



UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS RUSSAS
CURSO DE GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

HERVERSON DE SOUSA SILVA

BEHAVIOR DRIVEN DEVELOPMENT COM AUTOMAÇÃO DE TESTES
UTILIZANDO CYPRESS: UM ESTUDO DE CASO

RUSSAS

2022

HERVERSON DE SOUSA SILVA

BEHAVIOR DRIVEN DEVELOPMENT COM AUTOMAÇÃO DE TESTES UTILIZANDO
CYPRESS: UM ESTUDO DE CASO

Trabalho de Conclusão de Curso apresentado ao
Curso de Graduação em Engenharia de Software
do Campus Russas da Universidade Federal do
Ceará, como requisito parcial à obtenção do
grau de bacharel em Engenharia de Software.

Orientadora: Profa. Dra. Jacilane de Ho-
landa Rabelo

RUSSAS

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas

Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S58b Silva, Herverson de Sousa.

Behavior driven development com automação de testes utilizando cypress: um estudo de caso / Herverson de Sousa Silva. – 2022.

55 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Russas, Curso de Engenharia de Software, Russas, 2022.

Orientação: Profa. Dra. Jacilane de Holanda Rabelo.

1. Desenvolvimento orientado ao comportamento. 2. Automação de testes. 3. Cypress. 4. Teste de software. I. Título.

CDD 005.1

HERVERSON DE SOUSA SILVA

BEHAVIOR DRIVEN DEVELOPMENT COM AUTOMAÇÃO DE TESTES UTILIZANDO
CYPRESS: UM ESTUDO DE CASO

Trabalho de Conclusão de Curso apresentado ao
Curso de Graduação em Engenharia de Software
do Campus Russas da Universidade Federal do
Ceará, como requisito parcial à obtenção do
grau de bacharel em Engenharia de Software.

Aprovada em: 12 de Junho de 2022

BANCA EXAMINADORA

Profa. Dra. Jacilane de Holanda
Rabelo (Orientadora)
Universidade Federal do Ceará (UFC)

Prof. Dr. Alexandre Matos Arruda
Universidade Federal do Ceará (UFC)

Prof. Dr. Williamson Alison Freitas Silva
Universidade Federal do Pampa (UNIPAMPA)

RESUMO

Este trabalho objetivou os impactos e resultados da automação de testes com o auxílio da metodologia do BDD (Behavior Driven Development ou Desenvolvimento Orientado ao Comportamento), cenários de testes e histórias de usuários em um aplicativo de logística. Alguns dos passos galgados para chegar lá foram analisar como o BDD ajuda o time de qualidade e desenvolvimento no processo de criação de testes utilizando a ferramenta Cypress(Framework para automação de testes), bem como, compreender como a equipe de qualidade e desenvolvimento automatiza os cenários de testes; e elaborar sugestões de melhorias, ou apontar fragilidades, através da análise do estudo de caso, comparando com as teorias apresentadas. A partir da análise de dados, foi possível perceber um aumento na qualidade do software através da automação dos cenários de testes. Os participantes também identificaram vantagens no uso do BDD, como a redução do retrabalho relacionado ao esforço de desenvolvimento.

Palavras-chave: Desenvolvimento orientado ao comportamento; Automação de testes; Cypress; Teste de software.

ABSTRACT

This work aimed to analyze the impacts and results of test automation with the aid of the BDD (Behavior Driven Development) methodology, test scenarios, and user stories in a logistics application. Some of the steps taken to achieve this were to analyze how BDD helps the quality and development team in the process of creating tests using the Cypress tool (a framework for test automation), as well as to understand how the quality and development team automates test scenarios; and to elaborate suggestions for improvements or point out weaknesses, through the analysis of the case study, comparing it with the presented theories. From the data analysis, it was possible to perceive an increase in software quality through the automation of test scenarios. The participants also identified advantages in the use of BDD, such as the reduction of rework related to the development effort.

Keywords: Behavior driven development; test automation; Cypress; software testing.

LISTA DE FIGURAS

Figura 1 – Testes ponta a ponta testam todo o seu sistema totalmente integrado	14
Figura 2 – Exemplo de cenário de testes	19
Figura 3 – Logo Cypress	20
Figura 4 – Tela cypress	21
Figura 5 – Cenário de teste	38
Figura 6 – Código Cypress	38

LISTA DE QUADROS

Quadro 1 – Quadro comparativo dos trabalhos relacionados (Fonte Própria).	23
Quadro 2 – Questionário.	28

LISTA DE ABREVIATURAS E SIGLAS

E2E	<i>End-To-End</i>
GUI	<i>Graphical User Interface</i>
NLP	<i>Natural Language Program</i>
PO	<i>Product owner</i>
QA	<i>Quality Assurance</i>
TI	Tecnologia da Informação
US	<i>User Stories</i>

SUMÁRIO

1	INTRODUÇÃO	10
1.1	Problemática	10
1.2	Justificativa	11
1.3	Estrutura do trabalho	11
2	OBJETIVOS	12
2.1	Objetivo geral	12
2.2	Objetivos específicos	12
3	FUNDAMENTAÇÃO TEÓRICA	13
3.1	Teste de software	13
3.2	Teste ponta a ponta	14
3.2.1	<i>Tipos de teste ponta a ponta</i>	15
3.3	Automação de testes	15
3.4	Behaviour-Driven Development	16
3.4.1	<i>Histórias de usuários</i>	18
3.4.2	<i>Cenários de teste</i>	18
3.5	Cypress	19
4	TRABALHOS RELACIONADOS	22
4.1	Comparação entre os trabalhos relacionados	23
5	METODOLOGIA	24
5.0.1	APRESENTAÇÃO DA EMPRESA	24
5.0.2	<i>Delineamento da pesquisa e técnica para a coleta de dados</i>	24
5.1	Apresentação e análise dos resultados	29
5.2	Aplicação BDD e automatização	37
6	CONSIDERAÇÕES FINAIS	39
	REFERÊNCIAS	41
	APÊNDICES	43
	APÊNDICE A – Instrumento de coleta de dados	43
	APÊNDICE B – Respostas dos colaboradores	47

1 INTRODUÇÃO

A exigência da qualidade de software vem crescendo e se tornando o centro das atenções para o desenvolvimento de sistemas, antes era apenas como uma vantagem no mercado agora virou um processo principal e indispensável para que a empresa consiga se destacar no mercado de software que está sempre em constantes atualizações (Warmup, 2021) .

A qualidade de software é um processo que busca garantir a perfeita execução de processos e artefatos concebidos para um produto, buscando eliminar os diversos tipos de defeitos que surjam em todas as fases do processo de desenvolvimento (Bartié, 2002) .

Para um desenvolvimento com um maior nível de qualidade é necessário a adoção de padrões, introduzindo assim ao projeto uma série de definições já replicadas em outros cases de sucesso a um projeto que se pretende iniciar. Entre os padrões mais conhecidos pela comunidade quando se pensa em teste de software, (Smart, 2014) BDD (*Behavior Driven Development*) é praticado com sucesso em um grande número de organizações de todos os tamanhos ao redor do mundo, em uma variedade de maneiras diferentes. Em especificação, por exemplo, estudos de caso para mais de 50 dessas organizações.

1.1 Problemática

A automação de testes tem se tornado uma atividade essencial para a entrega de um produto com um alto nível de qualidade. O processo de automação busca a melhoria da qualidade baseado em teste de software aplicando a realização de testes antes manuais em fluxos automatizados (Delamaro *et al.*, 2013). Neste sentido, as empresas começaram a usar essas ferramentas e métodos em busca da garantia da qualidade do software.

Um caso real de problemática de qualidade de testes acontece recorrentemente na empresa estudada, onde para um dos seus principais produtos, em que eram criados casos de testes em linguagem natural seguindo o padrão BDD, no entanto, a cobertura dos cenários por meio da automação não era realizada.

Assim, a proposta deste trabalho é apresentar um estudo de caso de aplicação prática de técnicas utilizadas no modelo BDD como forma de buscar uma maior taxa de nível de cobertura de testes e nível de qualidade da aplicação cujas técnicas foram aplicadas.

1.2 Justificativa

Este trabalho apresenta um estudo de caso na automação de testes de ponta a ponta em um app mobile por meio do desenvolvimento orientado a comportamento, buscando analisar os dados obtidos em conjunto ao time de qualidade e desenvolvimento responsável pela aplicação. A empresa, em que este estudo será aplicado, atua no segmento de logística há 13 anos, possuindo clientes em diferentes países. Devido ao seu impacto global, onde sua solução afeta diretamente o consumo de diferentes cadeias, é preciso que exista uma administração eficaz do nível de qualidade e desenvolvimento de software, havendo assim a satisfação dos clientes.

1.3 Estrutura do trabalho

Esse trabalho é constituído por 6 capítulos organizados da seguinte forma:

- O segundo capítulo aborda os objetivos gerais e específicos deste trabalho.
- O terceiro capítulo apresenta a fundamentação teórica.
- O quarto capítulo mostra os trabalhos relacionados com esta pesquisa.
- O quinto capítulo apresenta os procedimentos metodológicos utilizados.
- O sexto capítulo capítulo é apresentado as conclusões finais deste trabalho.

2 OBJETIVOS

2.1 Objetivo geral

O objetivo geral deste trabalho é apresentar os impactos e resultados da automação de testes com o auxílio da metodologia BDD na empresa estudada.

2.2 Objetivos específicos

- Analisar como o BDD ajuda o time de qualidade e desenvolvimento no processo de criação de testes;
- Compreender como a equipe de qualidade e desenvolvimento automatiza os cenários de testes;
- Elaborar sugestões de melhorias, ou apontar fragilidades, através da análise do estudo de caso, comparando com as teorias apresentadas.

3 FUNDAMENTAÇÃO TEÓRICA

Esse capítulo apresenta a fundamentação teórica necessária para um melhor entendimento e desenvolvimento desta pesquisa: Desenvolvimento orientado a comportamento, automação de testes, histórias de usuário, cenários de testes, testes de ponta a ponta e cypress.

3.1 Teste de software

Segundo (Sommerville, 2011) teste de software é um processo que busca defeitos antes do uso do software final, ele busca analisar possíveis anomalias ou erros antes de ser entregue ao cliente, contudo, é constatar que testes realizados de forma errônea pode custar o ciclo de vida do projeto e da empresa. Como bem nos assegura (Pressman; Maxim, 2016), teste de software é um procedimento para revelar erros no software. Trata-se inegavelmente de que sem eles, a chance do software ter grandes falhas é maior.

Para Rios (2006, p.10) teste de software facilita na avaliação do comportamento com base no que foi especificado, visando executar de forma controlada. Neste contexto, fica claro que é considerado uma forma de validação de software. O mais preocupante, contudo, é constatar que não há possibilidade de executar todo o software e garantir que ele está livre de anomalias. Pois é quase impossível testar todas as combinações do sistema:

Teste de software permite Muitas outras definições poderiam ser ainda citadas, porém em essência, teste de software é um processo que visa a sua execução de forma controlada, com o objetivo de avaliar o seu comportamento baseado no que foi especificado. A execução dos testes é considerada um tipo de validação.

Como se pode verificar nessa citação, Teste de software é aplicado em projetos de softwares em fase de desenvolvimento. Evidentemente a aplicação pode ser utilizada como forma de garantia de qualidade na validação do software.

