INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA SUL-RIO-GRANDENSE - CÂMPUS PASSO FUNDO
CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET

IEDA ROSANA KOLLING WIEST

Benefícios da automação de testes com o uso da ferramenta Katalon Studio

Orientador: André Rollwagen

PASSO FUNDO

2021

IEDA ROSANA KOLLING WIEST

Benefícios da automação de testes com o uso da ferramenta Katalon Studio

Monografia apresentada ao Curso de Tecnologia em Sistemas para Internet do Instituto Federal Sul-rio-grandense, Câmpus Passo Fundo, como requisito parcial para a obtenção do título de Tecnólogo em Sistemas para Internet.

Orientador (a): André Rollwagen

PASSO FUNDO

2021

À Deus, por sempre iluminar meu caminho
e a minha família, pela compreensão e o estímulo
em todos os momentos

AGRADECIMENTOS

Agradeço, primeiramente, a Deus, por sempre iluminar meu caminho por todos esses anos e não me deixar desistir mostrando que, apesar das dificuldades, tudo acontece em seu tempo certo.

Toda minha gratidão ao meu esposo Roberto, pela ajuda e apoio nos momentos em que eu mais precisei, e minha filha Gabriela, pelo carinho e amor incondicional que sempre me estimularam quando pensei em desistir, eu nunca teria conseguido sem o apoio de vocês. Vocês são o motivo do meu empenho e dedicação.

Aos meus pais, que apesar de todas as dificuldades, e mesmo de longe sempre me incentivaram na realização do meu sonho.

Agradeço também a todos os professores do curso que colaboraram com seus ensinamentos e me incentivaram ao longo desse processo de aprendizado em especial ao professor André Rollwagen que me orientou e contribuiu com seu conhecimento para a conclusão deste trabalho.

Minha gratidão a todos que de alguma forma me auxiliaram e contribuíram para a conclusão desta etapa.

RESUMO

O tema proposto neste trabalho está ligado à melhoria da qualidade dos sistemas especificamente no processo de testes manuais e dos diferentes tipos de testes funcionais. A automação de teste é importante, pois em alguns casos reduz tempo e trabalho dos analistas, onde estes irão dedicar maior tempo em análises mais complexas e com maior valor agregado, deixando os testes mais simples para serem realizados automaticamente. Uma falha em um sistema pode gerar muitos prejuízos para uma organização, dependendo do produto e de sua abrangência. O objetivo deste trabalho foi avaliar os ganhos e dificuldades que o processo de automação de testes funcionais em aplicações web possui. Realizar a automação de diferentes tipos de teste funcional no sistema de inscrição de eventos do Instituto Federal Sul-Rio-Grandense – IFSUL Câmpus Passo Fundo. Para automação do processo de testes de software utilizou-se a ferramenta Katalon Studio. Por fim, realizou-se uma comparação entre os testes manuais e os testes automatizados mostrando através dessa execução as vantagens e possíveis desvantagens na automatização em cada um dos testes executados.

Palavras-chave: Automação de testes. Katalon Studio. Testes funcionais.

ABSTRACT

The theme proposed in this work is linked to the improvement of the quality of the systems specifically in the process of manual tests and of the different types of functional tests. Test automation is important, as in some cases it reduces the time and work of analysts, where they will dedicate more time to more complex analyzes and with greater added value, making tests simpler to be performed automatically. A failure in a system can generate millions in losses for an organization, depending on the product and its scope, these failures are unacceptable, as they can be fatal. The objective of this work was to evaluate the gains and difficulties that the process of automation of functional tests in web applications has. Perform the automation of different types of functional testing in the event registration system of the Instituto Federal Sul-Rio-Grandense – IFSUL Câmpus Passo Fundo. To automate the software testing process, the Katalon Studio tool was used. Finally, a comparison was made between manual tests and automated tests, showing through this execution the advantages and possible disadvantages of automation in each of the tests performed.

Key words: Test automation. Katalon Studio. Functional tests.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API Application Programming Interface (interface de programação de aplicação)
BDD – Behavior Driven Development (Desenvolvimento Guiado por Comportamento).
CSV -comma-separated-values
IDE – Integrated Development Environment (Ambiente de Desenvolvimento Integrado)
IFSUL – Instituto Federal Sul-rio-grandense.
ISTQB – International Software Testing Qualifications Board.
QAs – Quality Analyst (Analista de Qualidade)
SISU – Sistema de Seleção Unificada.
SQL- Standard Query Language
Web – World Wide Web

Sumário

1 INTRODUÇÃO

A qualidade dos sistemas desenvolvidos é uma preocupação constante nos times de desenvolvimento de software. Teste de software é uma área que tem crescido muito nos últimos tempos, principalmente a automação desses testes.

A (NBR ISO 9000:2005) define que a "qualidade é o grau no qual um conjunto de características inerentes satisfaz aos requisitos" e ainda pode-se afirmar que se algum produto ou serviço atende aos requisitos especificados, este mesmo produto ou serviço possui a qualidade que foi esperada para ele.

A qualidade pode ser medida através do grau de satisfação em que as pessoas avaliam determinado produto ou serviço. Contudo, esse produto ou serviço pode ter qualidade para algumas pessoas e para outras nem tanto, ou seja, a qualidade é algo subjetivo (NBR ISO 9000:2005).

Quando se fala em qualidade de software, Pressman (2016) recomenda que ela seja de fato implementada e não somente uma ideia ou desejo que uma organização tenha para seus projetos. Ainda segundo o autor, a qualidade de software consiste, em um conjunto de requisitos, por um produto que atende as necessidades dos clientes, cumprindo todos os requisitos solicitados da melhor forma possível.

Segundo Bastos et. al (2012), até pouco tempo eram realizados somente os testes unitários, em que os testes eram executados pelos próprios programadores. Por conta disso, não permitiam que muitos problemas fossem encontrados. Estes problemas apareciam quando o sistema já estava em ambiente de produção, onde a correção tem seu custo elevado e a credibilidade do sistema abalada.

Segundo o guia Software Engeneering Body of Knowledge (SWEBOK, 2013), os testes são uma atividade desempenhada para avaliação da qualidade do produto e para melhorar, identificando defeitos e problemas.

Um programa é testado com o propósito de encontrar defeitos a serem corrigidos ou minimizados. Cada cenário de teste requer um resultado esperado e a partir da definição de “certo” se determina se o programa está em conformidade com as especificações e se ele serve ao seu propósito (TSUI; KARAM, 2013).

Em geral, testar um sistema é um trabalho que exige muito esforço. O tempo de execução dessas tarefas pode ser encurtado por meio do uso de ferramentas apropriadas. Um analista de testes pode usar uma variedade de ferramentas, como por exemplo, um software que executa um script automaticamente a partir de um caso de teste gerado, sem necessidade de repetição manual ou um gerador de massa de dados de teste. Essas ferramentas são úteis para aumentar a eficiência e eficácia dos testes (NAIK; TRIPATHY, 2008).

Ao entrar na definição de qualidade, percebe-se quanto complexa ela é, e os projetos de desenvolvimento de softwares podem não alcançar resultados satisfatórios quando se trata do ciclo de vida do software e a qualidade do sistema. Um dos grandes desafios vem sendo conseguir cumprir prazos e custos estabelecidos e alcançar a qualidade desejada.

Assim, diante disso tem-se a seguinte questão de pesquisa: Quais são os benefícios da automação de diferentes tipos de testes e o impacto ao escolher uma ferramenta ao invés de realizar testes manuais e quando a automatização se torna vantajosa?

Pretende-se, portanto, a partir do desenvolvimento de um roteiro de diferentes tipos de teste automatizado aplicar em um sistema já existente e por fim avaliar quando é vantagem automatizar cada um dos tipos executados.

Em consequência disso este trabalho justifica-se pela importância do processo de automação de testes funcionais de software, os quais, quando aplicados corretamente, podem trazer inúmeros benefícios para o desenvolvimento de um software otimizando seu tempo e melhorando sua qualidade.

O objetivo deste trabalho é avaliar a importância da automação de teste funcional em diferentes tipos de testes. Demonstrar os benefícios da automação com uso da ferramenta Katalon Studio no sistema de Eventos do Instituto Federal-Câmpus Passo Fundo e fazer uma comparação dos métodos manuais de teste, mostrando como a ferramenta pode contribuir para obtenção da qualidade final esperada do produto.

Esse trabalho está organizado da seguinte forma: no capitulo 2 é apresentado referencial teórico, no capitulo 3 os trabalhos relacionados, capitulo 4 testes utilizando o Katalon Studio por fim, no capitulo 5 as considerações finais.

2 REFERÊNCIAL TEÓRICO

Neste capítulo são abordados conceitos e tecnologias que são empregadas no projeto.

2.1 Qualidade de Software

A demanda e a busca constante por um produto com a qualidade desejada é um grande desafio.

Quando se trata em qualidade de software refere-se em entregas bem sucedidas dentro do prazo proposto e sistemas que atendam os requisitos levantados e ainda, levando em conta sua usabilidade, confiabilidade e satisfação do cliente.

Conforme Andrade (2015), a qualidade tem sido um fator de diferenciação no mercado atual. A exigência por qualidade tem aumentado em todas as áreas e afetado também a indústria de software. Com os computadores, cada vez mais, fazendo parte da vida das pessoas, a produção de softwares vem aumentando e tornando os clientes mais exigentes. A grande exigência dos clientes por melhores softwares tem obrigado os desenvolvedores a aperfeiçoarem o seu produto final para continuarem competindo no mercado.

O autor destaca também que, ao realizar o aprimoramento da qualidade do processo de software, consequentemente aumentam-se as chances de adquirir um produto final mais adequado e que esteja a altura das expectativas do cliente. Para ser alcançado esse aprimoramento existem inúmeras abordagens que descrevem como a empresa pode avaliar o estado atual e com o resultado dessa avaliação procurar melhorar seu processo de maneira ideal e objetiva.

Segundo Sommerville (2011), a qualidade do software está diretamente relacionada a qualidade do processo de desenvolvimento de software onde parte do princípio que a qualidade do produto é intimamente relacionada ao processo de produção do mesmo. Define ainda que os padrões de software desempenham um

papel muito importante no gerenciamento de qualidade de software, onde uma parte muito importante é a definição de padrões que devem ser aplicados no processo de desenvolvimento.

Conforme a definição apresentada na Figura 1, após os padrões serem selecionados para uso, processos específicos de projeto devem ser definidos para monitorar o uso dos padrões e verificar se eles foram seguidos.

Figura 1 - Qualidade baseada em processos

Fonte: SOMMERVILLE (2011, p. 458)

A qualidade também pode ser vista na apresentação e usabilidade do software se ele é fácil de usar e de compreender.

No idioma inglês, a usabilidade é normalmente definida como a “capacidade de ser usada”, implicitamente a capacidade de uma entidade a ser usada. Conforme as normas de qualidade de software - (ISO 9241–11 (2015)), a usabilidade é a eficiência, eficácia e satisfação com a qual os públicos do produto alcançam objetivos em um determinado ambiente.

Usabilidade é o atributo de qualidade para avaliar a facilidade de uso de uma interface. A palavra “usabilidade” também se refere a métodos para melhorar a facilidade de uso durante o processo de design - (JAKOB; NIELSEN, 1999).

Segundo (JAKOB, 1999), a usabilidade é definida por cinco componentes:

1. Facilidade de aprendizado: o quão fácil é para os usuários completar tarefas básicas a primeira vez que eles utilizam a interface;
2. Eficiência de uso: uma vez que os usuários aprenderam a utilizar a interface, quão rápido eles conseguem realizar as tarefas;
3. Facilidade de memorização: quando os usuários retornam à interface depois de um período sem usar, conseguem reutilizar com facilidade;
4. Erros: quantos erros os usuários cometem, quão graves são esses erros e qual a dificuldade para corrigi-los;
5. Satisfação subjetiva: a interface é agradável.

