

# O uso de exemplos trabalhados para facilitar a contribuição de novatos em projetos de software livre

Andressa S. S. Medeiros

Universidade Estadual de Maringá  
Maringá, Brazil  
andressasilva.0797@gmail.com

Igor S. Wiese

Universidade Tecnológica Federal do Paraná  
Campo Mourão, Brazil  
igor@utfpr.edu.br

Thelma E. Colanzi

Universidade Estadual de Maringá  
Maringá, Brazil  
thelma@din.uem.br

Igor F. Steinmacher

Northern Arizona University  
Flagstaff, USA  
igor.steinmacher@nau.edu

## RESUMO

O mercado de desenvolvimento de software está crescendo rapidamente, e projetos de software livre (SL) oferecem oportunidades valiosas para novatos. Contudo, as barreiras técnicas e não técnicas ainda dificultam as primeiras contribuições. Embora estudos busquem mitigar esses desafios, persistem as limitações. Exemplos trabalhados ajudam aprendizes a cometer menos erros, mas é difícil encontrar exemplos estruturados. Muitos desenvolvedores recorrem ao Stack Overflow, estratégia que nem sempre é eficaz. Recentemente, modelos de linguagem como o ChatGPT transformaram a busca por soluções. Considerando a reutilização frequente de códigos do Stack Overflow e o uso crescente do ChatGPT, este estudo investigou se respostas dessas fontes poderiam funcionar como exemplos trabalhados para apoiar o aprendizado e a resolução de problemas de novatos. Para isso, usamos uma *issue* do projeto Pandas como estudo de caso e conduzimos entrevistas individuais com o protocolo Thinking Aloud, seguidas de análise qualitativa. Os resultados sugerem que exemplos trabalhados dessas fontes podem auxiliar novatos em suas primeiras contribuições a projetos de SL.

## PALAVRAS-CHAVE

Educação em engenharia de software com OSS, Integração de novatos, Exemplos trabalhados, Stack Overflow, ChatGPT

## 1 Introdução

O crescimento acelerado do mercado de trabalho em desenvolvimento de software exige que a educação superior forme profissionais capazes de atender às demandas de diversos domínios de aplicação. Esse crescimento é acompanhado por um ritmo intenso de inovação tecnológica, o que impõe desafios significativos a desenvolvedores iniciantes, que precisam rapidamente adquirir habilidades práticas para se manterem competitivos. Nesse contexto, o ecossistema de projetos de software livre (SL) é uma oportunidade valiosa: ao mesmo tempo em que oferece espaço para a entrada de novos desenvolvedores, permite a aquisição de experiência prática em ambientes reais de desenvolvimento, sem a exigência de experiência prévia formal. Essa abordagem tem sido cada vez mais explorada em contextos educacionais e de formação profissional [2, 13, 29].

Alguns estudos apontam que a entrada de novos contribuidores traz benefícios concretos para as comunidades de SL, como, por exemplo, a introdução de novas ideias e perspectivas [21–24].

Apesar disso, o processo de entrada nos projetos ainda apresenta barreiras relevantes, tanto técnicas quanto sociais, que resultam na desistência de muitos dos potenciais novos contribuidores [3, 21–24]. Estudos na literatura têm explorado soluções para diminuir essas barreiras, como a recomendação de tarefas apropriadas para novatos [29], que além de sugerir uma lista de *issues* apropriadas, ainda buscou reduzir a carga dos mantenedores na avaliação dessas *issues*, uma vez que a seleção prévia permitia que eles apenas confirmassem sua adequação para iniciantes. No entanto, Tan et al. [25], mostraram que a simples recomendação de tarefas pode não ser suficiente para atrair e manter novos contribuidores. Mentoria é outra forma de apoiar novatos, como mostra Balali et al. [2]. Eles mostraram que tal abordagem pode suprir a falta de conhecimento dos novatos, entretanto a mentoria sofre limitações de escalabilidade, adesão cultural e falta de treinamento apropriado. Assim, apesar dos esforços já realizados, ainda permanece o desafio de reduzir as barreiras enfrentadas pelos novatos em suas primeiras contribuições para projetos de SL. Nesse contexto, identificar estratégias pedagógicas eficazes para apoiar esses iniciantes torna-se uma prioridade de pesquisa.

Uma estratégia pedagógica destacada na literatura para apoiar iniciantes é o uso de exemplos trabalhados — composto pela formulação de um problema, a descrição detalhada das etapas de sua solução e a apresentação da resposta final, tornando explícitas as decisões e o raciocínio por trás da solução [1]. Estudos mostram que esses recursos melhoram o desempenho e reduzem o tempo de aprendizagem [19], embora ainda haja escassez de exemplos estruturados e contextualizados [26]. Diante dessa limitação, desenvolvedores novatos recorrem a exemplos de código como estratégia complementar de aprendizado [4, 15]. Plataformas como o Stack Overflow são amplamente utilizadas, oferecendo trechos reutilizáveis que aparecem com frequência em projetos reais [5, 6]. Embora úteis, enfrentam problemas como demora nas respostas e qualidade variável [30], reforçando a necessidade de meios mais acessíveis e sistemáticos de oferecer exemplos trabalhados, especialmente em contextos como o SL, onde o suporte pedagógico é limitado e a entrada pode ser desafiadora.

Nesse cenário, modelos de linguagem (LLMs), como o ChatGPT, emergem como ferramentas disruptivas que oferecem interação em linguagem natural, respostas imediatas e contextualizadas [9, 12, 17, 20]. Tais características tornam esses modelos particularmente promissores para apoiar novatos, ampliando o acesso a exemplos e

explicações com menor barreira de entrada e possibilidade de explorar mais detalhes. Embora estudos recentes explorem a comparação entre LLMs e Stack Overflow [8, 10, 11, 14, 30], há uma lacuna quanto à compreensão de seu papel como ferramenta pedagógica para a entrada de novatos em projetos de SL. Este é o ponto que busca-se investigar neste trabalho.

Considerando a ampla reutilização de trechos de código do Stack Overflow e o uso crescente de respostas geradas por LLMs, levantou-se a hipótese de que as respostas de ambas as fontes, podem funcionar como exemplos trabalhados com potencial para apoiar o aprendizado de desenvolvedores novatos, além de contribuir para mitigar a escassez de exemplos estruturados voltados a esse público. Nesse contexto, o objetivo deste trabalho foi investigar como exemplos trabalhados, gerados por um assistente de inteligência artificial (ChatGPT) ou extraídos de uma plataforma de pergunta e resposta (Stack Overflow), podem apoiar e orientar desenvolvedores novatos na resolução de problemas. Para orientar essa investigação, foram definidas duas questões de pesquisa que buscaram compreender a percepção de novatos e o impacto prático dos exemplos trabalhados no processo de contribuição:

- **QP1:** Qual a percepção de desenvolvedores novatos com relação a exemplos trabalhados do Stack Overflow e gerados pelo ChatGPT como apoio à análise de *issues* em projetos de SL?
- **QP2:** Como exemplos trabalhados contribuem para orientar e apoiar desenvolvedores novatos na compreensão e resolução de tarefas em projetos de SL?

Para responder às questões de pesquisa, foi conduzido um estudo observacional com 21 desenvolvedores, utilizando o protocolo *Thinking Aloud* (Seção 3). Cada participante analisou uma *issue* real do projeto Pandas<sup>1</sup>, acompanhada de um exemplo trabalhado do Stack Overflow ou gerado pelo ChatGPT, enquanto verbalizava seus pensamentos, buscava compreender a *issue* e relacioná-la ao exemplo e após a tarefa, os participantes responderam a dois questionários de percepção. Os dados foram analisados qualitativamente, com foco na identificação de evidências e padrões que respondessem às questões de pesquisa.

Os resultados evidenciam que exemplos trabalhados, do Stack Overflow ou ChatGPT, ajudam desenvolvedores novatos a compreender e resolver *issues* ao fornecerem suporte direto à interpretação do problema e à formulação de soluções. Os exemplos responderam dúvidas e incentivaram propostas de solução, com os do ChatGPT se destacando por maior clareza e impacto na confiança dos participantes. Ainda, o estudo demonstra que exemplos trabalhados complementam rótulos como “*good first issue*”, oferecendo orientação adicional que permite aos novatos avaliar se possuem o conhecimento necessário para contribuir. Essa abordagem reduz incertezas e facilita o processo de entrada em projetos de SL.

Com base nessas evidências, este estudo contribui com uma proposta concreta: uma estratégia pedagógica que integra o uso de exemplos trabalhados como ferramenta sistemática de apoio à entrada de novos colaboradores em projetos de SL, oferecendo apoio prático durante a resolução de *issues*. Tal estratégia contribui diretamente para tornar o processo de entrada mais acessível, estruturado e sustentável, com potencial impacto prático na sustentabilidade dos projetos de SL.

<sup>1</sup><https://github.com/pandas-dev/pandas>

## 2 Trabalhos Relacionados

Esta seção aborda trabalhos que discutem temas associados ao uso de exemplos trabalhados e à comparação entre as ferramentas Stack Overflow e ChatGPT, tópicos diretamente relacionados aos objetivos desta pesquisa.

