilizadas, maior possibilidade de validação e correção do código, compartilhamento de conhecimentos e experiências etc. Além disso, ela é capaz de tornar o trabalho ágil, facilitando assim a entrega de novos produtos e aplicações.

2.4 INTERAÇÃO HUMANO-COMPUTADOR

A Interação Humano-Computador é uma área da computação, que como o próprio nome diz, estuda as interações entre os humanos e computadores. Pois, enquanto as pessoas se comunicam com as outras por meio de texto, voz, gestos etc, as máquinas o fazem através de sinais elétricos, ou seja, bits.

Sendo assim, para que essas diferenças sejam traduzidas e essas duas classes possam se comunicar, existem interfaces que são responsáveis por realizar essa interação. Algumas delas são: teclado, mouse, tela, uma interface gráfica etc. Com isso, a IHC busca cada vez mais melhorar e abstrair essas diferenças. Desse modo, a interface de uma aplicação é o artifício pelo qual ocorre a comunicação entre um sistema e uma pessoa-usuária. De acordo com (PRESSMAN, 2011):

“A interface do usuário é indiscutivelmente o elemento mais importante de um produto. Se a interface for mal projetada, a capacidade de o usuário aproveitar todo o poder computacional e conteúdo de informações de uma aplicação pode ser seriamente afetada. Na realidade, uma interface fraca pode fazer com que uma aplicação, em outros aspectos bem projetada e solidamente implantada, falhe.”

Para o desenvolvimento do design de interação, (ROGERS; SHARP; PREECE, 2013) define 4 atividades principais que devem ser realizadas:

- Estabelecer os requisitos;
- Criar alternativas de design;
- Prototipar;
- Testar.

Com isso, ao projetar uma interface, a usabilidade deve ser vista como algo essencial, pois ela irá garantir que o sistema seja fácil, eficiente e amigável com o possível usuário (PREECE; ROGERS; SHARP, 2005). Eles definiram algumas metas de usabilidade para que uma boa experiência do usuário seja alcançada. Sendo elas:

- Eficácia no uso: está relacionado ao quanto o sistema é bom para realizar o que se espera dele;
- Eficiência no uso: corresponde o modo como o sistema ajuda os usuários durante a realização de suas tarefas;
- Fácil de lembrar como usar: visto como a facilidade de lembrar como usar o sistema;
- Fácil de entender: está relacionado a facilidade de um novo usuário entender como utilizar o sistema, ou seja o quão intuitivo ele é;
- De boa utilidade: corresponde a maneira com a qual o usuário consegue realizar tudo aquilo que ele precisa ou deseja;
- Seguro no uso: está relacionado a proteção do usuário de situações indesejáveis e que levem a erro.

Nos últimos anos, os usuários de aplicações digitais tem se tornado cada vez mais exigente com a aplicação utilizada, isso se dá devido a concorrência cada vez maior de soluções digitais similares e falta de paciência cada vez maior das pessoas. Sendo assim, empresas e organizações em geral estão investindo bastante em soluções digitais voltadas para o usuários, com interfaces fáceis de

utilizar, rapidez no desempenho e segurança. Apesar de muitos não saberem tais características estão relacionadas a usabilidade de um sistema.

2.4.1 Usabilidade ISO/IEC 9241-11

A usabilidade é uma medida definida pela ISO/IEC 9241 da seguinte maneira:

“A medida na qual um produto pode ser usado por usuários específicos para alcançar objetivos específicos com eficácia, eficiência e satisfação em um contexto específico de uso.” O termo designa o grau de facilidade com que o usuário realiza seus objetivos, sem o auxílio de um ajudante humano, ou seja, quão fácil é usar algo; mecânico, eletrônico ou digital.”

De acordo com (NIELSEN, 2012), a usabilidade é um atributo de qualidade que avalia a facilidade de utilização das interfaces do usuário. Ainda segundo o autor, a usabilidade é de extrema importância em um sistema, visto que se o usuário não consegue aprender a interagir com sua interface é como se o sistema não existisse pois o usuário não irá usá-lo.

Uma interface de um aplicação apresenta problema de usabilidade quando um ou mais usuários esbarram com alguma dificuldade referente a utilização do mesmo. Vários motivos podem levar a isso, sendo assim avaliações na interface devem ser feitas de maneira a diminuir ou evitar tais problemas. Pois, diversos problemas podem ser ocasionados por isso, como: diminuição de produtividade, perda de dados e até mesmo a desistência do uso (WINCKLER; PIMENTA, 2002).

2.4.2 Interface dos Ambientes Virtuais de Aprendizagem

De acordo com (ROSA; MATOS, 2015) a qualidade de interação desse tipo de ambiente tem sido preocupação de diversos estudiosos, visto que, esta qualidade pode influenciar nos processos de ensino e de aprendizagem. Sendo assim, é imprescindível que se elabore um bom design de interação para que além de ter a navegação e interfaces agradáveis, os usuários não gastem muito tempo aprendendo a utilizar o ambiente do que trabalhando no conteúdo da disciplina.

2.4.3 Avaliação Heurística

A avaliação heurística é uma técnica de avaliação na qual permite encontrar determinados problemas em interfaces de aplicações. Esse método faz uso de uma série de princípios denominados heurísticas, que representam propriedades comuns em interfaces vindas do conhecimento de questões computacionais, psicológicas e sociais (GLORIA, 2015).

Além de ser considerada como uma técnica barata, eficiente e rápida para detecção preventiva de problemas na interface, ela pode ser realizada em qualquer fase do desenvolvimento de uma aplicação (NIELSEN, 1990). Segundo (NIELSEN; MACK, 1994), ao aplicar esse método é possível identificar 60%, 70% e 75% de problemas na interface, tendo a presença de três, quatro e cinco avaliadores, respectivamente. De acordo com os autores, um número inferior ou maior do que esse pode acabar prejudicando o processo de avaliação.

2.4.3.1 Heurísticas de Nielsen

As heurísticas de Nielsen (1994) são um conjunto de princípios direcionados para avaliação da usabilidade em sistemas digitais, e deste modo capaz de detectar previamente possíveis problemas que possam ocorrer durante a utilização do mesmo. Elas são compostas por dez princípios que serão melhor descritos na tabela 1.

Tabela 1 - Heurísticas de Nielsen

Heurística	Descrição
1. Visibilidade do estado do sistema	O sistema deve sempre manter os usuários informados sobre o que está acontecendo, através de um feedback apropriado.
2. Compatibilidade entre o sistema e o mundo real	O sistema deve usar termos familiares a linguagem dos usuários, para que as informações sejam naturais e lógicas.
3. Controle e liberdade do usuário	O usuário tem que ser alertado das consequências das suas ações e saber se elas podem ser reversíveis ou não, para não ocorrer nenhuma ação indesejada.

4. Consistência e padronização	O sistema deve seguir padrões de interface para facilitar o seu uso e proporcionar ações de fácil memorização.
5. Prevenção de erros	O sistema deve prevenir a ocorrência de erros, apresentando mensagens antes do usuário se comprometer com a ação.
6. Reconhecimento em vez de memorização	O sistema deve tornar visíveis objetos, ações, informações e botões para minimizar a carga de memória dos usuários. Informações prévias devem ser apresentadas se necessário para aquele momento.
7. Flexibilidade e eficiência de uso	O sistema deve permitir ter atalhos, para usuários mais experientes conseguirem realizar ações de maneira mais rápida, sem comprometer usuários inexperientes.
8. Estética e design minimalista	As interfaces não devem conter informações irrelevantes ou totalmente desnecessárias para não diminuir a visibilidade de outras informações relevantes.
9. Ajude os usuários a reconhecer, diagnosticar e se recuperar de erros	As mensagens de erro devem ser expressas em linguagem simples, indicando o problema em específico e sugerindo uma solução de forma construtiva.
10. Ajuda e documentação	É melhor que o sistema não precise de explicações adicionais, mas caso haja necessidade, pode fornecer uma documentação para ajudar os usuários a entender como concluir as suas tarefas.

Fonte: (NIELSEN, 2021)

2.4.3.2 Heurísticas de Baker, Greenberg e Gutwin

Quando se trata de sistemas colaborativos, uma série de fatores devem ser considerados para que de fato ele possa promover a colaboração. Pensando nisso,

Baker, Greenberg e Gutwin (2002) desenvolveram um conjunto de heurísticas que dão suporte à avaliação de sistemas desse tipo.

Sendo assim, por meio delas é possível analisar se determinada interface garante a possibilidade de um grupo de pessoas, distantes fisicamente, colaborar por meio de uma interface digital. A lista proposta pelos autores é composta de oito heurísticas, conforme a tabela 2.

Tabela 2 - Heurísticas de Baker, Greenberg e Gutwin

Heurística	Descrição
1. Oferecer aos usuários os meios necessários para estabelecer comunicação verbal intencional e apropriada	As pessoas devem ser capazes de conversar e ouvir a conversa de outros membros do grupo.
2. Oferecer aos usuários os meios necessários para estabelecer comunicação gestual intencional e apropriada	Permitir o uso de gestos para apoiar a conversa como apontar a um objeto ao dizer “aquele”
3. Oferecer comunicação não intencional resultante da interação do indivíduo com o ambiente:	Permitir que outros percebam informações passadas de forma não intencional, como onde a pessoa está ou qual seu foco de atenção
4. Oferecer comunicação não intencional através do uso de artefatos compartilhados (feedthrough)	Feedback é a informação relativa ao uso de um elemento pelo usuário, feedthrough é a informação relativa ao uso do elemento por outro usuário. Por exemplo, saber que o outro membro clicou em um botão.
5. Oferecer segurança	Impedir que usuários atuem ao mesmo tempo sobre um elemento quando isso for gerar conflito ou interferir com a ação do outro.
6. Gerenciar transição entre colaboração mais flexíveis e mais estreitas	Permitir que membros alternem nas formas de trabalhos com outros membros
7. Apoiar os usuários na coordenação de suas ações	Permitir aos membros se comunicarem direta ou indiretamente e oferecer informações sobre as atividades dos demais membros.
8. Facilitar a identificação de colaboradores e ao contato entre eles:	Permitir aos membros saberem quem está presente no ambiente virtual e sua disponibilidade para comunicação.

Fonte: (Franklin, 2014)

2.4.3.3 Grau de severidade

Em uma avaliação heurística é necessário uma padronização quanto ao impacto que o problema encontrado terá na sua utilização. Pensando nisso, Nielsen (1994) sugere um conjunto de graus que auxiliem os avaliadores a metrificar-los. Sendo eles, exibido na tabela 3.