Seguindo os fatores de Sommerville (2011), a importância do software ser confiável, o termo confiança foi proposto por Laprie(1995) para cobrir os sistemas relacionados com atributos de disponibilidade, confiabilidade, segurança e proteção. Ainda segundo ele a confiança de um sistema é uma propriedade do sistema que reflete a sua fidedignidade. Fidedignidade aqui significa, essencialmente, o grau de confiança de um usuário no funcionamento esperado pelo sistema.

Ainda, quando se fala que o produto final foi entregue com a qualidade desejada pode-se incluir todas as atividades que direcionam e controlam uma organização em relação à qualidade. Entre outras atividades, a gestão da qualidade inclui a garantia de qualidade e o controle de qualidade. A garantia de qualidade é tipicamente focada na adesão a processos adequados, a fim de fornecer confiança de que os níveis apropriados de qualidade serão alcançados. Quando os processos são realizados adequadamente, os produtos de trabalho criados por esses processos são geralmente de maior qualidade, o que contribui para a prevenção de defeitos. Além disso, o uso de análise de causa raiz para detectar e remover as causas de defeitos, juntamente com a aplicação adequada das conclusões de reuniões retrospectivas para melhorar os processos, é importante para garantir a qualidade efetiva.

Para conseguir entregas de software com melhor qualidade, cada vez mais as empresas de software e usuários buscam soluções que proporcionam entregas de software especializadas no acompanhamento, elaboração, desenvolvimento, teste e homologação de sistemas. Para evitar falhas, é necessário desde o início do projeto, proceder com seu planejamento, pois grande parte do desenvolvimento com qualidade

de um sistema está associada aos testes, acertos, validações e homologações para de fato o produto ser entregue (GANDARA, 2012, p.11-12).

2.2 Teste de software

Para Pressman (2016), os testes de software são um grupo de atividades construídas e aplicadas metodicamente antes mesmo do desenvolvimento do software estar concluído, com o propósito de verificar se o produto realiza as ações requisitadas pelo cliente, desde sua interface e layout até mesmo se o código fonte foi implementado corretamente.

> *O teste é destinado a mostrar que um programa faz o que é proposto a fazer e para descobrir os defeitos do programa antes do uso. Quando se testa o software, o programa é executado usando dados fictícios. Os resultados do teste são verificados à procura de erros, anomalias ou informações sobre os atributos não funcionais do programa. (SOMMERVILLE, 2011, p. 144)*

Segundo Sommerville (2011) geralmente o sistema de software comercial tem de passar por três estágios de teste:

1. Teste em desenvolvimento, em que o sistema é testado durante o processo de desenvolvimento para descobrir bugs e defeitos;
2. Teste de release em que uma equipe de teste, independente testa uma versão completa do sistema antes de ele ser liberado ao usuário verificando assim se atende aos requisitos definidos:
3. Teste de usuário ou aceitação, em que os usuários testam o sistema em seu próprio ambiente.

Nos próximos tópicos são abordados os conceitos sobre os tipos de testes de software.

2.3 Tipos de Teste

Segundo o ISTQB (International Software Testing Qualifications Board) (Syllabus CTFL-2018) os testes podem ser classificados em função de:

1. Testes durante o ciclo de vida do produto.
    - Teste de integração: Caracteriza-se em testar tanto testes de integração entre componentes quanto de integração com outros sistemas;
    - Teste de sistema: Trata o comportamento de todo do sistema definido no levantamento de requisitos. Abrange os requisitos funcionais e não funcionais. Neste tipo de teste o ambiente de teste deve ser o mais semelhante possível ao ambiente de produção, a fim de potencializar a identificação de falhas específicas de ambiente;
    - Teste de aceite: Muitas vezes também chamado como teste beta ou teste de campo, é realizado com possíveis clientes ou usuários antes do software ser disponibilizado para comercialização ou para uso efetivo;
    - Teste de manutenção: Este teste é realizado quando sistemas ficam por muito tempo em funcionamento e onde são realizadas pequenas modificações de emergência ou há migrações e integrações com outros sistemas.

2. Testes conforme o objetivo:
    - Teste funcional: Esse tipo de teste é caracterizado por possuir o objetivo de validar se as funcionalidades disponíveis pelo software estão de acordo com o especificado pelo cliente na documentação do produto, por exemplo, requisitos de negócios e requisitos técnicos. O teste funcional considera o comportamento externo do software (teste de caixa preta);
    - Testes não funcionais. Refere-se em testar qualidades técnicas do sistema, por exemplo, testes de carga, teste de segurança. Os testes não funcionais podem ser realizados em todos os níveis de teste;
    - Teste estrutural. É o teste que visa validar a estrutura desenvolvida na parte do código, como por exemplo, fazendo uso da ferramenta jUnit para testar classes ou métodos desenvolvidos em Java (Syllabus ,CTFL,2018);
    - Teste de regressão. O teste de regressão é quando tem alteração ou inclusão de alguma funcionalidade nova em um sistema já em funcionamento, onde na maioria das vezes é necessário reteste do

sistema como um todo para garantir que todas as funcionalidades não foram impactadas.

2.4 Planos de Teste

No plano de teste consta todo o planejamento do teste envolvendo todas as atividades que definem os propósitos e a abordagem do teste, para se chegar a um objetivo previamente definido pelo contexto.

Como por exemplo, um plano de testes é especificar técnicas e tarefas de teste adequadas e formular um cronograma de teste para cumprir um prazo (Syllabus CTFL, 2018).

Ainda de acordo Rios e Moreira, (2013), um bom planejamento de teste é realizado por meio da análise e leitura das especificações e requisitos do sistema. Estes requisitos são necessários para definição da estratégia de testes e para estruturar o plano de teste, tendo em vista a criação dos casos de testes que serão executados pelos scripts ou procedimento de testes.

2.5 Casos de Teste

Após o planejamento de teste realizado uma das tarefas do analista de teste é pensar em quais são os tipos de casos de teste para uma determinada situação.

Segundo (Syllabus, CTFL, 2018), nos casos de teste estão descritos as condições a serem testadas, onde estas são compostas por valores de entrada, restrições para a sua execução e um resultado ou comportamento esperado. O processo de criação de casos de testes pode também ajudar a encontrar falhas de requisitos, ou seja, identificar cenários não tratados pelos requisitos do sistema.

Segundo Lages:

> Caso de teste é a representação de uma instância de teste para aquela funcionalidade que está sendo avaliada, de acordo com as entradas e o resultado esperado é percorrido um determinado caminho no algoritmo para que se avalie aquela situação. Logo, existem duas categorias de ferramentas, uma que gera conjuntos de casos de teste e a outra que os executam e verificam o resultado, ambos de modo automático *(LAGES, 2010, p.146).*

Um desafio específico é geralmente a definição das pós-condições do teste ou o resultado esperado de um teste. Na maioria das vezes os testadores estão preocupados não apenas com as saídas na tela, mas também com dados e pós-condições apresentadas. Se a base de teste estiver claramente definida, o resultado, teoricamente, deve ser simples. Contudo, as bases de teste são muitas vezes vagas, contraditórias, dificultando os resultados dos testes.

Conforme o exemplo apresentado na Figura 2 pode-se ter uma ideia através da descrição do passo a passo e da composição de um caso de teste executado de forma manual.

Figura 2 - Descrição passo a passo exemplo caso de teste manual

Caso de Teste	**Login sem informar usuário**	
Autor:	ieda.kolling	
Objetivo do Teste::		
1. Histórico: - 13/03/2014 - Ieda R. kolling Wiest - Criação do caso de teste. **2. Itens de Teste:** - EF: 2.5.1; - ET: 2.7.1. **3. Objetivo do Teste:** Verificar o comportamento ao tentar realizar login sem informar o usuário. **4. Pré-condições:** - Possuir iPad; - Possuir acesso URL do Checklist versão iPad; - Possuir uma senha válida. **5. Pós-condições:** O login não deve ser efetuado e deverá exibir a mensagem: "O(s) seguinte(s) campo(s) deve(m) ser preenchido(s): Login". **6. Documentação:** - EF - Checklist para iPad - v1.7; - ET - Checklist para iPad - v1.7.		
#:	Ações do Passo:	Resultados Esperados::
1	Acessar a url do Checklist indicada nas pré-condições;	Sistema exibe a tela de Login com os campos "Login" e "Senha" em branco;
2	Informar no campo "Senha" a senha indicada nas pré-condições contendo pelo menos 6 caracteres alfa numéricos;	Sistema aceita o valor;
3	Clicar em [Fazer Login];	Sistema deverá exibir a mensagem: "O(s) seguinte(s) campo(s) deve(m) ser preenchido(s): Login";
4	Clicar em [OK];	Sistema deverá permanecer na interface de Login com o campo "Senha" preenchido.
Tipo de Execução:	Manual	
Navegador para Execução:		
Requisitos	Nenhum	
Palavras-chave:	Nenhum	

Fonte: Da autora, 2014.

2.6 Automação de Teste

Segundo Sommerville (2011), na automatização, os casos de testes e seus cenários são todos codificados no software que estiver sendo utilizado e será realizada toda vez que for feita uma mudança na aplicação. Ainda, destaca que a automatização tem aumentado significativamente e destaca que os testes nunca poderão ser totalmente automatizados, pois o teste automatizado só irá fazer o que lhe foi proposto.

Muitas empresas possuem testadores para executar os testes manualmente, o que implica a necessidade de uma ou mais pessoas dedicando, planejando, efetuando e validando os resultados todas as vezes que houver essa necessidade. Ao fazer com o uso de um processo automatizado essas tarefas seriam necessárias apenas uma vez e quando surgir novamente à necessidade, os testes automatizados podem ser executados para gerar as validações, além de que o teste manual abre margens para erros, o que poderia ser evitado utilizando o processo automatizado (KUMAR; MISHRA, 2016).

Na hora de escolher entre teste manual e teste automatizado, de acordo com Marcus (2017) é preciso avaliar as características do cliente e do projeto, o tipo de orçamento disponível e qual é a metodologia de teste que beneficiará os prazos estabelecidos, as expectativas dos clientes e assim por diante. Destaca ainda de quando se trata de testes exploratórios e usabilidade, a melhor maneira de testar é através dos testes manuais. Quando se trata dos testes de desempenho, testes de carga, testes de regressão e testes repetidos, a melhor maneira é o teste automatizado, pois poderá ganhar no tempo de execução sendo que nestes tipos de teste o mesmo precisa ser repetido muitas vezes ou o caso de teste não tem mudança significativa.

Conforme os critérios apresentados na Figura 3 são possíveis fazer uma comparação de qual a melhor escolha no tipo de teste.

Figura 3 - Vantagens e desvantagens de testes manuais e testes automatizados.

Critério	Teste Manual	Automação de Teste
Premissa básica	Precisão menor e não executar tarefas repetidas	Precisão maior e altamente confiável para tarefas repetidas
Tempo de execução	Precisa de muito tempo	É um processo muito rápido
Investimento	Necessário nos colaboradores	Necessário nos colaboradores e em ferramentas de software
Indicado usar quando	Os casos de testes são executados uma ou duas vezes	Os casos de testes são executados sem limites
Interface com o Usuário	Altamente eficaz, pois, envolve a intervenção humana	Não é eficaz, uma vez que não há intervenção humana
Custo inicial	Pouco	Alto
Verificação	Não recomendado	Altamente recomendado

Fonte: MARCUS (2017)

No próximo item será explanado sobre uma ferramenta muito utilizada na automação de testes de software web e dispositivos móveis chamada Katalon Studio[1], que possibilita a automação seguindo um padrão de criar códigos através da gravação dos passos e reproduzir automaticamente os passos executados.

A ferramenta oferece suporte ao usuário através de seu site com documentação acessível a todos. Simples de manusear e *open source* na sua versão para uso individual, o que significa que a comunidade desenvolve constantemente novas funcionalidades, levando assim a escolha dessa ferramenta para desenvolvimento do trabalho.

