UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Gabriel Dias Ribeiro

Automação de testes aplicados ao DevOps

Uberlândia, Brasil

2019

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Gabriel Dias Ribeiro

Automação de testes aplicados ao DevOps

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Ronaldo Castro de Oliveira

Universidade Federal de Uberlândia – UFU

Faculdade de Ciência da Computação

Bacharelado em Sistemas de Informação

Uberlândia, Brasil

2019

Gabriel Dias Ribeiro

Automação de testes aplicados ao DevOps

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Sistemas de Informação.

Trabalho aprovado. Uberlândia, Brasil, 11 de julho de 2019:

Ronaldo Castro de Oliveira
Orientador

Professor

Professor

Uberlândia, Brasil
2019

Dedico a todos que apoiaram no desenvolvimento deste projeto, em especial o professor Ronaldo Castro de Oliveira

Agradecimentos

À Deus por ter me fortalecido em todo momento.

Aos meus pais Geraldo e Cleide por terem proporcionado toda a estrutura necessária nessa etapa, além do amor incondicional.

À minha namorada Vitória por todo apoio, incentivo e carinho.

Ao professor e orientador Ronaldo Castro de Oliveira pela disposição e ensinamentos passados.

À Universidade Federal de Uberlândia e todo o seu corpo docente, direção e administração, pela oportunidade de fazer o curso.

Aos que contribuiram de forma direta e indireta em toda minha formação.

Resumo

Este trabalho de conclusão de curso tem como objetivo a apresentação da aplicabilidade dos testes automatizados a metodologia DevOps. Para demonstrar a integração entre os conceitos foi implementada uma aplicação web, responsável por representar o produto, definido o escopo de testes, responsáveis por garantirem a qualidade esperada do sistema, e implementado os scripts de testes automatizados, responsáveis pela execução automática dentro do ciclo DevOps. Nessa monografia, foram abordados os principais cenários dentro do processo de desenvolvimento de software, resultando nos fluxos de trabalho da automação de testes junto ao DevOps.

Palavras-chave: Quality Assurance. Testes. Automação. DevOps.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

Com o contínuo avanço da tecnologia a grande importância dos computadores tem se tornado incontestável, a cada dia que passa ele participa mais na forma de se comunicar, relacionar, produzir, consumir e transmitir informação. Contendo grande capacidade de armazenamento e processamento de dados esse instrumento se consolidou como uma das principais ferramentas de trabalho, assim como forma de diversão e lazer.

Fato é, que o computador tem se apresentado indispensável na vida do ser humano, seus sistemas estão presentes nas tarefas desde as mais simples até as mais complexas, com isso tem se popularizado amplamente devido à, principalmente, comodidade, praticidade e facilidade de acesso.

Deparando-se com a necessidade de modernização e o alcance a todos os públicos sistemas antes mecânicos veem-se obrigados a migrar para a plataforma digital, fazendo com que cada vez mais sejam iniciados processos de desenvolvimento de software. O processo de criação de um software pode ser abordado de diversas maneiras, porém, algo comum à grande maioria das abordagens existentes é a busca por qualidade através do teste, sendo essa uma das etapas do processo de desenvolvimento de software, que consiste em buscar falhas no sistema para que as mesmas sejam corrigidas e o sistema alcance a qualidade esperada.(GSTI, 2018)

A busca por qualidade em um sistema pode ser algo complexo dado regras de negócio, custo e questões burocráticas. Pensando na solução desses problemas surgem os testes automatizados. Estes, por sua vez, tem se tornado cada vez mais comum, apresentando diversas plataformas e ferramentas para realização dessa atividade e mostrando-se extremamente eficaz e apresentando entre suas diversas vantagens as reduções de esforço e custo, confiabilidade e constância na cobertura dos testes.(OBJECTIVE, 2018)

No contexto de desenvolvimento de sistemas, o alinhamento de metodologias, ferramentas e tarefas entre a equipe responsável pelo desenvolvimento de software e a equipe de operações, com o intuito de tornar mais ágil as entregas do produto reduzindo os intervalos entre o desenvolvimento e operações, além de suprir a qualidade esperada surge. (IBM, 2018) Conhecido como DevOps, esse segmento tem se disseminado cada vez mais, principalmente devido à seu alinhamento de propostas aos métodos mais populares nos dias atuais, os métodos ágeis.

Pensando na aplicação dos testes ao DevOps sabe-se dos tipos de teste passíveis de utilização nessa metodologia: os testes integrados, de performance, visuais, funcionais e unitários (QUALITY, 2018) podem ser aplicados durante o desenvolvimento do produto visando garantia de qualidade alinhada a entrega ágeis.

Especificamente no que se tange aos conceitos de DevOps e de testes automatizados ainda há pouca informação sobre interação e, principalmente, aplicação dos mesmos. Desta forma, este trabalho busca encontrar soluções de unificação dos conceitos, apresentando a aplicação da automação de testes a metodologia DevOps, visando atingir o objetivo de melhoria do produto final com as vantagens do teste automatizado.

1.1 Trabalhos Correlatos

O trabalho desenvolvido por (MALDONADO et al., 2004) visa apresentar os fatores teóricos e práticos ligados as atividades de teste de software, apresentando um compilado das metodologias dos testes nos âmbitos de sua funcionalidade, estrutura e projeção de erros. Processos de avaliação dos critérios de teste são abordados, assim como a importação da automação dentro do cenário de qualidade.

Os autores destacam as técnicas e critérios de teste diferenciando-as pela fonte de dados da informação.

A técnica funcional ou caixa preta é apresentada juntamente com seus critérios de teste, sendo eles o Particionamento em Classes de Equivalência, Análise de Valor Limite e Grafo de Causa-Efeito e como utilizá-los para obter casos de testes capazes de verificar se as funcionalidades do software atendem ao padrão estabelecido. Os problemas e dificuldades existentes são abordados, dando grande ênfase a baixa qualidade das especificações que, muitas vezes, não está contida em um documento formal.

Quanto as técnicas estruturais são indicados os principais pontos da mesma, funcionamento, critérios e, assim como a técnica funcional, suas principais dificuldades. É dado grande importância ao processo estrutural e sua principal forma de trabalho, o fluxo de dados, exibindo como o critério pode ser aplicado para a verificação do software em construção. Para esta técnica é apresentada a ferramenta de teste PokeTool, sua utilização é demonstrada ela pode gerar valor agregado ao processo de teste.

É brevemente apresentada a técnica baseada em erros e como a mesma é utilizada na derivação dos requisitos de teste.

Os critérios Análise dos Mutantes e Mutação de Interface possuem amplo espaço no trabalho desenvolvido pelos autores, sendo apresentado seus processos, aplicações e variações. As ferramentas de trabalho Proteum e Proteum/IM são, respectivamente, apresentadas para os critérios acima.

Para os autores a automação do processo de teste é necessária, visto que a atividade de teste é um processo repetitivo e que pode tender a erros. É feito um comparativo com as ferramentas de automação apresentadas, destacando as abordagens de cada uma delas.

Estudos empíricos são utilizados, visto que segundo os autores a realização destes

chega a uma estratégia capaz de revelar falhas existentes no programa, porém com baixo custo de aplicação.

O trabalho desenvolvido por (OLIVEIRA, 2017) visa apresentar a metodologia DevOps e seus conceitos, paradigmas, processo e aplicações, seguindo a estrutura de estabelecer o conceito da abordagem, analisar os benefícios de sua implementação e expor os impactos e riscos da mesma.

Abrangendo também uma visão de entendimento do contexto de estratégias de mercado e negócio de desenvolvimento de software, o trabalho inicia com uma revisão da evolução das metodologias no mercado que migraram de abordagens incrementais como cascata até os modelos ágeis como lean, Kanban e SCRUM, e como, a partir da popularização da cultura ágil, o DevOps foi se estabelecendo.

É destacado no trabalho o grande potencial da metodologia DevOps e como ela pode criar valor automatizado e otimizado para o negócio. Também é demonstrando que apesar de possuir uma curva de implantação, a abordagem supera as expectativas e já passa a ser cada vez mais adotada. Outro ponto que destaca seu amplo valor agregado é o aspecto financeiro que, com a adoção da metodologia, fornece as empresas a possibilidade de obter vantagem competitiva no mercado.

O autor apresenta também uma contextualização do tema e uma linha do tempo da metodologia, demonstrando o surgimento e evolução da mesma, cronologicamente.

No trabalho, o DevOps é apresentado como o responsável pela otimização entre os setores de desenvolvimento e operações, tendo como características a entrega continua, consistência e escalabilidade dos serviços, automação dos processos e, principalmente, a agilidade no feedback. Sendo visto, também, como uma cultura a ser dissipada e estabelecida no ambiente de trabalho, um dos grandes objetivos dessa estratégia é a agilidade na construção e entrega nos produtos de trabalho. Para o autor, o processo de atuação do DevOps segue um ciclo com as etapas de Planejamento, Análise, Desenho, Implementação, Teste e Integração, e Manutenção. O autor (OLIVEIRA, 2017) coloca também a necessidade de adoção dos parâmetros de automação, gestão de ambientes, ciclo de vida e configurações, ferramentas e colaboração.

Existe grande destaque para o conceito de Integração Contínua, apresentado no trabalho, onde o mesmo é definido como a consolidação do processo de entrega contínua, integrando código e teste, e concretizando a entrega ágil e eficaz. Entrega Contínua que, por sua vez, é conceituada no trabalho como o processo de produção de software em curtos ciclos gerando maiores entregas do produto desenvolvido, também, é lembrada pelo autor. Ainda há espaço para a apresentação do conceito de Implantação Contínua, sendo definida como o progresso das entregas contínuas, visto que, após entregue o produto de trabalho automaticamente são efetuadas as verificações de qualidade e a subida para o ambiente de

produção. Apesar dos conceitos serem interligados e comporem a estrutura da metodologia DevOps um comparativo entre os tópicos é realizado pelo autor, mostrando os custos e vantagens de cada um dos mesmos.

O método utilizado pelo autor para comprovar a eficácia da abordagem segue o roteiro de definição do DevOps, conceitos associados ao DevOps, diferenças nas metodologias ágeis e vantagens econômicas e de processo geradas. Sendo executado em cada um dos estudos de caso apresentados no trabalho, o roteiro é capaz de gerar uma série de benefícios comprovando a produtividade do DevOps.

O autor conclui seu trabalho ressaltando três importantes pontos para o sucesso de um projeto que são obtidos através da metodologia DevOps, são eles a facilidade de alterações no produto, garantida pela simplicidade no rollback, a entrega mais rápida e a diminuição dos riscos.

Este trabalho apresenta os fatores relacionados a aplicação dos testes automatizados no cenário de desenvolvimento de software, apontando seus benefícios, funcionamento, ferramentas, e, até mesmo, comparativos com outros tipos de atividades de teste. Correlacionando, também, tal processo a metodologias ágeis de desenvolvimento, como Programação extrema (XP).

É apresentado uma visão do cenário atual de desenvolvimento, onde, na maioria dos casos, o processo é realizado de forma manual, possui processos rígidos e derivados do modelo Cascata. E apesar de comum, esse cenário pode dificultar muito a entrega de um produto que corresponda aos padrões solicitados.

Os autores dão amplo destaque ao surgimento de novas metodologias, tais como Lean, SCRUM e XP, e como tais metodologias se relacionam com testes automatizados, apresentando suas características, modelos de integração as estratégias propostas e como essa abordagem pode solucionar problemas oriundos dos testes manuais.

A princípio é apresentado a estratégia de implementação de testes unitários automatizados utilizando linguagem de programação Java, ferramenta Junit e IDE Eclipse.

Posteriormente, os autores apresentam a aplicação da automação de testes a interface web, onde é utilizada a ferramenta Selenium integrada ao JUnit e linguagem Java, porém abre-se espaço também para a flexibilidade da ferramenta, que pode ser utilizada também em outras linguagens.

O trabalho de (BERNARDO; KON, 2008) conclui salientando benefícios dos testes automatizados, como escalabilidade, confiabilidade e facilidade de manutenção. Dando grande ênfase para como a transformação do cenário atual, onde a maioria dos testes é executada de forma manual, para o cenário automatizado pode gerar valor agregado para uma organização.

O trabalho desenvolvido por (SACHDEVA, 2016) visa apresentar a integração do testes automatizados junto a metodologia DevOps aplicados a um estudo de caso, demonstrando a adaptação necessária nos processos para implantação das novas práticas da cultura ágil do DevOps e dos testes automatizados e principalmente, os benefícios identificados na análise dos resultados do novo processo.

O autor inicia seu trabalho contextualizado o modelo atual de desenvolvimento e entrega dentro da área onde a implantação dos testes automatizados será efetuada, o setor de tecnologia da informação da indústria da saúde. O cenário atual era composto de diversas técnicas consideradas arcaicas, onde grande parte do processo era efetuado de maneira manual e a integração entre os setores era de baixa qualidade. Iniciou-se um trabalho de conscientização e aplicação de novas técnicas onde era proposto um ambiente em nuvem com processos automatizados de construção, entrega e testes.

Difundiu-se amplamente a mudança do controle de qualidade para a engenharia de qualidade onde novos modelos foram propostos, avaliando a qualidade dos produtos de trabalho desde o princípio e a construção dos scripts de teste automatizados simultâneos a construção do software.

Dentro da organização, estabeleceu-se a setorização dos profissionais de testes em quatro áreas, engenharia de automação, desenvolvimento, especialização e engenheiros de qualidades. Para aplicação dessa estrutura foram estabelecidos diversos planos de formação para capacitar e adequar os profissionais. Também foi reformulada a estrutura da pirâmide de testes para compreender a nova abordagem, formulando novos setores de teste.