Tabela 3 - Graus de severidade

Severidade	Descrição
Problema cosmético/superficial	Precisa ser corrigido somente se houver tempo disponível.
Problema de usabilidade pequeno	Possui uma baixa severidade para ser solucionado.
Problema de usabilidade grande	Alta prioridade, importante solucionar o problema.
Catástrofe de usabilidade	Necessário solucionar o problema antes que o produto seja liberado para comercialização.

Fonte: Nielsen (1994)

2.5 TRABALHOS RELACIONADOS

Foram analisados algumas pesquisas, que de certa maneira, se relacionam como o objetivo deste trabalho, a seguir serão apresentados 3 trabalhos e seus resultados.

Em um estudo, SCHERER (2018) aplica, de maneira adaptada, às heurísticas de Nielsen para avaliar a usabilidade para softwares de design de interiores. Ou seja, como base na definição de cada heurística há uma apresentação de funcionalidades que aplicações desse tipo devem conter. Com isso, a autora buscou contextualizar os avaliadores sobre o que considerar durante a avaliação.

Já em outro trabalho, FRANKLIN (2014) formalizou um processo de avaliação de sistemas colaborativos remotos de realidade aumentada. Sendo assim, a autora propôs uma lista de heurísticas específicas para esse tipo de sistema. De maneira a validar e refinar a proposta foi concebido um protótipo funcional de baixo custo.

3 - MATERIAIS E MÉTODOS

Neste capítulo são apresentados os materiais e tecnologias utilizadas no trabalho, assim como detalhes da organização e aplicação da avaliação.

3.1 METODOLOGIA

Este trabalho fez uso de um método analítico chamado Avaliação Heurística, no qual especialistas em usabilidade analisam uma interface com o objetivo de encontrar possíveis problemas, considerando um conjunto de princípios determinados chamados heurísticas, podendo eles propor soluções para as violações encontradas.

Na literatura existem diversas heurísticas disponíveis, contudo como o foco deste trabalho foi analisar a experiência do usuário de um ambiente virtual colaborativo para ensino de programação, foram considerados 2 conjuntos: Nielsen (1994) e Baker, Greenberg e Gutwin (2002). Pois, o primeiro engloba questões de usabilidade de sistemas WEB, e o segundo, pontos relacionados a colaboração.

Devido ao cenário de pandemia, a avaliação ocorreu totalmente de maneira remota. Sendo a mesma dividida em 3 etapas:

- Pré-avaliação: Esta fase contou com a preparação do ambiente para dar suporte à avaliação e escolha dos avaliadores. Ou seja, ocorreu a definição das tecnologias, o desenvolvimento dos materiais a serem utilizados, definido o perfil de avaliadores, bem como o convite dos mesmos;
- Avaliação: Já nesta etapa contou com a realização da avaliação em si, no qual os avaliadores tiveram que analisar a interface da aplicação e descrever falhas e problemas em um planilha, de acordo com o guia da avaliação recebido.
- Pós-avaliação: Nesta última etapa, foram analisadas as descrições dos problemas apontados pelos avaliadores, bem como as sugestões de melhorias consideradas necessárias. Além disso, foram prototipadas algumas correções baseadas nelas.

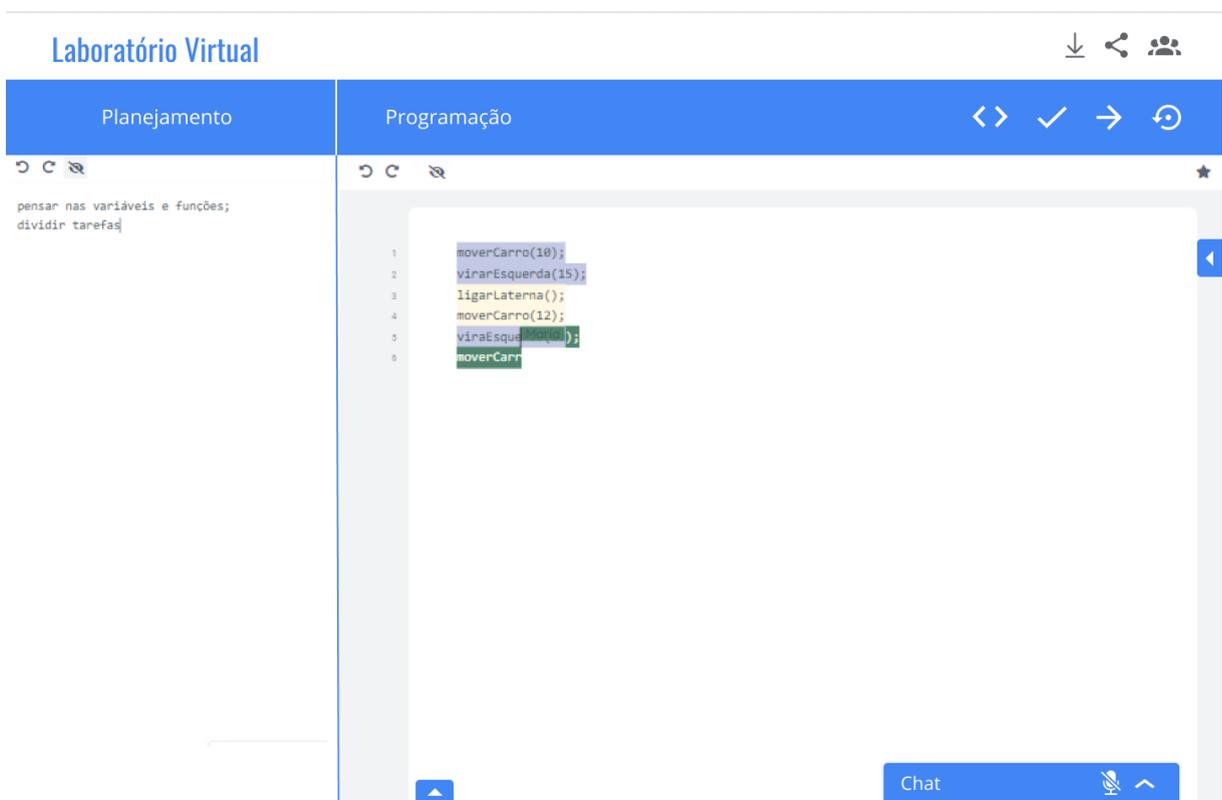
3.2 TECNOLOGIAS E MATERIAIS UTILIZADOS

Nesta seção serão apresentadas as tecnologias e materiais utilizados neste trabalho.

3.2.1 Interface analisada

Para esta avaliação foi escolhido analisar o protótipo de interface do ambiente virtual de aprendizagem colaborativa para o ensino de programação proposto pelo Laboratório Acadêmico em Redes de Aprendizagem (LARA), conforme figura . Visto que a proposta da mesma é oferecer ao usuário a possibilidade de aprender programação de maneira colaborativa, estando assim dentro do escopo considerado por este trabalho.

Figura 2 - Protótipo de interface analisado



Fonte: autor

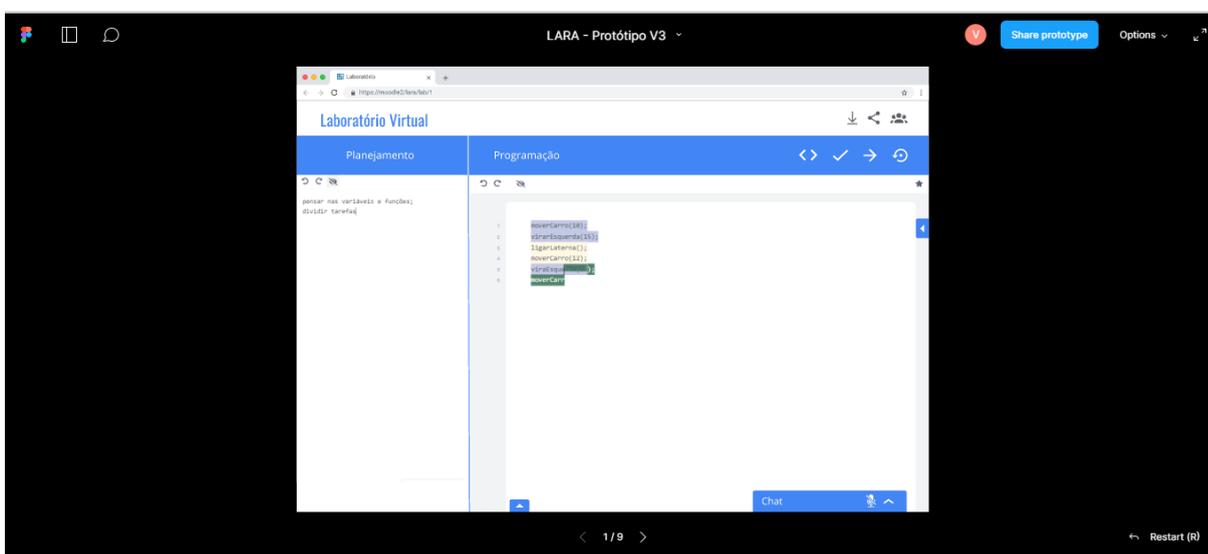
Com o objetivo de centralizar o máximo possível em uma única interface, sem a necessidade de usar outras aplicações para dar suporte à colaboração, esse protótipo oferece alguns componentes, tais como: editor de texto colaborativo, chat, comunicação por voz, gerenciamento de grupo, acompanhamento de histórico de alterações, assim como compartilhamento, download, compilação e execução do código.

Além disso, foi pensado em dois espaços voltados para o desenvolvimento dos algoritmos, um para o grupo planejar como será a aplicação e outro para a programação em si, para isso cada um contará com um editor de texto individual.

De maneira a oferecer ao avaliador uma maior interação com a interface proposta foi utilizado a aplicação web Figma¹. Por meio dela, o usuário pode interagir com a interface, bem como com o seu fluxo de interação. Sendo que, estes foram definidos previamente. Dentre os possíveis fluxos que os avaliadores poderiam analisar estão:

- Analisar as abas de planejamento e codificação;
- Abrir janela de experimento;
- Abrir terminal;
- Abrir a aba do chat e um conversa individual ou do grupo;
- Executar e compilar o código;
- Gerenciar o grupo;
- Acompanhar histórico de alterações;
- Baixar e compartilhar o código.