[1] Katalon Studio. Disponível em: </https://www.katalon.com/ >

2.7 Ferramenta para automação de testes de software Katalon Studio

Desenvolvimento de software está entrando em uma nova era da tecnologia: aplicativos sem código, acessíveis aos usuários em todos os níveis. O teste de software não é uma exceção ao movimento em direção a práticas sem código, acredita-se que as ferramentas de automação sem código possam resolver o desafio de simplificar os testes para equipes de controle de qualidade em diferentes níveis, além de lidar com a crescente complexidade do próprio software (KATALON BLOG, 2020).

2.8 Automação de teste sem código

Automação de teste sem código significa executar a automação de teste sem precisar escrever nenhum script. A maioria das ferramentas de automação sem código do mercado oferece diversos recursos integrados, sem necessidade de codificação complicada. Alguns desses recursos são: gravação e reprodução, interface de arrastar e soltar (KATALON BLOG, 2020).

2.9 Benefícios da automação de teste sem código

- Melhor cobertura do teste, aplicar a automação de teste em mais plataformas simultaneamente, em um tempo muito menor, incluindo aplicativos da Web, e desktop.
- Menor tempo de colocação no mercado onde a equipe de controle de qualidade pode se concentrar mais no valor e no resultado do teste, em vez de perder tempo revisando e executando práticas de teste repetitivas.
- Reutilização de teste aprimorada, ou seja, é capaz de reutilizar as etapas de teste em todo o projeto.
- Capacitar metodologia ágil, como as práticas de teste sem código requerem pouca ou nenhuma habilidade de programação, permite que membros da equipe com diversas funções e conhecimentos participem do processo de teste (KATALON BLOG, 2020).

2.10 Katalon Studio

É uma solução pronta, para automatizar testes onde cobre todo o ciclo de desenvolvimento de software, desde a geração de testes, a execução de testes até os relatórios de testes. Também suporta Cucumber Runner [1]para BDD[2] (Behavior Driven Development) e integração com Jira[3]. O Katalon Studio é uma ferramenta de automação de testes que faz uso do mecanismo principal do Selenium[4]. Onde os principais recursos são implantação, gerenciamento de teste suporte à linguagem de script, desempenho, usabilidade e integração com outras ferramentas (KATALON BLOG, 2020).

Conforme especificado no manual disponível no site, usuários do Katalon Studio não precisam ser conhecedores e especialistas em informática, podem ser os testadores com conhecimento limitado. A ferramenta fornece interface amigável, com isso o usuário pode arrastar e soltar, selecionar palavras-chave e testar objetos para formar etapas de teste. Oferece ainda uma interface gráfica com menus, visualizações em árvore, tabelas, etc. para gerenciar casos de teste, objetos e arquivos de dados.

Para criação de scripts oferece uma sintaxe simples com sugestão de código e depuração. É ideal para os testadores que querem apenas arrastar ou ter habilidades de programação limitadas (KATALON BLOG, 2020).

---

[1] Cucumber Runner. Coordenar e escrever cenários. Disponível em: https://cucumber.io/ >

[2] BDD. Desenvolvimento Orientado a Comportamento.

[3] Jira. Ferramenta que permite o monitoramento de tarefas e acompanhamento de projetos garantindo o gerenciamento de todas as suas atividades em único lugar. Disponível em: https://www.atlassian.com// >

[4] Selenium. Framework portátil para testar aplicativos web. Disponível em: https://www.selenium.dev/>

2.11 Estrutura dos testes no Katalon e sua execução

Nesta seção é mostrada a estrutura, funcionamento e como é realizada a construção e execução de testes na ferramenta proposta.

No Katalon Studio um **Test Case**, como é denominado, é um caso de teste. Apesar de se tratar de um caso, ele não precisa ser necessariamente um caso específico, deixando em aberto a possibilidade de montar qualquer modelo para o projeto de testes. Os **Object Repository** são objetos da tela onde ficam armazenados dados referentes à sua localização, desta maneira o Katalon pode encontrar o objeto em questão durante um caso de teste. Todos os testes podem ficar organizados em **Test Suítes ,** os Test Suítes são ambientes para a execução de Test Cases, é possível à utilização de um ou mais Test Cases em sequência, ou seja, poderá configurar para rodar toda a suíte de testes em sequência. Ainda tem a possibilidade de criar um **Test Suíte Collection**, assim como os Test Suítes são conjuntos de Test Cases, os Test Suíte Collection são conjuntos de Test Suítes, com eles é possível adicionar um ou mais Test Suítes, selecionar quais serão executados, qual navegador será utilizado para cada teste.

Além da organização em Suítes de testes a ferramenta possui um componente chamado **Data Files** onde Test cases geram variáveis, e apesar de possuir valores padrão, é muito útil que elas se alterem ao longo da sua reprodução, já que os Test suítes permitem um ambiente onde as variáveis possam ser retiradas um arquivo de valores para que possam ser testadas, chamadas Test Data, ele possui uma planilha de dados que serão posteriormente utilizados em um Test Suíte, estes dados podem ser escritos manualmente, como também podem ser importados de um arquivo em Excel, CSV ou de uma pesquisa SQL.

Semelhante ao Test Data, os Checkpoints são uma planilha de dados. Mas seu intuito é serem utilizados para comparação de dados em um Test case, firmando a certeza de valores consistentes e concluindo um teste de sucesso.

Toda vez que um Test Suíte ou Test Suíte Collection é executado, seu relatório é

armazenado organizadamente dentro de uma pasta chamada **Reports.**

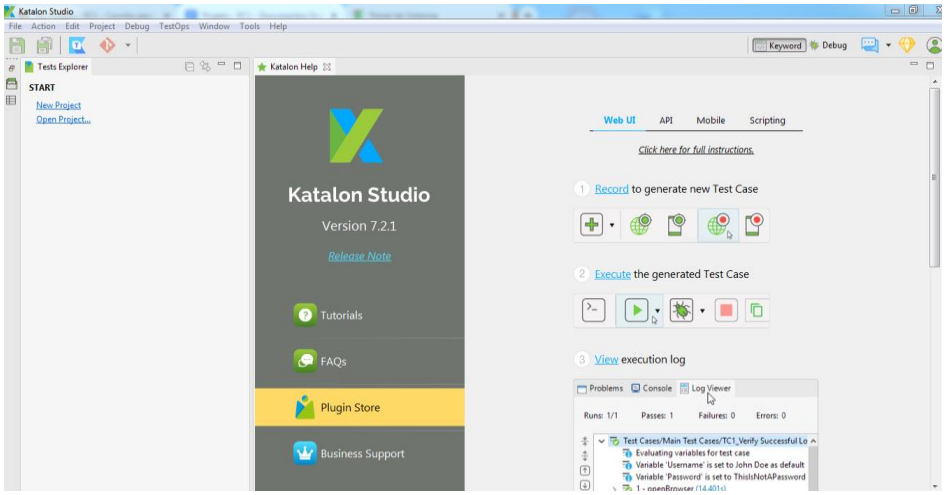
Na execução de um teste, perfis são variáveis globais associadas a usuários, por isso são denominados “Perfis”. Para a execução de testes independentes, sempre é utilizado o perfil Default, mas para Test Suíte Collections é possível selecionar um perfil específico para cada execução de Test Suíte.

2.12 Apresentação do ambiente de trabalho do Katalon Studio

A ferramenta Record do Katalon será usada para a montagem de um teste automatizado. Com o Record Web basta inserir a URL desejada e clicar no ícone do navegador, fazendo com que uma página da web seja aberta na URL especificada. Após isso, qualquer ação efetuada neste navegador será registrada como uma instrução (Keyword) do Katalon resultando em um script de teste automatizado ao fechar a janela, possibilitando assim a execução automática do mesmo teste posteriormente.

Assim como está sendo mostrada na Figura 4, a página inicial do Katalon Studio é composta por tutoriais, perguntas e respostas sobre a ferramenta, exibição de novos plugins e ainda as opções e tipos de teste que estão disponíveis para a escolha, que são Web, API, Mobile e Scripting, bem como também a barra de ferramentas para a criação e configuração dos projetos de teste.

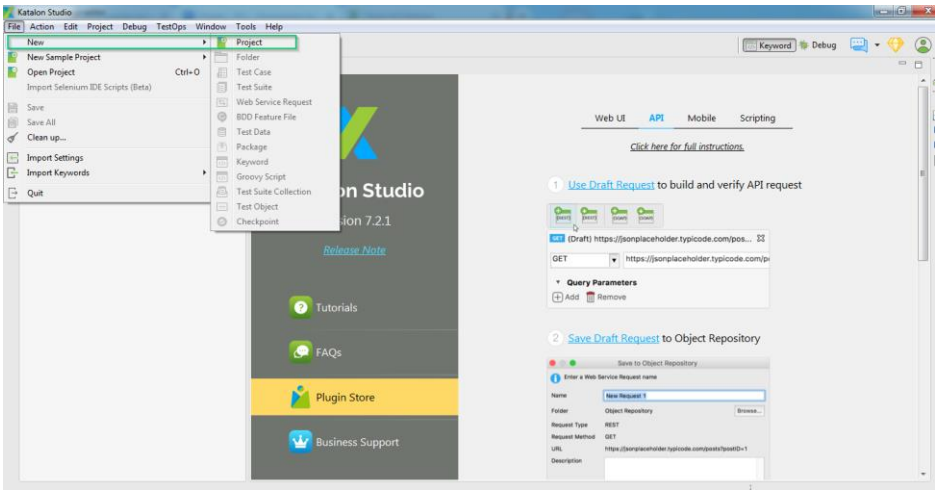
Figura 4 - Tela inicial Katalon Studio



Fonte: Katalon Studio

Na figura 5, está sendo mostrado que antes de iniciar a criação de um caso de teste será necessário primeiramente criar um projeto, para organização dos casos de teste.

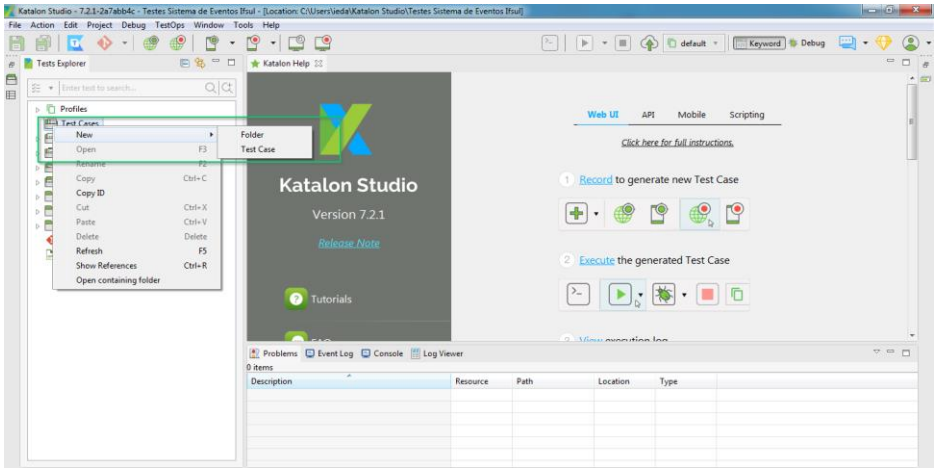
Figura 5- Criação de um novo Projeto.



Fonte: Katalon Studio

Quando um novo projeto é criado exibe a estrutura do projeto e a partir dele é iniciada a criação dos Test Case, como pode ser visto na Figura 6.