Este artigo utiliza exemplos trabalhados como estratégia de apoio. Neste contexto, os estudos apresentados a seguir se aproximam dessa proposta por também explorarem os exemplos trabalhados. Um estudo conduzido por Skudder and Luxton-Reilly [19] forneceu evidências de que os exemplos trabalhados representam uma abordagem superior em comparação às estratégias tradicionais de solução de problemas, melhorando tanto o tempo necessário para o aprendizado quanto o desempenho em questões similares. Segundo Tonhão et al. [7] e Wang et al. [27], exemplos trabalhados têm características que visam diminuir o excesso de carga cognitiva do estudante, favorecendo um aprendizado mais rápido, visto que os exemplos de código são uma das principais fontes de informações. No entanto, esses trabalhos não abordam a utilidade desses exemplos especificamente para apoiar a contribuição de novatos em projetos de software livre (SL), como é proposto neste estudo.

Já no estudo de Pham et al. [18] foi implementada uma ferramenta para recomendar exemplos de códigos de teste de software para desenvolvedores iniciantes, que apontou que os novos contribuidores têm mais confiança para escrever seus próprios casos de teste quando tem um código existente como referência. O trabalho de Tonhão et al. [26] tratou de observar o uso de exemplos trabalhados reais por docentes no ensino de engenharia de software. Os resultados revelaram que, embora docentes já utilizem exemplos em sala, na maioria das vezes não são exemplos reais e poucos conhecem formalmente o conceito. Tonhão et al. [7] implementaram um portal para catalogar exemplos trabalhados extraídos de projetos de SL. Em sua pesquisa, foi observado que a utilização de exemplos trabalhados ajudam os estudantes a praticar habilidades de *design*. Apesar desses estudos compartilharem o interesse em apoiar iniciantes por meio de exemplos, essa pesquisa se difere ao analisar comparativamente fontes específicas de exemplos (Stack Overflow e ChatGPT) e investigar sua utilidade prática para novatos contribuírem em projetos reais de SL.

Após revisar os estudos sobre exemplos trabalhados, destacam-se também pesquisas que comparam o Stack Overflow e o ChatGPT como ferramentas de apoio a desenvolvedores, tema que se alinha diretamente aos objetivos deste trabalho. Kabir et al. [11], conduziram uma análise da qualidade das respostas do ChatGPT para perguntas do Stack Overflow e identificaram que, embora sejam abrangentes, mais de 50% são incorretas devido à dificuldade do modelo em captar o contexto implícito. Já Delile et al. [8], compararam as duas plataformas em questões relacionadas à privacidade, apontando que o ChatGPT gera respostas mais longas e detalhadas, enquanto o Stack Overflow mantém maior precisão. Xu et al. [30] compararam respostas do ChatGPT e humanos no Stack Overflow sobre desenvolvimento de software, concluindo que as respostas humanas são melhores em seis aspectos (correção, utilidade, diversidade, legibilidade, clareza e concisão), embora o ChatGPT ofereça vantagem por trazer respostas imediatas.

Outros estudos, como o de Liu et al. [14] avaliaram o desempenho dessas ferramentas no aumento da produtividade de programadores, enquanto o de Garcia et al. [10], explorou as preferências dos programadores novatos entre o ChatGPT e o Stack Overflow como ferramenta de apoio em perguntas de programação, apontando para diferentes vantagens e limitações. Embora todos esses trabalhos apresentados nesta seção tenham investigado o Stack Overflow e o ChatGPT sob diferentes perspectivas como precisão, detalhamento e/ou corretude das respostas, impacto na produtividade ou uso como ferramenta de apoio em perguntas de programação ou outros contextos específicos, nenhum deles investigou a utilidade das respostas fornecidas por essas plataformas como exemplos trabalhados para facilitar a contribuição de novatos em projetos de SL. Essa lacuna justifica a proposta deste trabalho, que busca explorar justamente esse potencial ainda pouco investigado.

### 3 Método de Pesquisa

Esta seção apresenta o método de pesquisa seguido neste artigo, cujo objetivo principal é investigar como exemplos trabalhados—gerados pelo ChatGPT ou extraídos do Stack Overflow—podem apoiar e orientar desenvolvedores novatos em suas primeiras contribuições para projetos de software livre (SL). O método é apresentado na Figura 1 e detalhado nas subseções a seguir. Os instrumentos utilizados neste estudo estão disponíveis em [16] para possibilitar a replicação.

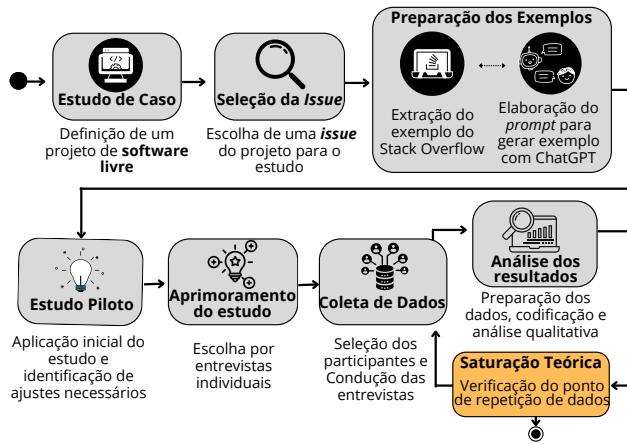


Figura 1: Visão geral do método de pesquisa

#### 3.1 Estudo de Caso: Projeto Pandas

Para esta pesquisa, foi adotado como estudo de caso o projeto Pandas, uma biblioteca de software livre amplamente utilizada para análise e manipulação de dados em Python. De acordo com o repositório oficial, Pandas oferece estruturas de dados flexíveis, rápidas e de alto desempenho, facilitam operações complexas, como agrupamento e manipulação de séries. Por meio de uma interface intuitiva, o Pandas permite o manuseio eficiente de grandes volumes de dados, oferecendo diversas funcionalidades para carregar, organizar, filtrar e transformar dados.

O projeto apresenta características alinhadas com os critérios recomendados para o uso em contextos educacionais [13], incluindo

um histórico consolidado de mais de dois anos de desenvolvimento ativo, é desenvolvido em Python (uma das linguagens mais utilizadas na atualidade), possui uma comunidade engajada, e possui uma quantidade considerável de *issues* com rótulos voltados para novatos (e.g., "good first issue"). Além disso, durante a análise do projeto, foram identificadas 46 *issues* que, além de conter esses rótulos, continham *links* que relacionavam essas *issues* a discussões no Stack Overflow, justificando a adequação do projeto Pandas para os objetivos desta pesquisa.

#### 3.2 Seleção da issue de interesse

Seguiu-se um processo criterioso e exploratório para a escolha da *issue* a ser utilizada como objeto de estudo. Durante a análise dos *links* nas *issues* do projeto, foram descartadas as *issues* em que as respostas no Stack Overflow não tinham votos ou a própria *issue* era apresentada como resposta. Também foram excluídas as *issues* cujos posts relacionados no Stack Overflow não possuíam respostas. Mantive-se apenas as *issues* que continham um *link* para o Stack Overflow com pelo menos uma resposta (não fosse a própria *issue*), e pelo menos um voto. Sete *issues* foram selecionadas para a análise juntamente com os exemplos extraídos do Stack Overflow por meio dos *links* indicados nas *issues*: 15058, 10586, 10336, 14135, 21651, 24306 e 51269. Os autores analisaram essas sete *issues* para escolher a mais apropriada para o estudo. Decidiu-se pela *issue* 10336 dada a facilidade de entendimento e pouca necessidade de conhecimento profundo no projeto.

A *issue* escolhida (10336) relatava o problema de que ao tentar filtrar no DataFrame apenas os valores falsos, um comportamento inesperado ocorria: a função retornava valores float ao invés de booleanos. Segundo relatado na *issue*, ao aplicar uma condição para exibir os valores False com `df_mask [df_mask == False]` ou `df_mask.where(df_mask == False)`, o usuário esperou que os valores False permanecessem booleanos. No entanto, observou-se que os valores exibidos eram 0 em vez de False na coluna CCC. A questão a ser analisada na *issue* era então: Por que os valores na coluna CCC apareciam como 0 em vez de False para os índices 1 e 3 resultando em um DataFrame com floats onde era esperado um DataFrame puramente booleano?

#### 3.3 Elaboração do Prompt para Geração do Exemplo

A elaboração do *prompt* para o ChatGPT gerar exemplos baseou-se no catálogo de padrões de *prompt* proposto por White et al. [28]. Com o objetivo de garantir que os exemplos gerados fossem adequados ao contexto da *issue* analisada, adotou-se um processo iterativo para o refinamento do *prompt*. Esse processo consistiu em três iterações com o ChatGPT, com ajustes baseados em três critérios avaliados pelos autores: (i) a resposta deve trazer um exemplo de código relevante; (ii) o exemplo deve incluir casos de teste quando necessário; e (iii) a resposta deve apresentar exclusivamente informações pertinentes ao contexto da *issue*.

Os *prompts* utilizados em cada uma das interações são fornecidos no pacote de replicação [16]. Na primeira iteração, verificou-se que as respostas não incluíam casos de teste, um elemento importante em resoluções de *issues*. Na segunda iteração, embora o ChatGPT tenha incluído os casos de teste, a resposta continha informações

## Exemplo Trabalhado - Stack Overflow

Python Pandas Boolean Dataframe Where Dataframe Equals False - Returns 0 instead of False?