Um importante ponto no trabalho foi a reestruturação do ciclo de desenvolvimento e entrega, utilizando como ferramenta de versionamento de código o git e de integração contínua o Jenkins, pontos cruciais para o sucesso da nova estrutura, dado a simplicidade de configuração, aplicação e integração das mesmas. Foi preparado um ambiente de entrega e implantação contínua, preparando o desenvolvimento, tanto do software quanto dos testes automatizados, para integrarem-se as configurações e implantações estruturadas.

Os resultados encontrados pelo autor destacam que a aplicação da automação de testes integradas a metodologia DevOps foi amplamente vantajosa para a organização, tornando possível a redução dos custos de infraestrutura em nuvem, flexibilidade na definição de tecnologias do projeto e diminuição no tempo de execução do teste, consequentemente, gerando uma entrega mais rápida.

2 Quality Assurance

2.1 Definição

Nos dias atuais, um dos principais pontos analisados em qualquer produto ou serviço entregue é a qualidade do mesmo. Este fator é de extrema relevância para a empresa, visto que, a qualidade de um produto é capaz de abrir ou fechar portas para a companhia, definir a imagem do produto no mercado e o nível de satisfação do usuário. Para o usuário, trata-se da capacidade de suprir, através de suas funcionalidades, as necessidades do mesmo de forma correta, prática e sem erros.

Dada a crescente relevância da qualidade em um produto surge uma das mais delicadas áreas do desenvolvimento de um sistema de software, o Quality Assurance ou Garantia de Qualidade que trata da gestão da qualidade de um produto, utilizando como base padrões de qualidade relevantes, administrando técnicas, processos e atividades com foco na detecção de falhas/erros que possam afetar o desempenho esperado do sistema.(GAEA, 2018a)

2.2 Relevância

O setor de qualidade possui a difícil tarefa de fazer com o produto seja entregue aos usuários nas condições esperadas pelos mesmos e sem apresentação de erros, ou seja, sua importância é extrema.

Sabe-se que o Quality Assurance é capaz de ser uma ferramenta de controle de custos e de marketing. (IGTI, 2018)

O marketing, pois o produto que é entregue está vinculado juntamente com a imagem da empresa, em caso de um sistema com alta qualidade pode trazer maior visibilidade e criar novas oportunidades para a produtora, enquanto que, um sistema defeituoso pode denegrir a visão de mercado da companhia, acabar com oportunidades futuras e diminuir significativamente o interesse dos usuários pelo produto.

Os custos pois, desenvolvimento do projeto e custos de correção de um defeito estão diretamente associados, ou seja, a medida que se avança nas etapas de desenvolvimento do produto, aumenta-se o custo para correção de um defeito que possa ser descoberto. Uma das principais razões para aumento dos custos de correção de uma falha é a adaptação necessária em todo material já desenvolvido para contemplar agora a funcionalidade existindo em sua forma correta, ou seja, não somente insere-se a correção de uma falha, é necessário analisar como a correção afetará aquilo que já existe e como projetar a inserção

de forma que não se crie nenhum outro erro.

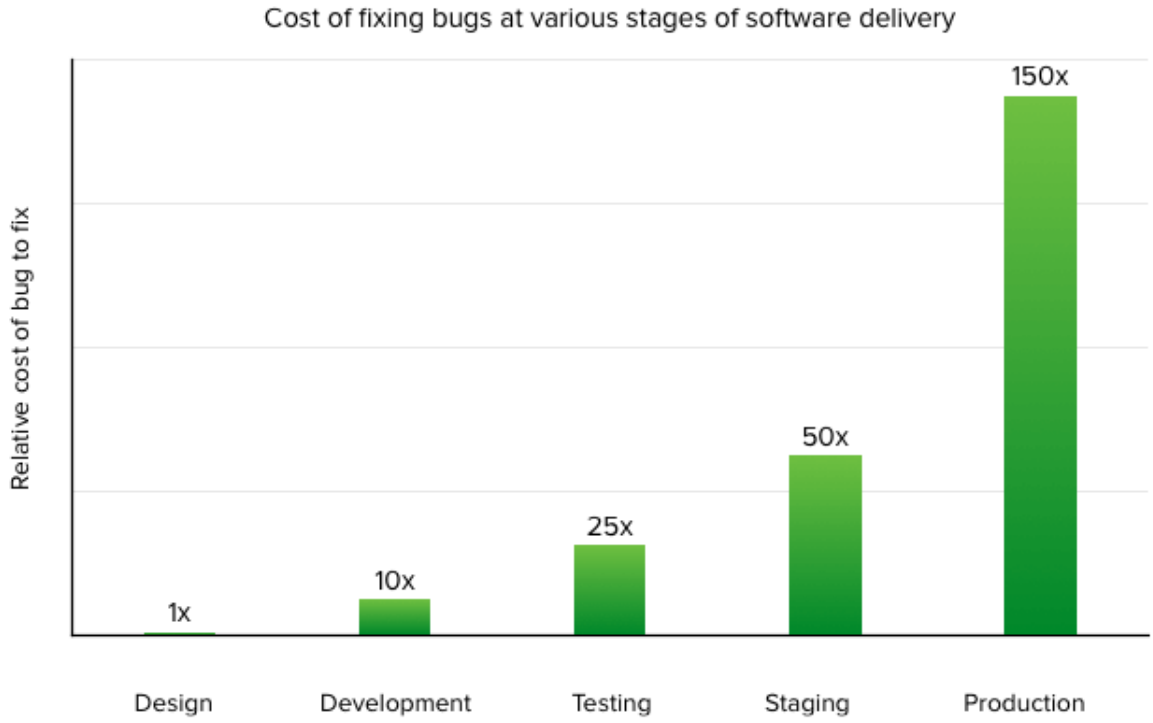

Figura 1 – Custo relativo de correção de uma falha (RAYGUN, 2014)

2.3 Funcionamento

É sabido que o papel do setor de Quality Assurance é a gestão de qualidade de um produto, para que isso seja possível é necessária atuação nas etapas de desenvolvimento do projeto certificando-se que o escopo esteja conforme o esperado, ou seja, a equipe de garantia de qualidade é capaz de criar estratégias de verificação de funcionalidades e realizá-las, conferindo seu resultado.

Para certificar-se da qualidade do sistema, o profissional responsável pela garantia de qualidade utiliza como principal ferramenta os testes. Como esse profissional possui conhecimento dos requisitos do sistema e de suas funcionalidades é capaz de definir um fluxo dentro do produto e observar se o resultado ocorre conforme o esperado, efetuando assim testes em todas as funcionalidades do sistema.

Para criação dos testes e certificação da qualidade do sistema, diversos padrões são utilizados, entre eles estão funcionalidade, confiabilidade e desempenho. (BASTOS et al., 2007)

A funcionalidade é utilizada como padrão de qualidade pois, a partir dela se percebe se o sistema é capaz de atender a necessidade do usuário, comportando-se da maneira correta e abrangendo todo o escopo do requisito funcional.

Confiabilidade trata-se da capacidade de resiliência do produto frente a possíveis falhas, ou seja, em um sistema com alto padrão de qualidade é possível utilizar-se do mesmo normalmente mesmo em um cenário onde existam erros.

Desempenho avalia se a performance do sistema é adequada, se não existem tempos de respostas fora de curva mesmo em cenários incomuns.

O profissional do setor de Quality Assurance utiliza das dimensões de qualidade para elaborar os testes necessários para verificar o sistema. Sendo assim, existem diversos níveis de testes de software, sendo os principais: teste de unidade, teste de integração, teste de sistema e teste de aceitação. Participantes do modelo V (variação do modelo de desenvolvimento em cascata) esse níveis de teste são relacionados as etapas do processo de desenvolvimento de software. (SOMMERVILLE; SAWYER, 1997) (PRESSMAN; MAXIM, 2016)

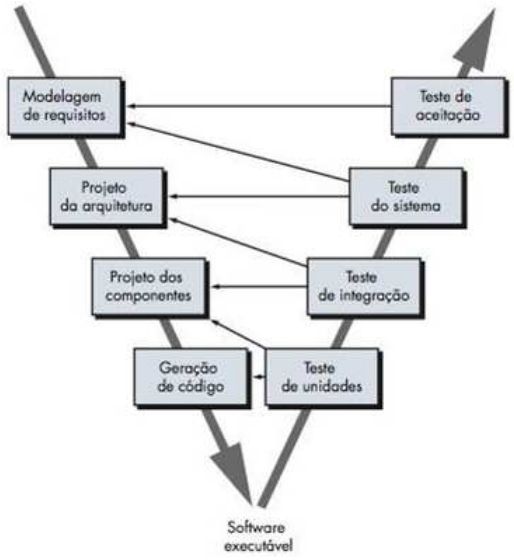

Figura 2 – Modelo 'V' (Variação do modelo Cascata) (DEVMEDIA, 2013)

O teste de componentes ou teste unitário tem como objetivo a avaliação de uma unidade do sistema, avaliando de forma isolada se a mesma oferece o funcionamento de acordo com o especificado. O responsável pela execução do teste avalia a entrada e saída dos dados, o funcionamento esperado do componente e o processamento do fluxo de dados, garantindo que possíveis sejam detectados em fases iniciais e que não se espalhem em níveis mais altos do produto. Geralmente, esse tipo de teste é de responsabilidade do próprio programador.

A etapa em que se testa a interface dos componentes é chamada de teste de integração. Nela são verificadas as combinações dos módulos do sistema, ou seja, a interação entre as partes do produto. Esse nível de teste possui escalabilidade, sendo possível ser entre componentes (caso em que é executado por desenvolvedores) ou entre sistemas (caso em que é executado por testadores).

O teste de sistema consiste na verificação do sistema como um todo, analisando se com todos os módulos trabalhando em conjunto o produto é capaz de atender os requisitos, sendo esses funcionais ou não funcionais, esse nível de teste visa avaliar os aspectos gerais do produto em questão. Ainda dentro do teste de sistema, existe um tipo de teste que possui grande importância, o teste de caixa-preta ou teste funcional. Essa técnica é responsável por verificar o comportamento do sistema, tendo como informações os parâmetros de entrada e o resultado esperado da aplicação, ou seja, quanto mais combinações de entrada para o teste, maior cobertura funcional teremos.

Geralmente de responsabilidade dos clientes, usuários ou donos do produto o teste de aceitação é o último nível de teste e visa validar se o produto se encontra apto para uso, analisando os requisitos originais e as necessidades do usuário e produzindo informações para avaliar a situação do sistema para implantação e entrega ao cliente. O teste de aceitação é dividido em três estratégias: teste alfa/beta, aceitação formal e aceitação informal.

2.4 Desafios

Apesar da grande relevância do setor de Quality Assurance para um produto ainda existem diversos desafios para a área. A falta de profissionais especializados, ferramentas para execução do trabalho, prazo e, principalmente, o custo, são alguns desses empecilhos.

Mesmo com o contínuo crescimento, o segmento de garantia de qualidade é considerado relativamente novo no ramo de desenvolvimento de sistemas, com isso temos uma pouca oferta de profissionais especializados no ramo, assim como ferramentas apropriadas para execução do mesmo, que por sua vez, podem acabar nem atendendo todas as necessidades do projeto.

Para manutenção e funcionamento deste setor são necessários recursos e tempo, fatores estes que resultam em um aumento no custo ou no prazo de entrega do projeto. Como o Quality Assurance não é tratado como prioridade e, ás vezes, rotulado como desnecessário, acaba-se não envolvendo tais atividades nos processos de desenvolvimento de sistemas de software. Porém, com o aumento nos padrões de qualidade do usuário final tal cenário vem apresentando mudança significativa e o setor ganhando cada vez mais força, superando os desafios existentes.

3 Testes automatizados

3.1 Definição

Sabendo da crescente relevância dada aos testes dentro do processo de garantia de qualidade surgem diversos conceitos com o objetivo de facilitar ainda mais essa tarefa, entre eles está a automação dos testes.

Analisando o contexto atual das empresas e da ampla necessidade de entregar um produto completo ao usuário a apresentação e utilização de estratégias que possam viabilizar ainda mais essa disciplina de qualidade de software não são dispensáveis, tal fato não deixa de se aplicar a automação de testes dentro de uma organização.

A automação de testes consiste na utilização de uma ferramenta para obter controle sobre a execução de testes utilizando-se da aplicação de processos e estratégias para validação do produto. (MEDIUM, 2019) Nela são verificados, de forma automatizada, se requisitos funcionais ou não funcionais atendem ao padrão especificado pelo cliente comparando os resultados esperados com aquilo que foi encontrado, com o objetivo de reduzir ao máximo o envolvimento humano nessas tarefas.

Apesar da eficiência dos testes manuais não ser baixa e essa tarefa permitir o descobrimento de diversas falhas no sistema, consequentemente, melhorando o nível de qualidade do produto, essa tarefa demanda bastante esforço, tempo e recursos, fatos tais que na área de desenvolvimento de software acabam se convertendo em despesas econômicas e influenciando a adoção do modelo automatizado. Considerados os grandes objetivos da automação de testes, a facilidade no controle da execução dos testes e redução de recursos são grandes atrativos quando se deseja instalar essa estratégia dentro de uma organização, porém esses não são as únicas vantagens desse modelo, sendo que a automação também é capaz de fornecer consistência e confiabilidade na repetição dos testes, uma vez que as validações são realizadas sempre da mesma maneira, facilidade no acesso à informações relacionadas as execuções, visto que com a utilização de ferramentas do tipo torna-se mais fácil extrair dados como progresso do teste, taxas de erros e performance, e, principalmente, redução no trabalho manual repetitivo, tarefa que com o passar do tempo pode perder confiabilidade, ocasionando uma queda de qualidade no produto final.