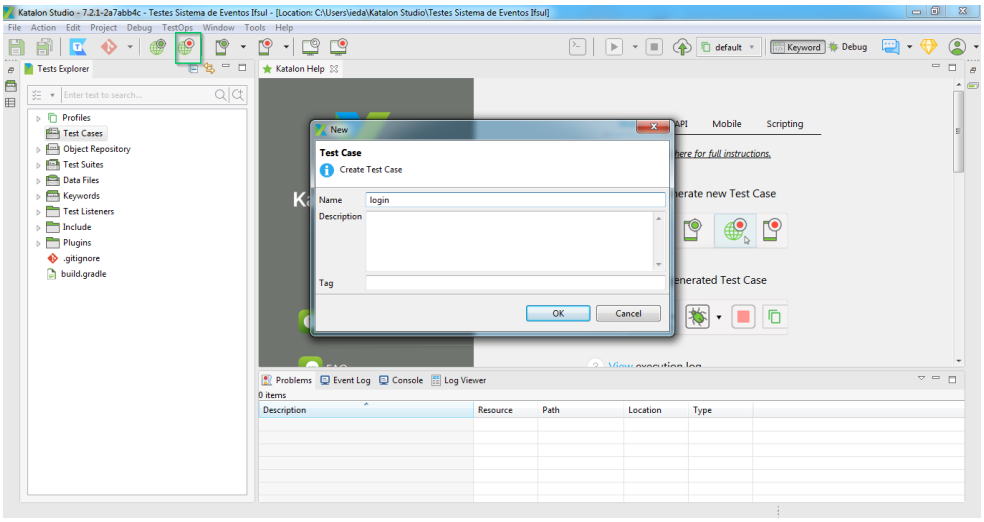
Figura 6- Estrutura de pastas



Fonte: Katalon Studio

Ao criar um novo caso de teste, Test Case deve ser definido um nome e logo após clicar na opção Record Web para iniciar gravação do passos executados no navegador, este passo pode ser visto na Figura 7.

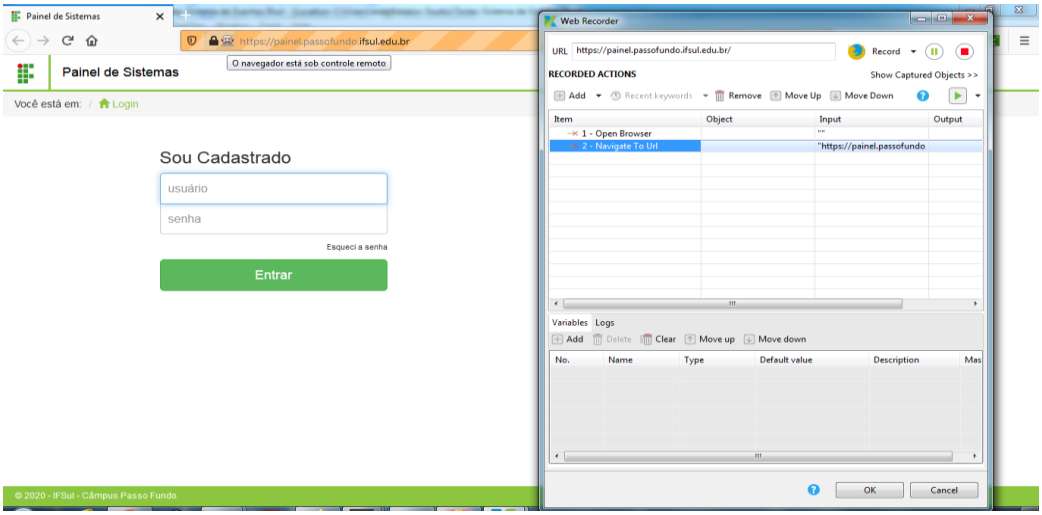
Figura 7- Criação do Test Case



Fonte: Katalon Studio

Na Figura 8, é demonstrada a criação e gravação do Test Case, bem como pode ser visualizado o passo a passo que está sendo criado.

Figura 8- Criação e gravação de cada passo da execução do Test Case.



Fonte: Katalon Studio

Diante de todas as especificações da ferramenta Katalon Studio, ela busca realizar uma solução de automação inteligente, robusta e escalável para mostrar sua facilidade de utilização tanto para testadores iniciantes ou especialistas em qualquer lugar.

3 TRABALHOS RELACIONADOS

Nesta seção serão apresentados alguns trabalhos que fazem uso do mesmo método de automação, porém com ferramentas diferentes.

O estudo de caso do trabalho de conclusão de Marques (2018) propõe a integração da linguagem Ruby,[5] juntamente com a utilização do framework Capybara[6], responsável pela automação de testes em aplicações web.

Utilizou-se framework Cucumber para gerar a arquitetura do projeto e gerenciar a execução dos casos de teste a fim de validar os benefícios da automação, onde também foi utilizado o sistema de inscrição de eventos do IFSUL e pôde concluir que apresentou a capacidade de executar testes em menor tempo do que o estimado durante a execução dos mesmos testes de forma manual. Porém, o custo beneficio de se utilizar é relevante, com a ressalva que é preciso ter-se a consciência de que o um automatizador de testes , muitas vezes poderá custar mais caro do que um testador manual, pois este profissional necessita obter conhecimento de programação e páginas web, além de conhecer os processos de teste de software.

Braga, (2019) apresenta um estudo em uma empresa com uma nova proposta de processo de desenvolvimento de software, baseada em um modelo ágil e no BDD (Behavior Driven Development), dando possibilidade para a automação de testes. Como resultado desse estudo foram listadas as principais ferramentas de automação e dentre elas foram escolhidas a ferramenta Cucumber e a linguagem Ruby para ser desenvolvida a automação no momento atual da empresa.. Neste contexto a pesquisa identificou que mesmo que a automação de testes traga diversos benefícios para a empresa, é constatado que ainda é necessário e recomendado serem realizados testes manuais e exploratórios.

---

[5] Ruby. linguagem de programação interpretada multiparadigma, de tipagem dinâmica e forte Disponível em: https://www.ruby-lang.org/pt/ >

[6] Capybara. Ferramenta para testar aplicativos da web, simulando como um usuário real interagiria com seu aplicativo. Disponível em: https://www.ruby-lang.org/pt/ >

Esses trabalhos auxiliam no entendimento da aplicação de testes automatizados, pois os estudos apresentam propostas semelhantes, porém com ferramentas diferentes, mas com mesmo proposito.

4 METODOLOGIA

Neste capítulo são descritos alguns passos básicos para melhorar os resultados obtidos com a prática da execução de testes manuais. Aqui, a execução de testes criada e executada em uma ferramenta de automação chamada Katalon Studio é a proposta, mostrando ao final ganho de tempo para futuros testes de regressão na mesma funcionalidade, ou seja, uma vez que o teste é executado de forma manual a ferramenta acima citada possibilita a gravação dos passos realizados pelo usuário em tempo real, reproduzindo-os em forma de código e convertendo em um caso de teste, sendo possível executá-lo de forma automática posteriormente.

Primeiramente foi realizado o planejamento dos testes onde será criado o Plano de Teste que será adicionado ao Apêndice A, deste projeto, para diferentes tipos de teste. Este será elaborado com base na análise de necessidades básicas e fluxos básicos do sistema de inscrição de eventos do Instituto Federal Sul Rio Grandense do câmpus Passo Fundo.

Neste planejamento foi definido o escopo de teste, onde serão testados, os requisitos funcionais do sistema, ou seja, será delimitada a execução dos fluxos principais.

A execução dos testes foi realizada pela ferramenta Katalon Studio, uma ferramenta gratuita de automação de teste tanto web quanto mobile, possui uma IDE própria a base da linguagem Java, uma solução com um conjunto abrangente e integrado de recursos, desde a gravação das etapas de teste e geração de scripts até a execução e a geração de relatórios dos resultados dos testes. Tal ferramenta foi escolhida justamente por propiciar facilidade no uso e também por oferecer uma interface amigável e de fácil entendimento tanto na criação dos testes bem como na sua execução, é gratuita, baseada em tecnologia open source, organiza os testes em projetos e possuir relatórios precisos e detalhados, permitindo também uma customização bem ampla para suas informações.

A automação e a montagem de um teste automatizado foram realizadas de forma manual no primeiro teste e a partir destes realizada a gravação dos passos, para isso será usado a opção Record do Katalon. Com o Record Web basta inserir a URL desejada e clicar no ícone do navegador, fazendo com que uma página da web seja aberta na URL especificada. Após isso, qualquer ação efetuada neste navegador é

registrada como uma instrução (Keyword) do Katalon resultando em um script de teste automatizado ao fechar a janela, possibilitando assim a execução automática do mesmo teste posteriormente.

Para alcançar o objetivo proposto neste trabalho, primeiramente foi desenvolvido um roteiro de testes, criar e executar casos de teste funcionais em uma ferramenta de automação chamada Katalon Studio e sua aplicação será um sistema utilizado junto aos cursos de informática do IFSUL Câmpus Passo Fundo.

No final da execução dos testes conseguimos apresentar soluções para problemas que acontecem na validação e execução de testes manuais de um sistema, fazendo um comparativo de tempo gasto com o teste manual e o mesmo teste sendo automatizado. Mostrar que é possível avaliar as funcionalidades de um sistema de maneira mais rápida eficiente e sem muito esforço, ou seja, esse processo é caracterizado pela facilidade de repetir esforços contínuos e repetidos para testar um sistema, além disso, mostrar que é possível um retorno mais rápido da equipe de testes, dessa forma bugs ocasionais podem ser resolvidos mais rapidamente.

Também realizar uma avaliação de quando é vantagem executar o teste manual e quando a automação é a melhor escolha. Ainda mostrar que pode ter um ganho de eficiência, pois testes manuais costumam demandar uma grande quantidade de tempo do ciclo de desenvolvimento de um sistema. Em consequência disso, até as melhorias mais simples podem fazer uma grande diferença no custo final de um projeto.

4.1 SISTEMA DE INSCRIÇÕES EM EVENTOS

O Sistema escolhido para a aplicação e execução dos testes é o {Sys Eve}, Sistema de Inscrição em Eventos, um sistema Web que foi desenvolvido com o objetivo de auxiliar nas Inscrições dos eventos da instituição, sendo também utilizado para aperfeiçoar a geração de certificados dos participantes, em relação aos eventos realizados pelo Instituto Federal Sul-rio-grandense - Câmpus Passo Fundo.

4.2 REQUISITOS DO SISTEMA DE INSCRIÇÕES EM EVENTOS

Os eventos podem ser classificados em dois tipos: Interno - disponível para alunos e servidores, e externo - disponível para o público em geral. Os requisitos do

sistema compõem:

- O cadastro do período de inscrição para cada evento;
- O registro de presenças nas atividades por código de inscrição ou nome;
- Gerar certificados para os participantes com presença registrada;
- O cadastro de atividades relacionadas ao evento ocorrido, ex: palestras e oficinas ofertadas;
- A geração de relatórios de inscritos por atividade, em relação às presenças computadas de acordo com seus crachás, código de barra, e presença por dia e turno;
- A listagem de eventos ativos aos usuários;
- A permissão do cadastro dos usuários nos eventos e atividades.

4.3 REQUISITOS POR PERFIS DE USUÁRIOS

O Sistema de Inscrições em Eventos possui quatro perfis de usuários: Alunos, Docentes, Técnicos Administrativos e Visitantes, sendo esses usuários Internos (Alunos, Docentes e Técnicos Administrativos) e/ou Externos (Visitante). Estes perfis possuem determinadas permissões e acessos a funcionalidades restritas ou específicas de cada perfil.

4.3.1 Perfil Todos os usuários - Permissões e funcionalidades padrões:

Quando falamos dos requisitos do sistema, todos os usuários possuem permissão para:

- Visualizar os eventos ativos;
- Realizar download do seu número de inscrição com código de barras;
- Visualizar as suas inscrições;
- Cancelar e realizar novas inscrições em atividades, desde que respeitando o período de inscrições;
- Imprimir os certificados dos eventos passados em que participou.

4.3.2 Perfil Usuários Externos

Caso usuário está designado com perfil de visitante, este poderá visualizar os eventos ativos e abertos ao público externo, se inscrever nas atividades destes eventos, e realizar download dos certificados os quais já tenha participado.