Not entirely sure why, but if you're looking for a quick fix to convert it back to bools you can do the following:

```
>>> df_bool = df_mask.where(df_mask == False).astype(bool)
>>> df_bool
AAA BBB CCC
0 True False True
1 True False False
2 True False True
3 True False False
```

This is because the returned dataframe has a different dtype: it's not longer a dataframe of bools.

```
>>> df2 = df_mask.where(df_mask == False)
>>> df2.dtypes
AAA float64
BBB bool
CCC float64
dtype: object
```

This even occurs if you force it to a bool dtype from the getgo:

```
>>> df_mask = pd.DataFrame([{'AAA': [True] * 4, 'BBB': [False] * 4,
   'CCC': [True, False, True, False]}, dtype=bool)
print(df_mask)
AAA BBB CCC
0 True False True
1 True False False
2 True False True
3 True False False
```

>>> df2 = df\_mask.where(df\_mask == False)
>>> df2

```
AAA BBB CCC
0 NaN False NaN
1 NaN False 0
2 NaN False NaN
3 NaN False 0
```

If you're explicitly worried about memory, you can also just return a reference, but unless you're explicitly ignoring the old reference (in which case it shouldn't matter), be careful:

<http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.astype.html>

<http://docs.scipy.org/doc/numpy/reference/generated/numpy.dtype.html>

Figura 2: Exemplo Trabalhado extraído do Stack Overflow

excessivas e fora do escopo da solução, o que prejudicava a objetividade. Por fim, na terceira iteração, a resposta atendeu aos critérios estabelecidos apresentando um exemplo de código relevante, incluindo casos de teste e foi concisa. Diante disso, o *prompt* final utilizado foi: (*From now on, I would like you to serve as an expert on the pandas project. When I ask questions, you must provide examples in Python code and the steps to solve. If the question doesn't require code, simply provide the solution steps. Include a test case.*).

### 3.4 Exemplos Trabalhados analisados no Estudo

Nesta subseção são apresentados os exemplos trabalhados utilizados no estudo – tanto extraído do Stack Overflow, quanto gerado pelo ChatGPT – disponíveis com fonte ampliada em [16].

**3.4.1 Exemplo 1 - Stack Overflow.** O Exemplo 1, apresentado na Figura 2, foi retirado diretamente do *link* pertencente a *issue* e apresenta alguns testes realizados na tentativa de resolver a *issue*. Logo no início, o exemplo sugere converter o DataFrame para o tipo booleano utilizando o `astype(bool)`. Em seguida, ele apresenta os tipos de cada coluna após a aplicação de `df_mask.where`, evidenciando que a coluna CCC está tipada como `float64`. Essa informação ajuda a entender por que a coluna imprime valores `float64` em vez de booleanos, como esperado, visto que a tipagem `float64` leva a uma conversão de valores `False` para `0`. Para finalizar, o exemplo demonstra que a conversão direta na declaração do DataFrame não resolve o problema, o que leva o colaborador a descartar essa abordagem. Como complemento, são fornecidos dois *links* relacionados à documentação oficial sobre `astype` e `dtype`, possibilitando ao desenvolvedor explorar mais sobre essas funcionalidades, caso tenha interesse.

**3.4.2 Exemplo 2 - ChatGPT.** O Exemplo 2, apresentado na Figura 3, explica detalhadamente por que o problema está acontecendo, mostrando que o projeto Pandas exibe `0` para `False` e `1` para `True` ao comparar DataFrames booleanos, porque utiliza uma representação inteira (`0` e `1`) para facilitar operações. Essa explicação ajuda o colaborador a entender por que está aparecendo `0` em vez de `False`. Para resolver isso e exibir `False` em vez de `0` no DataFrame resultante, o exemplo sugere substituir explicitamente os valores

## Exemplo Trabalhado - ChatGPT

Dataframe Where Dataframe == False Returns Dataframe with Floats

When you use `df_mask == False`, pandas returns a DataFrame where each element is compared against `False`. For columns where the values are boolean (`True` or `False`), pandas coerces these to integers (`1` for `True` and `0` for `False`) during comparison. This is a design choice in pandas to facilitate element-wise comparison and operations across DataFrames.

```
df_mask = pd.DataFrame([{'AAA': [True] * 4, 'BBB': [False] * 4,
   'CCC': [True, False, True, False]}, dtype=bool)
print(df_mask)
# Create the DataFrame with boolean values
```

```
AAA BBB CCC
0 True False True
1 True False False
2 True False True
3 True False False
```

```
# Replace the result of the comparison with booleans
```

```
result = df_mask.where(df_mask == False, False)
```

```
# Replace non-matching values with False
```

```
print(result)
```

```
AAA BBB CCC
```

```
0 NaN False NaN
```

```
1 NaN False 0
```

```
2 NaN False NaN
```

```
3 NaN False 0
```

```
# Expected output
```

```
expected_output = pd.DataFrame([{'AAA': [None] * 4, 'BBB': [None] * 4,
```

```
   'CCC': [None, False, None, False]}])
```

```
# Replace non-matching values with None
```

```
print(expected_output)
```

```
AAA BBB CCC
```

```
0 None False None
```

```
1 None False None
```

```
2 None False None
```

```
3 None False None
```

```
# Assert that the result matches the expected output
```

```
assert result.equals(expected_output)
```

```
print("Test case passed successfully.")
```

### Test Case

```
# Create the DataFrame with boolean values
df_mask = pd.DataFrame([{'AAA': [True] * 4, 'BBB': [False] * 4,
   'CCC': [True, False, True, False]}, dtype=bool])

# Replace the result of the comparison with booleans
result = df_mask.where(df_mask == False, False)
# Replace non-matching values with False

# Expected output
expected_output = pd.DataFrame([{'AAA': [None] * 4, 'BBB': [None] * 4,
   'CCC': [None, False, None, False]}])

# Assert that the result matches the expected output
assert result.equals(expected_output)
print("Test case passed successfully.")
```

Figura 3: Trecho do Exemplo Trabalhado gerado pelo ChatGPT

após aplicar a condição, utilizando `df_mask.where(df_mask == False, False)`. Desta forma, o DataFrame exibirá `False` nos locais esperados. O exemplo é encerrado com um caso de teste.

### 3.5 Condução do estudo

Este trabalho foi conduzido como um estudo observacional, com o objetivo de analisar como exemplos trabalhados podem apoiar desenvolvedores novatos na compreensão e resolução de tarefas reais em projetos de software livre. A coleta de dados foi baseada em sessões individuais com protocolo *Thinking Aloud* utilizando entrevista contextual. Isso possibilitou observar em tempo real o raciocínio dos participantes ao interagirem com uma *issue* e um exemplo de código associado. A seguir, descrevemos o processo adotado, desde as tentativas iniciais com um estudo piloto, até os procedimentos de recrutamento, coleta de dados e identificação da saturação teórica.

**3.5.1 Estudo Piloto.** Inicialmente, foi conduzido um estudo piloto com quatro participantes para testar a abordagem e os instrumentos de coleta. Os participantes analisaram uma *issue* real, seguida da avaliação de um exemplo extraído do Stack Overflow e outro gerado pelo ChatGPT. Cada participante respondeu a um questionário após a leitura da *issue* e, posteriormente, outro questionário para cada exemplo analisado. O estudo seguiu um protocolo do tipo *within-subject*, no qual todos os participantes analisaram ambos os exemplos. A ordem de apresentação foi alternada entre os grupos para controle de viés.

Os resultados do estudo piloto revelaram viés de aprendizagem: os participantes tendiam a avaliar o segundo exemplo como mais útil, independentemente da fonte. Além disso, o uso exclusivo de guias e questionários mostrou-se insuficiente para capturar adequadamente a percepção dos participantes. Como resultado, a abordagem foi reformulada para incluir observação com entrevistas semi-estruturadas individuais com apoio do protocolo *Thinking Aloud*, oferecendo maior profundidade nas análises. Os dados do estudo piloto não foram incluídos na análise final.

**3.5.2 Participantes.** O estudo contou com 21 participantes, incluindo estudantes de graduação, mestrado e doutorado, bem como profissionais atuando como professores e desenvolvedores de software. A seleção combinou conveniência com a técnica de *snowballing*. Inicialmente, 24 estudantes de pós-graduação em Ciência da Computação da instituição foram convidados por e-mail, resultando em dez confirmações. Esses participantes indicaram outras pessoas elegíveis, ampliando a base do estudo. Como critério, os participantes deveriam ter conhecimento prévio em programação.

Os participantes foram divididos em dois grupos: 11 (52,4%) avaliaram um exemplo retirado do Stack Overflow (Grupo 1) e 10 (47,6%) analisaram um exemplo gerado pelo ChatGPT (Grupo 2). Ambos os grupos trabalharam com a mesma *issue*; o que os diferenciava era apenas a fonte do exemplo apresentado como apoio. Buscou-se também um equilíbrio de gênero entre os grupos. Informações detalhadas sobre o perfil dos participantes estão na Tabela 1.