Figura 3 - Visualização do protótipo por meio do Figma¹



Fonte: autor

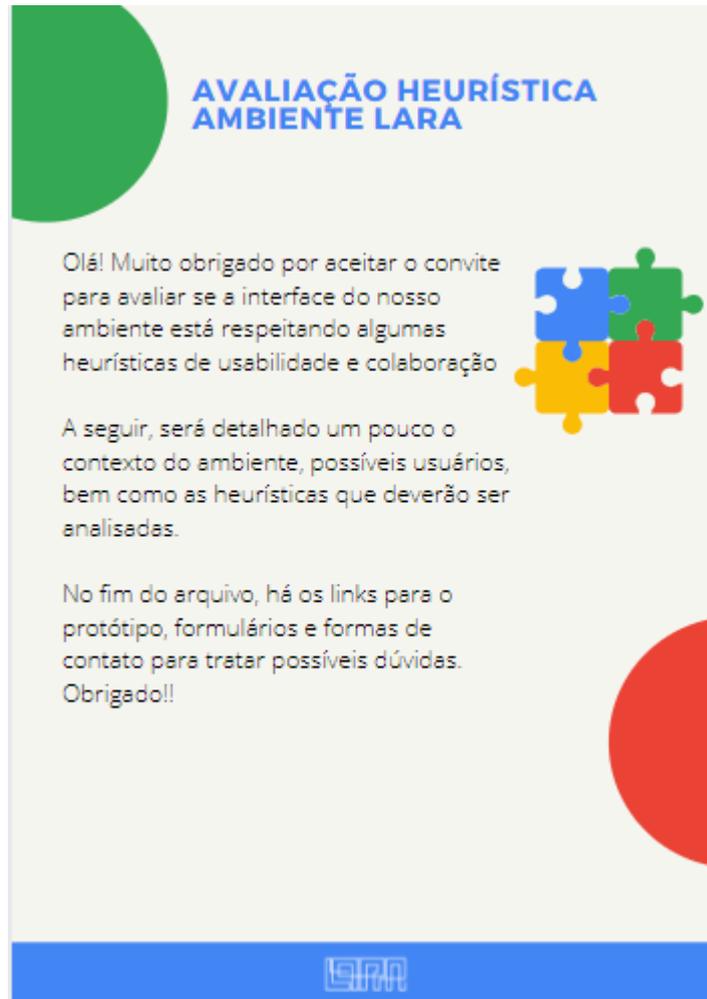
3.2.2 Guia para realização da avaliação

Visando auxiliar os avaliadores durante o processo de avaliação, foi elaborado um guia, sendo o mesmo estruturado da seguinte maneira:

1. Apresentação do projeto e contextualização: nesta seção foi apresentado um resumo da finalidade do projeto e as principais definições relacionadas a interface a ser analisada;
2. Personas: nesta parte foi apresentada 3 personas que são idealizadas como possíveis usuários deste sistema. De maneira a contextualizar melhor os avaliadores, foi apresentado características físicas comportamentais e a necessidade de cada um para com o sistema.
3. Heurísticas: foi exibido as heurísticas a serem consideradas bem como uma breve descrição de cada.
4. Grau de severidade: buscando padronizar o entendimento dos avaliadores sobre a classificação dos problemas encontrados, nesta parte são exibidos os graus de severidade considerados, bem como uma descrição.
5. Links úteis: contém os links para o protótipo a ser considerado e o formulário de avaliação individual de cada avaliador.
6. Informações de contato: seção informava algumas informações de contato, no caso de dúvidas ou problemas durante a avaliação.

Uma vez que a avaliação deveria ocorrer de maneira individual, para que a análise não fosse comprometida ou enviesada, cada avaliador recebeu um guia contendo um link diferente do formulário de avaliação. Sendo assim, apenas ele poderia visualizar e editar as informações nele contidas. Dado que a ferramenta utilizada para suportar o formulário foi o Google Planilhas ², foi possível ainda acompanhar, após a entrega, o histórico de alteração no documento e constatar quem fez as edições, trazendo assim um nível maior de segurança.

Figura 4 - Capa do guia da avaliação



Fonte: autor

De maneira geral, o guia teve boa aceitação pelos avaliadores, pois não houve nenhum contato para tirar dúvidas ou informar problemas. Fora isso, ao retornar feedbacks sobre o processo, alguns deles elogiaram a maneira como foi organizado.

3.3 APLICAÇÃO DA AVALIAÇÃO HEURÍSTICA

Para realizar a avaliação heurística, foram convidados cinco especialistas da área de Interação Humano-Computador, sendo que 4 deles possuem ou estão cursando pós-graduação na área. De acordo com (Nielsen,) esse tipo de avaliação deve possuir entre 3 a 5 pessoas, pois é o número que possibilita melhores resultados.

Uma vez que a avaliação seria realizada de maneira assíncrona, para cada avaliador foi enviado um convite, via e-mail, contendo um guia da avaliação (apêndice A).

Tabela 4 - Descrição dos avaliadores conforme grau de instrução e instituição de ensino

Participante	Grau de instrução	Instituição	Curso
1	Pós- Graduação (Mestrado)	Universidade Federal de São Carlos (UFSCar)	Ciência da Computação
2	Pós-Graduação (Mestrado)	Universidade de São Paulo (USP)	Sistema de informação
3	Pós-Graduação (Mestrado)	Universidade de São Paulo (USP)	Ciências de Computação e Matemática
4	Pós-graduação (Doutorado)	Universidade Federal da Bahia (UFBA)	Ciência da Computação
5	Graduação	Universidade Estadual do Sudoeste da Bahia (UESB)	Ciência da Computação

Fonte: autor

Dessa forma, cada avaliador, de maneira individual, navegou no protótipo da interface do ambiente até encontrar casos que fossem caracterizados como problemas de usabilidade e impeditivos na colaboração dos usuários. Feito isso, eles tiveram que descrever onde estava localizado o problema, realizar uma descrição, classificar de acordo com as heurísticas consideradas e definir um grau de severidade. Além disso, foi questionado possíveis soluções para o problema.

Uma vez identificado um problema de usabilidade, o passo seguinte é procurar uma solução adequada, é importante atribuir a cada problema encontrado um grau de severidade. O problema de usabilidade detectado é avaliado e classificado de acordo com o grau de severidade, ou seja, impacto causado na realização das tarefas (CARRION, 2008).

Para facilitar a análise, os avaliadores receberam um link de uma planilha para descrever tais informações.

4 - RESULTADOS E DISCUSSÃO

A avaliação resultou em 10 heurísticas violadas, sendo 7 das propostas por Nielsen e 3 da de Baker, Greenberg e Gutwin, e permitiu encontrar 18 possíveis problemas na interface do ambiente. Para facilitar o entendimento, a descrição das violações serão divididas de acordo com o conjunto de heurísticas consideradas. Em cada subtópico serão exibidas as heurísticas violadas, descrição do problema, sugestão apresentada e ajuste realizado.

Dessa forma, das 18 heurísticas consideradas, 8 não foram violadas. Contudo, isso não assegura que a aplicação está cobrindo todas essas classificações, apenas que não foi notado nenhum problema pelos avaliadores.

4.1 RESULTADOS DA AVALIAÇÃO HEURÍSTICA - NIELSEN

Este subtópico apresenta a análise dos 14 problemas de usabilidade encontrados durante a avaliação heurística, considerando apenas as heurísticas de Nielsen. Na tabela 5 é possível visualizar as heurísticas violadas, quantidade de ocorrências e os graus de severidade apontados. Para facilitar o entendimento, tais problemas serão melhor descritos de acordo com a heurística violada. Será apresentado uma descrição, local e sugestão apontada pelo avaliador e uma possível correção na interface.

Tabela 5 - Ocorrências de violações das heurísticas de Nielsen

Heurística violada	Problemas encontrados Porcentagem	Grau de severidade			
		1	2	3	4
1. Visibilidade do estado do sistema	1 (7,14%)		1		
2. Compatibilidade entre o sistema e o mundo real	0 (0%)				
3. Controle e liberdade do usuário	1 (7,14%)	1			
4. Consistência e padronização	1 (7,14%)	1			
5. Prevenção de erros	3 (21,49%)		1		2
6. Reconhecimento em vez de memorização	0 (0%)				
7. Flexibilidade e eficiência de uso	0 (0%)				
8. Estética e design minimalista	4 (28,57%)		1	2	1
9. Ajude os usuários a reconhecer, diagnosticar e se recuperar de erros	1 (7,14%)				1
10. Ajuda e documentação	3 (21,49%)			3	
Total	14 (100%)	2	3	5	4

Fonte: autor

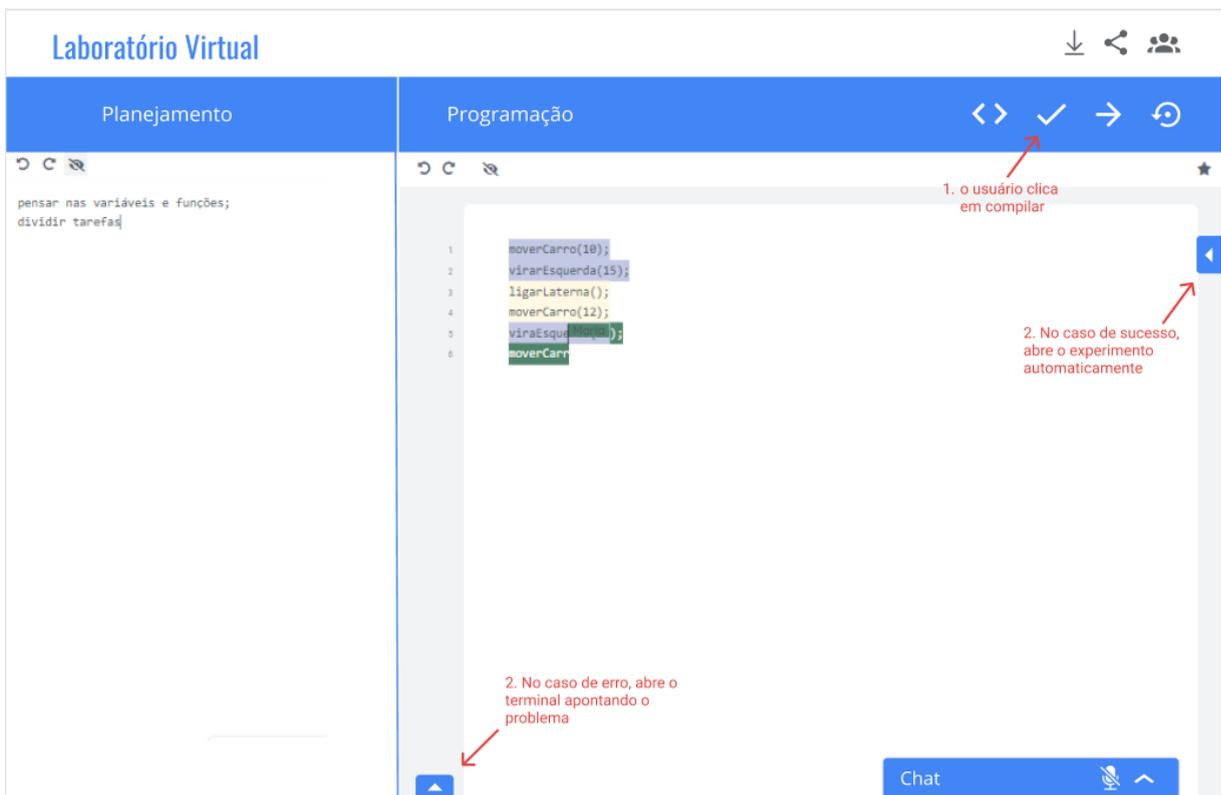
Nenhum dos avaliadores encontrou violações nestas heurísticas: 2 (Compatibilidade entre o sistema e o mundo real), 6 (Reconhecimento em vez de memorização), 7 (Flexibilidade e eficiência de uso). Entretanto, isso não garante que a interface não apresente problemas relacionados a essas 3 heurísticas. Visto que, outros especialistas poderiam encontrar violações que se encaixam nesses grupos.