4.3.3 Perfil - Usuários Internos (Alunos, Docentes e Técnicos Administrativos)

Para usuário com perfis internos, tanto Alunos, Docentes, Técnicos Administrativos, e demais usuários que possuam vínculos com a Instituição, poderão, visualizar os eventos ativos, realizar inscrições nas atividades e baixar os certificados dos eventos os quais já tenham participado.

4.3.4 Perfil Docente - Usuário Interno - Com permissão de cadastro de eventos e atividades.

O perfil Docente poderá realizar o cadastro de um novo evento e atividade, podendo também listar e obter relatórios das inscrições dos usuários, filtrando-as por: evento, categoria, atividade, e nome do inscrito. Ainda, poderá Registrar as presenças dos usuários, de acordo com o evento, atividade e a forma de identificação pessoal do participante, através do CPF, ou nome.

Além das demais permissões comuns a todos os usuários, como: Visualizar os eventos ativos, realizar inscrições nas atividades dos eventos e efetuar download dos certificados dos eventos ocorridos os quais já tenha participado.

5 RESULTADOS E DISCUSSÕES

Neste capitulo é apresentado o desenvolvimento do projeto, juntamente com as etapas necessárias para sua construção. Além disso, são apresentados os resultados obtidos a partir das etapas de criação e a execução dos casos de testes desenvolvidos.

5.1 CRIAÇÃO E ARQUITETURA DO PROJETO

Para a realização deste trabalho foi utilizada a ferramenta Katalon Studio assim como já mencionada e comentada no capítulo 2.10.

O Katalon Studio está disponível gratuitamente, sendo suportado para Windows 7, Windows 8, Windows 10, macOS 10.11+, Linux (Ubuntu based). Para a utilização da ferramenta primeiramente é necessário criar uma conta no site do Katalon Studio e posteriormente instalar a versão mais recente que pode ser baixado do site da Katalon. Para versões anteriores, também é possível o download na página de versões do GitHub.[7]

5.2 CRIAÇÃO DOS CASOS DE TESTE

Antes de iniciar a criação dos casos de teste é necessário criar um novo projeto para que tudo fique organizado por projeto.

Para uma melhor organização do projeto, a fim de manter todos os cenários dos casos de testes reunidos em uma só pasta, se criou uma pasta para cada um dos tipos de teste executado.

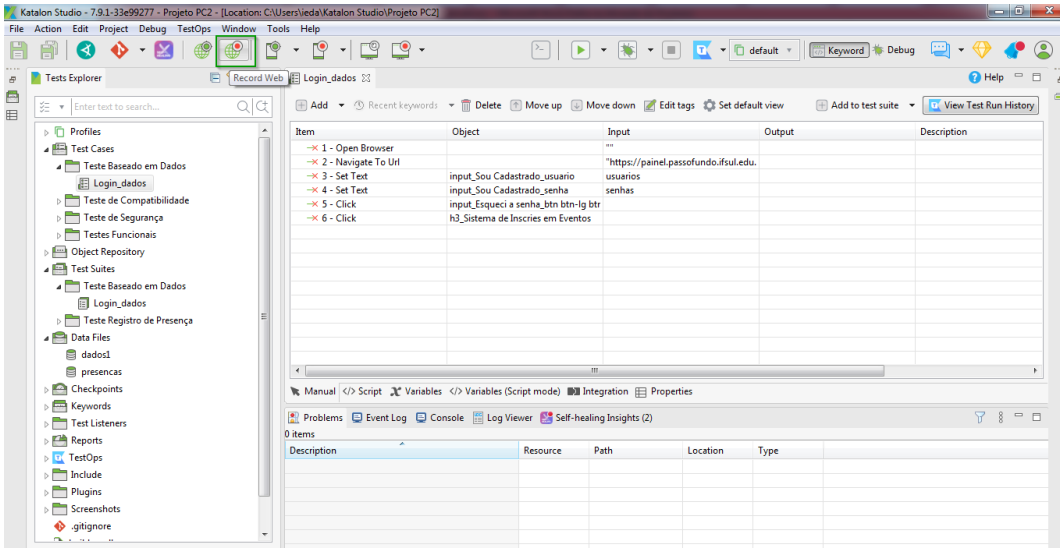
5.2.1 Casos de Teste Baseado em Dados

Primeiramente foi executado o teste manual juntamente com a sua gravação conforme a Figura 10, onde se tem o passo a passo de um teste de login.

---

[7] GitHub.. uma plataforma de hospedagem de código-fonte e arquivos com controle de versão. Disponível em: https://github.com// >

Figura 10 - Execução de um caso de teste manual.



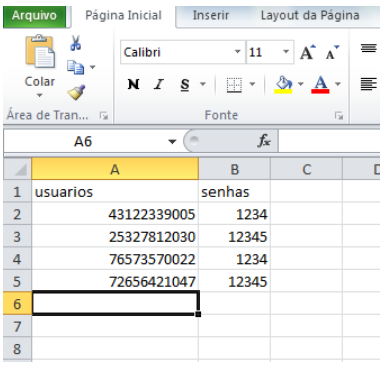
Fonte: Da autora, 2021.

Neste tipo de teste é possível buscar dados de uma tabela para realizar testes dinâmicos, repetitivos e automáticos para web.

Para esse recurso o Katalon fornece o objeto *Data File*, que pode consultar dados de fontes externas: arquivo CSV, arquivo Excel, banco de dados relacional, suporta testes baseados em dados e objeto de arquivo de dados. Como neste caso de teste, ao testar o recurso de login, pode-se especificar contas predefinidas em uma planilha do Excel ou em um banco de dados, neste caso foi utilizado dados de usuário e senha oriundos de uma planilha Excel.

Primeiramente foi necessário criar um arquivo utilizando o Microsoft Excel com os dados que o caso de teste vai usar com as colunas usuários e senhas conforme Figura 11.

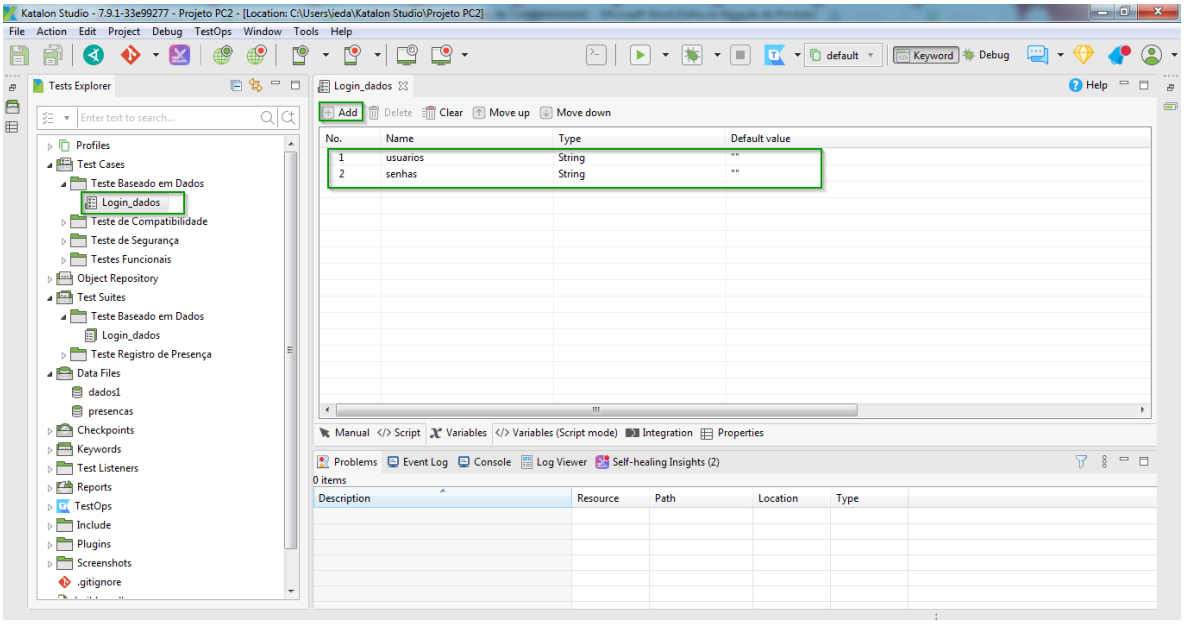
Figura 11 - Tabela de Dados caso de teste baseado em Dados.



Fonte: Da autora, 2021.

Após a criação da tabela é preciso acessar o caso de teste e adicionar as variáveis criadas no documento acima, neste caso, usuários e senhas. Acrescentar em ***'Variables' >>+Add*** **,** conforme evidência da Figura 12.

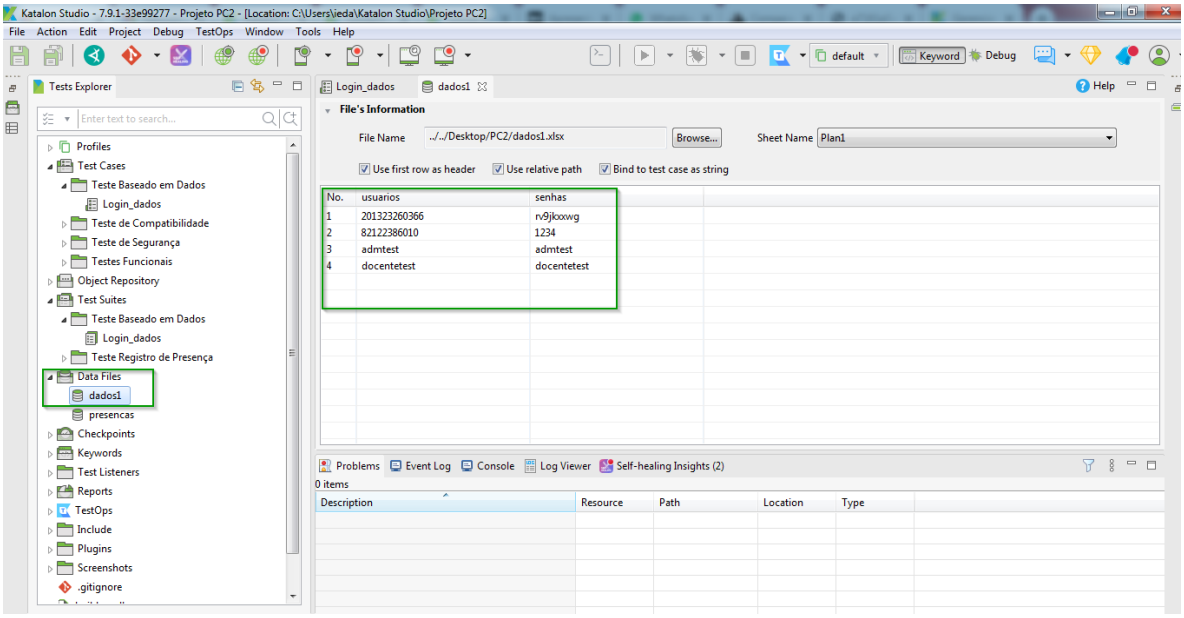
Figura 12 – Variáveis da Tabela.



Fonte: Da autora, 2021.

Depois da criação das variáveis deverá ser incluído o arquivo de dados à seção *Test Data* e prosseguir para vincular as variáveis e respectivas colunas do arquivo de dados como pode ser visualizado na Figura 13.

Figura 13 – Vinculação da Tabela.