**Tabela 1: Perfil dos Participantes**

Participante	Gênero	Escolaridade	Atuação	Exemplo avaliado
P1	Mulher	Mestre	Estudante	Stack Overflow
P2	Homem	Mestre	Desenvolvedor de software	ChatGPT
P3	Homem	Mestre	Estudante	Stack Overflow
P4	Mulher	Mestre	Desenvolvedor de software	ChatGPT
P5	Mulher	Mestre	Professor	Stack Overflow
P6	Homem	Mestre	Professor	ChatGPT
P7	Homem	Graduado	Desenvolvedor de software	Stack Overflow
P8	Homem	Mestre	Professor	ChatGPT
P9	Homem	Mestre	Professor	Stack Overflow
P10	Homem	Mestre	Desenvolvedor de software/ Professor	ChatGPT
P11	Homem	Graduado	Desenvolvedor de software	Stack Overflow
P12	Mulher	Graduado	Desenvolvedor de software	ChatGPT
P13	Homem	Mestre	Desenvolvedor de software/ Professor	Stack Overflow
P14	Homem	Graduado	Professor	ChatGPT
P15	Homem	Graduado	Desenvolvedor de software	Stack Overflow
P16	Homem	Graduado	Outro	ChatGPT
P17	Homem	Mestre	Estudante	Stack Overflow
P18	Mulher	Mestre	Professor	ChatGPT
P19	Mulher	Mestre	Professor	Stack Overflow
P20	Mulher	Ensino superior em andamento	Estudante	ChatGPT
P21	Homem	Graduado	Desenvolvedor de software	Stack Overflow

Quanto à experiência com software livre, 19 participantes (90,5%) relataram nunca ter contribuído com esse tipo de projeto. A maioria (76,2%) se classificou como novato inexperiente. Dois (9,5%) se identificaram como novatos com contribuições em progresso (*pull requests* enviados, mas ainda não aceitos), enquanto outros dois (9,5%) preferiram não responder diretamente à pergunta, embora tivessem indicado, durante a entrevista, que nunca haviam considerado contribuir. Apenas um participante (4,8%) relatou ter uma contribuição aceita, mas ainda se considerava inseguro quanto ao processo. Esses dados confirmam que o estudo envolveu majoritariamente novatos, alinhados ao público-alvo da investigação.

**3.5.3 Coleta de Dados.** As sessões com os participantes foram realizadas remotamente via *Google Meet*. Antes da sessão, os participantes deveriam responder a um questionário de caracterização enviado via e-mail, com o objetivo de coletar informações sobre

experiência e perfil. Durante a sessão, foi aplicado o protocolo *Thinking Aloud*, e foi utilizado um protocolo com questões pré-definidas (segundo os moldes de uma entrevista semiestruturada contextual) a serem feitas durante a observação (instrumento disponível no material suplementar [16]). Cada participante analisou uma *issue* e um único exemplo, para evitar viés de comparação. Durante as sessões, foi aplicado o protocolo *thinking aloud*, no qual os entrevistados verbalizam suas reflexões e pensamentos enquanto o pesquisador fazia perguntas relacionadas à *issue* e ao processo.

No início de cada sessão, foram explicados o objetivo do estudo e as instruções iniciais. Em seguida, foi solicitado o consentimento do participante para a gravação da reunião e o compartilhamento de tela, para que fosse possível acompanhar suas percepções. A sessão era composta por duas partes: na primeira, o participante lia a *issue* e respondia a um questionário sobre sua compreensão inicial; na segunda, analisava o exemplo e respondia a um segundo questionário, relacionando o conteúdo do exemplo com a *issue*. Durante toda a atividade, os participantes verbalizavam seus pensamentos, e o compartilhamento de tela era registrado com consentimento prévio. As instruções e materiais (*issue*, exemplos e questionários) estão disponíveis em [16].

As quatro primeiras entrevistas foram avaliadas quanto à eficácia da nova abordagem. Esta abordagem se mostrou mais adequada do que a utilizada no estudo piloto, principalmente por possibilitar a identificação mais precisa das percepções dos participantes sem viés. No total, foram entrevistados 21 participantes.

### 3.6 Análise dos dados

Para a análise dos dados, foi utilizada a técnica de análise qualitativa de conteúdo. As sessões foram gravadas, transcritas integralmente e enriquecidas com as respostas aos formulários preenchidos pelos participantes. As respostas foram integradas às transcrições para contextualizar os comentários feitos durante a escrita e garantir uma interpretação mais precisa das percepções expressas.

A análise foi conduzida de forma iterativa e alternada à coleta de dados, com o objetivo de monitorar a ocorrência de saturação teórica. Um dos autores realizou a codificação inicial das transcrições com base em leitura detalhada dos dados. A codificação foi realizada com o auxílio da ferramenta *Atlas.ti*<sup>2</sup> (versão web) e buscou identificar evidências alinhadas às questões de pesquisa. Alguns temas foram definidos a priori com base nas perguntas das entrevistas (por exemplo, “compreensão da *issue*”, “utilidade do exemplo” e “nível de confiança”), enquanto a maior parte dos códigos emergiu da análise aberta do conteúdo. Os dados foram analisados manualmente com foco na relevância das respostas, preservando a perspectiva dos participantes.

Para aumentar a confiabilidade das interpretações, os demais autores revisaram os códigos e participaram de sessões colaborativas de discussão, nas quais divergências foram examinadas e as categorias ajustadas quando necessário. Esse processo contribuiu para reduzir vieses individuais e fortalecer a validade da análise.

**3.6.1 Saturação Teórica.** A saturação teórica foi monitorada continuamente ao longo do estudo. Após cada sessão, o material era

<sup>2</sup><https://atlasti.com/atlas-ti-web>

transcrito, codificado e comparado com os dados previamente analisados. A saturação foi considerada atingida na décima sétima sessão, quando nenhum novo código relevante emergiu. Ainda assim, foram realizadas quatro sessões adicionais para confirmação, totalizando 21 participantes. A visualização do ponto de saturação está disponível no material suplementar [16].

## 4 Resultados

Esta seção apresenta e discute os principais resultados deste estudo. Primeiramente, são apresentadas as percepções iniciais dos participantes sobre a *issue*, seguido da resposta às questões de pesquisa.

### 4.1 Percepções iniciais sobre a *Issue*

**4.1.1 Compreensão.** Durante a codificação e análise de compreensão da *issue*, dois tipos de percepções foram analisados: (i) a percepção dos pesquisadores sobre o entendimento demonstrado pelos participantes; e (ii) as percepções relatadas pelos participantes. Esses dois olhares contribuíram para identificar até que ponto os participantes conseguiam compreender o problema da *issue*.

A partir da codificação e análise das falas, identificou-se que, entre os 21 participantes entrevistados, 15 (71,4%) compreenderam o problema descrito na *issue* mesmo antes de receberem o exemplo trabalhado como apoio. Por outro lado, seis participantes (28,6%) apresentaram algum nível de dificuldade, seja por falta de compreensão completa, compreensão parcial ou interpretação equivocada. Ainda que parte do grupo tenha demonstrado dificuldade, essa limitação não foi considerada uma ameaça relevante ao estudo, pois a maioria demonstrou entender o conceito central do problema.

Nas falas, alguns participantes evidenciaram compreensão clara e segurança, como P17, que afirmou: *"Eu achei simples de entender o que tá acontecendo"*, e complementou *"Não achei difícil [...] eu tenho pouca experiência, e entendi tudo que tá acontecendo bem tranquilo."* Outros participantes relataram que a dificuldade estava mais relacionada à falta de familiaridade com a linguagem Python, como P11: *" [...] eu não entendi completamente porque eu não sou familiarizado com a sintaxe da linguagem do Python."* Apesar disso, foi recorrente o relato de que o contexto da *issue* poderia estar mais claro. P6 comentou: *"o problema está claro, mas o contexto onde esse problema é observado não"*. Houve também relatos sobre a necessidade de realizar testes ou consultar documentação para validar hipóteses (P12: *"Acredito que é preciso realizar testes e consultar a documentação da função para descobrir porque não é exibido o resultado esperado."*). Além das dúvidas específicas, alguns participantes expressaram insegurança quanto à própria compreensão: P3 disse *"Tendo em vista que eu se eu entendi certo"*, e P4 comentou *"Isso se eu entendi certo, né? Eu vou, considerando que eu entendi."* Houve ainda um relato de autocrítica mais negativa, como no caso de P14: *"Não sei, eu tô me sentindo burro, se tá na minha cara, não estou enxergando, eu não sei"*.

Essa análise conjunta evidencia que, embora a maioria tenha conseguido compreender o problema principal, muitos participantes sentiram necessidade de mais informações de contexto e demonstraram insegurança, especialmente devido à familiaridade limitada com a linguagem ou ao receio de não ter entendido completamente.