Especifica determinadas condições que serão executadas com base nos resultados esperados. Cita-se, como exemplo, uma abordagem prática de teste de software pode ser vista em um software onde se deseja verificar um fluxo correto com base em uma especificação de requisitos. Ainda para Rios (2006, p. 14):

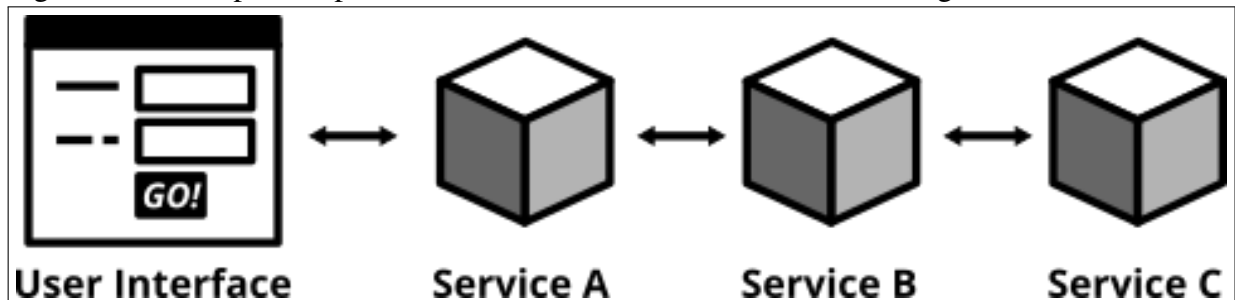
Conforme Boehm (1976), quanto mais tarde um defeito for identificado mais caro fica para corrigi-lo e mais, os custos de descobrir e corrigir defeitos no software aumentam exponencialmente na proporção em que o trabalho evolui através das fases do projeto de desenvolvimento. Nesse sentido, Teste de software permite um projeto que se tem um investimento inicial em testes de software aumentam as chances do ciclo de vida do projeto ser maior pois cada vez que for evoluindo os erros serão mais rastreáveis.

Logo, é importante compreender que um desenvolvedor que não utiliza o testes de software, pode impactar negativamente na qualidade do produto e da experiência com o cliente final. Nesse sentido, vamos exemplificar Teste de software como uma estratégia de verificar e validar um software assim como aumentar sua qualidade eliminando todos os erros existentes no mesmo.

3.2 Teste ponta a ponta

O termo ponta a ponta *End-To-End* (E2E) significa do início ao fim. O teste E2E é definido como um tipo de teste que valida todos os componentes do aplicativo e sua arquitetura, ele testa se a interface do usuário do software funciona corretamente. A entrada do usuário deve acionar os passos corretos, os dados devem ser mostrados ao usuário e o estado da aplicação deve mudar conforme o esperado. Também envolvem todas as funcionalidades de back-end, como ilustrado na Figura 1. (Selenium, 2021).

Figura 1 – Testes ponta a ponta testam todo o seu sistema totalmente integrado



Fonte: Fowler (2018)

O teste E2E pode ser realizado através de ferramentas como cypress, Selenium. Testar o comportamento da interface de usuário pode ser bem simples. Clica em um determinado componente, insere dados e deseja que o estado da interface do usuário mude de acordo com a expectativa. O teste E2E é crucial na garantia da qualidade do seu software, à medida que o seu software aumenta, é necessário não apenas garantir que ele funcione em uma vasta gama de dispositivos, ambientes e sistemas operacionais, mas também realizar a validação de serviços externos que o aplicativo utiliza funcionem conforme o esperado. Felizmente, testar o comportamento da sua interface de usuário é bem simples. Você clica aqui, insere dados lá e deseja que o estado da interface do usuário mude de acordo. As estruturas modernas de aplicativos de página única (react , vue.js , Angular e similares) geralmente vêm com suas próprias ferramentas e auxiliares que permitem testar completamente essas interações em um

nível bastante baixo (teste de unidade). Mesmo se você rolar sua própria implementação de frontend usando javascript vanilla, você pode usar suas ferramentas de teste regulares como Jasmine ou Mocha . Com um aplicativo renderizado no lado do servidor mais tradicional, os testes baseados em Selenium serão sua melhor escolha.

3.2.1 Tipos de teste ponta a ponta

Existem dois tipos de teste E2E: vertical e horizontal:

- Em testes E2E horizontal: o tipo mais comumente usado, que passa pelo usuário fluxo de trabalho ou transação em vários aplicativos do início ao fim para garantir que cada processo ocorra conforme o esperado (Rocha, 2021). Na pirâmide de testes E2E médio e horizontal acontecem no nível mais alto, testando o todo sistema da forma como será usado, como um usuário do mundo real.
- Em testes E2E vertical: o teste será implementado em todos os níveis da pirâmide, de testes de unidade, em seguida, continuando até os testes de integração. Verifique se cada camada está funcionando de forma independentemente. (Rocha, 2021).

3.3 Automação de testes

A automação de testes tem se tornado uma atividade essencial para a entrega de um produto com um alto nível de qualidade. O processo de automação busca a melhoria da qualidade baseado em teste de software aplicando a realização de testes antes manuais em fluxos automatizados (Delamaro, 2013).

Segundo Bernardo *et al*(2008) os testes automatizados podem ser definidos como programas ou scripts simples que exercitam funcionalidades do projeto sendo testado e fazem verificações automáticas nos efeitos colaterais obtidos. Portanto, por meio da automação é possível verificar rapidamente sem a realização dos fluxos de forma manual a introdução de problemas recorrentes da introdução de novas funcionalidades ou modificações em funcionalidades existentes. A automação permite a execução contínua em um ambiente de testes controlado. Sempre que possível, os testes de unidade devem ser automatizados para que sejam executados e verificados sem intervenção manual (Sommerville, 2011).

Em testes automatizados, os casos de testes e os seus cenários de testes são todos

desenvolvidos no software que a empresa está utilizando e serão realizados toda vez que for uma alteração no sistema (Sommerville, 2011), o autor deixa claro que ao incluir uma nova funcionalidade no software é preciso documentar o fluxo da funcionalidade, os cenários e os casos de testes ajudam nesse processo.

Por isso, a tarefa de automatizar os casos de testes, visando aumentar a produtividade em relação aos testes manuais e a qualidade do produto, é utilizar técnicas que facilitam a construção dos cenários de teste e casos de testes, como por exemplo o BDD.

O problema é que, muitas vezes, os testes ainda são feitos exclusivamente de forma manual. Não que testes manuais não sejam importantes, pelo contrário, são extremamente valiosos, mas eles são muito lentos se compararmos com testes automatizados. Testes manuais dependem muito da habilidade do testador, que muitas vezes precisa realizar os mesmos testes diversas vezes. Essa tarefa se torna cansativa, tediosa, lenta e pode fazer com que falhas passem despercebidas (Peixoto, 2018).

Segundo Bartié (2002) os testes automatizados podem ser definidos como programas ou scripts simples que exercitam funcionalidades do projeto sendo testado e fazem verificações automáticas nos efeitos colaterais obtidos. Portanto, por meio da automação é possível verificar rapidamente sem a realização dos fluxos de forma manual a introdução de problemas recorrentes da introdução de novas funcionalidades ou modificações em funcionalidades existentes. A automação permite a execução contínua em um ambiente de testes controlado. Sempre que possível, os testes de unidade devem ser automatizados para que sejam executados e verificados sem intervenção manual (Delamaro *et al.*, 2013).

3.4 Behaviour-Driven Development

Segundo North (2003) Behavior Driven Development o foi introduzido por Dan North em meados de 2003. O BDD trabalha como um processo colaborativo que envolve os membros do time envolvidos na parte da criação de uma história, geralmente os desenvolvedores, *Quality Assurance* (QA) e *Product owner* (PO). Com foco no alinhamento dos membros e no entendimento compartilhado. Como bem nos assegura (Silva, 2022), Behavior Driven Development é uma técnica de desenvolvimento ágil que visa juntar as regras de negócios com o desenvolvimento de software, focando no comportamento do mesmo.

Para Smart (2015, p. 12) Behavior Driven Development facilita fornecer um software mais valioso com menos desperdício. Mas em consideração ao valor para o cliente de uma nova melhoria não é percebida até que a mesma esteja sendo usada pelo cliente em ambiente de

produção. Com suas histórias de usuário criasse as features com os membros da equipe gerando uma documentação viva do projeto.:

Behavior Driven Development permite o Behavior-Driven Development (BDD) é um conjunto de práticas de engenharia de software projetadas para ajudar as equipes a criar e entregar softwares mais valiosos e de maior qualidade com mais rapidez. Ele se baseia em práticas ágeis e enxutas, incluindo, em particular, Desenvolvimento Orientado a Testes (TDD) e Design Orientado a Domínio (DDD). Mas o mais importante é que o BDD fornece uma linguagem comum baseada em frases simples e estruturadas expressas em inglês (ou no idioma nativo das partes interessadas) que facilitam a comunicação entre os membros da equipe do projeto e as partes interessadas do negócio.

Como se pode verificar nessa citação, Behavior Driven Development é aplicado em toda área onde será preciso melhorar a validação das regras de negócio com o desenvolvimento, e integrar os membros da equipe para uma melhor compreensão técnica e não técnica. Evidentemente a aplicação pode ser utilizada para descrever comportamentos em uma única linguagem que seja de fácil entendimento a QA, desenvolvedores, e partes interessadas no projeto, para melhorar a comunicação e documentação. nele os casos de testes podem ser implementados com uma linguagem onipresente usada pelas partes interessadas no projeto, deixando uma comunicação mais clara e com pouco jargões possibilitando uma maior colaboração entre os membros da equipe. Ainda para Smart (2015, p. 28):

O BDD tem tudo a ver com concentrar o esforço de desenvolvimento na descoberta e entrega dos recursos que fornecerão valor comercial e evitar aqueles que não fornecerão. Quando uma equipe cria um recurso que não está alinhado com os objetivos de negócios subjacentes ao projeto, o esforço é desperdiçado para os negócios. Da mesma forma, quando uma equipe escreve um recurso que o negócio precisa, mas de uma forma que não é útil para o negócio, a equipe precisará retrabalhar o recurso para se adequar ao projeto, resultando em mais desperdício. O BDD ajuda a evitar esse tipo de esforço desperdiçado, ajudando as equipes a se concentrarem em recursos alinhados às metas de negócios. Nesse sentido, Behavior Driven Development permite o compartilhamento do conhecimento entre os membros da equipe, como QA, DEV e PO. Focando no comportamento dos requisitos na visão do cliente na hora do desenvolvimento.

Logo, é importante compreender que o BDD é baseado em conversas e feedbacks. Pois, essas conversas impulsionam e geram o entendimento da equipe sobre os requisitos e como eles podem fornecer valor ao cliente com base nesses requisitos. Sem esses feedbacks será difícil extrair os benefícios do BDD. Nesse sentido, vamos exemplificar Behavior Driven Development como uma importante ferramenta de testagem e apesar de não ser uma framework de testes automatizados. O objetivo é melhorar as funcionalidades, escrevendo códigos que vão validando os requisitos dos clientes.

3.4.1 Histórias de usuários

As histórias de usuários (*Users Stories*) são feedbacks obtidos dos usuários que usam o sistema (Cohn, 2004) descreve histórias de usuários consistindo em três partes:

- uma descrição escrita usada para identificar a história em um contexto curto;
- conversas sobre a história para esclarecer a história em detalhes;
- testes que listam os detalhes do documento são válidos quando uma história é finalizada.

As histórias de usuários seguem um modelo simples: Como um (*type of user*), eu quero (*some goal*) para (*some reason*) (Cohn, 2004).

3.4.2 Cenários de teste

Segundo Wynne (2017) Cenários de teste é depois que o Product Owner finaliza o refinamento das histórias de usuário, o analista de testes deve revisar essas histórias e requisitos e, em seguida, escrever os cenários de validação para as funcionalidades que são selecionadas para serem implementadas. Para estruturar esses cenários o BDD utiliza as palavras chaves: Dado algum contexto inicial, Quando um evento ocorre, Em seguida, certifique-se de alguns resultados. Como bem nos assegura (Smart, 2014), cenários de teste é um formato simples adotado pelos praticantes de BDD desde então em torno de uma estrutura básica "Dado... Quando... Então". Um exemplo: Com um pouco de prática, as partes interessadas rapidamente se sentem confortáveis com esse formato, ajudando pessoas que não conhecem muito da regra de negócio do projeto consegue se adaptar rapidamente ao projeto. No BDD, os cenários de teste de aceitação são explicitamente escritos com o seguinte sintaxe simples:

- Dado algum contexto inicial;
- Quando um evento ocorre;
- Em seguida, certifique-se de alguns resultados.