4.1.1 Visibilidade do estado do sistema

Para esta heurística foi apontado um possível problema de usabilidade, segundo o avaliador 3, ao clicar em compilar o código o usuário deve receber um feedback automático na interface, tanto no caso de sucesso ou de erro. Segundo ele, “vocês podem colocar um tickzinho com certo ou errado caso não abra automaticamente para o usuário para que eles saibam caso haja sucesso ou erro na compilação”. Entretanto, devido ao grau de severidade apontado e o número baixo

de ocorrência, a opção considerada será abrir automaticamente a janela deslizante do experimento no caso de sucesso, ou abrir o terminal no caso de erro, como o próprio avaliador indica.

Figura 5 - Proposta de solução para o problema apontado referente a heurística “visibilidade do estado do sistema”



Fonte: autor

Dessa forma, em qualquer cenário o usuário terá um feedback instantâneo da ação de compilar o código.

4.1.2 Controle e liberdade do usuário

Foi apontado a falta de controle do usuário sobre seu status no sistema. Para o avaliador 5, há uma violação dessa heurística ao não permitir que o usuário escolha ficar inativo, mesmo estando presente. Entretanto, o mesmo pontuou como uma questão que não precisa ser modificada, apenas como entendimento da heurística.

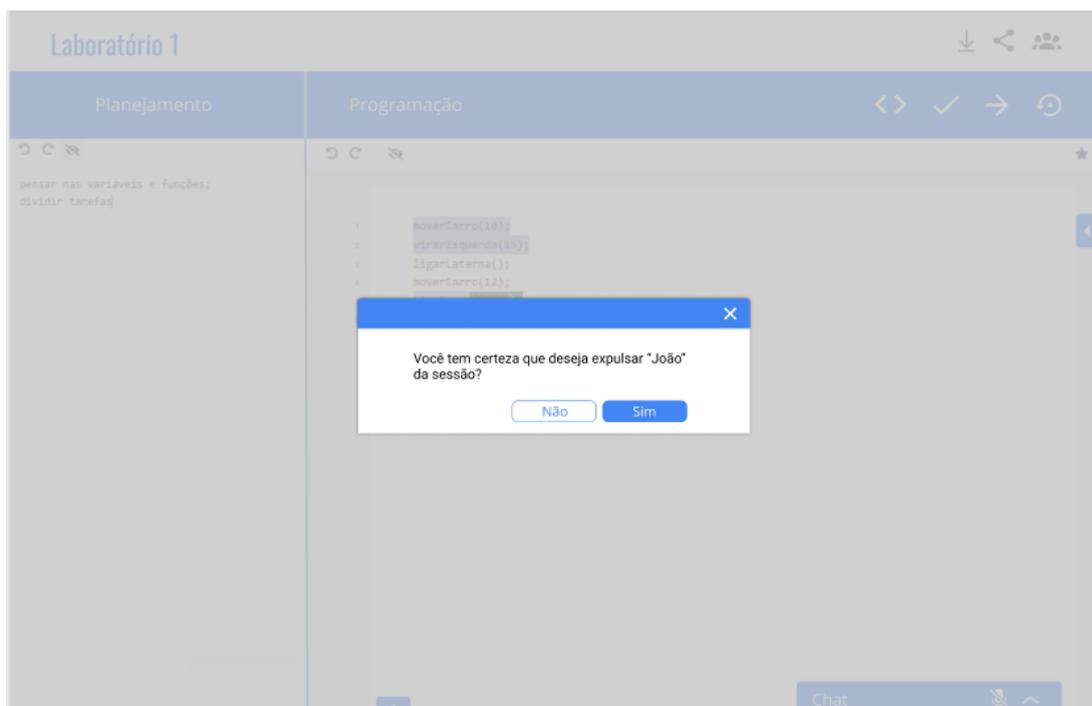
4.1.3 Consistência e padronização

Referente a esta heurística houve apenas uma ocorrência, sendo o grau de severidade apontado extremamente baixo, sendo considerado apenas um problema cosmético. Para o avaliador 3, no espaço de planejamento não há a exibição de linhas tal como no de código, infringindo assim a padronização. Entretanto, para ele isso não se torna algo a ser alterado, devido ao tamanho dedicado ao planejamento.

4.1.4 Prevenção de erros

Essa heurística apresentou uma taxa alta de ocorrência, sendo que dois avaliadores apontaram a falta de mecanismos de confirmação em ações críticas, sendo elas: silenciar algum usuário e expulsar alguém da sala. Para eles, devem ser exibidos caixas de confirmação antes da ação ser concluída, evitando assim possíveis erros. De acordo com o avaliador 4, “alguma tela que apresente aos usuários uma opção de confirmação antes que eles se comprometam com a ação”. Dessa forma, foi adicionado um modal para confirmação dessas ações mais críticas, conforme figura 5.

Figura 6 - Proposta de solução para o problema apontado referente a heurística “prevenção de erros”



Fonte: Autor

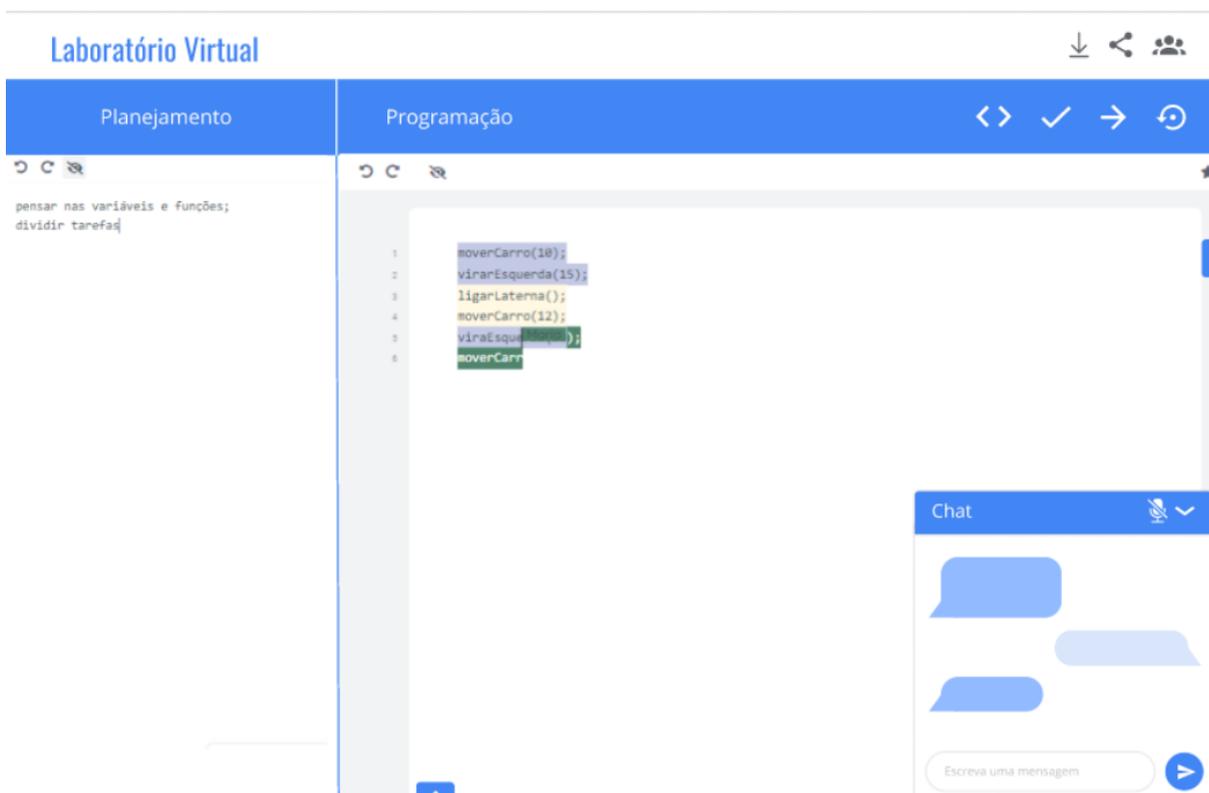
Além disso, outro problema apontado foi a inexistência de um guia para auxiliar o usuário a entender as ações que o sistema é capaz de realizar e uma explicação sobre cada uma delas, evitando assim possíveis erros. Entretanto, como essa característica está relacionada à heurística “Ajuda e documentação”, a solução apresentada será detalhada no tópico específico.

4.1.4 Estética e design minimalista

Considerando esta heurística os avaliadores encontraram 4 violações, sendo assim a com mais ocorrências na avaliação.

Segundo o avaliador 3, no editor de texto fica bastante confusa a identificação de alguns autores, pois há pouco contraste entre as determinadas cores, que acabam dificultando o entendimento, conforme imagem abaixo. De acordo com ele, “é necessário uma mudança no tom de azul e verde, tomar cuidado com o contraste de branco e cinza claro, está baixo”. Dessa forma, é necessário definir uma paleta de cores com um maior contraste.