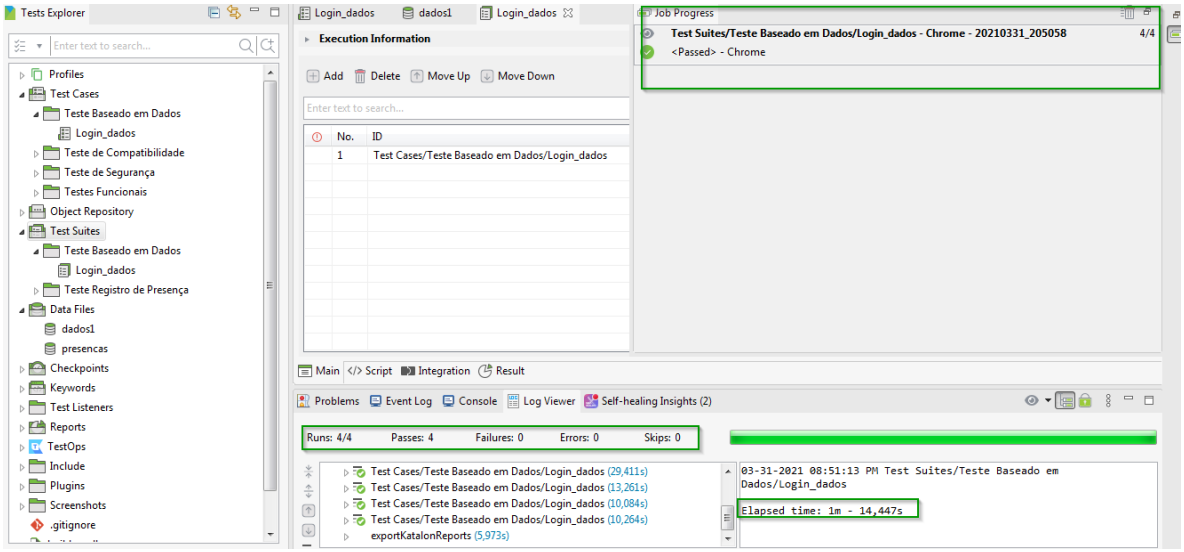


Fonte: Da autora, 2021.

Por fim o caso de teste poderá ser executado na quantidade de vezes usando os usuários e senhas definidas no arquivo de dados de teste, para isso uma nova "*Test Suites*" deverá ser adicionada. Também realizar a vinculação das variáveis em '*Show Data Binding'*, vincular o caso de teste que irá fazer uso das variáveis, vincular o arquivo de dados em '*Teste Data' >> Add*, então ele será executado a quantidade de vezes usando os dados definidos no arquivo de dados de teste.

Após a execução pode-se analisar os resultados obtidos como mostra a Figura 14. Assim, é possível verificar o progresso, status da execução e tempo gasto, entre outros dados referentes à sua execução.

Figura 14 – Resultados da Execução



Fonte: Da autora, 2021.

Ao finalizar a execução manual e a execução automática do tipo e teste baseado em dados, foi possível identificar que a utilização do Katalon para realizar testes que precisam ser repetidos inúmeras vezes, obtém-se um tempo menor do que o estimado durante a execução destes mesmos testes de forma manual, onde o tempo gasto foi de 3m34s para acesso com três usuários e o tempo de toda execução automatizada foi de 1m14s, esse tempo poderá variar dependendo da quantidade de execuções, para cada caso de teste e também do desempenho da máquina no momento da execução. Além do ganho de tempo pode-se levantar também a geração automática de relatórios que facilitam a verificação de possíveis erros caso falhar. Neste requisito não foram encontradas falhas, mas caso fosse encontrado o status do caso mudaria.

Ainda, é possível através da plataforma de armazenamento dos dados na nuvem chamada *DevOps*, coordenar várias atividades, ciclos e estruturas em teste de software para garantir a qualidade do software em cada estágio, sem sacrificar a velocidade.

Esta plataforma fornece painéis e relatórios sobre o status de qualidade do que está sendo executado.

Pode-se perceber também que a precisão é maior do que a executada manualmente onde pode se afirmar que é confiável para essas tarefas repetitivas.

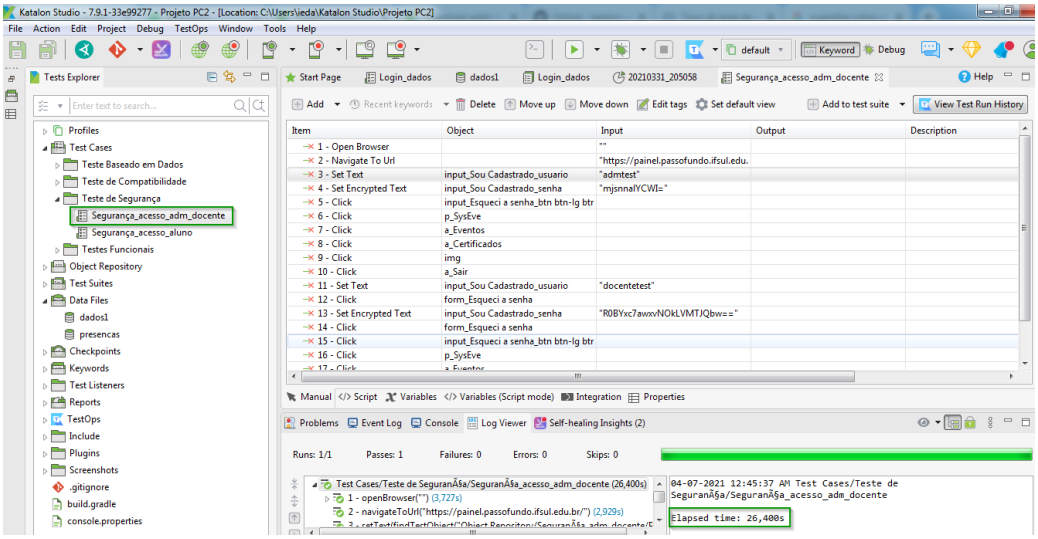
5.2.2 Casos de Teste de Segurança

Para estes cenários se pensou em testar a segurança do sistema verificando o acesso e permissões de cada um dos diferentes perfis de usuário do sistema.

5.2.2.1 Segurança acesso com diferentes perfis

Neste caso verificou-se o acesso dos perfis administrativo, docente (Figura 15) e aluno (Figura 16) a fim de validar as devidas permissões descritas no item 4.3.3.

Figura 15 – Caso de teste manual perfis administrativo e docente

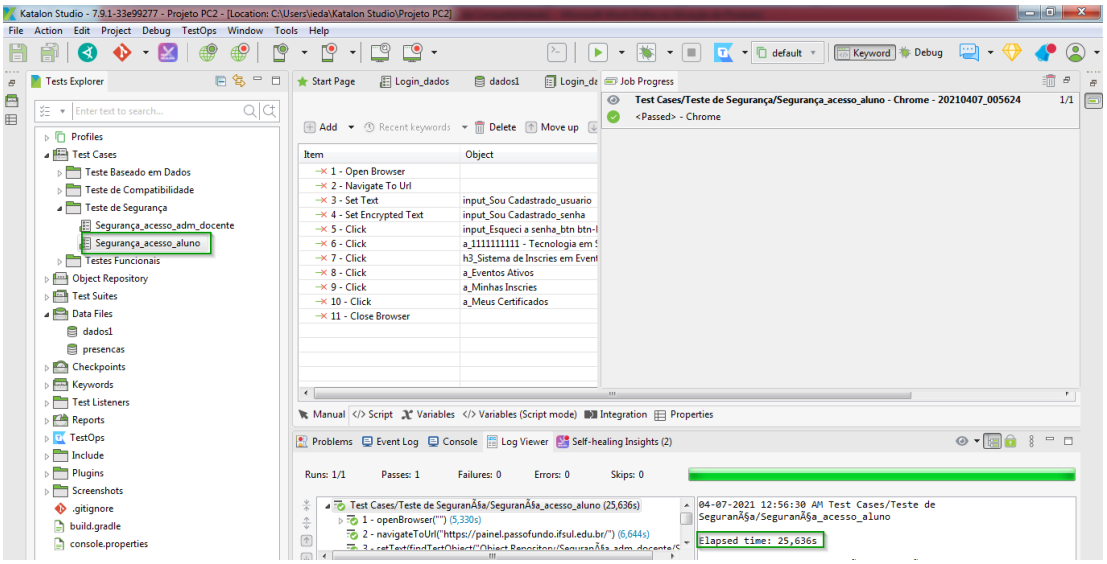


Fonte: Da autora, 2021.

Na figura 16 segue a mesma lógica do caso de teste perfil docente, porém com perfil aluno, a fim de validar que para alunos a permissão de acesso é apenas como

descrita no item 4.3.2 Perfil - Usuários Internos (Alunos, Docentes e Técnicos Administrativos).

Figura 16 – Caso de teste manual perfil aluno



Fonte: Da autora, 2021.

Ao finalizar a execução manual e a execução automática deste tipo de teste de validação da segurança, foi possível identificar que a utilização do Katalon em tipos de testes que necessitam de verificações constantes, evitando as brechas de segurança e reduzindo os riscos de uma possível invasão ou outros tipos de ataque e ainda, neste contexto, onde o vazamento de um dado sigiloso pode causar danos, obtém-se novamente um ganho no tempo execução, onde manual foi de 48s e automatizado foi de 25s. Também apresentou uma garantia maior na qualidade, pois, o risco em esquecer algum dos requisitos, tende a ser menor quando já estão automatizados e ainda torna exaustivo o teste manual. Além das vantagens acima citadas pode-se levantar também a geração automática de relatórios, assim como já apresentado no caso de teste baseado em dados, que facilitam a verificação de possíveis erros. Neste requisito também não foram encontradas falhas.

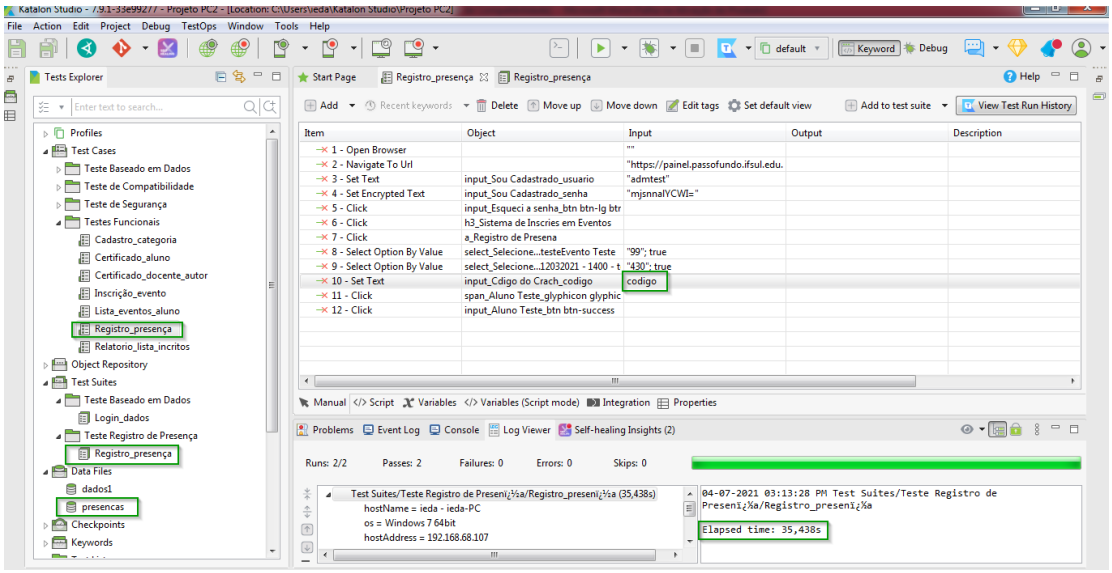
5.2.3 Casos de Teste Funcionais

Nesta seção serão listados os resultados dos casos de teste funcionais.

5.2.3.1 Registro presença evento dados tabela

O objetivo deste tipo de teste é medir a qualidade funcional dos requisitos principais do sistema de eventos. Uma das funcionalidades para o perfil administrador é a possiblidade de registrar presenças nas atividades por código de inscrição ou nome. Na Figura 17, é apresentado o caso de teste que fará o registro de presença de um evento especifico de todos os códigos de inscrição oriundos de uma tabela de dados, simultaneamente.

Figura 17 – Caso de teste registro presenças por código



Fonte: Da autora, 2021.