Cenários de teste permite que você escreva especificações executáveis no idioma nativo de seus usuários e produzirá resultados de teste que relatam sucesso ou falha não em termos de classes e métodos, mas em termos dos recursos solicitados pelas partes interessadas. As partes interessadas poderão ver suas próprias palavras aparecendo na documentação viva, o que faz maravilhas ao aumentar sua confiança de que você entende seus problemas. É isso que ferramentas BDD com Cucumber, JBehave e SpecFlow trazem para a mesa.

Como se pode verificar nessa citação, os cenários de testes são usados para docu-

mentar principalmente os requisitos do cliente em forma de etapas para que o desenvolvedor e o QA consigam entender o fluxo do que é para ser testado e implementado. Como mostrado no exemplo na Figura 2:

Figura 2 – Exemplo de cenário de testes

Feature: Humpty Dumpty
Scenario: Fall
Given I am on a wall
When I lose my balance
Then I should have a great fall
Scenario: Reassembly
Given all the king's horses
And all the king's men
When they attempt to put me back together again
Then I should be in one piece

Fonte: Dees *et al.* 2013.

3.5 Cypress

Segundo Mwaura (2021) framework cypress¹ é uma estrutura de automação de teste end-to-end construída e projetada para aplicativos web. Seu foco é na eliminação de erros em testes, garantindo que o desenvolvedor possa escrever, testar e executar testes no navegador web sem precisar de ajustes ou bibliotecas adicionais. Como bem nos assegura cypress (2021), framework cypress é uma estrutura de teste ponta a ponta baseada na linguagem de programação JavaScript. Ele é construído em cima do Mocha, uma sintaxe do BDD que se encaixa perfeitamente com teste de integração e teste de unidade, que é uma estrutura de teste de JavaScript rica em recursos em execução no navegador, tornando o teste assíncrono de fácil uso.

Para Mwaura (2021, p. 19) framework cypress facilita testar aplicativos da web e como a única maneira de automatizar a web é usando JavaScript, ele foi escrito especificamente para equipes que trabalham com JavaScript para web no desenvolvimento de seus produtos, também ele é usado em equipes que precisam começar rapidamente com o processo de testes de unidade, integração e testes de ponta a ponta e sem as dificuldades de configuração da estrutura do teste.:

O Cypress foi criado para desenvolvedores JavaScript e engenheiros de QA que desejam configurar rapidamente uma estrutura de automação e testar seus aplicativos da web de ponta a ponta, sem gastar muita largura de banda na

¹ <https://cypress.io/>

Figura 3 – Logo Cypress



Fonte: Cypress (2021).

configuração de estruturas de teste ou no entendimento das tecnologias por trás da construção de estruturas de teste. Com o Cypress, é fácil para os desenvolvedores ir além de escrever testes de unidade, para escrever testes de integração e até mesmo testes de aceitação com recursos como stubbing de dependências externas e teste de comportamento de seus aplicativos.

Como se pode verificar nessa citação, framework cypress cypress se aplica em projetos de front-end que utilizam o JavaScript como linguagem principal do projeto, ele foi criado por desenvolvedores e engenheiros de QA que necessitam de um framework que analisa e testa uma aplicação para a garantia da qualidade. Evidentemente a aplicação pode ser utilizada para testar seus aplicativos web de ponta a ponta, testes de unidade, testes de integração ou até mesmo testes de aceitação.

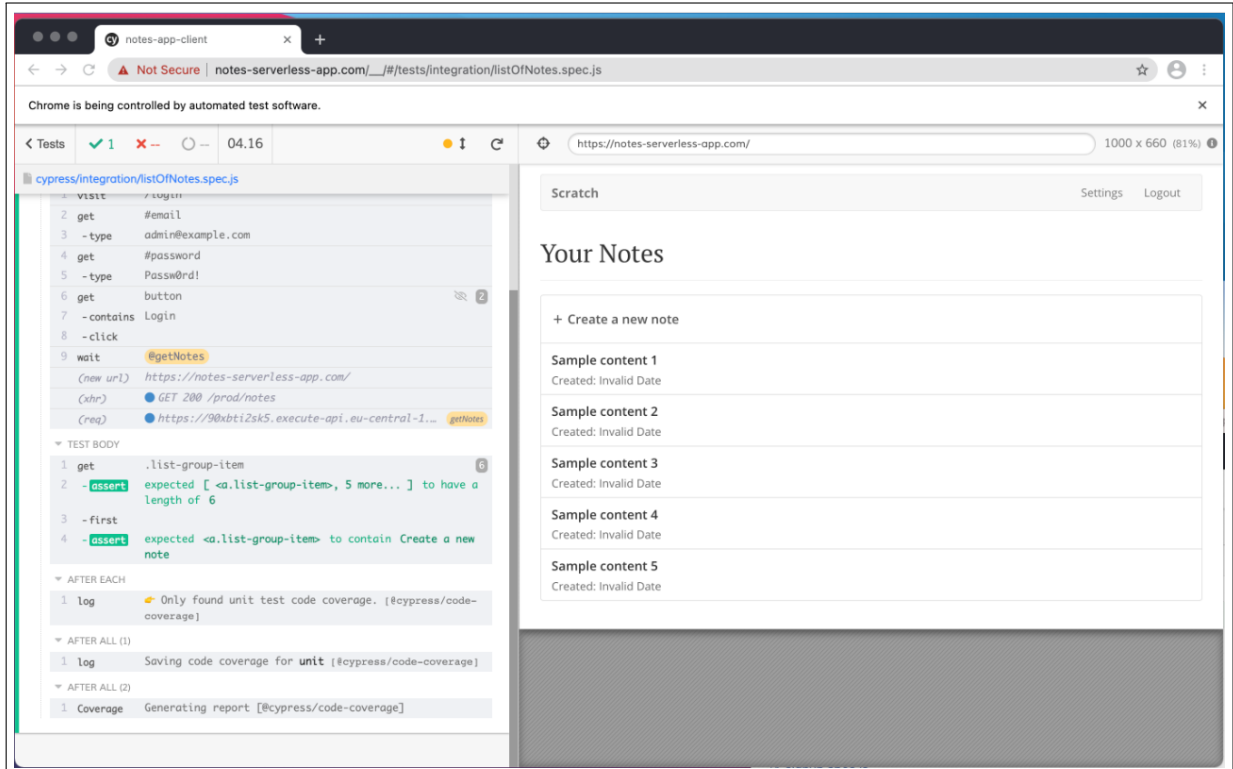
O cypress é usado para testar o fluxo da aplicação analisando os passos que são realizados através de clicks e seus retornos, comparando o que foi recebido da aplicação com o que o desenvolvedor ou QA deseja ser esperado, exemplificado na figura 4. Cita-se, como exemplo, quando se quer realizar uma entrega o app mostra um cartão com os produtos que serão entregues, então deseja-se verificar se todos os produtos estão sendo mostrado nele, basta selecionar o cartão com a tag do HTML e o cypress irá analisar todos os componentes verificando o que foi especificado na verificação.

Ainda para Mwaura (2021, p. 21):

O Cypress é executado no navegador e isso significa que a linguagem que ele suporta será sempre JavaScript, pois o código de teste será sempre avaliado no navegador. Ser capaz de rodar no navegador significa que para se conectar a um banco de dados ou a um servidor, só podemos usar os comandos Cypress de `cy.exec()`, `cy.request()` ou `cy.task()`, que fornecem uma maneira de expor o banco de dados ou o servidor, o que pode dar mais trabalho do que se pudéssemos definir explicitamente suas configurações e o Cypress as entendesse. Ter os testes executados no navegador cria uma ótima experiência para a execução de testes, mas é um pouco complicado conectar funcionalidades que precisam ser executadas fora do navegador.. Nesse sentido, framework cypress permite

facilitar as asserções de BDD e um navegador que pode ser emparelhado com qualquer estrutura de teste de JavaScript.

Figura 4 – Tela cypress



Fonte: elaborado pelo autor (2022).

Logo, é importante compreender que o cypress ajuda os desenvolvedores e os analistas de QA na criação dos testes, principalmente nos testes de integração onde é possível analisar todos os passos realizados pelo cypress. Nesse sentido, vamos exemplificar framework cypress como uma ferramenta que recebe as ações a serem realizadas e são executadas através de clicks realizando suas asserções.

4 TRABALHOS RELACIONADOS

Paula (2019) a sua pesquisa apresentou os resultados e conclusões do acompanhamento da aplicação da técnica de BDD - Desenvolvimento Dirigido por Comportamento - e de histórias de usuários - *User Stories* (US) como melhoria de processo em um projeto que utiliza Kanban. Foi utilizada a pesquisa-ação como método de pesquisa, coleta de dados via questionários com os membros do projeto e pela observação para a análise da qualidade do software em uma startup de Tecnologia da Informação (TI), abordando os principais conceitos de qualidade de software como o BDD ajudou no processo de criação de software. Na conclusão do estudo verificou-se que as práticas de BDD são bastante utilizadas durante a fase de desenvolvimento empírico, mas também foi mostrado que existe uma grande dificuldade em aderir tanto às práticas de testes, quanto às práticas de usabilidade pela equipe de desenvolvimento. Os membros da equipe perceberam o valor no uso do BDD no projeto em que estavam trabalhando e identificaram vantagens no seu uso.

(Chiavegatto *et al.*, 2013) aplicou a metodologia (BDD) junto com a técnica de automatização de testes com o auxílio da ferramenta para criação de cenário de testes JBehave, e para a automatização desses cenários foi utilizado as ferramentas de teste Selenium e JUnit. Foi realizado um estudo de caso em um sistema hospitalar contendo 9 módulos onde apenas 2 módulos foram automatizados pois tinham um grau maior de problemas em produção, em virtude de constantes mudanças, requerendo testes de regressão. Antes os testes de regressão, que antes eram custosos e executados manualmente, passaram a ser executados de forma automática. Foi realizado um comparativo entre o processo tradicional em relação ao orientado a comportamento.

(Bahaweres *et al.*, 2020) aplicou o método de desenvolvimento orientado por comportamento (BDD) como auxílio para automação de testes de ponta a ponta com a estrutura Cucumber nos aplicativos web CURA e Swag Labs. Com o auxílio da ferramenta Katalon Studio com base no conceito de *Natural Language Program* (NLP) nos aplicativos. O objetivo da sua pesquisa foi o de encontrar os resultados da automação de testes da *Graphical User Interface* (GUI) com a implementação de Cucumber usando o katalon Studio e os tempos de respostas gastos na automação dos testes usando o Katalon Studio. No final os casos de testes foram automatizados com base nos resultados médios dos cálculos, os resultados do tempo de resposta na aplicação web Swag Labs, obtiveram uma média de 2,122 minutos.

4.1 Comparação entre os trabalhos relacionados

São apresentados no quadro a seguir os aspectos que diferenciam os trabalhos relacionados a este. Os aspectos assinalados com um “Sim” indicam sua existência nos trabalhos, aqueles que apresentam o “Não” - indica a ausência do aspecto no trabalho. Este trabalho é apresentado na coluna com o título “Esta pesquisa”.

Quadro 1 – Quadro comparativo dos trabalhos relacionados (Fonte Própria).