Grandes são as vantagens da aplicação dos testes automatizados em uma organização, porém a utilização dessa estratégia também apresenta alguns riscos, tais como expectativas irreais de resultados, a subestimação de recursos como tempo, esforço e custo, a negligência no controle de versão dos produtos de trabalho, além da premissa existente da dependência de ferramentas e frameworks para execução do processo.

3.2 Abordagens

Apesar de crescente a aplicação dos testes automatizados nos cenários atuais está bem distante de ser trivial. Dentro da automação de testes possuímos duas grandes vertentes: automação de testes via gravação/execução e baseadas em scripts. (DEVMEDIA, 2015)

Na automação de teste baseada em gravação/reprodução de comandos via interface gráfica, também conhecida como record and play, uma ferramenta é utilizada para simular os eventos de entrada e interações com o navegador, observando sempre os resultados das ações. Para criação deste tipo de automação é necessário executar o procedimento de teste, enquanto o software guarda as ações executadas para uma posterior execução. Esta abordagem é aplicável a qualquer tipo de sistema que utilize interface gráfica, porém é, comumente, aplicada a sistemas web. Uma grande vantagem desse tipo de estratégia é a facilidade de implementação, dado que a necessidade de codificação é pouca ou inexistente, todavia apresenta também alguns pontos negativos, entre eles a eminente dificuldade de manutenção do projeto, sendo que se algum componente da interface é alterado o funcionamento do teste é comprometido, a difícil aplicabilidade e controle no caso de grandes projetos e a execução individual, ou seja, não é possível executar testes paralelamente.

Na abordagem baseada em código, também é simulada a interação de comandos (click, preenchimento de campos, navegação, etc.) com o navegador, porém sua execução está condicionada a um script de código desenvolvido em alguma linguagem de programação. A estrutura do projeto de automação de teste segue a estrutura de um projeto de desenvolvimento de sistema comum, incluindo boas práticas de programação e utilização de padrões de projeto, se necessário. O grande diferencial está na utilização de um framework, utilizada em conjunto com alguma linguagem de programação, que será o responsável por executar os passos do procedimento de teste, analisando entrada e saída, além do comportamento esperado do software. O ponto principal dessa abordagem é a escalabilidade, tornando possível a aplicação a grandes projetos, a confiabilidade, podendo abranger maior número de validações, e a fácil manutenção, devido à utilização de uma linguagem de programação é possível alterar somente o componente defeituoso do script.

3.3 Critérios necessários

Não há dúvidas de que a aplicação da automação de testes é uma boa prática dentro do processo de teste de uma organização, contudo diversos fatores são necessários para que a adoção dessa estratégia obtenha sucesso.

O principal fator para que o estabelecimento dessa prática obtenha êxito é o nível de maturidade da organização nos procedimentos, sobretudo, o de teste. (MEDIUM,

2019) É fundamental que a empresa tenha seus processos bem estruturados e definidos, possuindo também experiência e bom gerenciamento, tais fatores são relevantes, dado que a automação de teste exige uma análise rigorosa na demanda a qual será aplicada, onde projetos com frequentes mudanças podem ser prejudicados com essa abordagem. Outro ponto da automação que espera um nível elevado de maturidade na organização é na priorização dos testes. Exceto em casos triviais, a cobertura total de testes em um projeto não é viável, pois exige que recursos como tempo e custo sejam elevados ao extremo, por isso nos casos onde se decide adotar a automação de testes espera-se que os cenários principais de teste sejam os aptos a automação.

4 DevOps

4.1 Definição

No mercado atual de desenvolvimento de produtos de tecnologia da informação é cada vez mais comum a aproximação ao setor de negócios, onde padrões individuais passam a agir em integração com um mesmo objetivo, a entrega rápida de um produto de qualidade. Durante a atividade de gestão dos processos de desenvolvimento de software é necessária grande atenção aos conceitos que surgem, sendo de grande relevância o DevOps.

É relevante destacar que o DevOps não se trata de uma tarefa, tecnologia ou processo a ser adotado ou integrado, sendo considerado como uma cultura ou movimento entre os adeptos ao conceito.

O termo DevOps surgiu pela combinação das palavras "desenvolvimento" e "operações", palavras estas que estão relacionadas a processos na atividade de construção de um software. Tradicionalmente desiguais, tais setores possuem motivações distintas dentro de uma organização, sendo a área de desenvolvimento responsável pela evolução do projeto utilizando cada vez mais a adoção de metodologias ágeis, torna-se possível a realização de entregas constantes e com tempo reduzido, ou seja, atende-se a expectativa do demandante de que trabalho está em constante progresso. Enquanto o setor de operações, que é o responsável pela implantação e manutenção no ambiente produtivo, deseja a menor quantidade de alterações possíveis, visando reduzir o risco de instabilidade e erros no produto e, consequentemente, desvalorização do software. Fato é, que as áreas, tendem a seguir caminhos divergentes devido à, principalmente, seus objetivos distintos. (GAEA, 2018b)

O conceito de DevOps pode ser descrito como um conjunto de práticas que visam a integração, através da comunicação e colaboração, entre as equipes de desenvolvimento e operações, simplificando processos, alcançando um nível de qualidade mais alto e reduzindo o tempo das entregas. Utilizando como ponto principal a adoção de automação de processos, o DevOps visa tornar possível o desenvolvimento de aplicações com eficiência e agilidade em um modelo de gestão de estrutura bem definidos. Esse modelo aborda conceitos como integração e implantação contínua dada sua grande semelhança de características, onde torna-se extremamente prático e estruturado o lançamento de novas versões do produto, gerenciamento e documentação. Outro grande pilar do DevOps é o feedback contínuo, presente em todas as etapas do ciclo de vida de desenvolvimento de software essa tarefa é capaz de mapear as dificuldades existentes nos processos para uma futura mitigação, assim como os pontos positivos de cada etapa.

Para a (AMAZON, 2019) o DevOps trata-se da "Combinação de filosofias culturais, práticas e ferramentas que aumentam a capacidade de uma empresa de distribuir aplicativos e serviços em alta velocidade: otimizando e aperfeiçoando produtos em um ritmo mais rápido do que o das empresas que usam processos tradicionais de desenvolvimento de software e gerenciamento de infraestrutura".

Fato é que o DevOps visa aperfeiçoar o modelo de desenvolvimento do produto solucionado as lacunas existentes entre os setores de desenvolvimento e operações, permitindo que essa comunicação seja facilitada, modificando as fronteiras de responsabilidade e tornando favorável o trabalho de ambos os setores, resultando em uma maior fluidez na entrega de valor ao cliente.

4.2 Funcionamento

Com a adoção do modelo DevOps em uma organização, os times de desenvolvimento e operações passam a não atuar mais isoladamente e com objetivos distintos, mas sim a utilizar a simplificação e padronização de processos nesses setores para atuar em sintonia durante todo o ciclo de desenvolvimento do produto.

Vale ressaltar que assim como todas as culturas existentes o DevOps também possui suas variações, sendo adaptado a realidade e estrutura de cada organização. Porém existem aspectos comuns a todas essas variações, entre eles estão a integração contínua, automação de processos, comunicação e colaboração. (AMAZON, 2019)

A automação de processos possui um importante valor no correto funcionamento das práticas da cultura DevOps, visando ganho significativo na substituição de tarefas manuais por rotinas automatizadas, passa-se a responsabilizar equipes que antes possuíam tarefas como subida de correções e novas funcionalidades, apenas para acompanhamento, documentação e melhorias contínuas. Fato é, que o DevOps depende amplamente da automação, fazendo uso de diversas ferramentas para isso, tornando-se ajuizado pelo correto gerenciamento, funcionamento e integração das mesmas.

Com extrema importância no modelo DevOps, a comunicação é responsável por garantir não somente a integração entre os times de desenvolvimento e operações, mas também dissipar a aproximação e colaboração com os setores de negócio, gestão e administração dos produtos e serviços. O grande resultado alcançado com a melhoria na comunicação entre os setores está na criação e entendimento de soluções e melhorias no processo, com as áreas atuando em sinergia, visando a estrutura das atividades, não em segregação, mas sim como um todo, a redução no tempo de entrega é extremamente significativa. Atuando em fluxo de trabalho colaborativo a responsabilidade pelas atividades passa a ser compartilhada aumentando a eficiência e reduzindo desperdícios.

Considerado um dos grandes pilares da cultura DevOps, devido a sua semelhança de características com o movimento ágil, a integração contínua visa acelerar o ciclo de desenvolvimento e entrega de código eliminando as lacunas existentes entre as etapas e trabalhando na integração entre as mesmas, encontrando e mitigando os erros com maior eficiência e, consequentemente, elevando a qualidade do produto. Para obter tais resultados um conjunto de práticas de desenvolvimento é proposto, onde mudanças ou desenvolvimentos de código feitos são agrupados com frequência em uma base de código, de modo que, assim que finalizadas uma suíte de testes planejados são executadas sobre o produto de trabalho identificando e corrigindo um possível defeito o mais rápido possível, reduzindo o tempo de validação e lançamento de novas versões do produto.

O processo de integração contínua é adotado no DevOps na estrutura de um ciclo, onde cada etapa representa uma tarefa a ser melhorada continuamente. Estas etapas estão descritas nos próximos tópicos. É importante destacar que a nomenclatura das etapas podem variar conforme sua aplicação da cultura, porém em âmbito geral o contexto dos estágios deve ser sempre o mesmo.

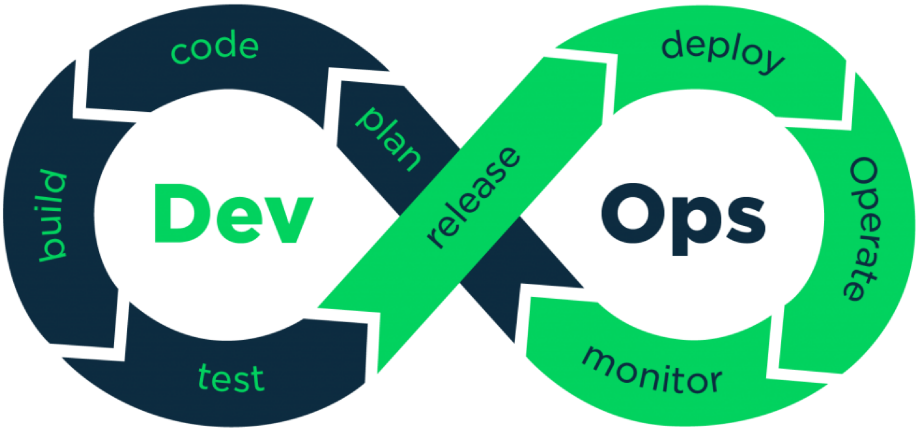

Figura 3 – Processo DevOps (Becode, 2017)

4.2.1 Plan (Planejamento)

A etapa de planejamento dentro do DevOps é responsável por definir o objetivo para todo o ciclo do projeto, estabelecendo o escopo das atividades para a fase em questão. Nessa etapa são analisadas as demandas existentes e suas prioridades, a capacidade de vasão de tarefas da equipe, riscos e pendências. Conforme a duração do ciclo varia a capacidade de entrega do time também sofre alteração, podendo receber menos ou mais atividades. Ao final dessa etapa obtemos a estratégia de trabalho, itens a serem desenvolvidos no ciclo atual e informações necessárias para desenvolvimento (requisitos, casos de uso, funcionalidades, etc.). Por ser a primeira etapa do ciclo DevOps, a etapa

de planejamento torna-se responsável por realizar a integração, principalmente através de ferramentas, com os ciclos anteriores e com as próximas etapas do ciclo atual. Para essa etapa podemos utilizar ferramentas responsáveis pelo gerenciamento e controle de tarefas como JIRA (ATLASSIAN, 2019) e Trello (TRELLO, 2019), além da aplicação de diversas metodologias ágeis.

4.2.2 Code (Codificação)

Dentro da cultura DevOps temos também a etapa de codificação, a qual é responsável por utilizar uma linguagem de programação para gerar o código-executável que corresponda ao software em desenvolvimento. A atividade de programação deve obter como resultado produtos correspondentes as tarefas mapeadas durante a etapa de planejamento, entregando parcialmente ou em totalidade o produto final, seguindo sempre padrões e estruturas estabelecidos para a organização e/ou projeto.

Também incluem-se nessa etapa tarefas comuns dentro da atividade de desenvolvimento de software, tais como a fusão de código (merge), inspeções e/ou revisões, interações com ferramentas responsáveis pelo gerenciamento de versões de código, como GIT (SCM, 2019) e Subversion (APACHE, 2019a). A etapa de codificação deve atuar no escopo definido para o ciclo atual, porém integrando seus produtos de trabalho aos demais entregues em ciclos anteriores.

4.2.3 Build (Construção)

Durante a etapa de construção, se tem como objetivo fundamental a preparação do ambiente para o produto, ou seja, é necessário garantir que as diversas partes do produto estejam em perfeita sintonia, seja uma chamada a um serviço externo ou o acesso remoto, fato é que nessa etapa é imprescindível que o produto esteja preparado para receber a fase seguinte do ciclo. Nessa etapa podemos ter o uso de ferramentas de integração contínua e grandes propostas de processos automatizados, melhorando consideravelmente a integração entre as etapas de codificação, construção e testes. Para essa etapa podemos utilizar ferramentas como Maven (APACHE, 2019b), que é um gerenciador de dependências, responsável pela compilação de projetos.

4.2.4 Test (Teste)

Juntamente com as etapas de codificação e construção, a etapa de teste forma o grupo dos processos de desenvolvimento dentro da cultura DevOps, essa etapa é responsável por assegurar que o produto entregue contenha a qualidade esperada pelo cliente. Dentro dessa etapa podemos ter diversos níveis de testes, podendo ir desde o analista desenvolvedor com o teste de unidade até uma equipe independentes de testes com testes

como de sistema, aceitação, e desempenho, que visam validar o correto funcionamento da aplicação em cada um dos aspectos dentro do escopo do projeto.