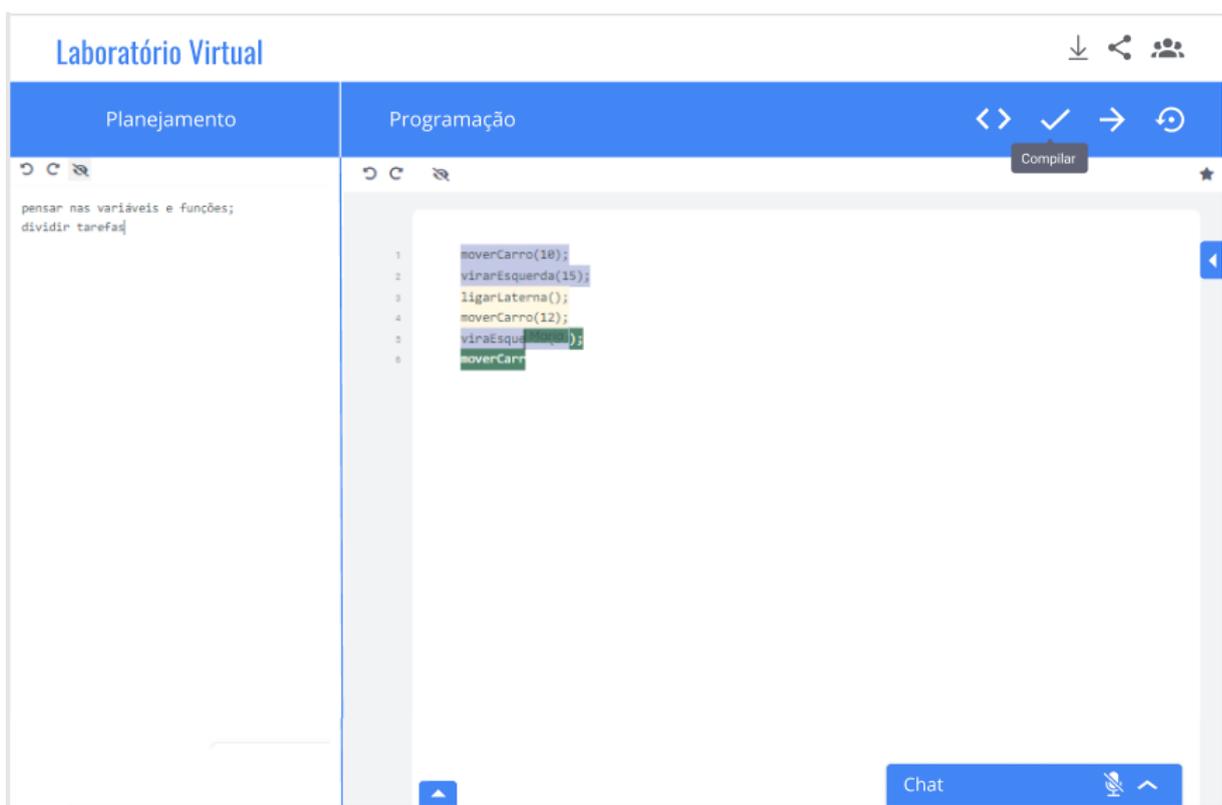
Figura 7 - Problema reportado por um dos avaliadores referente a heurística “Estética e design minimalista”



Fonte: autor

Outro ponto abordado foi o tamanho de determinados ícones e a opção de ícones escolhidos. Segundo o avaliador 4, alguns deles estão com tamanho desproporcional. Nas palavras dele, é necessário “diminuir um pouco os ícones, vai deixar a estética mais suave, está um pouco bruto pelo tamanho, o que também irá ajudar com ajuste à tela”. Sendo assim, o tamanho dos ícones serão redimensionados. Já para o avaliador 5, ficou difícil entender o que determinados ícones significavam, tais como o de compilar o código e histórico de alterações. Deste modo, foi sugerido a adição de tooltips ou dica de contexto, ou seja, aquelas molduras flutuantes que aparecem ao colocar o ponteiro do mouse em determinado ícone ou botão.

Figura 8 - Proposta de solução para o problema apontado referente a heurística “visibilidade do estado do sistema”

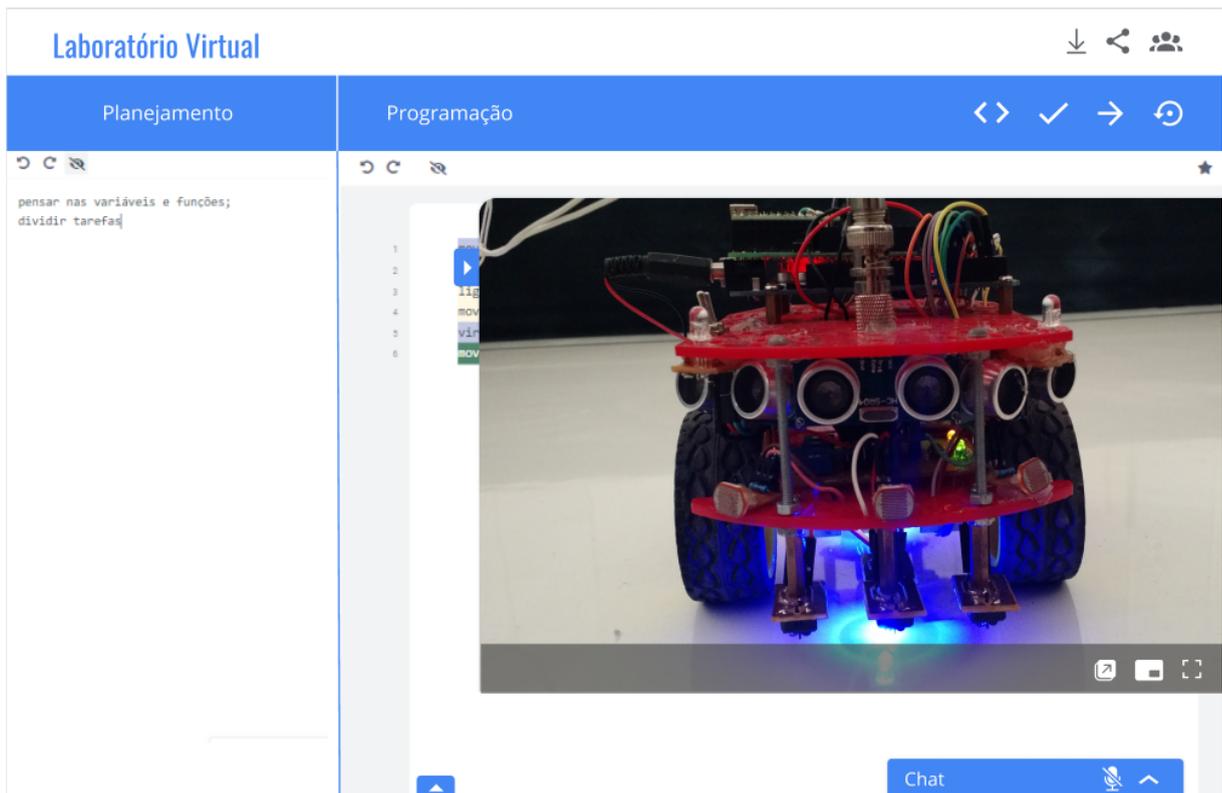


Fonte: autor

Por fim, o último problema relacionado a essa heurística, é a sobreposição da janela deslizante para a exibição do experimento. Para o avaliador 4, “A tela do objeto poderia ser exibida em uma tela paralela ao código ou com a possibilidade de ao clicar no objeto poder colocar ele na posição que desejar”. Sendo assim,

considerando esses dois pontos como possíveis novas funcionalidades, foi sugerido adicionar uma barra de comando dentro do player de vídeo, com as opções de miniplayer (sendo possível arrastar), tela cheia e abrir em outra janela.

Figura 9 - Proposta de solução para o problema apontado referente a heurística “visibilidade do estado do sistema”

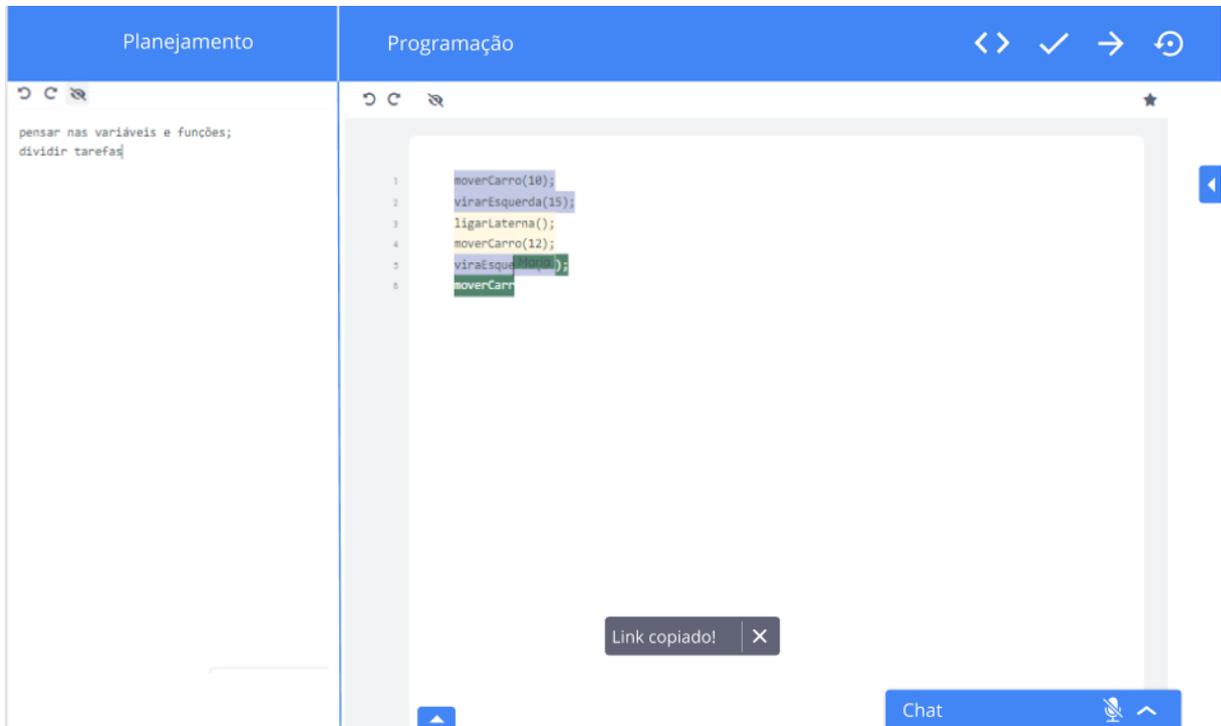


Fonte: autor

4.1.5 Ajude os usuários a reconhecer, diagnosticar e se recuperar de erros

Referente a essa heurística, foi apontado pelo avaliador 1, a falta de feedbacks ao usuário, seja após executar uma ação, ou na ocorrência de erro. Sendo sugerido a criação de algum componente de feedback para esses casos. Dessa forma, foi prototipado um componente de feedback, conforme o exemplo exibido na figura 9.