Para esse caso de teste foi necessário assim como no caso de teste baseado em dados do item 4.5.2.1, criar uma suíte de testes e uma tabela vinculada a ela contendo os códigos das inscrições. Ao executar o teste automatizado conforme os passos apresentados na Figura 17 serão registradas presenças para todos os códigos vindos da tabela para um evento também previamente configurado sem a necessidade de registrar individualmente manual para cada usuário. Resultando assim, novamente um ganho de tempo e agilidade em um processo que seria realizado de forma manual. O tempo irá depender da quantidade de vezes, ou, de dados que o caso de teste será executado, como neste caso foram dois códigos, o tempo manual foi de 1m15s e automatizado foi de 35s.

Ao total foram criados 7 cenários para os casos de teste funcionais 1 – Cadastrar uma nova categoria de um evento; 2 - Validar download do certificado de um evento de um usuário aluno; 3 - Validar download do certificado de um evento de um usuário docente autor; 4 - Realizar inscrição em um evento com usuário aluno; 5 – Registrar presença com dados oriundos de uma tabela. 6 - Validar exibição de eventos ativos com usuário aluno; 7 - Validar a geração de relatórios de inscritos por atividade, em relação às presenças computadas de acordo com seus crachás, código de barra, e presença por dia e turno.

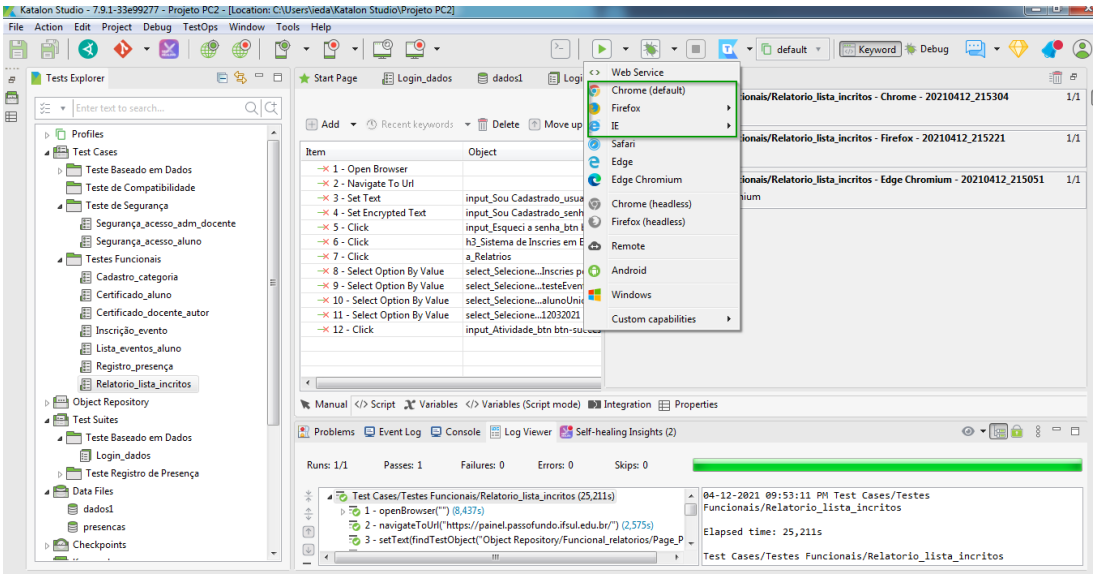
A descrição destes outros cenários não foi descrita porque estas seguem a mesma lógica, do caso de teste registro presença evento, especificadas no item 4.5.2.3.1, e foram prescritas na seção de apêndices deste trabalho.

5.2.4 Casos de Teste de Compatibilidade

Neste tipo de teste o objetivo foi realizar teste funcional de 1 caso de teste de função a fim de validar sua compatibilidade em diferentes navegadores, ou seja, validar se a navegação e a execução do teste funcional suportam diversos navegadores de Internet. Para isso, foram escolhidos três navegadores popularmente conhecidos e utilizados (Google Chrome, Firefox, e Microsoft Edge). Entretanto é possível executar em qualquer navegador.

A Figura 18 mostra o caso de teste manual e como proceder para execução em diferentes navegadores.

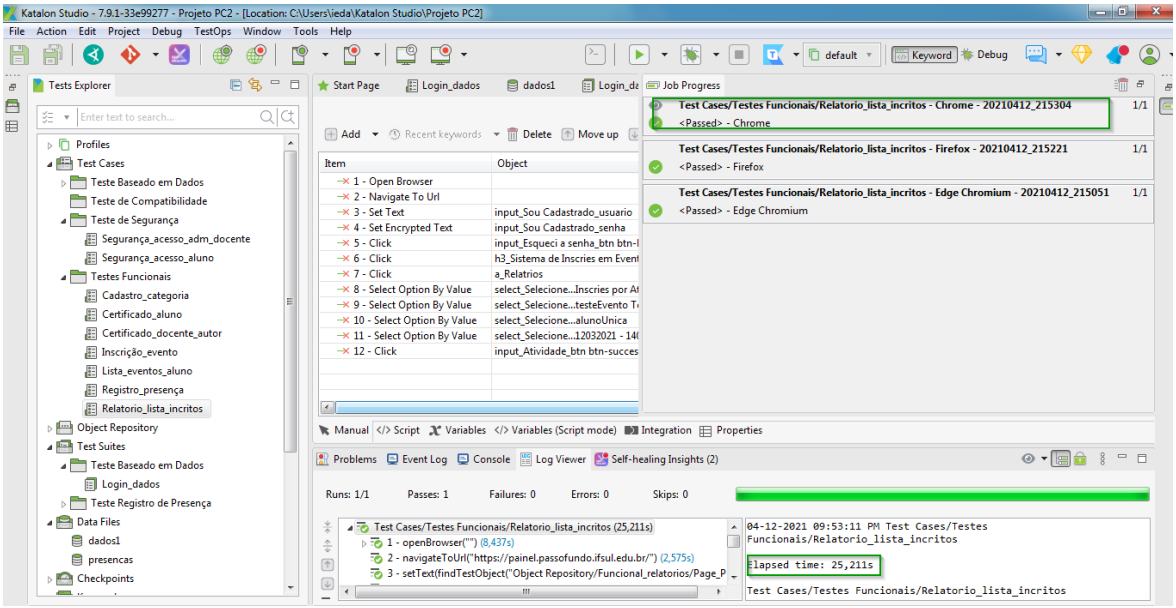
Figura 18 – Execução em navegadores diferentes



Fonte: Da autora, 2021.

Para a seleção dos navegadores basta escolher o navegador e executar o caso de teste sem a necessidade de recriar ele novamente, assim como mostra na Figura 18. No navegador Chrome o tempo de execução manual foi de 51s e automatizada de 25s e o teste executou sem falhas, assim como pode ser visto na Figura 19.

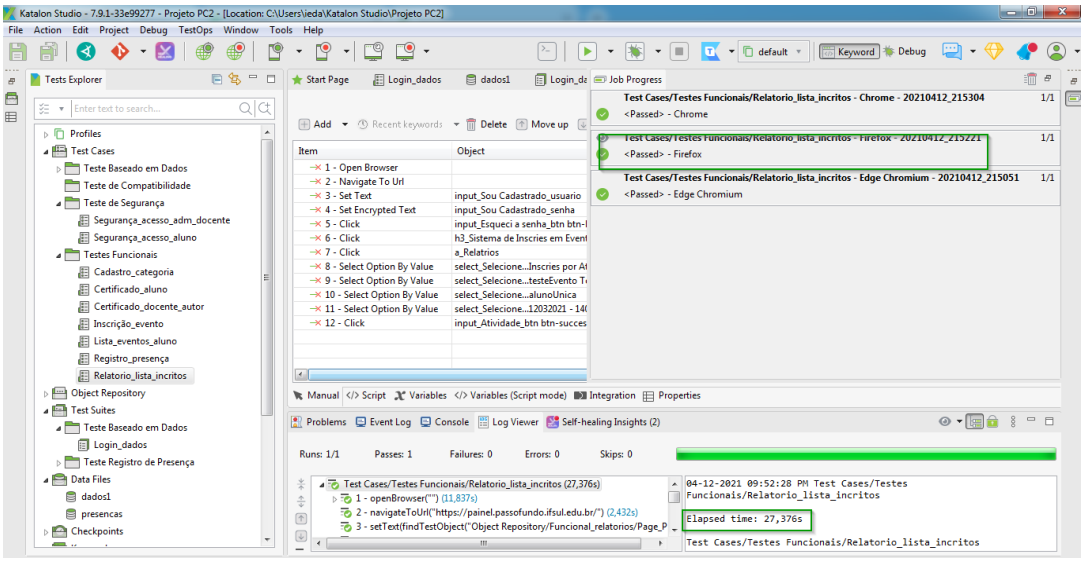
Figura 19 – Execução Chrome



Fonte: Da autora, 2021.

Já no navegador Firefox o tempo de execução foi um pouco maior comparado com o Chrome, 58s manualmente e 27s, automatizado e o teste executou sem falhas, conforme verificado na Figura 20.

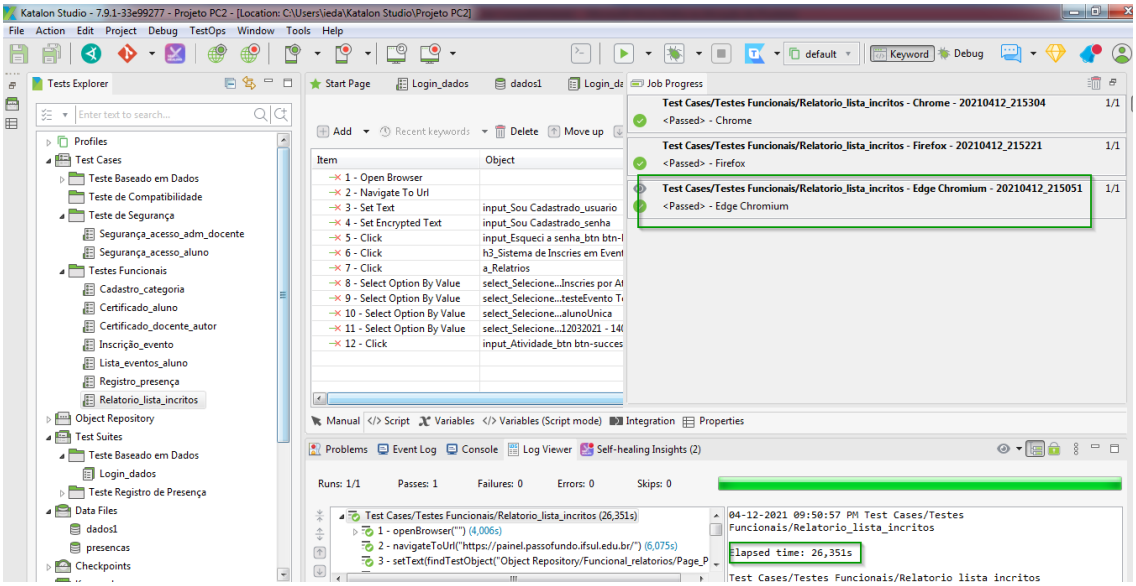
Figura 20 – Execução Firefox



Fonte: Da autora, 2021.

E, no navegador Microsoft Edge o tempo de execução foi 55s manualmente e 26s, automatizado e o teste executou sem falhas, conforme verificado na Figura 21.

Figura 21 – Execução Microsoft Edge



Fonte: Da autora, 2021.

Como resultado para os testes de compatibilidade nota-se que os testes automatizados entre navegadores verificarão rapidamente como o site funciona em vários navegadores, sem intervenção e a necessidade de repetir os testes para cada navegador. O teste é executado manual uma única vez, em qualquer um dos navegadores e após isso é possível escolher um navegador diferente e executar o teste novamente. A automação oferece resultados precisos, com risco baixo de cometer erro humano na sua execução.

Para todos os tipos executados podemos elencar que a automação permite que o teste seja repetido várias vezes, sendo mais fácil encontrar novos erros através da repetição e da simulação de cenários específicos.