	CHIAVEGATTO (2013)	PAULA (2019)	BAHAWERES (2020)	Esta pesquisa
Automação de testes	Sim.	Não.	Sim.	Sim.
Teste de ponta a ponta	Sim.	Não.	Sim.	Sim.
Tempo de resposta	Não.	Não.	Sim.	Opcional.
BDD	Sim.	Sim.	Sim.	Sim.
Cypress	Não	Não	Não	Sim.

Fonte: Elaborado pelo autor (2022).

5 METODOLOGIA

5.0.1 APRESENTAÇÃO DA EMPRESA

O local escolhido para o desenvolvimento desta pesquisa foi uma empresa no ramo de logística de roteirização, situada na cidade de Fortaleza, no estado do Ceará. Devido a dados sigilosos de clientes e funcionários, nomes não serão usados explicitamente nesta monografia. Daqui pra frente, a empresa será identificada apenas com "Empresa X". Há mais de 8 anos vem oferecendo informações detalhadas sobre seu plano de rota para seus clientes e empresas fornecedoras. Além disso, possui uma excelente infraestrutura técnica, profissionais técnicos capacitados para desenvolvimento, pois investe constantemente na especialização em software de logística, software de gerenciamento da cadeia de suprimentos e serviços baseados em nuvem para empresas de logística. Para a realização dos estudos propostos neste trabalho, os autores selecionaram o aplicativo da Empresa X, cujo objetivo é fornecer visibilidade sobre rotas e status de atendimento a clientes, bem como fornecer aos motoristas, comerciantes, representantes de vendas informações detalhadas sobre seu plano de rota.

O software realiza a troca de informações com uma aplicação servidora, funcionando em uma arquitetura de client-server. Essa troca de informação é necessária para que o aplicativo tenha acesso aos registros no sistema, este cadastro é realizado por meio de uma aplicação Web, nela é cadastrado uma série de informações, como o equipamento que é utilizado pelo motorista, paradas que o motorista irá atender e outras informações que são necessárias para uma rota logística.

Assim, a realização deste estudo teve como sujeitos de pesquisa os participantes do projeto de automação de testes como desenvolvedores e analistas de QA, o que facilitou o acesso aos membros do projeto. Corroborando com isso, em dezembro deste ano os 7 participantes, objeto que esta pesquisa visou explorar. Destes, 7 responderam ao questionário. Dos 7, 5 são QA e 2 destes são desenvolvedores, todos os 7 membros iniciaram juntos no início do projeto, no caso 2 anos.

5.0.2 Delineamento da pesquisa e técnica para a coleta de dados

Conforme verificado por Gil (2008), um procedimento formal e sistemático com o objetivo de investigação de fenômenos e aumento do conhecimento que induz a reflexão. Trata-se inegavelmente de que busca o progresso da ciência investigando novos conhecimentos. O mais

preocupante, contudo, é constatar que o tempo de planejamento pode ser demorado e dificultoso.

"Estuda um problema relativo ao conhecimento científico ou à sua aplicabilidade." (Marconi e Lakatos, 2003, p.160). Devido aos fins para melhoria da compreensão, essa pesquisa se enquadra na natureza básica.

Como bem nos assegura Fonseca (2012, p. 25), pode-se dizer que a pesquisa descritiva tem como fonte a observação do meio. Neste contexto, fica claro que sua busca é a coleta e registro de dados para um estudo de caso.

A pesquisa se classifica em descritiva devido ao envolvimento de questionários. A base da pesquisa será os questionários realizados no time de qualidade da empresa estudada.

Segundo Gil (2008), a forma de abordagem qualitativa tem como base a interpretação do pesquisador, trata-se inegavelmente de uma pesquisa mais adotável em projetos. O mais preocupante, contudo, é constatar que não existe apenas essa forma de abordagem. Assim, reveste-se a importância da abordagem quantitativa que usa as métricas estatísticas como instrumento de medida. Devido ao uso de questionários com predominância em perguntas abertas para essa coleta dos dados.

Conforme citado acima, a pesquisa tem a abordagem qualitativa devido a interpretação dos questionários respondidos pelo time de qualidade e desenvolvimento do projeto. O método de análise escolhido foi o hipotético-dedutivo devido ao manter a postura do pesquisador em permanente crítica.

Pode-se dizer que o estudo de caso é um estudo que realiza um estudo aprofundado e exaustivo. Neste contexto, para Gil (2008) fica claro que o estudo tem como base um estudo sobre um determinado assunto específico, os casos são referências para não se começar do zero. O mais preocupante, contudo, é constatar que há uma dificuldade na generalização.

Esta pesquisa tem como procedimentos de coleta de dados questionários e fontes bibliográficas. Nesse caso, a pesquisa usará os questionários como base para a investigação e análise. Também fez o uso de materiais já elaborados: livros, artigos científicos, documentos online e revistas em busca de embasamento sobre o BDD e a automação de testes.

Como instrumento para coleta de dados utilizou-se questionários, a fim de se obter uma melhor apreciação do conteúdo apresentado no trabalho. Os dados serão captados através de questionário construído e aplicado com o time de qualidade. Com esse tipo de resumo é possível obter informações em um questionário padronizado para a tabulação dos dados.

De acordo com Marconi e Lakatos(2003, p. 159): Os principais tipos de documentos são:

a) Fontes Primárias - dados históricos, bibliográficos e estatísticos; informações, pesquisas e material cartográfico; arquivos oficiais e particulares; registros em geral; documentação pessoal (diários, memórias, autobiografias); correspondência pública ou privada etc.

b) Fontes Secundárias - imprensa em geral e obras literárias.

Este trabalho teve como base dados que já se encontram disponíveis, pois já foram publicados (livros, teses e etc) e também dados que serão obtidos através dos questionários. Nesse sentido classifica como fontes secundárias e primárias.

Esta pesquisa foi realizada na empresa com os questionários para realizar a aplicação. Onde se escolheu os técnicos do time de qualidade e desenvolvimento que trabalharam na automatização dos cenários de testes no projeto para que se garanta a qualidade do software. Tendo como objetivo: identificar como o BDD foi usado e como ele melhorou nesse processo de automação do projeto. Os resultados, os dados obtidos dessa pesquisa serão analisados com relação ao sucesso ou não da prática do BDD.

A população do estudo foi composta por toda a literatura relacionada ao tema de estudo, indexada nos bancos de dados Scielo (Scientific Electronic Library OnLine), Google Livros, Google Scholar, IEEE(Institute of Electrical and Electronics Engineers), ACM(Association for Computing Machinery).

A seleção foi realizada a partir de literatura criteriosa dos artigos, teses e dissertações encontradas nas bases de dados, selecionada apenas a literatura que atendia aos critérios de inclusão definidos neste estudo. Foram incluídas apenas as publicações que responderam à questão do estudo, publicadas no período de 2002 a 2022, no idioma português e inglês, todos os tipos de delineamentos metodológicos foram aceitas.

De forma geral, o objetivo das perguntas era observar e analisar, entre outros aspectos, o processo de desenvolvimento de software e automação de testes pela introdução do BDD; quais os impactos na qualidade do software, na relação com os clientes, os membros da equipe do projeto. O estudo do pesquisador teve objetivo de colaborar com a comunidade acadêmica universitária explicando como o BDD ajuda na automação de testes em um software. A seguir as etapas para a coleta dos questionários:

- Fez-se um levantamento do time do projeto que ficou responsável pela criação dos cenários de testes e automação dos mesmos, além das melhorias no processo, desde a introdução do BDD.
- Foi realizado um primeiro contato com os membros do time pedindo autorização e sensibilizando sobre a importância desse processo de coleta dos dados.

- Com o consentimento do colaborador, de posse do questionário gerado em formato de formulário, estes foram enviados via Google Forms (Apêndice A). O colaborador, assim, poderia responder eletronicamente. Na maioria das questões, se fez necessário um questionamento usando o "Por quê?" Tanto para quando a resposta fosse "Sim" quanto para a resposta fosse "Não", sendo que todas as questões eram obrigatórias, com algumas respostas abertas.
- Após o envio do questionário aos colaboradores, foi estabelecido um prazo de retorno (10 dias), sem perguntar o nome do colaborador apenas a função dentro do projeto.

As questões foram adaptadas do estudo de caso do trabalho de (Anderle, 2015).

O quadro 2 apresenta as questões respondidas e as suas referências teóricas.

Quadro 2 – Questionário.

Questão:	Referência:
1 - Você utiliza BDD e de que forma?	Elaborada pelo autor com o intuito de saber se o colaborador utiliza o BDD em seu projeto.
2 - Você vê valor no uso do BDD no seu projeto? Sim / Não. Por quê?	Smart (2015) - o BDD ajuda as equipes a concentrar seus esforços na identificação, compreensão e na construção de aplicações que agreguem valor para o negócio, e certifica-se que estas aplicações são bem projetadas e bem implementadas.
3 - No seu projeto a US (user story) é escrita em formato de BDD? Se sim, tendo as US escritas no formato BDD, você acredita que isso trouxe um ganho de qualidade no desenvolvimento do software?	Rocha (2013) - Evita-se erros de compreensão e interpretação das histórias de usuário; North (2006) - BDD traz uma “linguagem única” para análise;
4 - Você acredita que com o uso do BDD todos da equipe tiveram um melhor entendimento do que era requisitado? Sim / Não. Por quê?	Rocha (2013) - O que antes eram requisitos funcionais e requisitos não funcionais são transformados em comportamentos funcionais do software e critérios de aceitação; North (2006) - Isso facilitará a comunicação e entendimento do time como um todo.
5 - Você acredita que com o uso do BDD melhorou a comunicação entre os membros da equipe? Sim / Não. Por quê?	Smart (2015) - Comunicação (encoraja analistas de negócios, desenvolvedores de software e testadores a colaborar mais de perto) Soares I (2011) - Comunicação entre equipes.
6 - Você acredita que com o uso do BDD o compartilhamento do conhecimento entre os membros da equipe (por ex: QA, DEV, PO, Scrum Master) melhorou? Sim / Não. Por quê?	Soares I (2011) - Compartilhamento de conhecimento. Rocha (2013) - Às reuniões de planejamento, revisão e diárias tornam-se mais eficazes.
7 - Quais as vantagens e desvantagens que você vê com o uso do BDD?	Smart (2015) - Releases mais rápidas; North (2006) - Os testes são focados no que realmente tem valor para o usuário;

8 - Relacionado ao esforço do desenvolvimento, você acredita que houve menos retrabalho com o uso do BDD? Sim / Não. Por quê?	Smart (2015) - Redução do desperdício; Redução de custo.
9 - De uma maneira geral, após a automação dos testes com o auxílio do BDD no projeto, houve uma melhoria na qualidade no processo de desenvolvimento de software?	Elaborado pelo autor - com o intuito de saber se houve uma melhoria;
10 - De uma maneira geral, após a introdução e a aplicação do BDD no projeto, você acredita que foi possível transformar os requisitos em um produto que agrega valor ao cliente?	North (2006) - BDD tenta ajudar os desenvolvedores no foco sobre o real valor a ser agregado ao cliente;
11 - Qual a sua função dentro da equipe? Desenvolvedor/QA	Elaborada pelo autor - com o intuito de saber qual a função do colaborador dentro do projeto.

Fonte: Elaborado pelo autor (2022).

5.1 Apresentação e análise dos resultados

A pesquisa foi realizada por meio de um questionário virtual, disponibilizado em um link (por meio de um formulário web elaborado a partir do Google Forms, aplicativo do Google que permite a criação, compartilhamento e disponibilização de formulário na web) e encaminhado aos membros da equipe do projeto, e respondido durante os meses de abril e maio de 2022. O universo de pesquisa compreendeu 7 respostas obtidas pelos membros 2 desenvolvedores e 5 QA. Este questionário foi a ferramenta metodológica que norteou a pesquisa, sendo que sua elaboração visou discutir questões relativas a percepção dos participantes quanto ao auxílio do BDD na automação de testes no projeto.