Figura 10 - Proposta de solução para o problema apontado referente a heurística “Ajude os usuários a reconhecer, diagnosticar e se recuperar de erros”

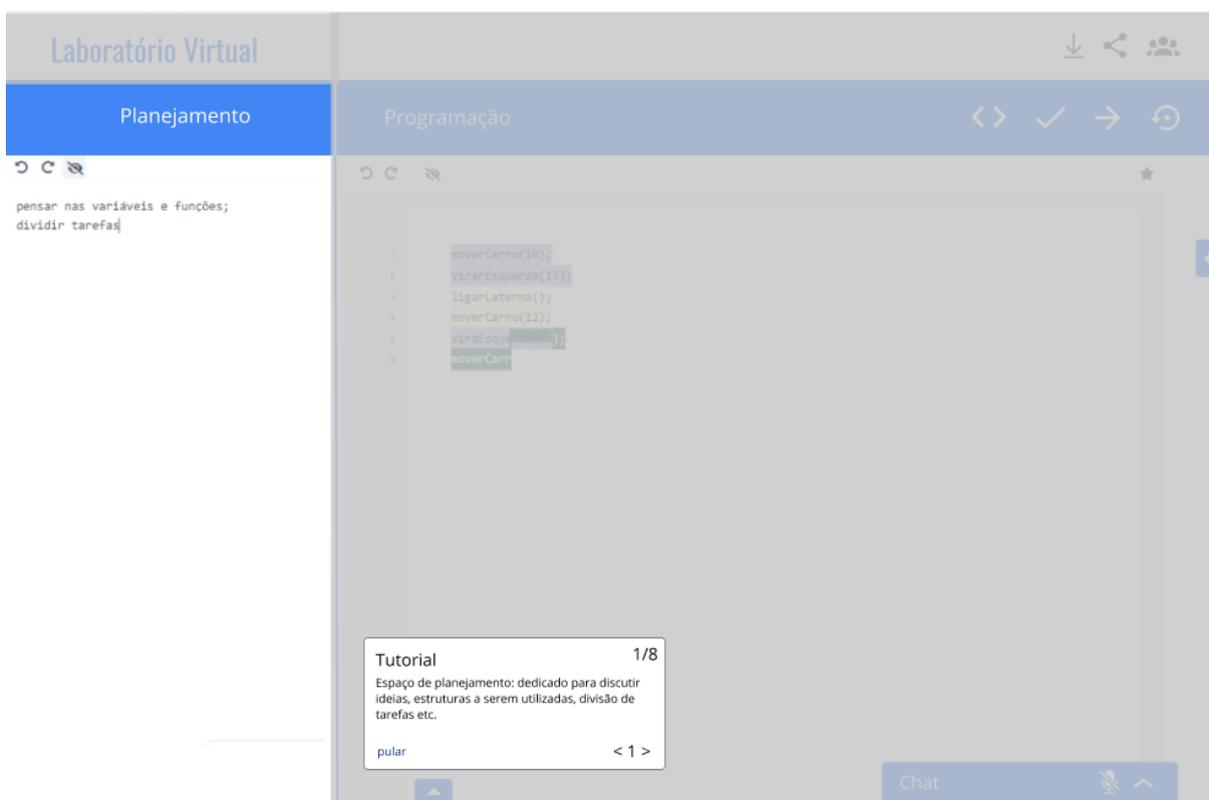


Fonte: autor

4.1.6 Ajuda e documentação

Essa heurística apresentou uma considerável taxa de ocorrência de violações, sendo 3 ocorrências de grau 3, um problema na interface dos sistemas. Visto que, 3 avaliadores apontaram problema na mesma questão, ajuda e suporte. Nas palavras do avaliador 3, “Não encontrei a função de ajuda ou suporte no protótipo”. Desses, dois sugeriram criar alguma espécie de manual de primeiro uso, para ambientar novos usuários a utilizar o sistema, já o outro sugeriu criar uma página de ajuda. Pensando em além de ajudar o usuário e dar mais informações sobre os componentes do ambiente, foi prototipado um sistema de guia de primeiro acesso, que auxilia os usuários a entender as funcionalidades da aplicação, conforme figura 10.

Figura 11 - Proposta de solução para o problema apontado referente a heurística “Ajude os usuários a reconhecer, diagnosticar e se recuperar de erros”



Fonte: autor

Este mecanismo é capaz de auxiliar o usuário, dando mais informações sobre determinado elemento, bem como dicas de uso. Para pessoas com uma maior facilidade ou experiência com a aplicação, é possível pular e ir direto para o uso, respeitando assim o controle e a liberdade do usuário.

4.2 RESULTADOS DA AVALIAÇÃO HEURÍSTICA - BAKER, GREENBERG E GUTWIN

Este subtópico detalha a análise dos 4 problemas de usabilidade encontrados durante a avaliação heurística, considerando apenas as heurísticas de Baker, Greenberg e Gutwin. Na tabela 6 é possível visualizar as heurísticas violadas, quantidade de ocorrências e os graus de severidade apontados. Para facilitar o entendimento, tais problemas são melhor descritos de acordo com a heurística violada. Será apresentado uma descrição, local e sugestão apontada pelo avaliador e uma possível correção na interface.

Tabela 6 - Ocorrências de violações das heurísticas de Baker, Greenberg e Gutwin

Heurística violada	Problemas encontrados	Grau de severidade			
	Porcentagem	1	2	3	4
1. Oferecer aos usuários os meios necessários para estabelecer comunicação verbal intencional e apropriada	0 (0%)				
2. Oferecer aos usuários os meios necessários para estabelecer comunicação gestual intencional e apropriada	0 (0%)				
3. Oferecer comunicação não intencional resultante da interação do indivíduo com o ambiente:	0 (0%)				
4. Oferecer comunicação não intencional através do uso de artefatos compartilhados (feedthrough)	1 (25%)	1			
5. Oferecer segurança	0 (0%)				
6. Gerenciar transição entre colaboração mais flexíveis e mais estreitas	0 (0%)				
7. Apoiar os usuários na coordenação de suas ações	1 (25%)		1		
8. Facilitar a identificação de colaboradores e ao contato entre eles:	2 (50%)		1	1	
Total	4 (100%)	1	2	1	

Fonte: autor

Nenhum dos avaliadores encontrou violações nestas heurísticas: 1 (Oferecer aos usuários os meios necessários para estabelecer comunicação verbal intencional e apropriada), 2 (Oferecer aos usuários os meios necessários para estabelecer comunicação gestual intencional e apropriada), 3 (Oferecer comunicação não intencional resultante da interação do indivíduo com o ambiente), 5 (Oferecer segurança) e 6 (Gerenciar transição entre colaboração mais flexíveis e mais estreitas). Porém, isso não garante que a interface não apresente problemas relacionados a essas 5 heurísticas. Visto que, outros especialistas poderiam encontrar violações que se encaixam nesses grupos.

4.2.1 Oferecer comunicação não intencional através do uso de artefatos compartilhados (feedthrough)

Para o avaliador 5, “seria interessante acompanhar o ponteiro do mouse dos colegas da equipe, pois seria possível ver o que cada um está fazendo”. Segundo ele, isso permitirá também explicar melhor uma parte do código ou função da aplicação. Entretanto, ele pontuou apenas com grau 1, ou seja, algo apenas estético. Visto que é possível se comunicar de outras formas e acompanhar a escrita dos demais.

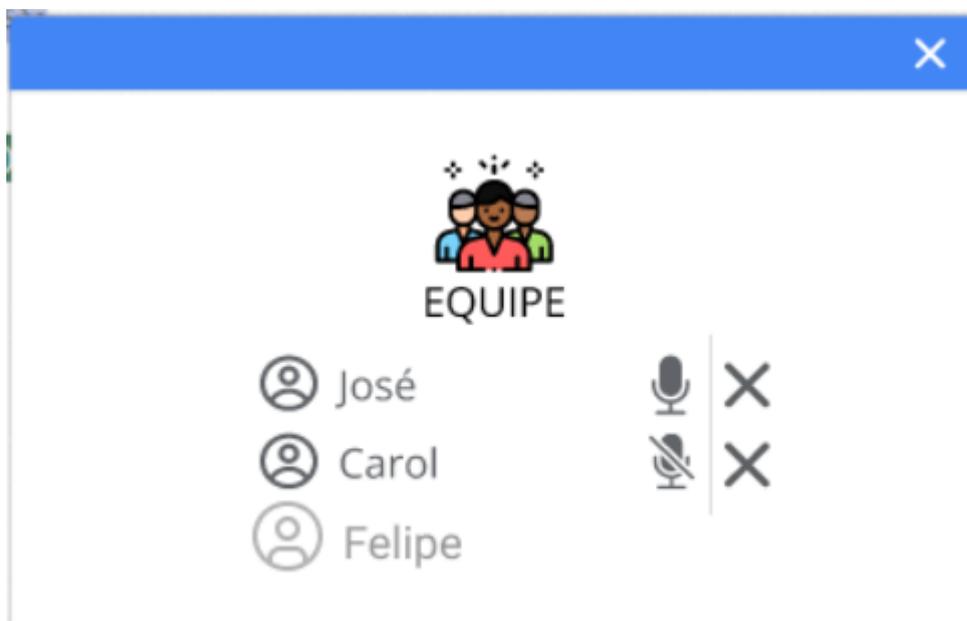
4.2.2 Apoiar os usuários na coordenação de suas ações

De maneira a auxiliar informar uma pessoa específica do grupo, para o avaliador 5, seria bastante útil a possibilidade de marcar uma pessoa na conversa do grupo, sendo esta notificado de algum modo na interface. Entretanto, ele definiu um grau de severidade 2, pois é possível chamar a atenção pela comunicação oral (quando síncrona), ou mandar uma mensagem diretamente para a pessoa (assíncrona).