Sabendo das práticas da cultura DevOps, a etapa de teste visa elevar-se ao nível máximo de automação, para que o ciclo possa ser o mais dinâmico, integrado e com menor número de lacunas possível. Automatizando essa etapa também é possível extrair informações de grande relevância para o projeto como riscos de negócio, nível de cobertura dos testes e de qualidade atingida. Para essa etapa podemos utilizar diversas ferramentas como Selenium (SELENIUMHQ, 2019), responsável por automatizar testes em aplicações web, e o Appium (APPIUM, 2019), responsável pela automação de testes em dispositivos movéis, entre outras ferramentas.

4.2.5 Release (Lançamento)

O lançamento também se trata de um estágio presente no ciclo do DevOps, onde nele é efetuada a homologação do produto, realizando o gerenciamento de mudanças do mesmo, ou seja, quaisquer alterações sejam no produto ou na infraestrutura da organização são tratadas nessa importante etapa. Também são realizadas nessa etapa as aprovações necessárias para a liberação do software, e como no DevOps se tem como objetivo a integração contínua dos processos a proposta é que essas aprovações sejam realizadas sempre de forma automatizada tornando a ligação dos processos cada vez menos onerosa. Assim como a etapa de planejamento, a etapa de lançamento é responsável por fazer a ligação entre as partes de desenvolvimento e operações, dado que são estágios intermediários no ciclo completo do DevOps. Para essa etapa, podemos temos algumas ferramentas à disposição, tais como o Docker (DOCKER, 2019), ferramenta responsável por prover uma camada de virtualização para ambientes operacionais.

4.2.6 Deploy (Implantação)

Posterior a etapa de lançamento, em que temos a liberação dos produtos de trabalho, temos a etapa de implantação, na qual o principal objetivo é a instalação do software no ambiente produtivo, ou seja, é realizada a configuração e instalação da aplicação no ambiente em que o usuário final desejar. Juntamente a atividade de instalação inclui-se a entrega das documentações da nova versão do sistema e apresentação das novas funcionalidades.

Com uma complexidade relativamente alta, esse estágio do ciclo requer bastante cautela para que algum problema na configuração não gere para o usuário um comportamento inadequado do sistema. Sabendo da necessidade de preparação por parte do cliente para recebimento do produto essa etapa costuma ocorrer em períodos estratégicos para a empresa, visando a diminuição de impacto nas atividades diárias.

4.2.7 Operate (Operação)

Após finalizar a etapa de implantação, em que a aplicação é instalada no ambiente de produção, temos a etapa de operação, a qual é responsável por prestar suporte aos usuários do produto e atuar na correção de defeitos. A atividade de prestar manutenção pode ocorrer dentro da etapa de operação tanto de forma evolutiva, onde melhorias são levantadas para o sistema, quanto de forma corretiva, onde atua-se na correção de falhas encontradas. Para essa etapa podemos utilizar ferramentas de computação em nuvem, responsáveis por execução de serviços e aplicações, como o Microsoft Azure (MICROSOFT, 2019), Amazon Web Service (AMAZON, 2019), entre outras.

4.2.8 Monitor (Monitoramento)

A última etapa do ciclo DevOps é a etapa de monitoramento, na qual o funcionamento do sistema é constantemente analisado, onde defeitos, divergências no resultado esperado da aplicação, queda de performance ou, simplesmente, melhorias e adaptações são diagnosticadas e levantadas para posterior desenvolvimento. Como ferramentas de monitoramento, podemos utilizar ZABBIX, AppDynamics, dentre diversas outras.

4.3 Aspectos

É sabido que com a adoção desse novo modelo torna-se possível desfrutar de vantagens bastante notáveis, entre eles estão a redução nos custos de produção, melhoria na comunicação e integração entre os times gerando, consequentemente, maior fluidez e eficiência organizacional, e aumento na satisfação dos colaboradores e clientes, visto que, os resultados mostram-se mais facilmente visíveis devido à entrega contínua de pequenos pacotes, mas de grande valor agregado. O contexto de aplicação dessa cultura também é de extrema importância, dado que os resultados obtidos podem ser variáveis de organização para organização.

O fato é que a cultura DevOps já é realidade em muitas empresas e a tendência é que cresça cada vez mais, seu sucesso se deve a sintonia entre os times, diminuição de burocracia graças a simplificação dos processos, diminuição no tempo das entregas de valor e a grande busca por melhoria contínua nos processos estabelecidos.

Porém, assim como outras culturas e metodologias, o DevOps pode gerar alguns paradigmas dentro de uma organização, entre eles está o fato de que a mudança estrutural e cultural de DevOps, caso não seja feita de forma gradativa e organizada, pode resultar em uma aversão por parte dos times, criando-se um bloqueio quanto aos novos processos. Essa transformação pode impactar também nos projetos profissionais futuros dos funcionários da empresa, necessitando uma capacitação por partes dos mesmos,

especializando-se nas práticas e processos definidos pela cultura, fato esse que pode não convergir com as expectativas dos membros da organização.

Outro importante estereótipo que pode ser gerado é que a automação dos processos resultará na redução numérica do time, onde processos antes desempenhados por membros da equipe que agora passam a ser automáticos culminam na demissão em massa do time. Porém tal fato acaba não condizendo com a realidade encontrada, sendo que a automação de processos tende a absorver apenas as atividades de cunho extremamente repetitivo. Também espera-se que a adoção da cultura DevOps apresente uma redução significativa na interação humana dentro da organização devido à automação, fato que também não espelha a realidade, visto que um dos grandes pilares do DevOps é a comunicação e colaboração entre os times, necessitando que a equipe atue em um mesmo fluxo de trabalho.

Alguns anos após seu surgimento já é possível observar que o DevOps surgiu para se estabelecer entre os grandes modelos de trabalho dentro de uma organização, seu nível de adesão é cada dia maior e torna-se difícil até mesmo de caracterizar, o fato que resulta nessa crescente onda são os ganhos proporcionados com a adoção da cultura. O DevOps tornou-se capaz de agradar desde os clientes finais devido à constante geração de entregas de valor, membros dos times de desenvolvimento, operação e negócio que passaram a se comunicar em uma mesma língua, até os diretores e administradores responsáveis pela organização, que notam grande aumento econômico com a utilização desse modelo. Porém para obter todos esses ganhos a tarefa não é trivial, é necessário acompanhar ativamente os processos garantindo que as práticas não se dispersem.

5 Desenvolvimento

5.1 Aplicação

Para este trabalho foi desenvolvida uma aplicação web, utilizando as linguagens de programação HTML, CSS e JavaScript. A aplicação em questão vem como solução para um problema de um cenário fictício, nesse cenário um profissional do corpo docente de uma determinada instituição acadêmica necessitava de uma ferramenta de interação, comunicação e apresentação de informações referentes ao seu âmbito acadêmico e pessoal.

A aplicação foi estruturada em quatro telas, cada uma delas seguindo um determinado propósito. A apresentação das telas é feita a seguir:

5.1.1 HOME

A tela HOME (Inicial) trata-se da tela de entrada do sistema, ou seja, seu principal objetivo é fazer apresentações sobre a figura do docente, pequenas mensagens e informações de cunho básico. A tela contém um espaço destinado à biografia do docente, na sessão 'Sobre mim' é apresentada toda a história do professor em questão, também está presente nessa tela uma sessão destinada aos alunos, trazendo apenas uma breve mensagem.

Figura 4 – Apresentação da tela HOME do sistema

5.1.2 DISCIPLINAS

A tela DISCIPLINAS foi inserida na aplicação para atender a necessidade do docente de que seja apresentada as informações aos alunos das últimas seis disciplinas mi-

nistradas pelo professor na instituição acadêmica. A tela apresenta uma tabela em ordem decrescente, ou seja, do semestre atual até os últimos seis semestres decrescentemente, trazendo informações do semestre em questão, do código e nome da disciplina.

Figura 5 – Apresentação da tela DISCIPLINAS do sistema - parte 1

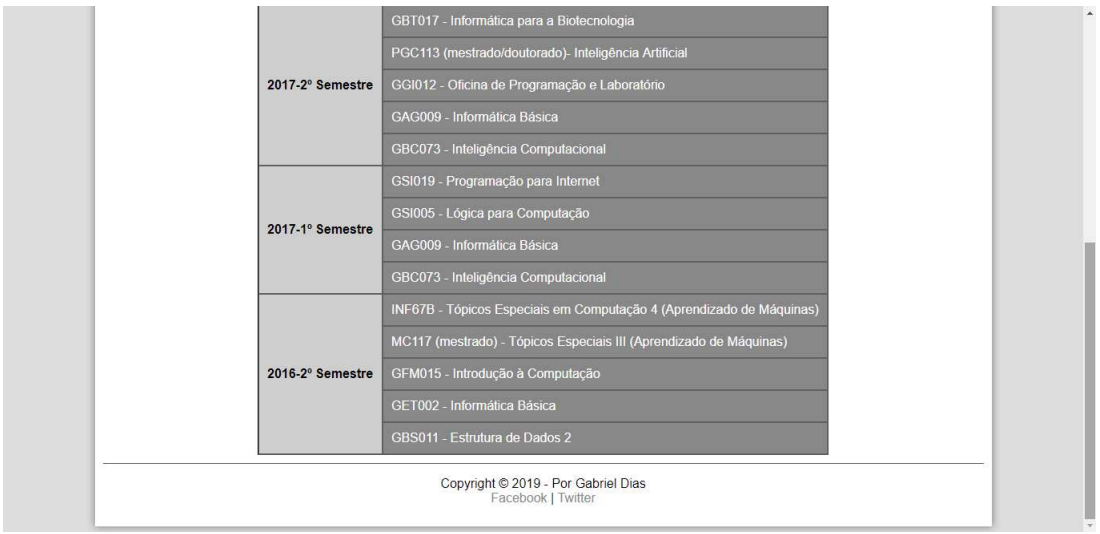

Figura 6 – Apresentação da tela DISCIPLINAS do sistema - parte 2

5.1.3 PESQUISA

A tela PESQUISA foi adicionada ao sistema com a finalidade de suprir a ausência de estudos referentes a área de pesquisa do docente, ou seja, são compiladas informações de apresentação sobre o ramo e os trabalhos já realizados ou de interesse. Nessa tela estão presentes, o espaço de 'Galeria de imagens' onde estão contidas imagens de trabalhos relacionados ao ramo de pesquisa do docente, e o espaço 'Pesquisa' onde é apresentado um breve roteiro científico e profissional dentro da área de pesquisa equivalente. Ainda

na sessão 'Pesquisa', é apresentado ao usuário da aplicação um apanhado de informações sobre o tema central.

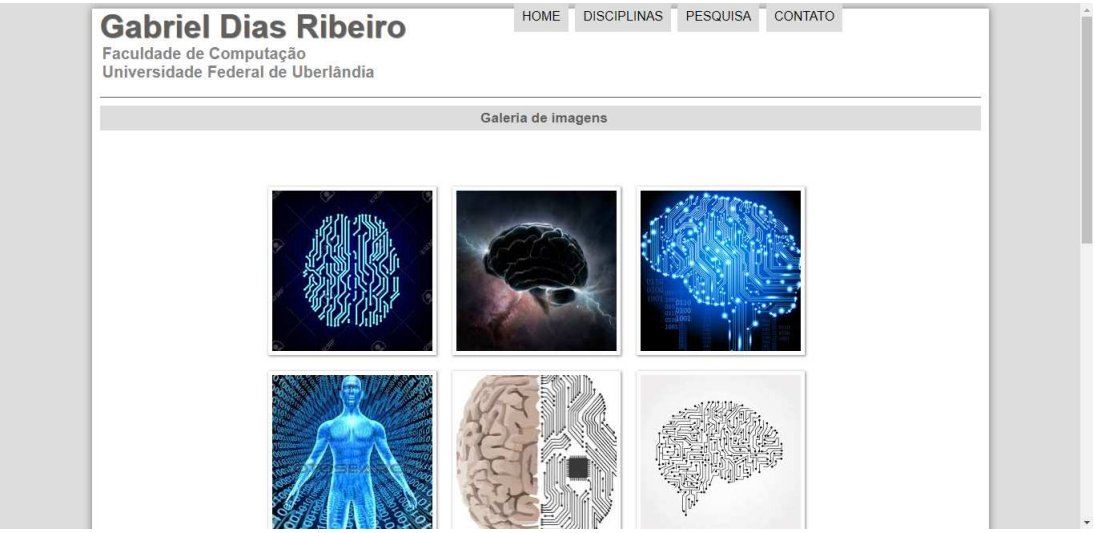

Figura 7 – Apresentação da tela PESQUISA do sistema - parte 1

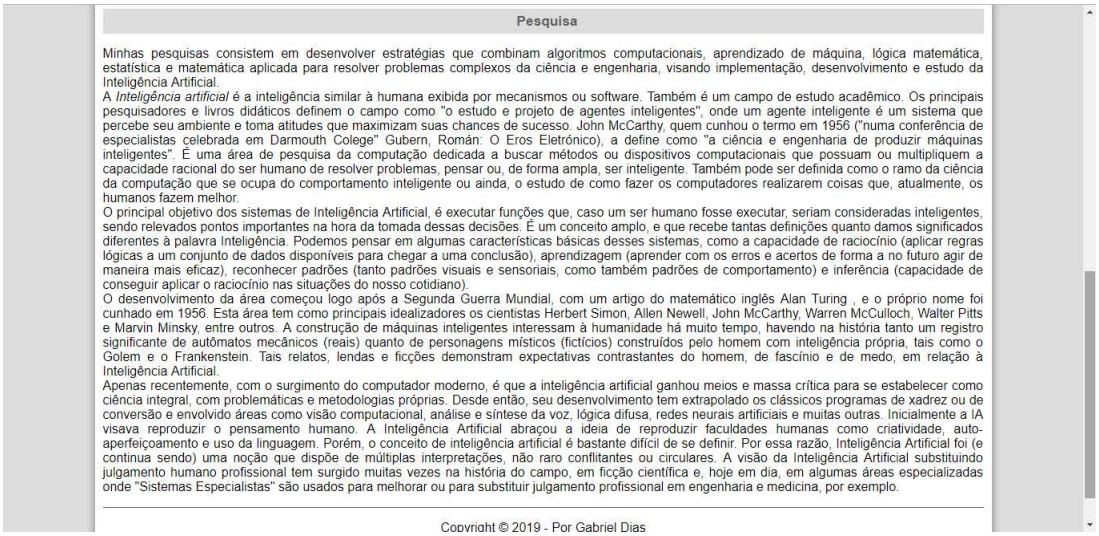

Figura 8 – Apresentação da tela PESQUISA do sistema - parte 2

5.1.4 CONTATO

A tela de CONTATO vem ao sistema para suprir a principal necessidade do docente, a de melhorar a comunicação com alunos ou interessados. A tela traz somente um formulário simples de identificação e um campo destinado ao corpo da mensagem a qual se deseja enviar ao docente.