Logo abaixo, são exibidos os resultados obtidos com esta pesquisa. Obteve-se 100% dos questionários respondidos pelos membros do projeto da empresa X, totalizando 7 questionários respondidos.

1 - Você utiliza BDD e de que forma?

Dos 7 participantes, todos utilizam o BDD no processo de desenvolvimento, mas em alguns casos como os desenvolvedores e QA's onde participam apenas na parte de criação dos cenários e na automação dos mesmos, não utilizam o BDD por completo como relatam os DEV1 E DEV2, QA1 e QA2:

DEV1 - "Sim, utilizo apenas no processo de criação dos cenários de teste e automação".

DEV2 - "Eu não utilizo o BDD completo, o que a gente utiliza é mais aquelas etapas

iniciais do BDD [...]”.

QA1 - "Em parte, utilizado durante a especificação dos casos de teste".

QA2 - "Escrita de cenários/caso de teste".

QA3 - "onde participa apenas no desenvolvimento de cenários de testes a partir das US e os scripts são realizados na ferramenta cypress."

QA3: "Sim, de forma passiva desenvolvendo Cenários de testes a partir de US escritas em Gherkin. Porém os scripts automatizados não seguem o mesmo formato e nem usamos o Cucumber, na nossa ferramenta de automação, o usado é o Cypress".

Foi utilizado apenas na criação dos cenários que serão usados na automação de testes, também existe iniciativa para ser implementado também nos critérios de aceite das tarefas que serão desenvolvidas, como relata QA4 e QA5: QA4 - "[...] para primeiro definir Quais são os cenários de teste das features que vão ser feitas e a partir do cenário começar a escrita dos casos de teste com o cenário definido [...]".

QA5 - "Atualmente meu projeto utiliza o BDD para criação dos cenários que serão utilizados na automação do projeto. Existem iniciativas para que possamos utilizar o BDD nas fases iniciais, para que possamos utilizar o BDD para criar os critérios de aceite das tarefas que serão desenvolvidas".

Nesse contexto, os desenvolvedores não participam de forma completa no processo do BDD, diferente dos QA que a maioria utilizam todas as fases do projeto.

2 - Você vê valor no uso do BDD no seu projeto? Sim / Não. Por quê?

Todos os participantes acreditam que o BDD agrega valor ao projeto, pois ajuda na documentação das necessidades do cliente, assim como uma principal vantagem a rapidez no processo de liberação de uma nova entrega de funcionalidade ou uma resolução de algum bug. Outro ainda corroborou com o repasse de conhecimento para novos membros no projeto e também ajuda a entender como o sistema se comporta, como relatam os participantes:

QA1 - “O BDD ajuda a entender as necessidades dos clientes e permite visualizar todos os cenários que a solução desenvolvida será aplicada, permite conhecer o cotidiano do cliente”

QA2 - “Pois é uma metodologia que encoraja membros do time a discutir pontos como cenários de testes a serem considerados na aplicação.” DEV1: “Acho que fica mais fácil de entender toda a interação do sistema como um todo.”

QA3 - "Facilita o entendimento uma vez que a US tá escrita da forma que sugere essa prática, bem como agiliza a entrega da demanda pois não exige muita documentação."

QA4 - "Porque facilita normalmente quando eu quero repassar para alguém que não tem tanto entendimento do domínio do negócio do que se trata aquela feature que tá sendo feita ou até mesmo o que o produto faz [...]".

Ajuda no processo de lançamento de novas features e também traz um ganho de tempo mantendo o projeto de automação organizado, como relata o DEV2 e QA5:

DEV2 - "Ajuda porque a gente consegue identificar alguns cenários alguns fluxos que a gente não tinha pensado, já consegue pouco mais cedo tem essa vantagem e também acelera o fluxo de lançamento de versões".

QA5 - "Sim, acredito que com o uso do BDD seguimos uma estrutura que traga ganho de tempo e mantenha o projeto de automação organizado".

3 - No seu projeto a US (user story) é escrita em formato de BDD? Se sim, tendo as US escritas no formato BDD, você acredita que isso trouxe um ganho de qualidade no desenvolvimento do software?

Dos 7 participantes, 6 afirmaram que sim, foi destacado também que com essa escrita ajuda na comunicação e traz uma grande vantagem para o aumento de cobertura dos cenários, também foi relatado que através desse formato o resultado esperado é definida de forma simples e enxuta como relata os participantes, QA2, DEV1, DEV2 e QA5:

QA2 - "a escrita em user story facilita pelo time o entendimento da funcionalidade(critérios que devem ser atendidos)".

DEV1 - "Sim pois é possível estabelecer com mais facilidade todos os fluxos possível para uma funcionalidade, com isso tem-se uma grande vantagem para o aumento da cobertura na escrita de cenários de teste".

DEV2 - "A equipe segue um padrão para facilitar a comunicação entre os membros no projeto".

QA5 - "Porque conseguimos definir bem a ação e o resultado que esperamos para aquela funcionalidade de forma simples e enxuta" Já o QA3 respondeu que sim, entretanto precisaria de mais práticas para ajudar como TDD".

QA3 - "Em termos de qualidade não consigo dizer ao certo, por conta que precisamos (ainda hoje) de muita informação e de outras práticas associadas ao BDD. A ex. o TDD. A

aplicação desenvolvida nem sempre consegue atender os cenários felizes (e especificados na US).”

Outro respondeu que não por não participar diretamente da criação das US, como relata o QA1:

QA1 - "No meu caso, não chego a ter contato direto com as US e não sei se elas são escritas no formato BDD”.

4 - Você acredita que com o uso do BDD todos da equipe tiveram um melhor entendimento do que era requisitado? Sim / Não. Por quê?

Todos responderam que sim, destacando, principalmente, sobre como o BDD ajuda no entendimento da funcionalidade/requisitos que devem ser alcançados. Pela forma de escrita das US e dos cenários de testes, como afirmam os participantes QA1, QA2, DEV1 e QA3.

QA1 - "É possível ter uma noção de como a aplicação será utilizado no cenário real".

QA2 - "Com o BDD definimos alguns, um melhor entendimento baseado na escrita de casos de teste que auxiliassem no entendimento da funcionalidade, um complemento as user story."

DEV1 - "Por conta do detalhamento das "user stories" e critérios de aceite fica mais fácil estabelecer uma fronteira para os requisitos que devem ser desenvolvidos".

QA3 - "Todos entendem, o único problema é que no contexto do projeto, nunca sabia o que iria impactar, se tivesse alguma alteração não saberia se houve essa alteração".

5 - Você acredita que com o uso do BDD melhorou a comunicação entre os membros da equipe? Sim / Não. Por quê?

Todos os participantes afirmaram que sim, destacando que a comunicação melhorou por que os membros são incentivados a participar da construção dos cenários e também por gerar uma documentação viva do projeto através das especificações durante as sprints, como afirmam os participantes QA1, QA2, DEV1, QA3, QA4 e DEV2:

QA1 - "Todos ficam cientes do que será implementado, assim ficam menos dúvidas, e em caso de dúvidas, é possível recorrer a documentação de especificação".

QA2 - "É uma comunicação que fica bem definida desde o início da sprint com a planning, e no ciclo de desenvolvimento durante a sprint".

DEV1 - "Acredito que por ficar meio que como uma documentação dos próprios

requisitos, facilita a comunicação das partes envolvidas".

QA3 - "Pelo motivo de que fica fácil de assimilar o que está faltando no projeto no momento do desenvolvimento. Já aconteceu comigo quando eu vi que estava faltando um requisito no projeto quando estava testando uma tarefa".

QA4 - "porque o cenário de teste no caso dos casos é para ser mais uma definição final do que a gente acabou definindo para o produto ou não [...]".

DEV2 - "Depois que começou a adotar aqui no projeto, a gente tem um bom entendimento. Todo mundo fica à frente do projeto".

Para o QA5 os benefícios do BDD são potencializados quando todos do time participam na construção dos cenários juntos.

QA5 - "Acredito que meu time posso potencializar os benefícios do uso do BDD ao trazer ele para a construção de cenários juntos. Atualmente utilizamos mais na fase de construção dos cenários do teste (quando a funcionalidade já foi desenvolvida, o que não é o ideal)".

6 - Você acredita que com o uso do BDD o compartilhamento do conhecimento entre os membros da equipe (por ex: QA, DEV, PO, Scrum Master) melhorou? Sim / Não. Por quê?

Todos os participantes, afirmam que melhorou pois a escrita por não ter uma linguagem técnica na escrita dos cenários, pelos membros principalmente DEV e QA estarem presentes nas discussões e documentando essas features destacando também as descrições das users stories que facilitaram o entendimento da funcionalidade como afirmam os participantes QA1, QA2, QA3, QA4, QA5, DEV1 e DEV2:

QA1 - "Sim, todos podem dar suas opiniões a respeito dos cenários que a solução será implementada, permitindo a criação de casos de teste mais específicos, evitando trabalho desnecessário".

QA2 - "O QA em algumas situações conseguem replicar técnicas ou cenários que devem ser testados na construção de alguma funcionalidade".

DEV1 - "Acho que principalmente nos que estão mais ligados com o desenvolvimento, eu colocaria DEV + QA".

DEV2 - "porque quando eu entrei não tinha uma comunicação eficiente entre o dev e o QA depois da adoção ajudou bastante".

QA4 - "[...] Tem a questão de você ir lá e fazer uma explicação da feature mas tem uma questão de você simplesmente pegar alguns cenários de teste de um cenário que já estão

agrupados uma ferramenta tipo Cucumber passar para ele para ele ver mais ou menos quais são os cenários que tem e só com aqueles cenários ele já tem a visão como um todo do comportamento correto nela que ela fez o é tipo ele não fica muito amarrado ao conhecimento [...]"

QA5 - "Acredito que se o BDD for utilizado como meio colaborativo sim, principalmente por ter diversos papéis e expertises onde podemos encontrar muitas visões e ideias para construção da funcionalidade".

QA3 - "Com certeza. Mesmo eu acabando de sair da faculdade consegui entender muito bem os requisitos, principalmente pelas descrições das user history".

7 - Quais as vantagens e desvantagens que você vê com o uso do BDD?

Das vantagens e desvantagens, destacam-se, respectivamente:

1. Entregas mais rápidas através das validações das funcionalidades do cliente, principalmente por conhecer o cotidiano do cliente. Por possuir uma linguagem formal fica mais fácil de validar as funcionalidades e possuir feedbacks rápidos, como relata os participantes QA1, DEV2 e QA5:

QA1 - "a vantagem é a possibilidade de conhecer o cotidiano do usuário".

DEV2 - "Lançar features mais rápido do que antigamente. Desvantagem é na parte de planejamento nas partes iniciais do que desenvolvendo".

QA5 - "organização, colaboração na criação de validações das funcionalidades, feedbacks rápidos".

Outro ponto levantado foi a vantagem de comunicação dos membros da equipe em relação ao projeto, como relata o QA4:

QA4 - "entendimento de negócio ajuda bastante e é todo time que tem cenário de teste escrito em BDD ajuda a entender uma feature quando você chega não time novo que tem caso de teste ajuda bastante no entendimento quando não tenho dificuldade um pouco você tem que estar se comunicando com pessoas".

Aumento na qualidade do produto final como afirma o QA2:

QA2 - "[...] é um aumento da qualidade produto final entregue [...]".

2. Inserir esse método em um time com o processo já engessado pode ter uma resistência, como relata o DEV1: Um ponto que talvez seja uma desvantagem seria estabelecer esse novo pensamento diante de um time junto com a implementação, seria meio que organizar

os papéis.