**4.1.2 Dúvidas.** Durante a entrevista, os participantes relataram diferentes dúvidas sobre a *issue* apresentada. Essas dúvidas não

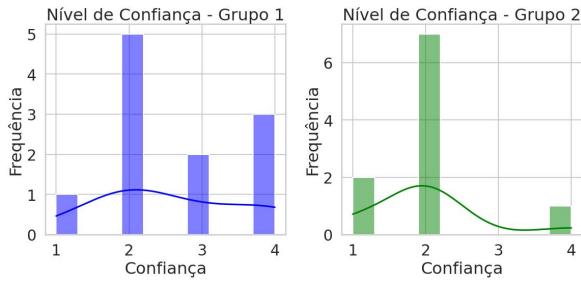
foram esclarecidas pelos pesquisadores, pois o objetivo era avaliar se os exemplos trabalhados poderiam cumprir esse papel. De modo geral, as incertezas identificadas se agruparam em três categorias principais: falta de contextualização do problema; dificuldades com a sintaxe da linguagem Python; e dúvidas pontuais ou superficiais. Entender essas dúvidas iniciais foi importante para analisar se poderiam dificultar a definição de uma estratégia de resolução da *issue* e, posteriormente, verificar se os exemplos apresentados ajudariam a superar essas barreiras.

A dúvida mais frequente esteve relacionada ao **contexto de uso** do trecho de código, como apontado por P6: *"Uma dúvida importante seria qual o contexto onde aquele dataframe pode ser considerado e aquele problema pode ser observado?"*. Essa carência de informações adicionais foi percebida por vários participantes como um obstáculo para entender completamente a *issue*. Também surgiram dúvidas relacionadas à **sintaxe da linguagem Python**, refletindo insegurança ou pouca familiaridade. P2 comentou: *"Não sei se é possível trabalhar com atribuição de valor para resolver a issue, não sei como se trabalhar com valores booleanos em python e por fim, eu não sei como funciona a chamada de print em python."* De forma semelhante, P11 destacou: *"Dúvidas sobre a sintaxe de python, não tendo total entendimento sobre as linhas de print que o usuário forneceu."* Por fim, alguns participantes levantaram **dúvidas mais pontuais**, ligadas a detalhes específicos do código. É o caso de P7, que questionou: *"Dúvida em relação ao funcionamento função where aplicada no dataframe."* Outros tiveram apenas **dúvidas superficiais**, geralmente associadas à falta de atenção ou detalhes menos centrais da *issue*.

**4.1.3 Orientação para resolução.** Esta etapa da análise buscou compreender se os participantes sabiam o que precisava ser feito para resolver a *issue* e se tinham estratégias definidas para iniciar a solução. Embora os participantes tivessem uma noção do que poderia ser feito para resolver a *issue*, muitos não possuíam uma ideia clara de como iniciar a solução.

Os dados revelaram que apenas cinco participantes (23,8%) demonstrou ter clareza quanto aos passos necessários e apresentou uma estratégia concreta desde o início. Por outro lado, oito participantes (38,1%) admitiram não saber por onde começar ou relatou não ter uma estratégia definida. Outros cinco participantes (23,8%) destacaram a importância de compreender melhor o contexto antes de propor uma solução, como exemplificado por P6: *"Eu penso o seguinte, que se eu entendo o contexto eu consigo supor uma saída correta para ele, e aí, supondo essa saída correta, eu posso comparar com esse trecho de código apresentado e pensar em uma solução."* Já P11 mencionou que recorreria a fontes externas para obter apoio, afirmando: *" [...] Buscaria em sites de fóruns. [...] seria no Stack Overflow, se alguém já passou por esse problema ou utilizaria o ChatGPT para questionar o motivo do retorno inesperado."*

Além disso, houve participantes que só se sentiram aptos a propor uma solução após a realizar testes práticos, como no caso de P17, que mencionou: *"Eu não diria que eu tenho uma ideia de como resolver, eu tenho ideia de como testar [...]"*. Alguns ainda sugeriram alternativas mais imediatistas, priorizando que a solução funcionasse, mesmo que não fosse a mais adequada, como afirmou P15: *"Meu objetivo é sempre fazer as coisas funcionar. Então se eu colocar um condicional funcione [...], não vai ser a maneira mais elegante, mas muitas vezes funciona."*



**Figura 4: Nível de confiança dos participantes para propor uma solução à issue agrupados**

De forma geral, embora a maioria dos participantes tivesse uma noção inicial sobre o que poderia ser feito, poucos apresentaram segurança para planejar os próximos passos de forma estruturada, evidenciando que ainda havia lacunas na definição de uma estratégia clara de resolução.

**4.1.4 Confiança para propor uma solução.** Os resultados também revelaram um nível geral de confiança baixo entre os participantes para propor uma solução à *issue*. Conforme mostrado na Figura 4, a maioria das respostas ficou concentrada na pontuação dois, em uma escala de um a cinco. Essa baixa confiança pode estar relacionada a diversos fatores já identificados nas análises anteriores, como a falta de familiaridade com a linguagem Python, a necessidade de compreender melhor a lógica envolvida e a ausência de um contexto mais detalhado para orientar a resolução. Observando os grupos separadamente, percebeu-se que no Grupo 1, as respostas mostraram uma distribuição um pouco mais equilibrada, com maior concentração entre dois e quatro, sugerindo uma confiança maior nesse grupo. Já no Grupo 2, as respostas ficaram mais concentradas entre um e dois, evidenciando maior insegurança para propor uma solução. De forma geral, esses dados indicam que, antes de receberem os exemplos trabalhados, os participantes apresentavam baixa confiança podendo comprometer sua capacidade de avançar na resolução da *issue*.

## 4.2 Percepções dos novatos sobre os exemplos como apoio à análise de issues

Nesta subseção, comparamos as percepções dos participantes em relação aos exemplos apresentados. A análise abrangeu a clareza dos exemplos, mudanças na compreensão da *issue* após sua apresentação e sua eficácia em esclarecer dúvidas. Também foram avaliados os aspectos considerados mais úteis e inspiradores, além da capacidade dos exemplos de orientar a formulação de soluções e de influenciar o nível de confiança dos participantes.

**4.2.1 Clareza dos Exemplos.** A análise da clareza dos exemplos trabalhados indicou avaliações positivas em ambos os casos, com algumas diferenças em relação à facilidade de compreensão. O **exemplo do Stack Overflow** foi considerado claro pela maioria dos participantes. Apesar disso, alguns relataram a necessidade de esforço adicional para compreender integralmente o motivo do

problema descrito na *issue*. Ainda assim, destacaram como aspectos positivos a capacidade do exemplo de corrigir interpretações equivocadas, confirmar hipóteses previas e, em alguns casos, trazer informações novas que ampliaram o entendimento sobre o tema. O **exemplo gerado pelo ChatGPT**, por sua vez, foi classificado como claro e, na maioria das vezes (60% dos participantes), fácil de compreender. Os participantes destacaram sua explicação detalhada, que contribuiu para confirmar pensamentos e a utilidade didática, além de gerar *insights* sobre como resolver o problema. Outro ponto positivo foi a capacidade do exemplo de suprir informações ausentes na descrição da *issue*, facilitando o entendimento do funcionamento do código e do contexto, como relatado por P4: "*O exemplo ajuda porque supre essa parte de entender como o Pandas funciona. Acho que era essa a informação que faltava para eu conseguir ajudar no problema relatado.*"

**4.2.2 Compreensão da issue após os exemplos.** A análise dos exemplos contribuiu para a compreensão da *issue* por parte dos participantes. No caso do **exemplo do Stack Overflow**, dez participantes afirmaram que ele os ajudou a entender melhor o problema, destacando que o exemplo complementava a descrição da *issue* ao apontar o erro de forma concreta (P17). P1 acrescentou que o exemplo descreveu melhor trechos do código e sugeriu possíveis abordagens. Apenas um participante (P7) relatou que ainda se sentia confuso mesmo após a leitura. Em relação ao **exemplo do ChatGPT**, os participantes destacaram que a explicação detalhada sobre a lógica do Pandas foi essencial para compreender a causa do problema. P4 afirmou: "*Consegui entender o funcionamento [...] então ajudou bastante porque estava bem explicado.*" P12 reforçou: "*O exemplo consegue detalhar bem a causa do problema da issue.*" Outros aspectos mencionados incluíram a clareza da estrutura do código (P2) e P8 destacou: "*O exemplo me deu o contexto da biblioteca e mais informações sobre o problema.*"

**4.2.3 Capacidade de responder dúvidas.** Os exemplos trabalhados foram percebidos como recursos eficazes para esclarecer dúvidas durante a análise das *issues*. O **exemplo do Stack Overflow** ajudou a resolver dúvidas em 82% dos casos, incluindo algumas não mencionadas previamente. Participantes relataram que o exemplo contribuiu para ajustar interpretações e ampliar o entendimento. P19 comentou: "*Não tinha percebido que os elementos tinham tipos diferentes. Ficou mais claro quando a explicação mostrou os tipos [...]*" Apenas um participante apontou dificuldade em entender o contexto, mesmo após analisar o exemplo. O **exemplo do ChatGPT** também foi teve bom desempenho, esclarecendo 80% das dúvidas mencionadas e abordando outras não explicitadas, especialmente sobre a lógica do código e o motivo do problema. Como disse P14: "*O exemplo me ajudou a compreender como funcionam as comparações de tipos booleanos no Pandas.*" Esses resultados indicam que ambos os exemplos contribuíram positivamente para tornar o raciocínio por trás do problema mais acessível.