Figura 9 – Apresentação da tela CONTATO do sistema - parte 1

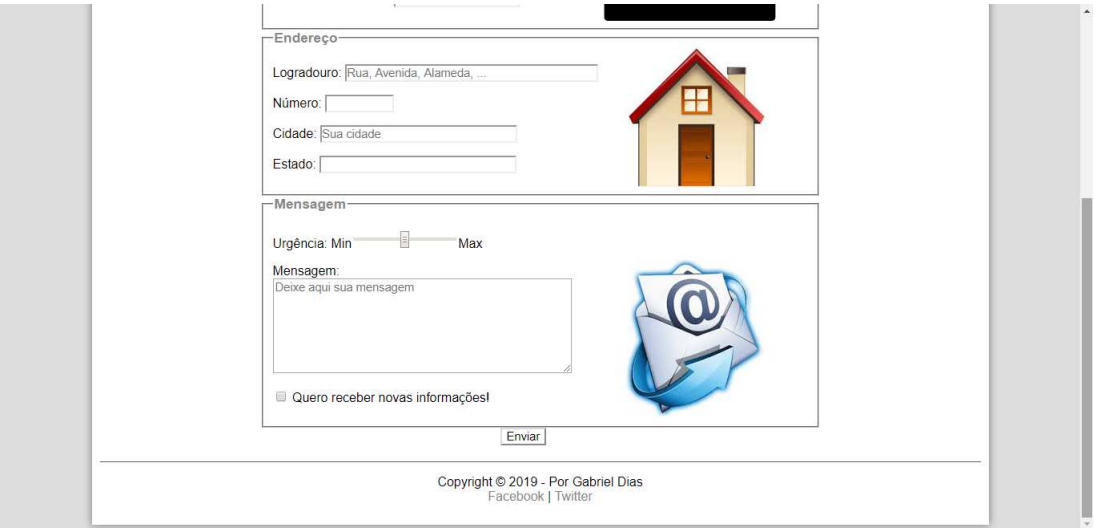

Figura 10 – Apresentação da tela CONTATO do sistema - parte 2

5.2 Testes

5.2.1 Escopo dos testes

Para verificar a aplicação e garantir que a mesma tenha sido desenvolvida conforme o padrão esperado, são estabelecidos os testes necessários para isso. Neste caso, a definição do escopo dos testes, ou seja, aquilo será testado, foi elaborada a partir dos principais requisitos funcionais do sistema.

A partir das funcionalidades é possível estabelecer os testes de caixa-preta que darão cobertura necessária para o software, visto que se sabe quais serão as entradas e os resultados esperados do sistema, mesmo não tendo conhecimento do processo interno.

A apresentação dos principais requisitos funcionais da aplicação é feita na tabela

a seguir:

Tabela 1 – Requisitos funcionais do sistema agrupados por tela

Tela	Funcionalidade
Home	O sistema deverá ser capaz de navegar entre as telas através de abas (cada uma correspondente a uma tela) localizadas no canto superior direito da aplicação
	O sistema deverá exibir um texto referente a biografia do professor em uma sessão destinada
	O sistema deverá exibir um texto para os alunos em uma sessão destinada
	O sistema deverá conter um link no rodapé com direcionamento a página do Facebook do professor.
	O sistema deverá conter um link no rodapé com direcionamento a página do Twitter do professor.
Disciplinas	O sistema deverá exibir uma tabela com informações sobre as disciplinas ministradas pelo professor nos últimos seis semestres
Pesquisa	O sistema deverá ser capaz de exibir imagens interativas em uma sessão destinada
	O sistema deverá exibir um texto referente a área de pesquisa do professor em uma sessão destinada
Contato	O sistema deverá ser capaz de realizar o preenchimento e envio de um formulário
	O sistema deverá ser capaz de apresentar um alerta caso os campos obrigatórios do formulário não tenham sido preenchidos
	O sistema deverá ser capaz de apresentar um alerta de confirmação caso seja marcado o checkbox de novas informações

Após a definição do escopo dos testes, teremos a construção dos casos de testes, onde são escolhidos dados que servirão para teste do programa e os objetivos que se deseja alcançar com os mesmos.

O produto final de trabalho com os processos destacados é o plano de testes da aplicação, onde estará contido o escopo, propósito e estratégia dos testes, assim como os casos de testes, detalhados passo a passo.

5.2.2 Testes automatizados

Para realizar a elaboração da automação dos testes planejados para a aplicação foram utilizadas as ferramentas Selenium WebDriver e JUnit, juntamente a linguagem de programação orientada a objetos Java.

O Selenium WebDriver (SELENIUMHQ, 2019) é uma ferramenta de automação de testes de sistema que provê ao desenvolvedor a execução de ações interativas com o navegador, dada sua integração direta com o mesmo. Com essa ferramenta é possível criar

scripts de testes, em diferentes linguagens de programação, responsáveis por simular ações de um usuário no browser, ou seja, é possível reproduzir, via código, ações como cliques, preenchimento de campos, seleção de menus, confirmação de mensagens de alerta, entre outros. A reprodução dos comandos pela ferramenta é possível através da interação com os elementos HTML da página web.

Já o JUnit (JUNIT, 2019) é um framework de código aberto responsável por facilitar a criação e automatização de testes de unidade em projetos de código Java. Devido a sua grande versatilidade, fornecendo uma completa aplicação para construção dos testes, sua utilização geralmente está ligada a integração com outras ferramentas. Através do JUnit é possível estruturar projetos de testes unitários ou automatizados, sendo a ferramenta, responsável pela parte de execução dos testes, análise dos resultados obtidos e estruturação do script em si.

```
@Test
public void Home_CT001() throws InterruptedException {
    if (!utils.acessarAbaSuperior("Home", "Sobre mim")) {
        return;
    }
    if (!utils.acessarAbaSuperior("Disciplinas", "Tabela de Disciplinas")) {
        return;
    }
    if (!utils.acessarAbaSuperior("Pesquisa", "Galeria de imagens")) {
        return;
    }
    if (!utils.acessarAbaSuperior("Contato", "Identificação")) {
        return;
    }
}
```

Figura 11 – Exemplo - script de teste automatizado

A estruturação do projeto foi definida avaliando os requisitos funcionais da aplicação, onde os mesmos eram agrupados por tela, assim foi definido no projeto de automação. Para cada página do sistema, existe uma classe no projeto contendo os métodos responsáveis pela execução automatizada dos comandos que compõem o teste, assim como um script que contém todos os testes da página em verificação. Pensando em um âmbito geral, há também um script responsável pela execução de toda a suíte de testes, ou seja, todos os testes desenvolvidos para a aplicação independente da página.

Para a criação dos testes automatizados foram utilizados como artefato de entrada os casos de teste contidos no documento de plano de testes. Onde para cada caso de testes existia o detalhamento dos passos necessários para sua realização foi reproduzida a automatização dos comandos.

A execução dos testes é realizada de forma bem simples e intuitiva, sendo feita através do próprio JUnit e, de acordo com a estrutura definida para o projeto, pode ocorrer de duas maneiras. O primeiro cenário aborda a execução dos testes referentes a

somente uma página da aplicação e o segundo cenário aborda a execução de todos os testes já desenvolvidos para o sistema, podendo corresponder a um teste de regressão.

Os resultados obtidos pela execução dos testes podem ser apresentados de duas maneiras, a apresentação gráfica e a apresentação consolidada dos dados, através de um documento. Na apresentação gráfica, que ocorre por meio da própria ferramenta de desenvolvimento e execução dos casos, são apresentadas somente as principais informações dos casos e do projeto em geral, além do grande destaque dado aos resultados. Já na apresentação consolidada, é gerado um documento de extensão do tipo xml através do JUnit. Este documento é responsável por conter informações referentes a execução dos testes automatizados, são apresentados dados do projeto em geral, como nome do projeto, quantidade de casos de teste executados, quantidades de falhas, entre outros, e informações relativas a cada um dos testes individualmente, como nome do caso de teste, tempo de execução, entre outros dados.

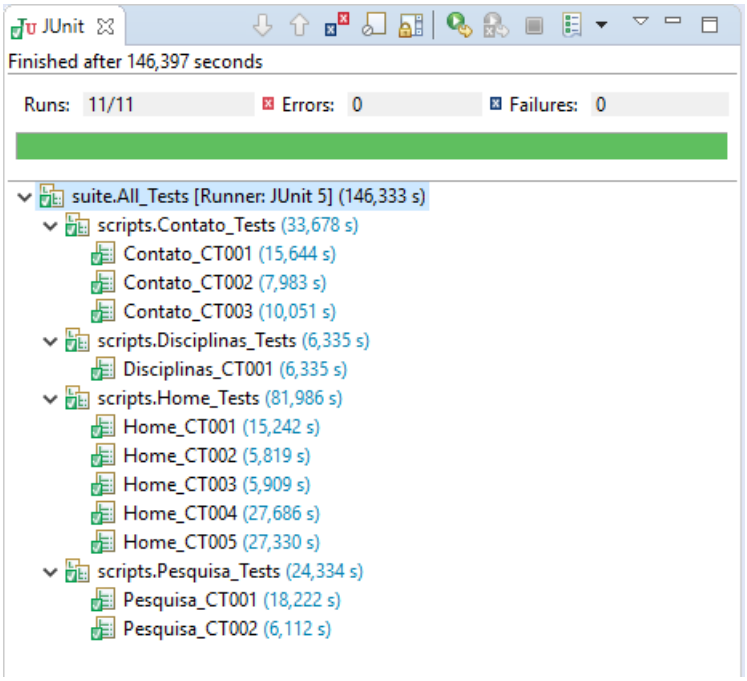

Figura 12 – Resultados da execução dos testes automatizados - Modelo visual

```
<?xml version="1.0" encoding="UTF-8"?>
<testrun ignored="0" errors="0" failures="0" started="11" tests="11" project="Automacao" name="All_Tests">
  <testsuite name="suite.All_Tests" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]" time="146.333">
    <testsuite name="scripts.Contato_Tests" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Contato_Tests]" time="33.678">
      <testcase name="Contato_CT001" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Contato_Tests]/[test:Contato_CT001
        (scripts.Contato_Tests)]" time="15.644" displayname="Contato_CT001" classname="scripts.Contato_Tests"/>
      <testcase name="Contato_CT002" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Contato_Tests]/[test:Contato_CT002
        (scripts.Contato_Tests)]" time="7.983" displayname="Contato_CT002" classname="scripts.Contato_Tests"/>
      <testcase name="Contato_CT003" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Contato_Tests]/[test:Contato_CT003
        (scripts.Contato_Tests)]" time="10.051" displayname="Contato_CT003" classname="scripts.Contato_Tests"/>
    </testsuite>
    <testsuite name="scripts.Disciplinas_Tests" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Disciplinas_Tests]" time="6.335">
      <testcase name="Disciplinas_CT001" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Disciplinas_Tests]/
        [test:Disciplinas_CT001(scripts.Disciplinas_Tests)]" time="6.335" displayname="Disciplinas_CT001" classname="scripts.Disciplinas_Tests"/>
    </testsuite>
    <testsuite name="scripts.Home_Tests" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Home_Tests]" time="81.986">
      <testcase name="Home_CT001" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Home_Tests]/[test:Home_CT001
        (scripts.Home_Tests)]" time="15.242" displayname="Home_CT001" classname="scripts.Home_Tests"/>
      <testcase name="Home_CT002" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Home_Tests]/[test:Home_CT002
        (scripts.Home_Tests)]" time="5.819" displayname="Home_CT002" classname="scripts.Home_Tests"/>
      <testcase name="Home_CT003" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Home_Tests]/[test:Home_CT003
        (scripts.Home_Tests)]" time="5.909" displayname="Home_CT003" classname="scripts.Home_Tests"/>
      <testcase name="Home_CT004" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Home_Tests]/[test:Home_CT004
        (scripts.Home_Tests)]" time="27.686" displayname="Home_CT004" classname="scripts.Home_Tests"/>
      <testcase name="Home_CT005" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Home_Tests]/[test:Home_CT005
        (scripts.Home_Tests)]" time="27.33" displayname="Home_CT005" classname="scripts.Home_Tests"/>
    </testsuite>
    <testsuite name="scripts.Pesquisa_Tests" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Pesquisa_Tests]" time="24.334">
      <testcase name="Pesquisa_CT001" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Pesquisa_Tests]/
        [test:Pesquisa_CT001(scripts.Pesquisa_Tests)]" time="18.222" displayname="Pesquisa_CT001" classname="scripts.Pesquisa_Tests"/>
      <testcase name="Pesquisa_CT002" uniqueid="[engine:junit-vintage]/[runner:suite.All_Tests]/[test:scripts.Pesquisa_Tests]/
        [test:Pesquisa_CT002(scripts.Pesquisa_Tests)]" time="6.112" displayname="Pesquisa_CT002" classname="scripts.Pesquisa_Tests"/>
    </testsuite>
  </testsuite>
</testrun>
```


Figura 13 – Resultados da execução dos testes automatizados - Modelo detalhado (Arquivo xml com consolidado das informações)

6 Testes automatizados aplicados ao DevOps

Para esse trabalho foi adotada a utilização do framework DevOps devido seu alinhamento de priorização de atividades automatizadas, porém para o mesmo poderiam ser aplicadas diversas metodologias e frameworks resultando também nos ganhos potencias obtidos pelos testes automatizados.