Outro ponto levantado foi a dificuldade na manutenção dos testes e inserção de dados desnecessários da equipe como banco de dados, como relataram os participantes QA3 e QA2:

QA3 - "vejo algumas dificuldades em relação aos cenários por conta que ainda são inseridas informações de terceiros como banco de dados".

A maioria dos processos são realizados apenas pelos QA's como afirma o QA2:

QA2 - "[...] a desvantagem é que muitos dos processos que devem ser executados com BDD acabam ficando de responsabilidade apenas do QA".

8 - Relacionado ao esforço do desenvolvimento, você acredita que houve menos retrabalho com o uso do BDD? Sim / Não. Por quê?

Dos 7 participantes, apenas um QA que não concordou, pois o BDD não foi aplicado completamente às suas fases no projeto, acreditando que precisaria de outras técnicas como o TDD como relata o QA3:

QA3 - "Ainda existe retrabalho por conta que o BDD ainda não foi implementado totalmente no projeto precisando de outras metodologias de auxílio como o TDD".

O QA5 destaca a colaboração entre os membros do time que ajuda na definição das funcionalidades do cliente, facilitando no entendimento de todos os envolvidos.

QA5 - "sim, principalmente por ter usado ele em COLABORAÇÃO com o time. E dessa forma conseguimos pensar/questionar e detalhar nossas funcionalidades chegando em um nível de entendimento mais simples para todos".

Principalmente com adição de novas validações fica mais fácil de acompanhar, como relata o DEV1:

DEV1 - "Principalmente para a parte adição de novas validações, porque se torna mais fácil acompanhar o que só foi validado e simplesmente incorporar algo novo".

9 - De uma maneira geral, após a introdução e a aplicação do BDD no projeto, você acredita que foi possível transformar os requisitos em um produto que agrega valor ao cliente?

Todos os participantes concordam que a afirmação, pois o refinamento dos requisitos já existentes com a escrita das US e com o fluxo de validações gerando valor para o cliente como afirmam os participantes QA2, QA3 e DEV1:

QA2 - "Sim. Foi possível refinar os requisitos já existentes, e então, ver novas

oportunidades que geram valor para o cliente".

QA3 - "Com certeza. O cliente especificou no sow e transforma em US e é entregue pra ele na maioria dos casos e quando está estável eles ficam satisfeitos".

DEV1 - "Sim, simplesmente por trazer essa estrutura e fluxo de validações".

Após a introdução teve uma queda nos problemas encontrados em ambiente de produção, como relata o QA4:

QA4 - "[...] gente tava conhecendo o projeto então foi basicamente duas sprints que o pessoal fazia as features sem ter caso de teste para as features que a gente tava elaborando e tinha muito problema de bug que ia para produção que só era identificada na produção, o usuário que tava fazendo o uso da aplicação era que encontrava o problema, e acaba relação com o pessoal, pessoal tinha que corrigir e a partir do momento que a gente começou assim já ajustar a questão dos cenários de teste a nossa inicial dos movimentos que é uma coisa que foi teve um próximo tempo assim de adaptação a gente percebeu uma diminuída bastante na questão de problemas que chegam em ambiente de produção".

Para o DEV2 após a introdução do BDD, começou a ir bem fundo em relação de como o cliente teria uma melhor experiência.

DEV2 - "eu acredito sim, a gente tenta sempre pensar como o cliente vai usar, quando a gente começou a aplicar fomos bem a fundo de como ele ia usar como ele teria uma melhor experiência".

10 - De uma maneira geral, após a automação dos testes com o auxílio do BDD no projeto, houve uma melhoria na qualidade no processo de desenvolvimento de software?

Todos os participantes, afirmam ter uma melhoria na qualidade com a automação através do auxílio do BDD. Pois há uma cobertura dos testes principalmente com os cenários de aceitação levantados e foi de grande auxílio na construção da automação desses cenários, como relata o QA2 e QA5:

QA2 - "Sim, a definição de cenários de teste utilizando BDD foi de grande auxílio na construção de uma automação de testes".

QA5 - "Sim, principalmente por usarmos os cenários levantados para a construção da automação e assim construir uma cobertura dos nossos testes aceitável".

Melhoria no código no processo de desenvolvimento e uma definição junto com uma implantação a adoção do BDD foi produtiva, como relata o DEV1 e DEV2:

DEV1 - "Sim, dado que tenhamos uma definição completa junto a uma implantação acredito que a adoção foi muito produtiva para o desenvolvimento".

DEV2 - "Sim, ficou bem menos refatoração no desenvolvimento, melhoria no código como código mais limpo. Menos tarefas de bug sendo retornada pelo time de qualidade".

Impactou na melhora da qualidade do software e na redução do esforço gerado pelos testes que antes eram realizados apenas de forma manual pelo analista de teste como relata o Q4:

QA4 - "Sim. Quando não tinha os casos você tinha que fazer o teste manualmente, e o teste manualmente necessita de esforço uma pessoa e nem sempre o pessoal tem tempo para esse esforço, com automação sim tem uma diminuída boa, porque você consegue executar com mais facilidade mais rapidez os cenários que já foram definidos, já a automação é baseada nos cenários de teste. Então sim, houve uma melhora na qualidade".

Com os testes de ponta a ponta sendo construído para a validação dos cenários de testes, tendo uma visibilidade muito rápida caso apareça alguma inconsistência no teste garantindo que a feature vai estar sempre funcionando no processo de liberação de uma versão, como relata o QA3:

QA3 - "Teve uma melhora principalmente nos testes de E2E que conseguimos garantir que os cenários estão sendo acordados através da automação, caso apareça algum problema vai ter uma visibilidade muito rápida. A vantagem da automação é a garantia de sempre que a feature vai estar funcionando, quando for executado o processo de regressão".

Para o QA1 impactou na garantia da qualidade do processo através dos cenários automatizados executados frequentemente validando os cenários de testes especificados pelo time do projeto.

QA1 - "Sim, pois os cenários automatizados são executados frequentemente, garantindo que aquela solução continue se comportando conforme o esperado, consequentemente, garantindo a qualidade do processo".

5.2 Aplicação BDD e automatização

Com o intuito de agilizar e reduzir o processo de execução de testes, foi utilizada a técnica de automatização do projeto através dos cenários de testes, principalmente quando são necessários testes de regressão, assim simplificando a manutenção do software e entendimento dos membros com a documentação viva que o BDD gera.

Para automatizar os cenários de testes foi utilizada as ferramentas do framework

cypress através dos testes de ponta a ponta. Nesse exemplo será apresentado um dos cenários que foram especificados no projeto, um dos cenários escolhido foi o resumo financeiro de uma entrega de pedidos. A figura 5 ilustra o arquivo que contém o cenário levantado para ser automatizado, a figura 6 com a parte do arquivo .js com seu respectivo mapeamento e seus métodos implementados com o framework Cypress, que por sua vez irá validar através dos clicks e asserções os passos descritos no cenário na tela do aplicativo.

Figura 5 – Cenário de teste

Narrativa: Como um motorista,
Desejo realizar a entrega de pedidos,
Então conseguirei validar o total dos pedidos.

Cenário: 01 - Verificar se os valores mostrados no resumo financeiro estão corretos
Dado que o motorista acessa a página inicial do sistema e validando o servidor
E efetua o login no sistema com (driver) e senha (driver)
E insere o equipamento
E o motorista acessa as stops
Quando o usuário clica realiza o arrive na stop
E o motorista clica no Botão Sair
E o motorista seleciona a stop atendida
E o motorista clica no card de resumo financeiro
Então deve exibir o montante correto de 'US\$ 500.00'

Fonte: elaborado pelo autor (2022).

Figura 6 – Código Cypress

```

1
2 describe('Verification of the values displayed in the financial summary', function () {
3
4     it('Verification of the values', function () {
5         const driver = this.route.driverAssignments[0].driver;
6         const equipment = this.route.equipmentAssignments[0].equipment;
7
8         cy.visit('/');
9         cy.setAndSaveServer();
10        cy.skipWelcomeTour();
11        cy.login(driver);
12        cy.loadRoute(equipment);
13        cy.startAndLeaveDC();
14
15        cy.arriveStop(this.route.stops[0].location);
16        cy.validateTotalOrder(this.route.stops[0].orders[0].number, 'US$ 500.00');
17    });
18 });
19

```

Fonte: elaborado pelo autor (2022).

6 CONSIDERAÇÕES FINAIS

Esta pesquisa possibilitou uma análise aprofundada sobre o desenvolvimento de software com a aplicação da metodologia *Behavior Driven Development* (BDD) associada à automação de testes. O estudo buscou compreender, de forma prática, como essas abordagens influenciam o entendimento das funcionalidades, a elaboração de cenários de teste e a redução do esforço em testes manuais.

Os resultados indicaram que a adoção do BDD proporcionou melhorias expressivas no processo de desenvolvimento e teste. A utilização de *User Stories* (US) em formato estruturado contribuiu para alinhar a compreensão entre todos os membros da equipe e aumentar a precisão dos critérios de aceitação. Observou-se, no entanto, que a atribuição de responsabilidades ainda se concentrou excessivamente nos analistas de QA, limitando a participação integral dos desenvolvedores na construção dos cenários de teste.

Principais constatações:

- Todos os participantes relataram redução de retrabalho no desenvolvimento com a utilização do BDD.
- A comunicação e o compartilhamento de conhecimento entre os membros da equipe melhoraram de forma unânime.
- Houve reconhecimento geral do valor do BDD e das vantagens da sua aplicação.
- A automação dos cenários de teste contribuiu diretamente para o aumento da qualidade do software.
- Os ganhos de qualidade foram percebidos tanto na prevenção de defeitos quanto na agilidade das entregas.

O estudo de caso confirmou a viabilidade e a eficácia do BDD como prática integrada ao ciclo de desenvolvimento, evidenciando benefícios como maior cobertura de testes, alinhamento entre áreas técnicas e de negócio e ganho de eficiência nas entregas. Ainda assim, foram observadas resistências pontuais, especialmente em cenários com prazos curtos para liberação de funcionalidades, o que reforça a necessidade de integração contínua dessas práticas no planejamento do projeto.

Conclui-se que o BDD, aliado à automação de testes, constitui uma abordagem estratégica para o desenvolvimento de software, capaz de gerar benefícios tangíveis como redução de retrabalho, melhoria na comunicação e aumento da qualidade do produto final. A experiência relatada reforça a importância de manter um processo colaborativo e automatizado

de forma contínua, garantindo que as soluções entregues estejam alinhadas às expectativas e necessidades dos clientes.