4.2.3 Facilitar a identificação de colaboradores e ao contato entre eles

Segundo o avaliador 2, esta heurística é violada uma vez que a interface não apresenta o status do usuário no momento, ou seja, se ele tá online ou não. De acordo com ele, uma sugestão de melhoria seria “Ao clicar no ícone de usuários, mostrar os usuários inativos com uma cor mais clara, enquanto que usuários ativos uma cor escura”. Tornando assim, mais fácil a indicação de quem está disponível no momento.

Figura 12 - Proposta de solução para o problema apontado referente a heurística “Facilitar a identificação de colaboradores e ao contato entre eles”



Fonte: autor

4.3 CONSIDERAÇÕES FINAIS

A avaliação heurística no protótipo da interface possibilitou que problemas de usabilidade e na questão de colaboração fossem encontrados antes da aplicação ser desenvolvida, ou seja, evitando assim problemas e retrabalho no futuro.

Em relação às heurísticas de Nielsen, o problema mais apontado pelos especialistas foi em relação à heurística “Estética e design minimalista”, que em termos gerais foi direcionada à falta de contraste entre determinadas cores, tamanho e sobreposição de determinados componentes junto a outros. Contudo, de modo geral, alterações foram realizadas para evitar possíveis problemas no futuro.

Quanto às de Baker, o número de ocorrências foi relativamente baixo, sendo a heurística “Facilitar a identificação de colaboradores e ao contato entre eles” com 2 casos, sendo que ambos tratavam da mesma questão: a visibilidade do status do usuário no ambiente. Porém, conforme mencionado acima, as questões foram tratadas.

Com isso, os dois conjuntos de heurísticas proporcionaram avaliar a experiência do usuário como um todo neste tipo de aplicação. Visto que, a usabilidade foi coberta por meio das heurísticas de Nielsen, já as de colaboração por Baker.

5 - CONCLUSÃO

Este trabalho teve como objetivo propor uma avaliação de sistemas colaborativos para ensino de programação voltado para a experiência do usuário como um todo, tanto nas questões de usabilidade quanto nas de colaboração. dessa forma, diferente das pesquisas citadas em Trabalhos relacionados, ele utilizou de maneira simultânea dois conjuntos de heurísticas distintos, cobrindo assim o escopo considerado.

Os resultados dessa avaliação apontaram diferentes tipos de erros, bem como possíveis soluções para sanar tais violações. Os principais problemas encontrados foram referente a questões estéticas e status do usuário, que para os avaliadores iriam prejudicar a utilização da aplicação referente ao objetivo proposto.

De modo geral, a maior contribuição deste trabalho foi desenvolver um modo de avaliação assíncrono que considera esses dois conjuntos de heurísticas para melhor avaliar sistemas colaborativos para ensino de programação, dado cenários de baixo investimento, pouco tempo para execução ou dificuldades em interagir com possíveis usuários. Entretanto, vale lembrar que ele não exclui a possibilidade de utilizar outros métodos.

Como trabalhos futuros, propõe-se avaliar as alterações realizadas com possíveis usuários para obter feedbacks ou até mesmo encontrar outros problemas na experiência do usuário. Pois, tal avaliação não deve ser considerada auto suficiente, e quanto mais métodos de avaliação mais coberto a falhas a aplicação estará.

REFERÊNCIAS

ALVES, G; REBOUÇAS, A; SCAICO, P. Coding Dojo como Prática de Aprendizagem Colaborativa para Apoiar o Ensino Introdutório de Programação: Um Estudo de Caso. **WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO (WEI)**, Belém, p. 276-290, 2019.

BARANAUSKAS, Maria Cecília Calani; SOUZA, Clarisse Sieckenius de. Desafio 4: **Acesso participativo e universal do cidadão brasileiro ao conhecimento**. Computação Brasil, v. 7, n. 23, p. 7, 2006.

BAX, Marcello Peixoto. **Design science**: filosofia da pesquisa em ciência da informação e tecnologia. Ciência da Informação, v. 42, n. 2, 6 ago. 2015.

BRITTO, Nathalino P.; FURTADO, Maria Elizabeth S.; CARDOSO, Rafaela PL. **Uma Estratégia para Institucionalização de Iniciativas para Interdisciplinaridade de IHC aplicada ao Ensino de Programação**. In: Anais Estendidos do XVII Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais. SBC, 2018.

CAMBRUZZI, Eduardo; SOUZA, R. M. O **uso da robótica educacional para o ensino de algoritmos**. Anais do Encontro Anual de Tecnologia da Informação e Semana Acadêmica de Tecnologia da Informação. Universidade Federal de Santa Maria, Frederico Westphalen-RS, 2014.

CARRION, Wellington. **Design para Webdesigners**. Rio de Janeiro : Brasport, 2008

CASTRO, Alberto; MENEZES, Crediné. **Aprendizagem colaborativa com suporte computacional**. Pimentel, M. e Fuks, H. Sistemas Colaborativos. Rio de Janeiro: Campus. ISBN, p. 978-85, 2011.

COELHO, WILLYANS GARCIA; TEDESCO, PATRICIA CABRAL DE AZEVEDO RESTELLI. **A percepção do outro no ambiente virtual de aprendizagem: presença social e suas implicações para Educação a Distância**. Revista Brasileira de Educação, v. 22, n. 70, p. 609-624, 2017.

CORNEY, M; TEAGUE, D; THOMAS, R.N. Engaging students in programming. **Conferences in Research and Practice in Information Technology Series**, v. 103, p. 63-72, 2010.

DRESCH, Aline; LACERDA, Daniel Pacheco; JÚNIOR, José Antonio Valle Antunes. **Design science research**: método de pesquisa para avanço da ciência e tecnologia. Bookman Editora, 2015.

ELLWANGER, Cristiane; SANTOS, Cristina Paludo; MOREIRA, Guilherme Jantsch. **Padrões de interface, gameificação e mobilidade no desenvolvimento de aplicações educacionais**. In: Proceedings of the 13th Brazilian Symposium on Human Factors in Computing Systems. 2014. p. 365-368.

ESTEVEES, Micaela et al. **Contextualização da aprendizagem da programação: estudo exploratório no Second Life**. Proceedings of IADIS Ibero-Americana WWW/Internet, p. 7-8, 2007.

FONSECA, David et al. **Informal interactions in 3D education: Citizenship participation and assessment of virtual urban proposals**. Computers in Human Behavior, v. 55, p. 504-518, 2016.

FRANKLIN, Fernanda Freire. **Heurísticas de usabilidade para sistemas colaborativos remotos de realidade aumentada**. Recife, 2014. 107 p. Dissertação - Universidade Federal do Pernambuco.

GLORIA, Heloisa de Souza. **Avaliação de um conjunto de heurísticas de usabilidade para aplicativos de smartphones na área da saúde por meio de testes de usabilidade**. 1-75, 2015.

HÄMÄLÄINEN, R. **Designing an evaluating collaboration in a virtual game environment for vocational learning**. Computers & Education, V. 50, P.98-109, 2008.

HELMINEN, J; MALMI, L. Jype - a program visualization and programming exercise tool for python. **Proceedings of the ACM Conference on Computer and Communications Security**, p. 153-162, 2010.

HU, L.L; TSENG, S.S; LEE, T.J. Towards scaffolding problem-solving implementation process in undergraduate programming course. **Proceedings - 2013 IEEE 13th International Conference on Advanced Learning Technologies**, v. icalt 2013, p. 417-418, 2013.

IBRAHIM, R; JAAFAR, A. Using educational games in learning introductory programming: A pilot study on students' perceptions. **Proceedings 2010 International Symposium on Information Technology - Visual Informatics**, v. 1, 2010.

JANTAN, S.Rb; ALJUNID, S.A. An experimental evaluation of scaffolded educational games design for programming. **2012 IEEE Conference on Open Systems**, v. icos 2012, 2012.

LACERDA, Daniel Pacheco et al. **Design Science Research: método de pesquisa para a engenharia de produção**. Gestão & produção, v. 20, n. 4, p. 741-761, 2013.

LAND, S.M., ZEMBAL-SAUL, C. **Scaffolding reflection and articulation of scientific explanations in a data-rich, project-based learning environment: An investigation of progress portfolio**. Educational Technology Research and Development, v. 51, p. 65-84, 2003.

LOISELLE, Jean. **A exploração da multimídia e da rede internet para favorecer a autonomia dos estudantes universitários na aprendizagem**. Ciberespaço e formações abertas: rumo a novas práticas educacionais, p. 107-116, 2002.

LOPES, Maisa Soares et al. **Web environment for programming and control of a mobile robot in a remote laboratory**. IEEE Transactions on Learning Technologies, v. 10, n. 4, p. 526-531, 2016.

MEERBAUM-SALANT, O; ARMONI, M; BENARI, M. Learning computer science concepts with scratch. **ICER'10 - Proceedings of the International Computing Education Research Workshop**, p. 69-76, 2010.

MURPHY, Elizabeth. **Recognising and promoting collaboration in an online asynchronous discussion**. British Journal of Educational Technology, v. 35, n. 4, p. 421-431, 2004.

NIELSEN, Jakob. **Usability 101: Introduction to usability**. NNGroup, 2012. Disponível em: <<https://www.nngroup.com/articles/usability-101-introduction-to-usability/>>. Acesso em: 20/05/2021.

NIELSEN, J.; MOLICH, R.. Heuristic evaluation of user interfaces - Disponível em <http://www.acm.org/dl>. Seattle, Washington, United States, 1990.

NIELSEN, Jakob; MACK, Robert L. Usability Inspection Methods Computer. [S.l.]: John Wiley & Sons, 1994. ISBN 0471018775.

NIELSEN, Jakob. 10 Usability Heuristics for User Interface Design. Disponível em: <https://www.nngroup.com/articles/ten-usability-heuristics/#poster>.

NUNES, Felipe Becker et al. **Systematic review of virtual worlds applied in education**. In: Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE). 2016.

PANWONG, P; KEMAVUTHANON, K. Problem based learning framework for junior software developer: Empirical study for computer programming students. **Wireless Personal Communications**, v. 76(3), p. 603-6013, 2014.

PITEIRA, M; HADDAD, S.R. Innovate in your program computer class: An approach based on a serious game. **ACM International Conference Proceeding Series**, p. 49-54, 2011.

PITEIRA, M; COSTA, C. Learning computer programming: Study of difficulties in learning programming. **ACM International Conference Proceeding Series**, p. 75-80, 2013.

PRATES, Raquel Oliveira; DE SOUZA, Clarisse Sieckenius. **Extensão do teste de comunicabilidade para aplicações multi-usuário**. Cadernos do IME-Série Informática, v. 13, p. 51-62, 2002.