A aplicação dos testes automatizados a metodologia DevOps foi centralizada em dois cenários principais, o evolutivo e o corretivo. Os cenários são detalhados nos tópicos a seguir.

6.1 Cenário corretivo

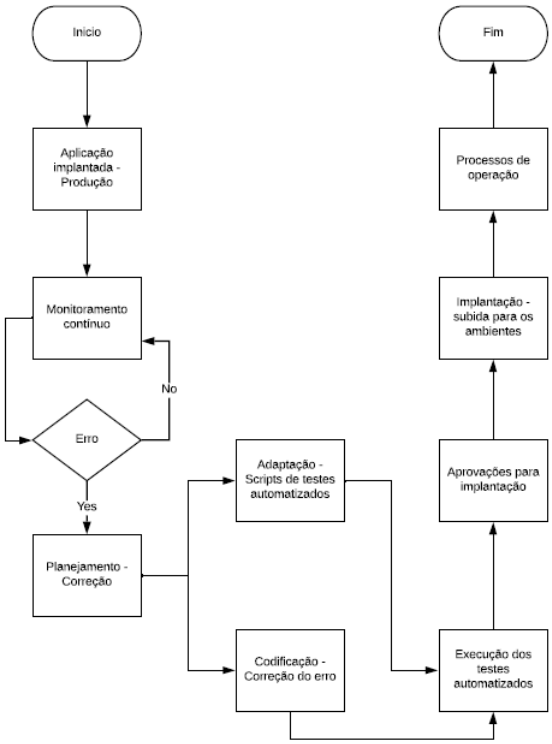

Figura 14 – Fluxograma - cenário corretivo

No cenário corretivo ou de manutenção, o ciclo DevOps inicia-se com a aplicação implantada em ambiente produtivo, ou seja, os desenvolvimentos parciais ou em totalidade encontram-se entregues. Posteriormente a esta etapa temos a etapa de monitoramento contínuo, onde o sistema é continuamente acompanhado, verificando-se a presença de erros, comportamentos inesperados ou mesmo possíveis pontos de melhorias, no momento em que algum desses pontos é identificado o sistema é encaminhado a próxima etapa do ciclo DevOps que será responsável por dar continuidade ao processo de adaptação.

Durante a etapa de planejamento, a principal atividade realizada é a análise da causa-raiz do problema, ou seja, procura-se identificar a origem do problema existente. Descoberto o motivo do comportamento inadequado da aplicação, passa-se a planejar o plano de ação para correção do erro, tendo este o objetivo de organizar as medidas a serem realizadas para solução da falha do sistema. Essa etapa é responsável também por estruturar as tarefas das próximas etapas, mapear riscos e impactos.

Após a etapa de planejamento da medida de correção, temos a execução paralela de duas tarefas, a codificação da correção do erro, ou seja, a implementação em linguagem de programação da solução do problema, e a adaptação, caso seja necessário, do script de teste automatizado. Como o script de teste automatizado é baseado na posição dos elementos na tela da aplicação, a correção do problema pode resultar na mudança da disposição dos elementos na página criando, consequentemente, a necessidade de adaptação, que será realizada nessa etapa do ciclo.

O processo seguinte a codificação da correção do erro e da adaptação do script de teste automatizado é a execução automatizada de todos os testes programados, visando garantir que a realização da tarefa de correção tenha sido efetuada com sucesso e nenhum outro problema tenha sido inserido no software.

Após a execução automatizada dos testes é analisado o resultado obtido através do documento exportado pela ferramenta. Posteriormente a análise dos resultados encontrados inicia-se a etapa de lançamento onde são realizadas as aprovações necessárias para implantação da medida corretiva.

Durante a etapa de implantação, são realizadas as subidas de versão para os ambientes da aplicação, normalmente dividido em três estruturas: ambiente de desenvolvimento, onde os programadores do software efetivamente desenvolvem e realizam testes unitários, o ambiente de homologação, ambiente mais estável e de menos liberdade para os desenvolvedores, pois são verificadas as funcionalidades e comportamentos do sistema, e o ambiente produtivo, que é o ambiente utilizado pelo usuário final da aplicação.

Figura 15 – Ambientes de um software

No ciclo DevOps o software deve passar pelos três ambientes em suas diferentes etapas, sendo que na etapa de implantação as implementações feitas são levadas ao ambiente produtivo.

Finalizada a etapa de implantação, dá-se início aos processos existentes na etapa

de operação do ciclo DevOps, ou seja, são realizadas atividades de documentação da correção, treinamentos e suporte aos usuários. Após a realização dos processos da etapa de operação o ciclo volta ao estado inicial.

6.2 Cenário evolutivo

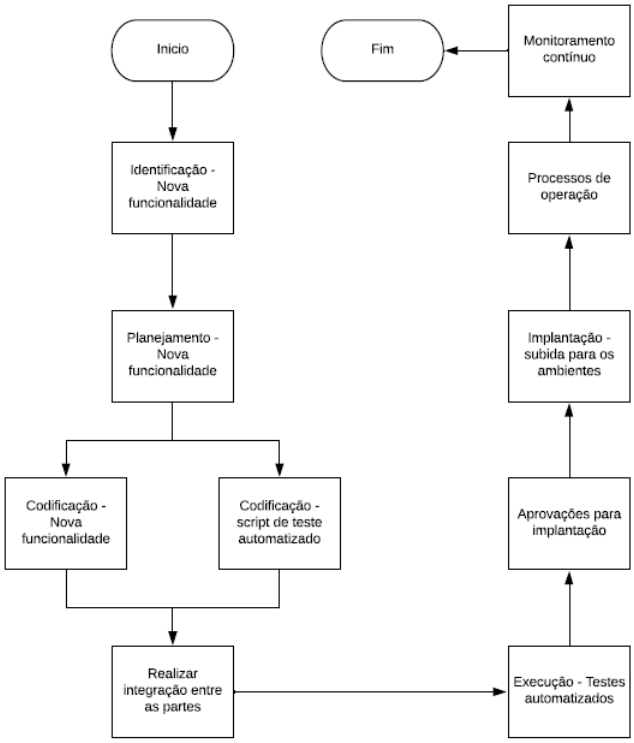

Figura 16 – Fluxograma - cenário evolutivo

Diferentemente do cenário corretivo, onde o ciclo se inicia com a aplicação em ambiente produtivo e em constante monitoramento, no cenário evolutivo o ciclo da metodologia DevOps é aplicado devido à necessidade de criação de uma tarefa evolutiva no sistema, ou seja, surge a necessidade de implementação de uma nova funcionalidade ou uma nova demanda referente a aplicação. Fato é que para satisfazer a carência do um novo recurso é necessário aplicar a metodologia, tendo como início a etapa de planejamento. Durante a etapa de planejamento são estudadas as necessidades do cliente e da aplicação, analisada a regra de negócio e o requisito funcional envolvente, é realizada também a estruturação das tarefas do ciclo, assim como a organização dos recursos existentes.

Após a etapa de planejamento da nova demanda é realizada paralelamente a execução de duas tarefas da etapa de codificação, a implementação da nova funcionalidade necessária na aplicação, tarefa essa executada pela equipe de desenvolvimento, e a implementação dos scripts de testes automatizados responsáveis por fornecer cobertura de qualidade do novo ponto do sistema, tarefa essa realizada pelo time de testes automa-

tizados. Como a demanda se trata de uma nova funcionalidade não há possibilidade de adaptação como no cenário corretivo, visto que não existe produto de trabalho já desenvolvido. A codificação dos scripts ocorre a partir de acordo com a entrega da equipe de desenvolvimento do software.

Durante a etapa de construção, tem-se como responsabilidade a preparação do projeto para a chegada da nova funcionalidade construída, ou seja, é realizada a integração entre a nova parte do produto entregue e o restante do produto, que já está desenvolvido. A atividade de integração ocorre tanto para o projeto da aplicação em si, quanto para o projeto de testes automatizados que também demandam a necessidade de integração com o que já existe desenvolvido para garantia de qualidade do produto.

Finalizada a etapa de construção, inicia-se a etapa de testes, na qual é realizada a execução dos testes automatizados, já desenvolvidos, na aplicação. Como a metodologia DevOps caminha junto a processos ágeis, a ideia principal é que a transição entre etapa de construção e execução dos testes ocorra com a menor lacuna de tempo possível, utilizando para tal, ferramentas de integração.

Ao final da etapa de testes, colhe-se o relatório de resultados da execução onde inicia-se a etapa de lançamento concedendo as aprovações necessárias, em caso de sucesso nos testes, para o lançamento da aplicação no ambiente subsequente (produtivo). Assim como ocorre no cenário corretivo, no cenário evolutivo a implementação realizada também necessita de lançamento de versões para os ambientes do software (desenvolvimento, homologação e produção), sendo que nessa etapa o projeto é configurado e instalado em seu ambiente final.

Na etapa de operação, as tarefas existentes dentro do processo se coincidem as do cenário corretivo, ou seja, é fornecido o suporte necessário para a nova funcionalidade, tais como treinamento, documentação, entre outros.

Encerrando o ciclo DevOps dentro do cenário evolutivo existe a tarefa de monitoramento. Realizada de forma contínua, são constantemente verificados e analisados os comportamentos do sistema para, em caso real de falha, reporte aos responsáveis por fornecer manutenção ao software e início do ciclo do cenário corretivo.

7 Conclusão

O seguinte trabalho de conclusão de curso teve como objetivo a demonstração da aplicação dos testes automatizados de software a metodologia DevOps de desenvolvimento de software. Produzir o mesmo foi de extrema importância para o autor apresentando e estabelecendo conceitos práticos e teóricos tão relevantes para a área em questão.

Dentro da área de tecnologia de informação tem se estabelecido cada vez mais a importância do setor de Quality Assurance ou Garantia de Qualidade, dada sua responsabilidade de gerir os processos de qualidade de um produto. Responsável pelo planejamento, execução e controle das atividades de verificação de um sistema, o setor de Quality Assurance utiliza como principal ferramenta a execução de testes, sendo esses, incorporados cada dia mais nas etapas do ciclo de desenvolvimento e não apenas estabelecidos dentro de uma única etapa ao final da implementação da aplicação.

Visando obter ganho através da redução de recursos, a automação dos testes tem se estabelecido como uma estratégia bastante viável. Com a crescente propagação dessa abordagem, as vantagens desse processo têm alcançado a atenção e curiosidade dos responsáveis pelos projetos disseminando seu estabelecimento como processo no mercado.

Acompanhando a busca por melhorias de qualidade e redução de custo e tempo, a metodologia DevOps também tem se estabelecido no mercado, propondo principalmente a integração dos setores de desenvolvimento e operação. Desse modo, entender os conceitos e sua aplicação no mercado pode resultar na junção das abordagens em busca de um mesmo objetivo.

Durante a elaboração desse trabalho foram implementadas a criação de uma aplicação web para associação a um produto de software, juntamente a criação de scripts automatizados de teste visando demonstrar os cenários principais de trabalho oriundos da junção dos conceitos de testes automatizados e DevOps, e a estrutura e integração entre as partes. O resultado dessa integração foram os fluxos de trabalho dos cenários de desenvolvimento, verificando-se a diminuição das lacunas entre as etapas do ciclo de desenvolvimento, além da organização das tarefas e ganhos significativos com automação de processos.

Por fim, é possível observar a partir do conteúdo desse trabalho que a automação de processos dentro dos modelos de desenvolvimento de software é produtiva, sendo possível a criação de novas linhas de pesquisa visando a automação de diversas outras tarefas dentro do ciclo de desenvolvimento, assim como a utilização de outras ferramentas a fim de obter ganhos, tais como Sonar (APACHE, 2019c) para validação de código, JMeter (SONARQUBE, 2019) para testes de performance/stress, entre outras.

Referências

AMAZON. [S.l.], 2019. Disponível em: <https://aws.amazon.com/pt/>. Acesso em: 19 jun. 2019. Citado na página 27.

AMAZON. *O que é DevOps?* [S.l.], 2019. Disponível em: <https://aws.amazon.com/pt/devops/what-is-devops/>. Acesso em: 05 mai. 2019. Citado na página 23.

APACHE. [S.l.], 2019. Disponível em: <https://subversion.apache.org/>. Acesso em: 19 jun. 2019. Citado na página 25.

APACHE. [S.l.], 2019. Disponível em: <https://maven.apache.org/>. Acesso em: 19 jun. 2019. Citado na página 25.

APACHE. [S.l.], 2019. Disponível em: <https://jmeter.apache.org/>. Acesso em: 19 jun. 2019. Citado na página 41.