REFERÊNCIAS

- ANDERLE, A. Introdução de bdd (behavior driven development) como melhoria de processo no desenvolvimento ágil de software. Universidade do Vale do Rio dos Sinos, 2015.
- BAHAWERES, R. B.; OKTAVIANI, E.; WARDHANI, L. K.; HERMADI, I.; SUROSO, A.; SOLIHIN, I. P.; ARKEMAN, Y. Behavior-driven development (bdd) cucumber katalon for automation gui testing case cura and swag labs. In: IEEE. **2020 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)**. [S.l.], 2020. p. 87–92.
- BARTIÉ, A. **Garantia da qualidade de software**. [S.l.]: Gulf Professional Publishing, 2002.
- BERNARDO, C.; SILVA, A.; SANTOS, P. Testes automatizados: uma abordagem prática. **Revista de Engenharia de Software**, Editora Universitária, v. 12, n. 3, p. 45–62, 2008.
- BOEHM, B. W. Software engineering. **IEEE Transactions on computers**, IEEE, v. 25, n. 12, p. 1226–1241, 1976.
- CHIAVEGATTO, R. B.; SILVA, L.; ANDRÉIA, V.; MALVEZZI, W. Desenvolvimento orientado a comportamento com testes automatizados utilizando jbehave e selenium. In: **Anais do Encontro Regional de Computação e Sistemas de Informação Manaus**. [S.l.: s.n.], 2013.
- COHN, M. **User stories applied: For agile software development**. [S.l.]: Addison-Wesley Professional, 2004.
- DEES, I.; WYNNE, M.; HELLESoy, A. **Cucumber Recipes: Automate Anything with BDD Tools and Techniques**. [S.l.]: Pragmatic Bookshelf, 2013.
- DELAMARO, M.; JINO, M.; MALDONADO, J. **Introdução ao teste de software**. [S.l.]: Elsevier Brasil, 2013.
- FONSECA, R. C. V. d. **Metodologia da pesquisa científica**. Fortaleza: UFC, 2012.
- FOWLER, M. The practical test pyramid. **martinfowler.com**. [Online]. Available: <https://martinfowler.com/articles/practical-test-pyramid.html>, 2018.
- GIL, A. C. **Métodos e técnicas de pesquisa social**. 6. ed. São Paulo: Atlas, 2008.
- MARCONI, M. d. A.; LAKATOS, E. M. **Metodologia científica**. 5. ed. São Paulo: Atlas, 2003.
- MWAURA, J. Cypress: End-to-end testing framework for modern web applications. **Journal of Software Testing and Quality Assurance**, Software Testing Institute, v. 15, n. 3, p. 18–25, 2021.
- NORTH, D. **Introducing bdd** dan north & associates. 2006.
- PAULA, W. H. d. **Qualidade de software e desenvolvimento dirigido por software-um estudo de caso**. 2019.
- PEIXOTO, J. Importância dos testes automatizados na qualidade de software. **Revista de Tecnologia da Informação**, Editora Científica, v. 8, n. 2, p. 1–15, 2018.
- PRESSMAN, R.; MAXIM, B. **Engenharia de Software-8ª Edição**. [S.l.]: McGraw Hill Brasil, 2016.

RIOS, E. **Teste de Software: Um Guia Prático**. Rio de Janeiro: Ciência Moderna, 2006.

ROCHA, A. C. **A Pirâmide de teste e os Testes end-to-end**. 2021. Disponível em: <<https://medium.com/gtsw/a-pir%C3%A2mide-de-teste-e-os-testes-end-to-end-38f77ad3d137>>. Acesso em: 4 jun. 2022.

SELENIUM. **Tipos de testes**. 2021. Disponível em: <https://www.selenium.dev/pt-br/documentation/guidelines/types_of_testing/>. Acesso em: 8 ago. 2021.

SILVA, G. **O que é BDD?** 2022. Disponível em: <<https://coodesh.com/blog/dicionario/o-que-e-bdd/>>. Acesso em: 4 jun. 2022.

SMART, J. **BDD in Action: Behavior-driven development for the whole software lifecycle**. [S.l.]: Simon and Schuster, 2014.

SOMMERVILLE, I. **Engenharia de Software 9ª Edição**. [S.l.]: Pearson Universidades, 2011.

WARMUP. A importância da qualidade de software na vida das pessoas. 2021. Disponível em: <<https://warmupweb.com.br/2021/01/08/a-importancia-da-qualidade-de-software-na-vida-das-pessoas/>>.

WYNNE, M.; HELLESOY, A.; TOOKE, S. **The cucumber book: behaviour-driven development for testers and developers**. [S.l.]: Pragmatic Bookshelf, 2017.

APÊNDICE A – INSTRUMENTO DE COLETA DE DADOS

Você utiliza BDD e de que forma? *

Sua resposta

Você vê valor no uso do BDD no seu projeto? Sim / Não. Por quê? *

☐ Sim

☐ Não

Por quê? *

Sua resposta

No seu projeto a US (user story) é escrita em formato de BDD? Se sim, tendo as US escritas no formato BDD, você acredita que isso trouxe um ganho de qualidade no desenvolvimento do software? *

☐ Sim

☐ Não

Por quê? *

Sua resposta

Você acredita que com o uso do BDD todos da equipe tiveram um melhor entendimento do que era requisitado? Sim / Não. Por quê? *

☐ Sim

☐ Não

Por quê? *

Sua resposta

Você acredita que com o uso do BDD melhorou a comunicação entre os membros da equipe? Sim / Não. Por quê? *

☐ Sim

☐ Não

Por quê? *

Sua resposta

Você acredita que com o uso do BDD o compartilhamento do conhecimento entre os membros da equipe (por ex: QA, DEV, PO, Scrum Master) melhorou? *
Sim / Não. Por quê?

☐ Sim

☐ Não

Por quê? *

Sua resposta

Quais as vantagens e desvantagens que você vê com o uso do BDD? *

Sua resposta

Relacionado ao esforço do desenvolvimento, você acredita que houve menos retrabalho com o uso do BDD? Sim / Não. Por quê? *

☐ Sim

☐ Não

Por quê? *

Sua resposta

De uma maneira geral, após a introdução e a aplicação do BDD no projeto, você *
acredita que foi possível transformar os requisitos em um produto que agrega
valor ao cliente?

Sua resposta

De uma maneira geral, após a automação dos testes com o auxílio do BDD no *
projeto, houve uma melhoria na qualidade no processo de desenvolvimento de
software?

Sua resposta

Qual a sua função dentro da equipe? Desenvolvedor/QA *

Sua resposta

Enviar

[Limpar formulário](#)

Nunca envie senhas pelo Formulários Google.

Este conteúdo não foi criado nem aprovado pelo Google. [Denunciar abuso](#) - [Termos de Serviço](#) - [Política de Privacidade](#)

Google Formulários

APÊNDICE B – RESPOSTAS DOS COLABORADORES

1 - Você utiliza BDD e de que forma?

QA1: Em parte, utilizo durante a especificação dos cenários/casos de teste

QA2: Escrita de cenários/caso de teste

DEV1: Sim, utilizo apenas no processo de criação dos cenários de teste e automação

QA3: Sim, de forma passiva desenvolvendo Cenários de testes a partir de US escritas em Gherkin. Porém os scripts automatizados não seguem o mesmo formato e nem usamos o Cucumber, na nossa ferramenta de automação.

QA4: sim eu utilizo aí a forma que ele é utilizado ele é utilizado mais quando a gente está em início de Sprint o que no caso sempre que a gente faz um planejamento de uma Sprint uma planning e a gente tem um intervalo entre o início da Sprint em si e a etapa de planejamento, que é uma etapa que a gente chama de refinamento que aproveita como QA aproveito essa etapa de refinamento para primeiro definir Quais são os cenários de teste das features que vão ser feito e a partir do cenário começar a escrita dos casos de teste com o cenário definido isso numa planilha ou até mesmo em um TXT qualquer para depois passar para uma ferramenta de gestão né no caso é o xray, a gente utiliza sim o BDD

DEV2: Eu não utilizo o BDD completo o que a gente utiliza é mais aquelas etapas iniciais do BDD, a parte que todo mundo conversa e troca de ideias, depois o time de qualidade faz os cenários de testes e os desenvolvedores automatizam.

QA5: Atualmente meu projeto utiliza o BDD para criação dos cenários que serão utilizados na automação do projeto. Existem iniciativas para que possamos utilizar o BDD nas fases iniciais, para que possamos utilizar o BDD para criar os critérios de aceite das tarefas que serão desenvolvidas.

2 - Você vê valor no uso do BDD no seu projeto? Sim / Não. Por quê?

QA1: Sim. O BDD ajuda a entender as necessidades dos clientes e permite visualizar todos os cenários que a solução desenvolvida será aplicada, permite conhecer o cotidiano do cliente

QA2: Sim. Pois é uma metodologia que encoraja membros do time a discutir pontos como cenários de testes a serem considerados na aplicação.

DEV1: Sim. Acho que fica mais fácil de entender o toda a interação do sistema como um todo.

QA3: Sim. Facilita o entendimento uma vez que a US tá escrita da forma que sugere essa prática, bem como agiliza a entrega da demanda pois não exige muita documentação.

QA4: Sim. Porque facilita normalmente quando eu quero repassar para alguém que não tem tanto entendimento do domínio do negócio do que se trata aquela feature que tá sendo feita ou até mesmo o que o produto faz, quando eu modelo o cenário e definir os cenários de teste é mais fácil de mostrar para pessoa. O que a funcionalidade que a gente está fazendo faz em si e quais são os fluxos de exceção é mais fácil de explicar.

DEV2: Sim. Ajuda porque a gente consegue identificar alguns cenários alguns fluxos que a gente não tinha pensado, já consegue pouco mais cedo tem essa vantagem e também acelera o fluxo de lançamento de versões.

QA5: Sim, acredito que com o uso do BDD seguimos uma estrutura que traga ganho de tempo e mantenha o projeto de automação organizado.

3 - No seu projeto a US (user story) é escrita em formato de BDD? Se sim, tendo as US escritas no formato BDD, você acredita que isso trouxe um ganho de qualidade no desenvolvimento do software?

QA1: Não. No meu caso, não chego a ter contato direto com as US e não sei se elas são escritas no formato BDD

QA2: Sim. A escrita em user story facilita pelo time o entendimento da funcionalidade(critérios que devem ser atendidos)

DEV1: Sim pois é possível estabelecer com mais facilidade todos os fluxos possível para uma funcionalidade, com isso tem-se uma grande vantagem para o aumento da cobertura na escrita de cenários de teste.

QA3: Sim. Em termos de qualidade não consigo dizer ao certo, por conta que precisamos (ainda hoje) de muita informação e de outras práticas associadas ao BDD. A ex. o TDD. A aplicação desenvolvida nem sempre consegue atender os cenários felizes (e especificados na US).

QA4: o modelo que a gente utiliza é um modelo mas comportamental seguindo já o próprio padrão de escrita do aqueles arquivos gherkin que sempre que você faz uma escrita por exemplo ele já demarca palavras reservadas como (dado... quando... então) para quem foi e saber que você está utilizando o BDD na escrita

DEV2: Sim. A equipe segue um padrão para facilitar a comunicação entre os membros no projeto.

QA5: Não. Porque conseguimos definir bem a ação e o resultado que esperamos para aquela funcionalidade de forma simples e enxuta

4 - Você acredita que com o uso do BDD todos da equipe tiveram um melhor entendimento do que era requisitado? Sim / Não. Por quê?

QA1: Sim. É possível ter uma noção de como a aplicação será utilizado no cenário real

QA2: Com o BDD definimos alguns, um melhor entendimento baseado na escrita de casos de teste que auxiliassem no entendimento da funcionalidade, um complemento as user story

DEV1: Sim. Por conta do detalhamento das "user stories" e critérios de aceite fica mais fácil estabelecer uma fronteira para os requisitos que devem ser desenvolvidos. DEV

QA3: Sim. Todos entendem, o único problema é que no contexto do projeto, nunca sabia o que iria impactar, se tivesse alguma alteração não saberia se houve essa alteração.

QA4: Sim. Sempre que tem uma feature de novo para ser feita a PO ela descreve pelo menos aqui na minha squad ela descreve o que a gente pensou para isso e ela coloca o comportamento mínimo comportamento mínimo aceitável para para feature. Se for uma feature de enviar e-mail ela espera que quando você inserir um e-mail e clicar em enviar esse e-mail seja enviado, só que para o dev fica muito limitado sabe ele tem a visão somente do que a PO escreve lá na história e como a escrita dos cenários na verdade a definição do cenário e a escrita dos cenários de teste com BDD aí eu vejo na Squad como um todo, desenvolvedores o pessoal que trabalha até na parte de teste tem uma visão maior do que está sendo feito sabe até na questão de se vai ter esse comportamento de exceção por exemplo eu já vou ter muito trabalho maior para se tratar isso, são coisas que talvez se fossem ser observadas ao longo do desenvolvimento quando você tem a definição esses cenários de teste antes com essa escrita mais comportamental ajuda no entendimento assim na visão de problemas.