**4.2.4 Utilidade dos Exemplos.** A utilidade dos exemplos foi analisada sob duas perspectivas: apoio para pensar em uma solução para a *issue* (Figura 5) e como fonte de inspiração para propor soluções (Figura 6).

No caso do **exemplo do Stack Overflow**, sete participantes (63,6%) relataram que o exemplo os ajudou a pensar em uma solução,

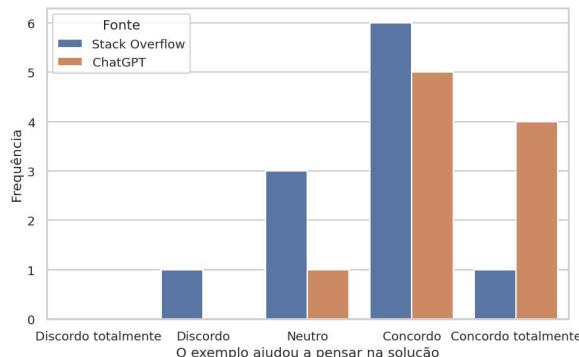


Figura 5: Impacto dos exemplos no pensamento da solução

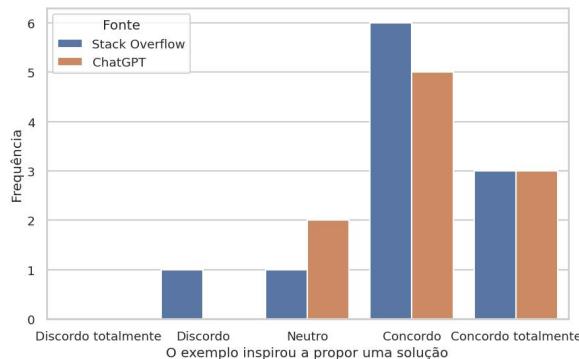


Figura 6: Impacto dos exemplos na inspiração

e nove (81,8%) afirmaram ter se sentido inspirados. As entrevistas indicaram que diferentes elementos contribuíram para isso, sendo a parte relacionada à conversão e tipagem dos dados a mais citada. P11 afirmou: "Estou repetindo o astype(bool) aqui, mas é isso que estava na minha cabeça", e P3 destacou: "Como funciona o processo de conversão". Mesmo P21, que afirmou não ter se sentido inspirado de forma geral, reconheceu: "Saber da informação da tipagem ajudou bastante". Outros participantes mencionaram aspectos como a simplicidade dos termos (P17), o uso da função dtype para entender os tipos resultantes (P21), e até mesmo trechos que indicavam caminhos a evitar como úteis e inspiradores. P3 valorizou os *links* para materiais complementares. Esses relatos demonstram que os participantes utilizaram diferentes partes do exemplo para orientar o raciocínio.

No caso do **exemplo do ChatGPT**, nove participantes (90%) afirmaram que ele os ajudou a pensar em uma solução, e oito (80%) se sentiu inspirados. Os aspectos mais mencionados foram a estrutura clara, o detalhamento, explicações sobre o funcionamento do código, os exemplos de uso, casos de teste e comentários no código. Esses elementos contribuíram tanto para a compreensão da issue quanto para a motivação em propor soluções. P4 destacou que o detalhamento o ajudou a entender a lógica por trás do problema, "dando mais confiança para trabalhar com situações semelhantes." P12 reforçou a importância da organização e clareza: "Foi

*a explicação de como funciona o código e como o documento foi estruturado, deixando as informações bem organizadas.*" Já P8 mencionou o contexto, os exemplos de código e o teste como pontos-chave para sua compreensão. Mesmo não citando partes específicas, o exemplo como um todo foi reconhecido como útil e inspirador para a formulação de soluções.

**4.2.5 Capacidade dos exemplos orientarem a propor uma solução.** Os participantes relataram que, apesar de enfrentarem certa dificuldade para implementar uma solução completa com base no **exemplo do Stack Overflow**, o exemplo foi mais útil do que a *issue* isolada por trazer explicações mais detalhadas, reduzir o tempo de tentativa e erro e confirmar raciocínios. P13 comentou: "O exemplo ajudou a pensar em soluções em vez do processo de tentativa e erro ou passar horas procurando uma solução parecida." P9 destacou que o exemplo permitiu observar o comportamento da função em diferentes situações, enquanto outro participante afirmou que ele "ajuda a ter mais certeza sobre o que realmente pode ser feito no problema." Mesmo entre os participantes que não consideraram o exemplo mais útil que a *issue* sozinha, o exemplo ainda foi visto como um recurso válido para validar ideias ou acelerar o processo. P21, por exemplo, afirmou que, embora acreditasse que chegaria à mesma solução sem o exemplo, reconheceu que ele mostrou "qual caminho seguir", facilitando a tentativa de implementação de uma solução própria.

Em relação ao **exemplo do ChatGPT**, as respostas foram majoritariamente positivas. Os participantes afirmaram que o exemplo facilitou a compreensão da lógica do problema e apontou caminhos para a solução. P4 mencionou: "Com o exemplo, ficou mais fácil entender e solucionar a issue." P6 ressaltou que a apresentação a nível de código facilitou o entendimento, enquanto P14 destacou que o exemplo "direciona ao caminho da solução e apresenta os conceitos necessários". P12 complementou afirmando que o exemplo foi mais eficaz do que apenas o código da issue, por explicar seu funcionamento de forma clara e acessível.

Esses relatos indicam que ambos os exemplos ajudaram a estruturar o raciocínio para propor soluções, com destaque para o exemplo do ChatGPT, percebido como mais claro e estruturado para guiar o raciocínio até a solução.

**4.2.6 Nível de confiança dos participantes após a análise dos exemplos.** Após analisar os exemplos, observou-se um aumento significativo na confiança dos participantes para propor soluções (Figura 7). No Grupo 1 (exemplo do Stack Overflow), a média de confiança cresceu 30%. P17 reforçou essa percepção ao afirmar: "Ah, com certeza isso aqui [o exemplo] aumentou minha confiança." No Grupo 2 (exemplo do ChatGPT), o impacto foi ainda maior: nove participantes (90%) relataram maior confiança, elevando a média em 87%. P4 comentou: "Com o exemplo, eu me sinto mais confiante [...] aumentou bastante a confiança, porque antes eu não tinha muita certeza." P8 também destacou essa evolução: "Fiquei bastante confiante em comparação principalmente a como eu estava no início." Já P6 trouxe uma perspectiva interessante ao afirmar que o exemplo ajuda o colaborador a perceber se possui conhecimento suficiente para resolver a issue: "Fica mais fácil para o desenvolvedor saber se ele possui conhecimento suficiente, não só pela escrita do código, mas pelo entendimento do problema."

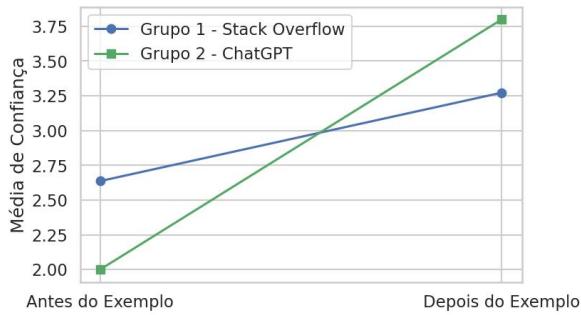


Figura 7: Comparativo da Média de Confiança Antes e Depois do Exemplo do Stack Overflow e ChatGPT

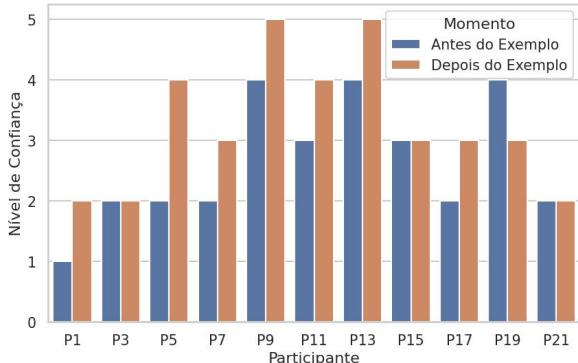


Figura 8: Nível de Confiança dos Participantes Antes e Depois do Exemplo (Grupo 1 - Stack Overflow)

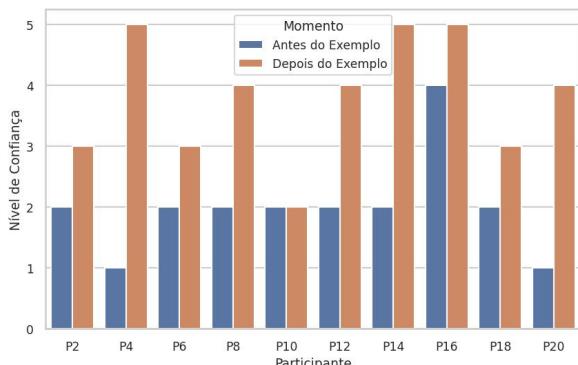


Figura 9: Nível de Confiança dos Participantes Antes e Depois do Exemplo (Grupo 2 - ChatGPT)

A partir desses dados, buscou-se compreender como essa informação foi distribuída entre os grupos. A análise individual mostrou que, no Grupo 1, sete participantes (63,6%) aumentaram sua confiança, três (27,3%) permaneceram iguais, e um (9,1%) teve uma leve

redução (Figura 8). Já no Grupo 2, nove participantes (90%) relataram aumento de confiança e apenas um (10%) manteve o mesmo nível inicial (Figura 9). Esses dados sugerem que o exemplo do ChatGPT teve um impacto mais positivo na confiança dos participantes, possivelmente por ser mais claro e detalhado, enquanto o exemplo do Stack Overflow exigiu maior esforço de interpretação, reduzindo parcialmente esse efeito.