APPIUM. [S.l.], 2019. Disponível em: <http://appium.io/>. Acesso em: 19 jun. 2019. Citado na página 26.

ATLASSIAN. [S.l.], 2019. Disponível em: <https://br.atlassian.com/software/jira>. Acesso em: 19 jun. 2019. Citado na página 25.

BASTOS, A. et al. Base de conhecimento em teste de software. *São Paulo*, 2007. Citado na página 16.

Becode. *O que é DevOps? Processo, origem e problemática!* 2017. [Online; accessed June 10, 2019]. Disponível em: <https://becode.com.br/o-que-e-devops/>. Citado 2 vezes nas páginas 6 e 24.

BERNARDO, P. C.; KON, F. A importância dos testes automatizados. *Engenharia de Software Magazine*, p. 1–7, 2008. Citado na página 13.

DEVMEDIA. *Introdução ao Modelo Cascata*. 2013. [Online; accessed June 19, 2019]. Disponível em: <https://www.devmedia.com.br/introducao-ao-modelo-cascata/29843/>. Citado 2 vezes nas páginas 6 e 17.

DEVMEDIA. *Testes automatizados com o Framework Selenium*. [S.l.], 2015. Disponível em: <https://www.devmedia.com.br/testes-automatizados-com-o-framework-selenium/32955>. Acesso em: 13 mar. 2019. Citado na página 20.

DOCKER. [S.l.], 2019. Disponível em: <https://www.docker.com/>. Acesso em: 19 jun. 2019. Citado na página 26.

GAEA. *Afinal, o que é Quality Assurance?* [S.l.], 2018. Disponível em: <https://gaea.com.br/afinal-o-que-e-quality-assurance/>. Acesso em: 23 jun. 2018. Citado na página 15.

GAEA. *Conceito: O que é DevOps?* [S.l.], 2018. Disponível em: <https://gaea.com.br/o-que-e-devops-conceito/>. Acesso em: 07 mai. 2019. Citado na página 22.

GSTI, P. *Testes, testes de software*. [S.l.], 2018. Disponível em: <https://www.portalgsti.com.br/testes-de-software/sobre/>. Acesso em: 08 mai. 2018. Citado na página 10.

IBM. *Devops, o que é*. [S.l.], 2018. Disponível em: <https://www.ibm.com/developerworks/community/blogs/rationalbrasil/entry/o_que_devops?lang=en>. Acesso em: 08 mai. 2018. Citado na página 10.

IGTI. *A importância dos Testes de Software*. [S.l.], 2018. Disponível em: <http://igti.com.br/blog/a-importancia-dos-testes-de-software/>. Acesso em: 10 oct. 2018. Citado na página 15.

JUNIT. [S.l.], 2019. Disponível em: <https://junit.org/junit5/>. Acesso em: 21 jun. 2019. Citado na página 34.

MALDONADO, B. et al. Introdução ao teste de software. *Instituto de Ciências Matemáticas e de Computação*, ICMC, v. 65, p. 1–56, 2004. Citado na página 11.

MEDIUM. *Automação de testes: o que é, quando e por que automatizar*. [S.l.], 2019. Disponível em: <https://medium.com/venturus/quais-as-razoees-para-automacao-de-testes-c177cbd9397>. Acesso em: 10 mar. 2019. Citado 2 vezes nas páginas 19 e 21.

MICROSOFT. [S.l.], 2019. Disponível em: <https://azure.microsoft.com/pt-br/>. Acesso em: 19 jun. 2019. Citado na página 27.

OBJECTIVE. *Testes automatizados, testes automatizados de software*. [S.l.], 2018. Disponível em: <https://www.objective.com.br/testes-automatizados-de-software/>. Acesso em: 09 mai. 2018. Citado na página 10.

OLIVEIRA, F. Devops em sistemas de informação. *Information Management School*, v. 1, n. 1, p. 1–61, 2017. Citado na página 12.

PRESSMAN, R.; MAXIM, B. *Engenharia de Software-8ª Edição*. [S.l.]: McGraw Hill Brasil, 2016. Citado na página 17.

QUALITY. *DEVOPS e Automação, A importância da automação dos testes no DevOps*. [S.l.], 2018. Disponível em: <http://blog.quality.com.br/importancia-da-automacao-dos-testes-no-devops-entenda-mais/>. Acesso em: 09 mai. 2018. Citado na página 10.

RAYGUN. *Massively reduce the cost of bugs with error tracking*. 2014. [Online; accessed October 10, 2018]. Disponível em: <https://raygun.com/blog/massively-reduce-the-cost-of-bugs-with-raygun-error-tracking/>. Citado 2 vezes nas páginas 6 e 16.

SACHDEVA, R. Automated testing in devops. *Optum technology Inc.*, v. 1, n. 1, p. 1–17, 2016. Citado na página 14.

SCM. [S.l.], 2019. Disponível em: <https://git-scm.com/>. Acesso em: 19 jun. 2019. Citado na página 25.

SELENIUMHQ. [S.l.], 2019. Disponível em: <https://www.seleniumhq.org/>. Acesso em: 19 jun. 2019. Citado 2 vezes nas páginas 26 e 33.

SOMMERVILLE, I.; SAWYER, P. *Requirements engineering: a good practice guide*. [S.l.]: John Wiley & Sons, Inc., 1997. Citado na página 17.

SONARQUBE. [S.l.], 2019. Disponível em: <https://www.sonarqube.org/>. Acesso em: 19 jun. 2019. Citado na página 41.

TRELLO. [S.l.], 2019. Disponível em: <https://trello.com/>. Acesso em: 19 jun. 2019. Citado na página 25.

A Casos de teste - versão 1

O documento abaixo apresenta os casos de testes responsáveis pela cobertura de qualidade da aplicação antes da elaboração de um novo caso de teste como medida de verificação da atividade corretiva.

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o correto funcionamento das abas do menu superior		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT001	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Posicionar mouse sobre a aba superior **Home**	A aba deverá mudar de cor
	3	Clicar na aba superior **Home**	A tela deverá permanecer em **Home**
	4	Posicionar mouse sobre a aba superior **Disciplinas**	A aba deverá mudar de cor
	5	Clicar na aba superior **Disciplinas**	A tela de **Disciplinas** deverá ser exibida
	6	Posicionar mouse sobre a aba superior **Pesquisa**	A aba deverá mudar de cor
	7	Clicar na aba superior **Pesquisa**	A tela de **Pesquisa** deverá ser exibida
	8	Posicionar mouse sobre a aba superior **Contato**	A aba deverá mudar de cor
	9	Clicar na aba superior **Contato**	A tela de **Contato** deverá ser exibida

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o texto presente na sessão 'Sobre mim'		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT002	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Validar exibição do texto na sessão **Sobre mim**	Texto da sessão **Sobre mim** exibido com sucesso

Figura 17 – Casos de teste referentes a tela HOME - parte 1

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o texto presente na sessão 'Aos Alunos'		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT003	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Validar exibição do texto na sessão **Aos Alunos**	Texto da sessão **Aos Alunos** exibido com sucesso

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento do link 'Facebook' presente no rodapé da página		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT004	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Clicar no link **Facebook** no rodapé da página	Página inicial do **Facebook** exibida com sucesso

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento do link 'Twitter presente no rodapé da página		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT005	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Clicar no link **Twitter** no rodapé da página	Página inicial do **Twitter** exibida com sucesso

Figura 18 – Casos de teste referentes a tela HOME - parte 2

Agrupamento	Tela Disciplinas		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar a exibição das disciplinas ministradas nos últimos seis semestres		
Caso de Teste	ID	Passos	Resultado esperado
Disciplinas_CT001	1	Acessar a página **Disciplinas** do sistema	Página exibida com sucesso
	2	Validar exibição das disciplinas do semestre **2019-1**	Disciplinas do semestre **2019-1** exibidas com sucesso
	3	Validar exibição das disciplinas do semestre **2018-2**	Disciplinas do semestre **2018-2** exibidas com sucesso
	4	Validar exibição das disciplinas do semestre **2018-1**	Disciplinas do semestre **2018-1** exibidas com sucesso
	5	Validar exibição das disciplinas do semestre **2017-2**	Disciplinas do semestre **2017-2** exibidas com sucesso
	6	Validar exibição das disciplinas do semestre **2017-1**	Disciplinas do semestre **2017-1** exibidas com sucesso
	7	Validar exibição das disciplinas do semestre **2016-2**	Disciplinas do semestre **2016-2** exibidas com sucesso

Figura 19 – Caso de teste referente a tela DISCIPLINAS

Agrupamento	Tela Pesquisa		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar a interação e exibição das imagens na sessão 'Galeria de imagens'		
Caso de Teste	**ID**	**Passos**	**Resultado esperado**
Pesquisa_CT001	1	Acessar a página **Pesquisa** do sistema	Página exibida com sucesso
	2	Posicionar o cursor do mouse sobre a **primeira imagem**	A **primeira imagem** deverá se expandir e sua legenda exibida com sucesso
	3	Posicionar o cursor do mouse sobre a **segunda imagem**	A **segunda imagem** deverá se expandir e sua legenda exibida com sucesso
	4	Posicionar o cursor do mouse sobre a **terceira imagem**	A **terceira imagem** deverá se expandir e sua legenda exibida com sucesso
	5	Posicionar o cursor do mouse sobre a **quarta imagem**	A **quarta imagem** deverá se expandir e sua legenda exibida com sucesso
	6	Posicionar o cursor do mouse sobre a **quinta imagem**	A **quinta imagem** deverá se expandir e sua legenda exibida com sucesso
	7	Posicionar o cursor do mouse sobre a **sexta imagem**	A **sexta imagem** deverá se expandir e sua legenda exibida com sucesso

Agrupamento	Tela Pesquisa		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o texto presente na sessão 'Pesquisa'		
Caso de Teste	**ID**	**Passos**	**Resultado esperado**
Pesquisa_CT002	1	Acessar a página **Pesquisa** do sistema	Página exibida com sucesso
	2	Validar exibição do texto na sessão **Pesquisa**	O texto da sessão **Pesquisa** deverá ser exibido com sucesso

Figura 20 – Casos de teste referentes a tela PESQUISA

Agrupamento	Tela Contato		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento e envio do formulário		
Caso de Teste	ID	Passos	Resultado esperado
Contato_CT001	1	Acessar a página **Contato** do sistema	Página exibida com sucesso
	2	Preencher o campo **Nome** com um nome completo válido	O campo deverá aceitar o valor inserido
	3	Preencher o campo **Senha** com uma senha válida	O campo deverá aceitar o valor inserido
	4	Preencher o campo **E-mail** com um e-mail válido	O campo deverá aceitar o valor inserido
	5	Selecionar uma opção para o campo **Sexo**	O campo deverá aceitar a opção escolhida
	6	Preencher o campo **Data de nascimento** com uma data de nascimento válida	O campo deverá aceitar o valor inserido
	7	Preencher o campo **Logradouro** com um logradouro válido	O campo deverá aceitar o valor inserido
	8	Preencher o campo **Número** com um número válido	O campo deverá aceitar o valor inserido
	9	Preencher o campo **Cidade** com uma cidade válida	O campo deverá aceitar o valor inserido
	10	Selecionar uma opção para o campo **Estado**	O campo deverá aceitar a opção escolhida
	11	Preencher o campo **Mensagem** com uma mensagem válida	O campo deverá aceitar o valor inserido
	12	Clicar no borão **Enviar**	Um alerta deverá ser exibido informando o sucesso no envio da mensagem.

Figura 21 – Casos de teste referentes a tela CONTATO - parte 1

Agrupamento	Tela Contato		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento do checkbox de novas informações		
Caso de Teste	ID	Passos	Resultado esperado
Contato_CT002	1	Acessar a página **Contato** do sistema	Página exibida com sucesso
	2	Marcar o checkbox **Quero receber novas informações!**	O sistema deverá exibir um alerta solicitando confirmação da opção
	3	Clicar em **OK** no alerta exibido pelo sistema	O sistema deverá exibir outro alerta informando sobre o envio de novas informações

Figura 22 – Casos de teste referentes a tela CONTATO - parte 2

B Resultados - versão 1

Abaixo, apresentado no modelo visual, a execução dos testes automatizados antes da elaboração de um novo caso de teste como medida de verificação da atividade corretiva.

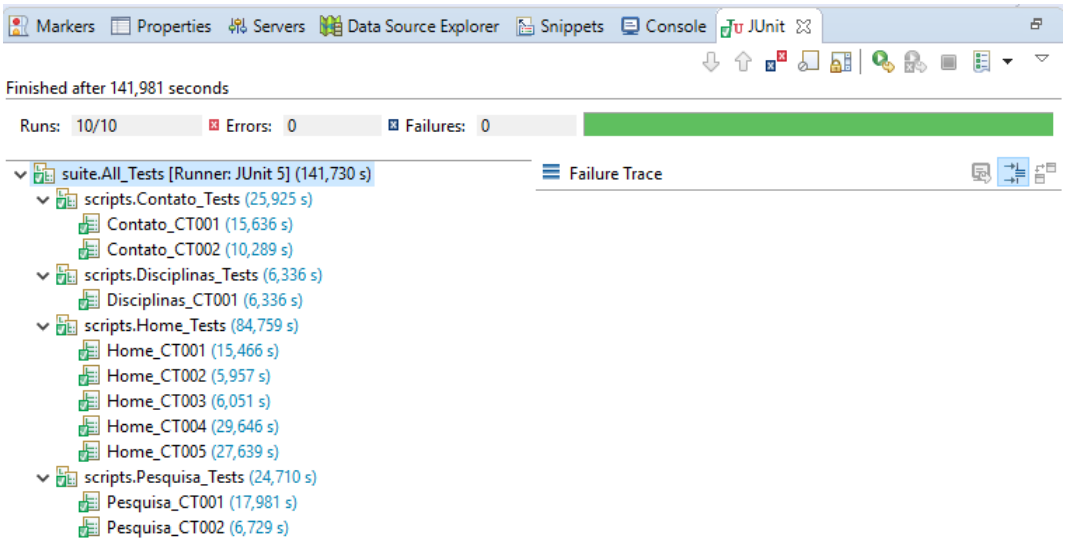

Figura 23 – Resultados da execução dos testes automatizados - versão 1

C Casos de teste - versão 2

O documento abaixo apresenta os casos de testes responsáveis pela cobertura de qualidade da aplicação após a elaboração de um novo caso de teste como medida de verificação da atividade corretiva.