DEV2: Sim. porque já no começo a gente senta todo mundo, entendeu e fala da sua opinião sobre o fluxo que pode acontecer.

QA5: Acredito que sim, já que quando criamos com a user story com estrutura COMO / EU QUERO / PARA construímos/geramos algumas dúvidas que ajudam na construção e geração de cenários de teste

5 - Você acredita que com o uso do BDD melhorou a comunicação entre os membros da equipe? Sim / Não. Por quê?

QA1: Sim. Todos ficam cientes do que será implementado, assim ficam menos dúvidas, e em caso de dúvidas, é possível recorrer a documentação de especificação

QA2: Sim. É uma comunicação que fica bem definida desde o início da sprint com a planning, e no ciclo de desenvolvimento durante a sprint.

DEV1: Acredito que por ficar meio que como uma documentação dos próprios requisitos, facilita a comunicação das partes envolvidas.

QA3: Sim. Pelo motivo de que fica fácil de assimilar o que está faltando no projeto no momento do desenvolvimento. Já aconteceu comigo quando eu vi que estava faltando um requisito no projeto quando estava testando uma tarefa.

QA4: Sim. porque o cenário de teste no caso dos casos é para ser mais uma definição final do que a gente acabou definindo para o produto ou não, só que tem um problema é um problema que acontece aqui recorrente que é de tipo alguém fez solicitou alguma modificação na feature ou alguém definiu alguma coisa, só que por exemplo essa conversa ficou muito reservada e ficou reservada entre as pessoas e não foi por exemplo documentada com comentários simples na história principal e os casos de teste que tinham sido já criados não foram atualizados, tem muito desse problema é que entre desenvolvedor e QA mas tem muito do problema de quando é alguém externo do time que principalmente Superintendência, que a gente pensa em fazer alguma coisa aí a Superintendência vai e não vamos fazer isso daqui assim agora. Mais normalmente eles não têm a visão que tem esse caso de teste, eles pegam só a história principal.

DEV2: Sim. Depois que começou a adotar aqui no projeto, a gente tem um bom entendimento. Todo mundo fica à frente do projeto.

QA5: Sim. Acredito que meu time posso potencializar os benefícios do uso do BDD ao trazer ele para a construção de cenários juntos. Atualmente utilizamos mais na fase de construção dos cenários do teste (quando a funcionalidade já foi desenvolvida, o que não é o ideal)

6 - Você acredita que com o uso do BDD o compartilhamento do conhecimento entre os membros da equipe (por ex: QA, DEV, PO, Scrum Master) melhorou? Sim / Não. Por quê?

QA1: Sim, todos podem dar suas opiniões a respeito dos cenários que a solução será implementada, permitindo a criação de casos de teste mais específicos, evitando trabalho desnecessário

QA2: Sim. O QA em algumas situações conseguem replicar técnicas ou cenários que devem ser testados na construção de alguma funcionalidade.

DEV1: Sim. Acho que principalmente nos que estão mais ligados com o desenvolvimento, eu colocaria DEV + QA.

QA3: Sim. Com certeza. Mesmo eu acabando de sair da faculdade consegui entender muito bem os requisitos, principalmente pelas descrições das user history

QA4: Sim. porque sempre que entra um novo membro do time e ele vai ter que tocar uma feature ou fazer manutenção, fazer um bug em uma feature porque ele não conhece. Tem a questão de você ir lá e fazer uma explicação da feature mas tem uma questão de você simplesmente pegar alguns cenários de teste de um cenário que já estão agrupados uma ferramenta tipo Cucumber passar para ele para ele ver mais ou menos quais são os cenários que tem e só com aquele cenários ele já tem a visão como um todo do comportamento correto nela que ela fez o é tipo ele não fica muito amarrado ao conhecimento que as pessoas têm uma coisa que já está documentada.

DEV2: Sim. Porque quando eu entrei não tinha uma comunicação eficiente entre o dev e o QA depois da adoção ajudou bastante.

QA5: Sim. Acredito que se o BDD for utilizado como meio colaborativo sim, principalmente por ter diversos papéis e expertises onde podemos encontrar muitas visões e ideias para construção da funcionalidade.

7 - Quais as vantagens e desvantagens que você vê com o uso do BDD?

QA1: Não consigo enxergar desvantagem, a vantagem é a possibilidade de conhecer o cotidiano do usuário

QA2: Principais vantagens é um aumento da qualidade produto final entregue, a desvantagem é que muitos dos processos que devem ser executados com BDD acabam ficando de responsabilidade apenas do QA.

DEV1: Considero vantagem a forma mais estruturada com do desenvolvimento usando BDD, justamente por trazer um maior detalhamento dos fluxos/cenários para um determinado requisito. Um ponto que talvez seja uma desvantagem seria estabelecer esse novo pensamento diante de um time junto com a implementação, seria meio que organizar os papéis.

QA3: vejo algumas dificuldades em relação aos cenários por conta que ainda são inseridas informações de terceiros como banco de dados.

QA4: as principais entrar no ponto de entendimento de negócio ajuda bastante e é todo time que tem cenário de teste escrito em BDD ajuda a entender uma feature quando você chega não time novo que tem caso de teste ajuda bastante no entendimento quando não tenho

dificulta um pouco você tem que estar se comunicando com pessoas ajuda no próprio teste da feature que uma vez que você definir o cenário escreve os cenários de teste utilizando o BDD uma forma numa linguagem mais formal é mais fácil de você ter definido de algo que você vai testar, quando dev concluiu o desenvolvimento ajuda bastante até já trazendo da pergunta anterior quando você quer passar isso para alguém né ou quer transmitir o que foi feito naquela feature para alguém nos casos de testes já basicamente explica o que foi feito o que tem que ser feito ponto negativo, era para ser uma coisa mais colaborativa tipo se teve uma mudança em uma feature e o cenário de teste já tava feito Nossa sinceramente o QA somente tem que ir lá dar manutenção, na manutenção que eu digo ele é correção, no cenários de teste que foi alterado no cenário, o Dev a partir do momento que ele faz alteração na funcionalidade ele poderia ter a ação de lá e corrigir o problema que eu vejo muita gente acha que a vamos escrever os cenários de teste aqui com o BDD pronto então vamos lá e utilizar cucumber no projeto.

DEV2: Lançar features mais rápido do que antigamente. Desvantagem é na parte de planejamento nas partes iniciais do que desenvolvendo.

QA5: Desvantagem: Acredito que existe um trabalho a mais quando usamos o BDD para construção dos testes automatizados. Já que a manutenção pode ser um pouco mais demorada para projetos que sigam esse padrão Vantagem: organização, colaboração na criação de validações das funcionalidades, feedbacks rápidos

8 - Relacionado ao esforço do desenvolvimento, você acredita que houve menos retrabalho com o uso do BDD? Sim / Não. Por quê?

QA1: Sim. No meu caso, quando temos os casos de testes definidos, fica mais fácil de realizar os testes, evitando perder tempo com cenários desnecessários QA2: Sim. Uma vez que temos especificado o que testar, os devs têm ciência de quais pontos eles devem ter atenção durante o desenvolvimento, isso garantindo uma menor incidência de bugs, e consequentemente menos retrabalho.

DEV1: Sim. Principalmente para a parte adição de novas validações, porque se torna mais fácil acompanhar o que só foi validado e simplesmente incorporar algo novo.

QA3: Não. Ainda existe retrabalho por conta que o BDD ainda não foi implementado totalmente no projeto.

QA4: Sim. Houve na questão de por exemplo a feature ele foi desenvolvida e eu só vou definir os cenário da funcionalidade e os casos de teste, depois desenvolvido no que o dev não tem a visão assim de fluxo de exceção, pode apresentar muito problema que durante

esse desenvolvimento ele não vai corrigindo e não vai ter atenção, mas tem sempre da questão de teste encontra problema e volta para ele e reteste pode acabar tendo algum outro problema fica muito nesse bate e volta que com a questão deles usar o BDD evita bastante essa parte de retrabalho, tem alguns problemas bem simples que dá para ser evitado principalmente quando você tem uma nova integração com uma API externa.

DEV2: Sim. Sim, bastante, porque a gente não precisa mais ficar voltando no código em um fluxo onde tinha pensado que tem que mudar essas coisas não acontecem, não acontece com muito mais frequência

QA5: Sim. Sim, principalmente por ter usado ele em COLABORAÇÃO com o time. E dessa forma conseguimos pensar/questionar e detalhar nossas funcionalidades chegando em um nível de entendimento mais simples para todos.

9 - De uma maneira geral, após a introdução e a aplicação do BDD no projeto, você acredita que foi possível transformar os requisitos em um produto que agrega valor ao cliente?

QA1: 100%

QA2: Sim. Foi possível refinar os requisitos já existentes, e então, ver novas oportunidades que geram valor para o cliente.

DEV1: Sim, simplesmente por trazer essa estrutura e fluxo de validações.

QA3: Com certeza. O cliente especificou no sow e transforma em US e é entregue pra ele na maioria dos casos e quando está estável eles ficam satisfeitos.

QA4: É sim, assim eu não sei se tem estudo, mais eu consigo observar pela quantidade de defeito a cada Sprint, no meu time é assim que eu cheguei eu ainda tava me adaptando a gente tava conhecendo o projeto então foi basicamente duas sprints que o pessoal fazia as features sem ter caso de teste para as features que a gente tava elaborando e tinha muito problema de bug que ia para produção que só era identificada na produção, o usuário que tava fazendo o uso da aplicação era que encontrava o problema, e acaba relação com o pessoal, pessoal tinha que corrigir e a partir do momento que a gente começou assim já ajustar a questão dos cenários de teste a nossa inicial dos movimentos que é uma coisa que foi teve um próximo tempo assim de adaptação a gente percebeu uma diminuída bastante na questão de problemas que chegam em ambiente de produção.

DEV2: eu acredito sim, a gente tenta sempre pensar como o cliente vai usar, quando a gente começou a aplicar fomos bem a fundo de como ele ia usar como ele teria uma melhor experiência.

QA5: acredito que sim

10 - De uma maneira geral, após a automação dos testes com o auxílio do BDD no projeto, houve uma melhoria na qualidade no processo de desenvolvimento de software?

QA1: Sim, pois os cenários automatizados são executados frequentemente, garantindo que aquela solução continua se comportando conforme o esperado, consequentemente, garantindo a qualidade do processo

QA2: Sim. a definição de cenários de teste utilizando BDD foi de grande auxílio na construção de uma automação de testes.

DEV1: Sim, dado que tenhamos uma definição completa junto a uma implantação acredito que a adoção foi muito produtiva para o desenvolvimento.

QA3: Teve uma melhora principalmente nos testes de E2E que conseguimos garantir que os cenários estão sendo acordados através da automação, caso apareça algum problema vai ter uma visibilidade muito rápida. A vantagem da automação é a garantia de sempre que a feature vai estar funcionando, quando for executado o processo de regressão.

QA4: a nossa aplicação é basicamente baseada em preenchimento do formulário com os campos daí então quando não tinha tinha os casos você tinha que fazer o teste manualmente, e o teste manualmente necessita de esforço uma pessoa e nem sempre o pessoal tem tempo para esse esforço, com automação sim tem uma diminuída boa, porque você consegue executar com mais facilidade mais rapidez os cenários que já foram definidos, já a automação é baseada nos cenários de teste. Então sim, houve uma melhora na qualidade.

DEV2: Sim, ficou bem menos refatoração no desenvolvimento, melhoria no código como código mais limpo. Menos tarefas de bug sendo retornada pelo time de qualidade.

QA5: Sim, principalmente por usarmos os cenários levantados para a construção da automação e assim construir uma cobertura dos nossos testes aceitável.