**Resposta à QP1:** Os resultados apontam que os exemplos trabalhados foram percebidos como elementos importantes no apoio à análise de issues por desenvolvedores novatos. Foi relatado **melhorias na clareza das informações, na compreensão do problema, na capacidade de propor soluções e na confiança para contribuir**. O exemplo do ChatGPT destacou-se pelos efeitos positivos sobre a confiança, clareza e estrutura explicativa. Já o exemplo do Stack Overflow, embora exigisse maior esforço de interpretação, também foi valorizado por validar raciocínios. Ambos demonstraram potencial para orientar e apoiar desenvolvedores novatos na resolução de tarefas em projetos de SL.

### 4.3 Como Exemplos Trabalhados Apoiam Desenvolvedores Novatos na Compreensão e Resolução de Tarefas

Os exemplos atuaram como facilitadores. Embora a *issue* apresentasse uma descrição clara do problema, ela por si só, não foi capaz de esclarecer a lógica ou contexto, gerando dúvidas e dificultando que os participantes pensassem em soluções, o que resultou em baixa confiança no início do estudo. Segundo os participantes, os exemplos trouxeram sentido para a *issue*, esclarecendo o contexto e a lógica, apresentou informações que ampliaram o conhecimento, clareou o problema e uma possível solução, além de ajudar os participantes a pensarem em uma solução, inspirá-los e orientá-los a propor uma solução. Após analisar os exemplos, houve aumento na confiança, como apresentado na Figura 7. Além disso, foi mencionado que os exemplos proporcionam ao colaborador novato saber se ele possui conhecimento suficiente para tentar resolver a *issue*.

Analisa-se durante as entrevistas a utilidade do exemplo incentivar os participantes a contribuir e, dos 21 (100%) participantes 20 (95,2%) consideraram útil ou extremamente útil o uso de exemplos, e que tê-los os incentivariam. P12 comentou: "Acho que os problemas serão resolvidos bem mais rápido com uma explicação dessas[...]" . Já P14 afirmou: "É extremamente útil justamente porque ele explica o que você não está entendendo. Então ele consegue exemplificar o que tá ali que você não conseguiu pegar, e se você não tem um conhecimento sobre a lógica do Pandas, com um exemplo você consegue entender porquê que aquilo aconteceu. E se você bater o olho novamente na issue, vai fazer sentido porquê que aquilo foi um resultado." P4 também afirmou que os exemplos deixariam as contribuições mais acessíveis: "[...] deixa mais democrático contribuir, porque às vezes a pessoa quer contribuir, mas não tem tanta experiência em determinada linguagem. [...] podendo entender primeiro todo o contexto, ela vai conseguir propor uma solução. Então faz com que todo mundo consiga ajudar de alguma forma, por exemplo, no começo eu não conseguia ajudar só pelo meu conhecimento, mas depois que

*tive o exemplo, me deixou um pouco mais confiante. Possibilita que muito mais gente contribua e apareça soluções melhores e incentiva que outras pessoas possam colaborar...".* Esse relatos indicam que exemplos não só facilitam a compreensão técnica, mas também ampliam a participação e confiança dos desenvolvedores novatos.

**Resposta à QP2:** Os exemplos contribuíram para a **compreensão do problema**, não apenas ao confirmar percepções iniciais, mas principalmente ao **esclarecer dúvidas, ampliar o entendimento do contexto e apresentar caminhos possíveis para a solução**. Foi relatado que os exemplos os ajudaram a pensar em alternativas viáveis, inspiraram propostas de solução e serviram como guia para estruturar seu raciocínio. Além de promoverem maior clareza, os exemplos também tiveram impacto direto na confiança dos novatos, ao permitir que avaliassem com mais precisão se possuíam o conhecimento necessário para contribuir. Embora rótulos como "good first issues" ajudem a sinalizar tarefas potencialmente acessíveis, eles não oferecem elementos suficientes para que o colaborador iniciante julgue sua capacidade de resolução. Nesse sentido, os exemplos trabalhados funcionam como um recurso adicional que **reduz barreiras cognitivas, facilita a tomada de decisão e estimula a participação ativa** dos novatos em projetos de SL.

## 5 Ameaças à Validade

A seguir, discutimos as ameaças à validade deste estudo, considerando aspectos internos, de construção, de conclusão e externos. Essa reflexão é fundamental para contextualizar os resultados obtidos e delimitar o alcance das evidências apresentadas.

**Validade Interna.** A ausência de um período prévio para que os participantes se familiarizassem com o projeto e sua documentação pode ter prejudicado a compreensão da *issue*, e reduzido a eficácia do exemplo extraído do Stack Overflow. Esse fator pode ter impactado os resultados observados. Entretanto, todos os participantes receberam o mesmo contexto inicial e instruções uniformes, minimizando variações introduzidas por intervenção direta.

**Validade de Construção.** A utilização de apenas um exemplo por fonte representa uma limitação, pois características específicas de cada exemplo (formato, clareza, linguagem) podem ter influenciado a avaliação. Também foi considerada apenas uma tarefa de análise de *issue*, o que restringe o escopo de aplicação dos resultados a outros tipos de atividade. Apesar disso, o uso de uma *issue* real e o equilíbrio de complexidade entre os exemplos visaram garantir relevância e comparabilidade direta entre os grupos.

**Validade de Conclusão.** Embora o estudo tenha seguido critérios para balanceamento entre os grupos, cada participante analisou apenas um exemplo, o que limita a comparação entre as abordagens. Como consequência, o perfil dos participantes pode ser um fator de confusão. Além disso, a análise qualitativa, embora conduzida com rigor e triangulação entre fontes (transcrições, verbalizações e questionários), depende da interpretação dos pesquisadores, o que pode introduzir viés. Para mitigar esse risco, as etapas de codificação foram discutidas entre os autores para validação de interpretações emergentes. Embora não seja possível eliminar completamente a

subjetividade nesse tipo de análise, essas estratégias aumentam a confiabilidade e a transparência do processo.

**Validade Externa.** O estudo foi conduzido com um único projeto (Pandas) e uma única *issue*, o que restringe a generalização dos resultados para outros contextos. Apesar do projeto escolhido atender critérios educacionais amplamente adotados na literatura, resultados semelhantes não são garantidos em projetos com menor maturidade, menos documentação ou outros tipos de tarefas (e.g., revisão de código ou implementação). Adicionalmente, a maioria dos participantes era estudante de pós-graduação, o que pode não refletir a diversidade de perfis encontrados em comunidades de SL.

Também é importante considerar outras limitações que complementam as ameaças já discutidas. O uso do protocolo *Thinking Aloud*, embora eficaz para acessar o raciocínio dos participantes, pode ter influenciado seu comportamento cognitivo natural, afetando as validades interna e de construção. Além disso, o tempo limitado para leitura e análise pode ter reduzido o engajamento com a *issue* e o exemplo, afetando as validades interna e externa.

## 6 Conclusão

Exemplos trabalhados são amplamente reconhecidos como recursos eficazes para apoiar o aprendizado de iniciantes. Este estudo investigou como exemplos gerados pelo ChatGPT e extraídos do Stack Overflow podem apoiar desenvolvedores novatos na análise e compreensão de *issues* em projetos de software livre (SL).

Os resultados da pesquisa mostram que ambos os exemplos foram percebidos como úteis, mas com algumas diferenças: o exemplo do ChatGPT foi avaliado como mais estruturado, claro e instrutivo, enquanto o do Stack Overflow exigiu maior esforço de interpretação por parte dos participantes. Ainda assim, os dois exemplos ajudaram os novatos a compreender a tarefa proposta, levantar hipóteses de solução e avaliar se possuíam o conhecimento necessário para contribuir. Esse apoio foi particularmente importante para reduzir a insegurança e reforçar a autoconfiança na decisão de contribuir.

A principal contribuição deste trabalho está em evidenciar que exemplos trabalhados, mesmo quando não elaborados com fins instrucionais, podem atuar como instrumentos de apoio pedagógico no processo de entrada em projetos de SL. Esses exemplos complementam abordagens tradicionais, como o uso de rótulos do tipo *good first issue*, oferecendo suporte adicional à tomada de decisão e à compreensão da tarefa.

Como trabalhos futuros, propõe-se considerar a análise de múltiplos exemplos por fonte, a exploração de diferentes tipos de tarefas (além da análise de *issues*) e o acompanhamento longitudinal de contribuições realizadas após o uso desses exemplos. Além disso, é importante investigar como essas estratégias podem ser integradas de forma sistemática nos processos de *onboarding* em comunidades de SL, potencializando seu papel formativo e inclusivo.