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o correto funcionamento das abas do menu superior		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT001	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Posicionar mouse sobre a aba superior **Home**	A aba deverá mudar de cor
	3	Clicar na aba superior **Home**	A tela deverá permanecer em **Home**
	4	Posicionar mouse sobre a aba superior **Disciplinas**	A aba deverá mudar de cor
	5	Clicar na aba superior **Disciplinas**	A tela de **Disciplinas** deverá ser exibida
	6	Posicionar mouse sobre a aba superior **Pesquisa**	A aba deverá mudar de cor
	7	Clicar na aba superior **Pesquisa**	A tela de **Pesquisa** deverá ser exibida
	8	Posicionar mouse sobre a aba superior **Contato**	A aba deverá mudar de cor
	9	Clicar na aba superior **Contato**	A tela de **Contato** deverá ser exibida

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o texto presente na sessão 'Sobre mim'		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT002	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Validar exibição do texto na sessão **Sobre mim**	Texto da sessão **Sobre mim** exibido com sucesso

Figura 24 – Casos de teste referentes a tela HOME - parte 1

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o texto presente na sessão 'Aos Alunos'		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT003	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Validar exibição do texto na sessão **Aos Alunos**	Texto da sessão **Aos Alunos** exibido com sucesso

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento do link 'Facebook' presente no rodapé da página		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT004	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Clicar no link **Facebook** no rodapé da página	Página inicial do **Facebook** exibida com sucesso

Agrupamento	Tela Home		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento do link 'Twitter presente no rodapé da página		
Caso de Teste	ID	Passos	Resultado esperado
Home_CT005	1	Acessar a página **Home** do sistema	Página exibida com sucesso
	2	Clicar no link **Twitter** no rodapé da página	Página inicial do **Twitter** exibida com sucesso

Figura 25 – Casos de teste referentes a tela HOME - parte 2

Agrupamento	Tela Disciplinas		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar a exibição das disciplinas ministradas nos últimos seis semestres		
Caso de Teste	ID	Passos	Resultado esperado
Disciplinas_CT001	1	Acessar a página **Disciplinas** do sistema	Página exibida com sucesso
	2	Validar exibição das disciplinas do semestre **2019-1**	Disciplinas do semestre **2019-1** exibidas com sucesso
	3	Validar exibição das disciplinas do semestre **2018-2**	Disciplinas do semestre **2018-2** exibidas com sucesso
	4	Validar exibição das disciplinas do semestre **2018-1**	Disciplinas do semestre **2018-1** exibidas com sucesso
	5	Validar exibição das disciplinas do semestre **2017-2**	Disciplinas do semestre **2017-2** exibidas com sucesso
	6	Validar exibição das disciplinas do semestre **2017-1**	Disciplinas do semestre **2017-1** exibidas com sucesso
	7	Validar exibição das disciplinas do semestre **2016-2**	Disciplinas do semestre **2016-2** exibidas com sucesso

Figura 26 – Caso de teste referente a tela DISCIPLINAS

Agrupamento	Tela Pesquisa		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar a interação e exibição das imagens na sessão 'Galeria de imagens'		
Caso de Teste	**ID**	**Passos**	**Resultado esperado**
Pesquisa_CT001	1	Acessar a página **Pesquisa** do sistema	Página exibida com sucesso
	2	Posicionar o cursor do mouse sobre a **primeira imagem**	A **primeira imagem** deverá se expandir e sua legenda exibida com sucesso
	3	Posicionar o cursor do mouse sobre a **segunda imagem**	A **segunda imagem** deverá se expandir e sua legenda exibida com sucesso
	4	Posicionar o cursor do mouse sobre a **terceira imagem**	A **terceira imagem** deverá se expandir e sua legenda exibida com sucesso
	5	Posicionar o cursor do mouse sobre a **quarta imagem**	A **quarta imagem** deverá se expandir e sua legenda exibida com sucesso
	6	Posicionar o cursor do mouse sobre a **quinta imagem**	A **quinta imagem** deverá se expandir e sua legenda exibida com sucesso
	7	Posicionar o cursor do mouse sobre a **sexta imagem**	A **sexta imagem** deverá se expandir e sua legenda exibida com sucesso

Agrupamento	Tela Pesquisa		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o texto presente na sessão 'Pesquisa'		
Caso de Teste	**ID**	**Passos**	**Resultado esperado**
Pesquisa_CT002	1	Acessar a página **Pesquisa** do sistema	Página exibida com sucesso
	2	Validar exibição do texto na sessão **Pesquisa**	O texto da sessão **Pesquisa** deverá ser exibido com sucesso

Figura 27 – Casos de teste referentes a tela PESQUISA

Agrupamento	Tela Contato		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento e envio do formulário		
Caso de Teste	ID	Passos	Resultado esperado
Contato_CT001	1	Acessar a página **Contato** do sistema	Página exibida com sucesso
	2	Preencher o campo **Nome** com um nome completo válido	O campo deverá aceitar o valor inserido
	3	Preencher o campo **Senha** com uma senha válida	O campo deverá aceitar o valor inserido
	4	Preencher o campo **E-mail** com um e-mail válido	O campo deverá aceitar o valor inserido
	5	Selecionar uma opção para o campo **Sexo**	O campo deverá aceitar a opção escolhida
	6	Preencher o campo **Data de nascimento** com uma data de nascimento válida	O campo deverá aceitar o valor inserido
	7	Preencher o campo **Logradouro** com um logradouro válido	O campo deverá aceitar o valor inserido
	8	Preencher o campo **Número** com um número válido	O campo deverá aceitar o valor inserido
	9	Preencher o campo **Cidade** com uma cidade válida	O campo deverá aceitar o valor inserido
	10	Selecionar uma opção para o campo **Estado**	O campo deverá aceitar a opção escolhida
	11	Preencher o campo **Mensagem** com uma mensagem válida	O campo deverá aceitar o valor inserido
	12	Clicar no borão **Enviar**	Um alerta deverá ser exibido informando o sucesso no envio da mensagem.

Figura 28 – Casos de teste referentes a tela CONTATO - parte 1

Agrupamento	Tela Contato		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento do alerta de campos obrigatórios		
Caso de Teste	ID	Passos	Resultado esperado
Contato_CT002	1	Acessar a página **Contato** do sistema	Página exibida com sucesso
	2	Clicar no borão **Enviar** sem preencher nenhum campo do formulário	O sistema deverá exibir um alerta informando que todos os campos devem ser preenchidos.

Agrupamento	Tela Contato		
Pré-condições	Acesso a aplicação		
Objetivo	Verificar o funcionamento do checkbox de novas informações		
Caso de Teste	ID	Passos	Resultado esperado
Contato_CT003	1	Acessar a página **Contato** do sistema	Página exibida com sucesso
	2	Marcar o checkbox **Quero receber novas informações!**	O sistema deverá exibir um alerta solicitando confirmação da opção
	3	Clicar em **OK** no alerta exibido pelo sistema	O sistema deverá exibir outro alerta informando sobre o envio de novas informações

Figura 29 – Casos de teste referentes a tela CONTATO - parte 2

D Resultados - versão 2

Abaixo, apresentado no modelo visual, a execução dos testes automatizados após a elaboração de um novo caso de teste como medida de verificação da atividade corretiva.

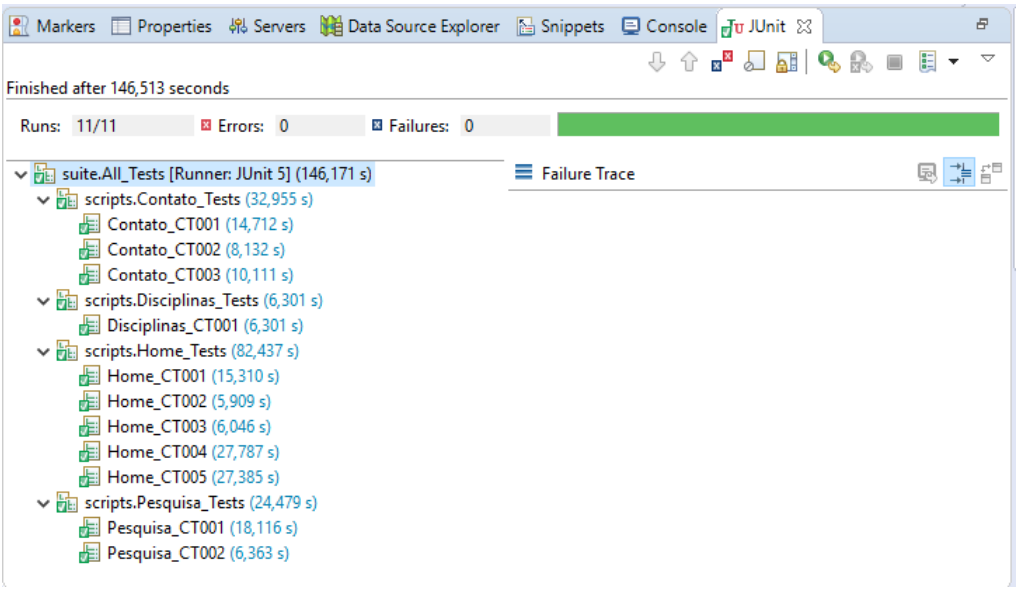

Figura 30 – Resultados da execução dos testes automatizados - versão 2

E Scripts de teste

Nas figuras abaixo são apresentadas a estrutura geral dos casos de teste referentes a tela CONTATO e a estrutura interna dos metódos utilizados na construção dos scripts dessa página.

```
@Test
public void Home_CT001() throws InterruptedException {
    if (!utils.acessarAbaSuperior("Home", "Sobre mim")) {
        return;
    }
    if (!utils.acessarAbaSuperior("Disciplinas", "Tabela de Disciplinas")) {
        return;
    }
    if (!utils.acessarAbaSuperior("Pesquisa", "Galeria de imagens")) {
        return;
    }
    if (!utils.acessarAbaSuperior("Contato", "Identificação")) {
        return;
    }
}
```

Figura 31 – Estrutura dos scripts de teste referentes a tela CONTATO

```java
/**
 * Metodo para marcar o checkbox de novas informacoes e aceitar os alerts
 *
 * @return void
 * @author gdiasrib
 * @throws InterruptedException
 */
public void marcarCheckboxInformacoes(String msgEsperada1, String msgEsperada2) throws InterruptedException {
    driver.findElement(By.xpath("//input[@name='inf']")).click();
    Thread.sleep(2000);
    String msgRecebida1 = driver.switchTo().alert().getText();
    if (!msgEsperada1.equals(msgRecebida1)) {
        return;
    }
    driver.switchTo().alert().accept();
    Thread.sleep(2000);
    String msgRecebida2 = driver.switchTo().alert().getText();
    if (!msgEsperada2.equals(msgRecebida2)) {
        return;
    }
    driver.switchTo().alert().accept();
}

/**
 * Metodo para clicar no botao enviar do formulario e aceitar o alert
 *
 * @return msgAlerta
 * @author gdiasrib
 * @throws InterruptedException
 */
public String clicarEnviar() throws InterruptedException {
    driver.findElement(By.xpath("//input[@value='Enviar']")).click();
    Thread.sleep(2000);
    String msgRecebida = driver.switchTo().alert().getText();
    driver.switchTo().alert().accept();
    return msgRecebida;
}
```

Figura 32 – Estrutura interna dos metódos utilizados nos scripts de teste referentes a tela CONTATO - parte 1

```java
public String preencherFormulario(String nome, String senha, String email, String sexo, String data,
        String logradouro, String num, String cidade, String estado, String mensagem) throws InterruptedException {
    driver.findElement(By.xpath("//input[@name='nome']")).sendKeys(nome);
    Thread.sleep(500);
    driver.findElement(By.xpath("//input[@name='senha']")).sendKeys(senha);
    Thread.sleep(500);
    driver.findElement(By.xpath("//input[@name='email']")).sendKeys(email);
    Thread.sleep(500);
    if (sexo.equalsIgnoreCase("Masculino")) {
        driver.findElement(By.xpath("//input[@id='masc']")).click();
    } else {
        driver.findElement(By.xpath("//input[@id='fem']")).click();
    }
    Thread.sleep(500);
    driver.findElement(By.xpath("//input[@name='nasc']")).sendKeys(data);
    Thread.sleep(500);
    driver.findElement(By.xpath("//input[@name='rua']")).sendKeys(logradouro);
    Thread.sleep(500);
    driver.findElement(By.xpath("//input[@name='num']")).sendKeys(num);
    Thread.sleep(500);
    driver.findElement(By.xpath("//input[@name='cid']")).sendKeys(cidade);
    Thread.sleep(500);
    driver.findElement(By.xpath("//input[@name='est']")).sendKeys(estado);
    Thread.sleep(500);
    driver.findElement(By.xpath("//textarea[@name='msg']")).sendKeys(mensagem);
    Thread.sleep(500);
    return clicarEnviar();
}
```

Figura 33 – Estrutura interna dos metódos utilizados nos scripts de teste referentes a tela CONTATO - parte 2