PREECE, J; ROGERS, Y; SHARP, H. **Design de Interação: Além da Interação Humano-Computador**. São Paulo: Bookman, 2005.

PRESSMAN, R.S. **Engenharia de Software: Uma abordagem profissional**. Tradução de Ariovaldo Griesi e Mario M. Fecchio. Porto Alegre: AMGH Editora, 2011.

ROGERS, Yvonne; SHARP, Helen; PREECE, Jennifer. **Design de Interação: Além da Interação Humano Computador**. Bookman Editora, 2013.

ROSA, Jean; MATOS, Ecivaldo. **Considerando aspectos culturais no (re) design da interação de Ambientes Virtuais de Aprendizagem**. In: Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE). 2015. p. 852.

SCHERER, N. **Avaliação heurística e teste de usabilidade para softwares de design de interiores**. Campo Mourão, 2018. 53 p. Monografia - Universidade Tecnológica Federal do Paraná

SERRANO-CÁMARA, Luis Miguel et al. **An evaluation of students' motivation in computer-supported collaborative learning of programming concepts**. Computers in human behavior, v. 31, p. 499-508, 2014.

SHI, J; WHITE, S. Work-in-progress: Learning to program in a connected world. **Proceedings - 2013 Learning and Teaching in Computing and Engineering**, p. 229-232, 2013.

SOUZA, Draylson Micael; DA SILVA BATISTA, Marisa Helena; BARBOSA, Ellen Francine. **Problemas e dificuldades no ensino de programação: Um mapeamento sistemático**. Revista Brasileira de Informática na Educação, v. 24, n. 1, p. 39, 2016.

STAHL, G.; KOSCHMANN, T.; SUTHERS, D. CSCL: **An Historical Perspective on Computer-supported collaborative learning**. Cambridge handbook of the learning sciences, p. 409-426, 2006.

WINCKLER, Marco; PIMENTA, Marcelo Soares. **Avaliação de usabilidade de sites web**. Escola de Informática da SBC SUL (ERI 2002) ed. Porto Alegre: Sociedade Brasileira de Computação (SBC), v. 1, p. 85-137, 2002.

AVALIAÇÃO HEURÍSTICA AMBIENTE LARA

Olá! Muito obrigado por aceitar o convite para avaliar se a interface do nosso ambiente está respeitando algumas heurísticas de usabilidade e colaboração



A seguir, será detalhado um pouco o contexto do ambiente, possíveis usuários, bem como as heurísticas que deverão ser analisadas.

No fim do arquivo, há os links para o protótipo, formulários e formas de contato para tratar possíveis dúvidas. Obrigado!!

SISTEMA COLABORATIVO

É um software que apoia o trabalho em grupo. Skip Ellis o definiu como um "sistema baseado em computador que auxilia grupos de pessoas envolvidas em tarefas comuns e que provê interface para um ambiente compartilhado".

AMBIENTE LARA

É um ambiente colaborativo voltado para o ensino de programação, ele está sendo projeto para integrar um Ambiente Virtual de Aprendizagem - AVA. Dessa forma, os estudantes matriculados em determinado curso terão a possibilidade de realizar atividades de maneira conjunta.

EDITOR DE TEXTO COLABORATIVO

É uma aplicação que permite que os autores editem simultaneamente um documento de texto, no nosso caso o código, e vejam todas as edições dos participantes em tempo real, com a capacidade de exibir o texto de cada autor em sua própria cor, posição do ponteiro do colega etc.



PERSONAS



Informações gerais

Idade: 21 anos

Formação: IV Semestre de

Ciências da Computação

Cidade: Guanambi – BA

Estado Civil: Solteira

Características

É uma jovem que adora filmes, é muito corajosa e independente e adora tomar frente de novas ideias

Camila Fontana Lordosa

Estudante Universitária

Descrição

Camila é universitária e já está na metade do curso, por causa da pandemia os métodos de ensino mudaram. Agora passa boa parte do tempo em casa conectada virtualmente. Mora com a avó, e costuma ir pra casa da amiga aos sábados para conversar.

Relação com o LARA

Camila faz sua primeira matéria por meio desta nova ferramenta. Camila se sente confortável com o sistema, acha bastante interativo as várias formas de se comunicar com os colegas.

Cleylbisson Silva Amorine

Estudante Universitário

Descrição

Cleylbisson, que gosta de ser chamado de “Cley” ou “Bil”, acaba de ingressar no curso, está cheio de expectativas. Encontra-se um pouco nervoso por que viu apenas um vídeo sobre programar, animação de objetos, não sabe como vai ser o curso.

Gosta de ler bastante nas horas vagas, além de ser competitivo e adorar fazer maratonas online.

Relação com o LARA

Cleylbisson tem como objetivo realizar uma atividade em grupo da disciplina programação I.



Informações gerais

Idade: 17 anos

Formação: I Semestre de

Ciências da Computação

Cidade: Itapicurú – BA

Estado Civil: Solteiro

Características

É orgulhoso tende fazer e aprender as coisas por si só.



PERSONAS



Informações gerais

Idade: 29 anos

Formação: VIII Semestre
de Ciências da
Computação

Cidade: Anajé - BA

Estado Civil: Casado

Características

João é competitivo, e quando faz algo gosta de fazer o melhor, só não é muito a fim de trabalho em equipe.

João Rafael Arcanjo dos Santos

Estudante Universitário

Descrição

João está para concluir o curso, além do TCC está com uma matéria atrasada, e por causa da pandemia faz estágio a distancia, o que tornam seus dias lotados de trabalho virtual.

João mora com a esposa, gosta de fazer jantares românticos para os dois. Tem muitos amigos, gosta de conversar bastante e costuma sempre fazer um churrasquinho com samba ao finais de semana.

Relação com o LARA

João está terminando o último trabalho da matéria. Por meio do chat e do ambiente colaborativo, ele consegue realizar os trabalhos em grupo.



HEURÍSTICAS CONSIDERADAS

A seguir, serão apresentadas as heurísticas que deverão ser consideradas, com uma breve descrição. Caso sinta alguma dúvida sobre alguma definição, pode consultar na Internet ou entrar em contato conosco.



Heurísticas de Nielsen

Heurística	Descrição
Visibilidade do status do sistema	O sistema sempre deve manter os usuários informados sobre o que está acontecendo, por meio de feedback apropriado dentro de um prazo razoável.
Compatibilidade entre o sistema e o mundo real	O sistema deve falar o idioma do usuário, com palavras, frases e conceitos familiares ao usuário, em vez de termos orientados ao sistema. Siga as convenções do mundo real, fazendo as informações aparecerem em uma ordem natural e lógica.
Controle e liberdade para o usuário	Os usuários geralmente escolhem as funções do sistema por engano e precisam de uma "saída de emergência" claramente definida para deixar o estado indesejado sem ter que passar por um diálogo prolongado.
Consistência e Padronização	Os usuários não devem ter que se perguntar se palavras, situações ou ações diferentes significam a mesma coisa.
Prevenção de erros	Melhor do que boas mensagens de erro, é um design que evita a ocorrência de um problema. Elimine condições propensas a erros ou verifique-as e apresente aos usuários uma opção de confirmação antes que eles se comprometam com a ação.
Reconhecimento em vez de memorização	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 do diálogo para outra. As instruções de uso do sistema devem ser visíveis ou facilmente recuperáveis sempre que apropriado.
Estética e design minimalista	Os diálogos não devem conter informações irrelevantes ou raramente necessárias. Cada unidade extra de informação em um diálogo concorre com as unidades relevantes de informação e diminui sua visibilidade relativa.

Heurística	Descrição
Ajude os usuários a reconhecerem, diagnosticarem e recuperarem-se de erros	As mensagens de erro devem ser expressas em linguagem simples, indicar com precisão o problema e sugerir uma solução.
Ajuda e documentação	Mesmo que seja melhor se o sistema puder ser usado sem documentação, pode ser necessário fornecer ajuda e documentação. Essas informações devem ser fáceis de pesquisar, focadas na tarefa do usuário, listar etapas concretas a serem executadas e não serem muito grandes.

Heurísticas de Baker, Greenberg e Gutwin

Heurística	Descrição
Oferecer aos usuários os meios necessários para estabelecer comunicação verbal intencional e apropriada	As pessoas devem ser capazes de conversar e ouvir a conversa de outros membros do grupo.
Oferecer aos usuários os meios necessários para estabelecer comunicação gestual intencional e apropriada.	Permitir o uso de gestos para apoiar a conversa como apontar a um objeto ao dizer "esse" ou "aquele".

Heurística	Descrição
Oferecer comunicação não intencional resultante da interação do indivíduo com o ambiente.	Permitir que outros percebam informações passadas de forma não intencional, como onde a pessoa está ou qual seu foco de atenção.
Oferecer comunicação não intencional através do uso de artefatos compartilhados (feedthrough).	Feedback é a informação relativa ao uso de um elemento pelo usuário, feedthrough é a informação relativa ao uso do elemento por outro usuário. Por exemplo, saber que outro membro clicou em um botão.
Oferecer segurança	Impedir que usuários atuem ao mesmo tempo sobre um elemento quando isso for gerar conflito ou interferir com a ação de outro.
Gerenciar transição entre colaboração mais frouxas e mais estreitas	Permitir que membros alternem nas formas de trabalhos com outros membros.
Apoiar os usuários na coordenação de suas ações	Permitir aos membros se comunicarem direta ou indiretamente e oferecer informações sobre as atividades dos demais membros.
Facilitar a identificação de colaboradores e o contato entre eles	Permitir aos membros saberem quem está presente no ambiente virtual e sua disponibilidade para comunicação.

GRAU DE SEVERIDADE

Para padronizar a avaliação, a escala de severidade que deverá ser adotada será a sugerida por Nielsen (1994):



Escala de Severidade

Severidade	Descrição
1. Problema cosmético/superficial	Precisa ser corrigido somente se houver tempo disponível.
2. Problema de usabilidade pequeno	Possui uma baixa severidade para ser solucionado.
3. Problema de usabilidade grande	Alta prioridade, importante solucionar o problema.
4. Catástrofe de usabilidade	Necessário solucionar o problema antes que o produto seja liberado para comercialização.

LINKS ÚTEIS

Protótipo: Ele foi desenvolvido no Figma, e pode ser visto como um protótipo de alta fidelidade.

Formulário do avaliador: Nele você poderá inserir suas considerações sobre a interface.



**DÚVIDAS?
ENTRE EM CONTATO VIA WHATSAPP
JOÃO GABRIEL (77) 99974-1647
VINICIUS (77)98838-9236**

OBRIGADO!