## DISPONIBILIDADE DE ARTEFATO

Os artefatos utilizados neste estudo, incluindo a *issue*, os exemplos, protocolo de entrevista e formulários estão disponíveis em [16].

## AGRADECIMENTOS

Os autores agradecem ao apoio da CAPES (Financiamento 001), da Fundação Araucária (Processos PRD2023361000135 e PRD2023361000043) e do CNPq (Processos 306774/2025-9 e 140446/2025-7).

## REFERÊNCIAS

- [1] Robert K. Atkinson, Sharon J. Derry, Alexander Renkl, and Donald Wortham. 2000. Learning from Examples: Instructional Principles from the Worked Examples Research. *Review of Educational Research* 70, 2 (2000), 181–214. <https://doi.org/10.3102/00346543070002181>
- [2] Sogol Balali, Umayal Annamalai, Hema Susmita Padala, Bianca Trinkenreich, Marco A. Gerosa, Igor Steinmacher, and Anita Sarma. 2020. Recommending Tasks to Newcomers in OSS Projects: How Do Mentors Handle It?. In *Proceedings of the 16th International Symposium on Open Collaboration* (Virtual conference, Spain) (*OpenSym '20*). Association for Computing Machinery, New York, NY, USA, Article 7, 14 pages. <https://doi.org/10.1145/3412569.3412571>
- [3] Sogol Balali, Igor Steinmacher, Umayal Annamalai, Anita Sarma, and Marco Aurelio Gerosa. 2018. Newcomers' barriers... is that all? an analysis of mentors' and newcomers' barriers in OSS projects. *Computer Supported Cooperative Work (CSCW)* 27 (2018), 679–714.
- [4] Sebastian Baltes and Stephan Diehl. 2019. Usage and attribution of Stack Overflow code snippets in GitHub projects. *Empirical Software Engineering* 24, 3 (2019), 1259–1295.
- [5] Sebastian Baltes, Christoph Treude, and Stephan Diehl. 2019. Sotorrent: Studying the origin, evolution, and usage of stack overflow code snippets. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, 191–194.
- [6] Xiangping Chen, Furen Xu, Yuan Huang, Xiaocong Zhou, and Zibin Zheng. 2024. An empirical study of code reuse between GitHub and stack overflow during software development. *Journal of Systems and Software* 210 (2024), 111964.
- [7] Simone de França Tonhão, Thelma Elita Colanzi, and Igor Steinmacher. 2020. A Portal for Cataloging Worked Examples Extracted from Open Source Software. In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering (Natal, Brazil) (SBES '20)*. Association for Computing Machinery, New York, NY, USA, 493–498. <https://doi.org/10.1145/3422392.3422471>
- [8] Zack Delile, Sean Radel, Joe Godinez, Garrett Engstrom, Theo Brucker, Kenzie Young, and Sepideh Ghanavati. 2023. Evaluating Privacy Questions From Stack Overflow: Can ChatGPT Compete? *arXiv preprint arXiv:2306.11174* (2023).
- [9] Amir Dirin and Teemu Henrikki Laine. 2024. Examining the Utilization of Artificial Intelligence Tools by Students in Software Engineering Projects. In *CSEDU* (2), 286–293.
- [10] Manuel B Garcia, Teodoro F Revano, Renato R Maliiw, Pitz Gerald G Lagrazon, Arlene Mae C Valderama, Ari Happonen, Basit Qureshi, and Ramazan Yilmaz. 2023. Exploring Student preference between AI-powered chatGPT and Human-curated stack overflow in resolving programming problems and queries. In *2023 IEEE 15th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*. IEEE, 1–6.
- [11] Samia Kabir, David N Udo-Imeh, Bonan Kou, and Tianyi Zhang. 2023. Who Answers It Better? An In-Depth Analysis of ChatGPT and Stack Overflow Answers to Software Engineering Questions. *arXiv preprint arXiv:2308.02312* (2023).
- [12] Alireza Kavianpour. 2024. Teaching Programming Languages by Two Teachers: Instructor and ChatGPT. In *2024 ASEE Annual Conference & Exposition*.
- [13] Moara Sousa Brito Lessa and Christina von Flach G. Chavez. 2020. An Approach for Selecting FLOSS Projects for Education. In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering*. 463–472.
- [14] Jinrun Liu, Xinyu Tang, Linlin Li, Panpan Chen, and Yepang Liu. 2023. ChatGPT vs. Stack Overflow: An Exploratory Comparison of Programming Assistance Tools. In *2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security Companion (QRS-C)*. IEEE, 364–373.
- [15] Saraj Singh Manes and Olga Baysal. 2021. Studying the change histories of stack overflow and github snippets. In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*. IEEE, 283–294.
- [16] Andressa S. S. Medeiros, Thelma Elita Colanzi, Igor Scaliante Wiese, and Igor Steinmacher. 2024. O uso de exemplos trabalhados para facilitar a contribuição de novatos em projetos de software livre. <https://doi.org/10.5281/zenodo.15167298>
- [17] JunSeong Moon, RaeEun Yang, SoMin Cha, and Seong Baeg Kim. 2023. chatGPT vs Mentor: Programming Language Learning Assistance System for Beginners. In *2023 IEEE 8th International Conference On Software Engineering and Computer Systems (ICSECS)*. 106–110.
- [18] Raphael Pham, Yauheni Stolar, and Kurt Schneider. 2015. Automatically recommending test code examples to inexperienced developers. In *Proceedings of the 2015 10th joint meeting on foundations of software engineering*. 890–893.
- [19] Ben Skudder and Andrew Luxton-Reilly. 2014. Worked examples in computer science. In *Proceedings of the Sixteenth Australasian Computing Education Conference-Volume 148*. 59–64.
- [20] Sandro Speth, Niklas Meißner, and Steffen Becker. 2023. Investigating the Use of AI-Generated Exercises for Beginner and Intermediate Programming Courses: A ChatGPT Case Study. In *2023 IEEE 35th International Conference on Software Engineering Education and Training (CSEE&T)*. 142–146.
- [21] Igor Steinmacher, Ana Paula Chaves, Tayana Uchoa Conte, and Marco Aurélio Gerosa. 2014. Preliminary empirical identification of barriers faced by newcomers to Open Source Software projects. In *2014 Brazilian Symposium on Software Engineering*. IEEE, 51–60.
- [22] Igor Steinmacher, Tayana Uchoa Conte, Christoph Treude, and Marco Aurélio Gerosa. 2016. Overcoming Open Source Project Entry Barriers with a Portal for Newcomers. In *Proceedings of the 38th International Conference on Software Engineering (Austin, Texas) (ICSE '16)*. Association for Computing Machinery, New York, NY, USA, 273–284. <https://doi.org/10.1145/2884781.2884806>
- [23] Igor Steinmacher, Marco Aurélio Graciotto Silva, and Marco Aurélio Gerosa. 2014. Barriers faced by newcomers to open source projects: a systematic review. In *Open Source Software: Mobile Open Source Technologies: 10th IFIP WG 2.13 International Conference on Open Source Systems, OSS 2014, San José, Costa Rica, May 6–9, 2014. Proceedings* 10. 153–163.
- [24] Igor Steinmacher, Igor Wiese, Ana Paula Chaves, and Marco Aurélio Gerosa. 2013. Why do newcomers abandon open source software projects?. In *2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*. IEEE, 25–32.
- [25] Xin Tan, Yiran Chen, Haohua Wu, Minghui Zhou, and Li Zhang. 2023. Is It Enough to Recommend Tasks to Newcomers? Understanding Mentoring on Good First Issues. In *Proceedings of the 45th International Conference on Software Engineering (Melbourne, Victoria, Australia) (ICSE '23)*. IEEE Press, 653–664. <https://doi.org/10.1109/ICSE48619.2023.00064>
- [26] Simone Tonhão, Thelma Colanzi, and Igor Steinmacher. 2021. Using real worked examples to aid software engineering teaching. In *Proceedings of the XXXV Brazilian Symposium on Software Engineering*. 133–142.
- [27] Wengran Wang, Yudong Rao, Archit Kwatra, Alexandra Milliken, Yihuan Dong, Neeloy Gomes, Sarah Martin, Veronica Catéte, Amy Isvik, Tiffany Barnes, et al. 2023. A Case Study on When and How Novices Use Code Examples in Open-Ended Programming. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education* V. 1. 82–88.
- [28] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. 2023. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382* (2023).
- [29] Wenxin Xiao, Hao He, Weiwei Xu, Xin Tan, Jinhao Dong, and Minghui Zhou. 2022. Recommending Good First Issues in GitHub OSS Projects. In *Proceedings of the 44th International Conference on Software Engineering (Pittsburgh, Pennsylvania) (ICSE '22)*. Association for Computing Machinery, New York, NY, USA, 1830–1842. <https://doi.org/10.1145/3510003.3510196>
- [30] Bowen Xu, Thanh-Dat Nguyen, Thanh Le-Cong, Thong Hoang, Jiakun Liu, Kisub Kim, Chen Gong, Changan Niu, Chenyu Wang, Bach Le, et al. 2023. Are We Ready to Embrace Generative AI for Software Q&A? *arXiv preprint arXiv:2307.09765* (2023).