Então a partir disso podemos afirmar que é possível minimizar os problemas da abordagem manual, o tempo despendido e, consequentemente, o custo final.

6 CONSIDERAÇÕES FINAIS

O presente estudo teve como objetivo avaliar a importância da automação de teste funcional em diferentes tipos de testes. Demonstrar os benefícios da automação de testes no sistema de Eventos do Instituto Federal Sul Rio Grandense - Câmpus Passo Fundo e fazer uma comparação com métodos manuais de teste.

Para inicio do desenvolvimento foi necessário construir um plano de testes disponível na seção de apêndices deste documento, para elencar os diferentes tipos de teste que seriam executados e para execução e automação deste trabalho foi escolhida a ferramenta de automação de testes chamada Katalon Studio.

Ao finalizar a execução dos testes manuais e automatizados nos tipos de testes levantados neste trabalho se tem redução do tempo de execução, e que a automatização é adequada, quando, há casos de teste que precisam ser executados repetidamente por um longo período de tempo, sendo mais confiável, pois ferramentas e / ou scripts realizam o teste automatizado, ainda, outras pessoas podem executar os mesmos testes e visualizar os resultados progressivamente.

Por fim, destacar a importância de avaliar a real necessidade de automatizar testes de software, pois quando há necessidade de identificar problemas relacionados à aparência visual do sistema e problemas de usabilidade os testes automatizados não conseguem identificar essas lacunas. Em relação ao trabalho de Marques (2018) também concluiu que os métodos manuais se tornam mais demorados em relação ao mesmo teste sendo executado automaticamente e que a automação pode se tornar mais complexa ou cara dependendo da situação.

É importante citar que a ferramenta pode não conseguir validar alguns campos ao gravar os passos para ser feita a automação o que prejudica a execução no desenvolvimento do teste e, consequentemente demorar mais que um teste manual, sendo necessária a alteração do script.

Como trabalhos futuros é sugerida a continuação desse estudo ampliando para outros tipos de testes não funcionais, como testes de desempenho, integrações com outras ferramentas de teste e testes de serviço, mobile, desktop, ou seja, explorar outras possibilidades que a ferramenta dispõe.

REFERÊNCIAS

ABNT – Associação Brasileira de Normas Técnicas. NBR ISO 9000:2000 – Sistemas de gestão da qualidade e garantia da qualidade – Fundamentos e Vocabulário. Rio de Janeiro: ABNT, 2001. Citado na página 7

BASTOS, A., RIOS, E., CRISTALLI, R., e MOREIRA, T., Base de conhecimento em teste de software (Rio de Janeiro: Alta Books, 2012). Citado na página 7

FELIPE AMILIBIA, BRAGA. Qualidade De Software: Proposta De Automação De Testes E De Um Processo Ágil Para Uma Empresa De Software. Disponível em: https://www.riuni.unisul.br/bitstream/handle/12345/8451/TCC%20Felipe%20Braga.pdf?sequence=1&isAllowed=y. Acesso em: 20 abr. 2021. Citado na página 32

GANDARA, Fernando. Qualidade e Teste em Software. Joinville: Clube dos Autores, 2012. Citado na página 10.

GOMES MEDEIROS, MARCUS. Teste manual vs automação de tester. Disponível em: https://medium.com/@mvmgmg/teste-manual-vs-automa%C3%A7%C3%A3o-de-teste-8fcf4f9832b1. Acesso em: 26 Mai 2020. Citado na página 19

ISO CD 9241-11: Ergonomia da interação homem-sistema - Parte 11: Usabilidade: definições e conceitos (2015)9.Citado na página 14

JAKOB, Nielsen. Usabilidade na Web. Projetando websites com qualidade, 2007. Citado na página 14

KATALON STUDIO-BLOG, 2020. Disponível em: <https://www.katalon.com/>.Acesso em: 01 Jun 2020. Citado na página 21.

KECHI, Hirama. Engenharia de Software. Rio de Janeiro: Elsevier, 2012.

KUMAR, D.; MISHRA, K. The impacts of test automation on softwareŠs cost, quality and time to market. Procedia Computer Science, Elsevier, v. 79, 2016. Citado na página 19

LAGES, Scaldaferri Daniel. Automação dos Testes: um lobo na pele de cordeiro?. Engenharia de Software Magazine, n. 29, p. 20-25, 2010. Citado na página 18

MARQUES DA COSTA, CAMILA. Caso De Uso: Integração De Ferramentas Para Automação De Testes Funcionais De Software. Disponível em: https://painel.passofundo.ifsul.edu.br/uploads/arq/20190221150108727886174.pdf. Acesso em: 20 abr. 2021. Citado na página 32

MEDIUM. Automação de testes: o que é, quando e por que automatizar: 2018. Disponível em: <https://medium.com/venturus/quais-as-raz%C3%B5es-para-automa%C3%A7%C3%A3o-de-testes-c177cbd9397>.Acesso em: 09 abr. 2020. Citado na página 10.

MYERS, G. J. The Art of Software Testing. 2. ed. Hoboken: John WileyeSons Inc., 2004. ISBN 0-471-46912-2. Citado na página 10.

NBR ISO 9000:2005. Sistemas de Gestão da Qualidade Fundamentos e vocabulário. Rio de Janeiro.

NAIK, K.; TRIPATHY, P. SOFTWARE TESTING AND QUALITY ASSURANCE. [S.l.]: Wiley, 2008. Citado na página 8.

PRESSMAN, Roger; MAXIM, Bruce. Engenharia de Software: 8ª Ed., São Paulo: McGraw-Hill, 2016. Citado na página 7.

RIOS, Emerson; MOREIRA FILHO, Trayahú. Teste de Software. 3. Ed. Rio de Janeiro, RJ: Alta Books, 2013. Citado na página 16

SOMMERVILLE, Ian. Engenharia de Software. 9ª Ed., São Paulo: Addison Wesley, 2011. Citado na página 14

SWEBOK2013. Disponívelem:<https://ieeecsmedia.computer.org/media/education/swebok/swebok-v3.pdf>. Acesso em: 08 abr 2020. Citado na página 7

SYLLABUS-CTFL, 2018. Disponível em:<https://www.bstqb.org.br/uploads/syllabus/syllabus_ctfl_2018br.pdf>>. Acesso em: 09 abr 2020. Citado na página 10.

Tian, J. *Software quality engineering: Testing, quality assurance, and quantifiable improvement.* John Wiley and Sons Ltda., 2005.Citado na página 8

TSUI, Frank; KARAMAN, Orlando. Fundamentos da Engenharia de Software. Tradução e Revisão de Edson Tanaka. 2.ª Edição. Rio de Janeiro: LTC, 2013. Citado na página 8

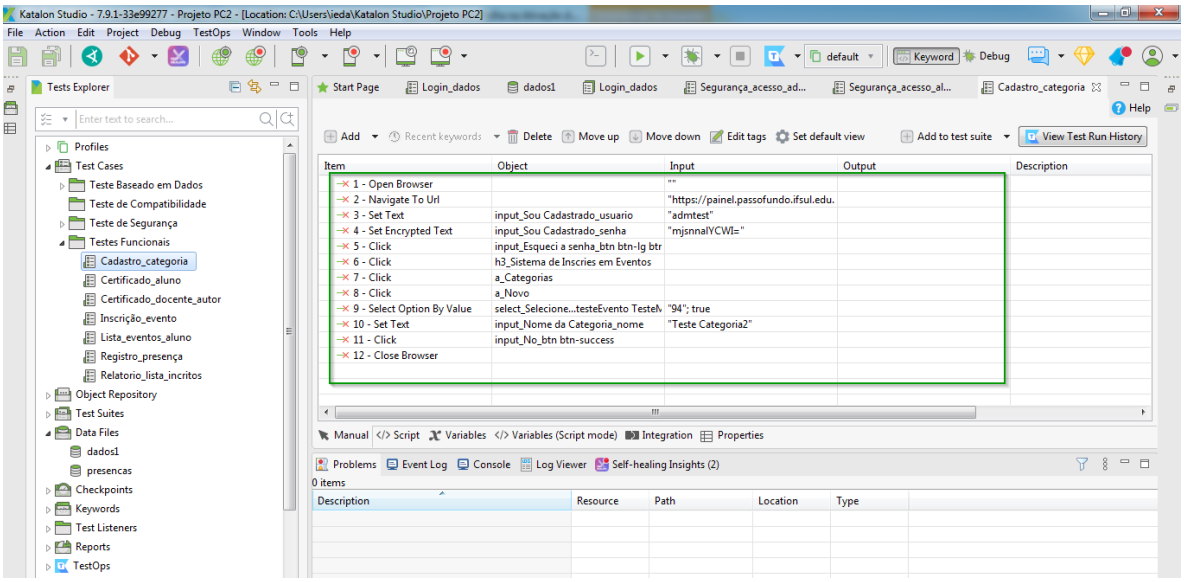
VIDAL SANTOS, Diego.; DE SANTANA OLIVEIRA, Katuche Varjão. Introdução à garantia de qualidade de software, 2017. Citado na página 11

WIKLUND, K. et al. Impediments for automated testing – an empirical analysis of a user support discussion board. In: 2014 IEEE Seventh International Conference on Software Testing, Verification and Validation. [S.l.: s.n.], 2014. Pag. 113–122. ISSN 2159-4848. Citado na página 11.

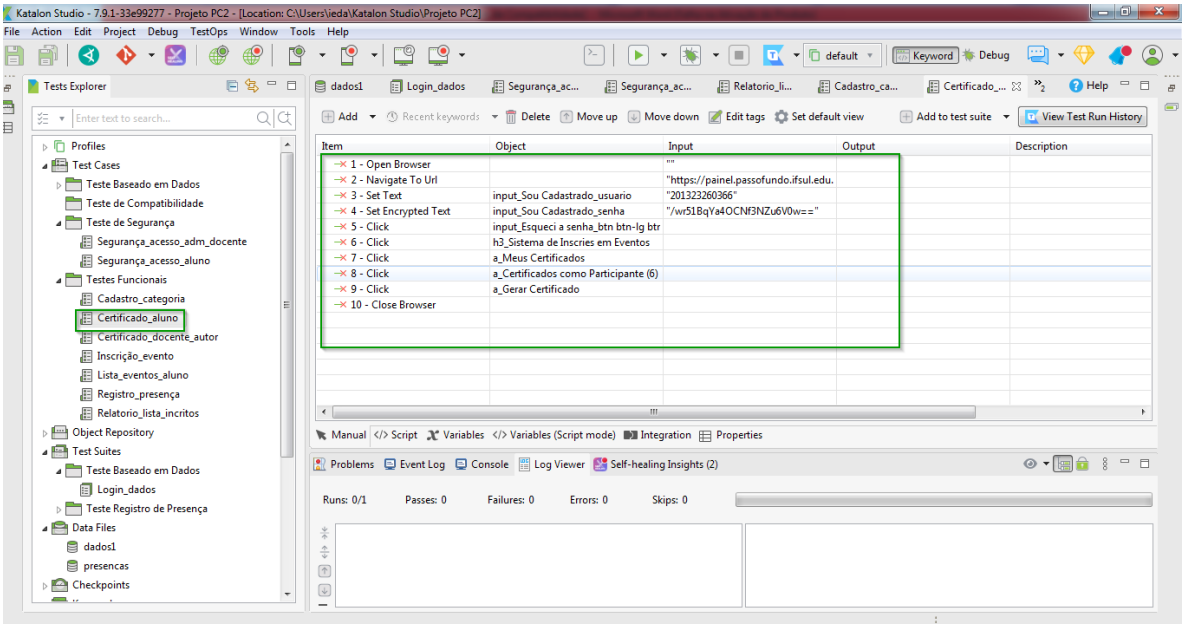
APÊNDICES

Abaixo serão apresentados os demais casos de teste funcionais, os quais foram criados e executados a fim de validar os reais benefícios da sua automatização.

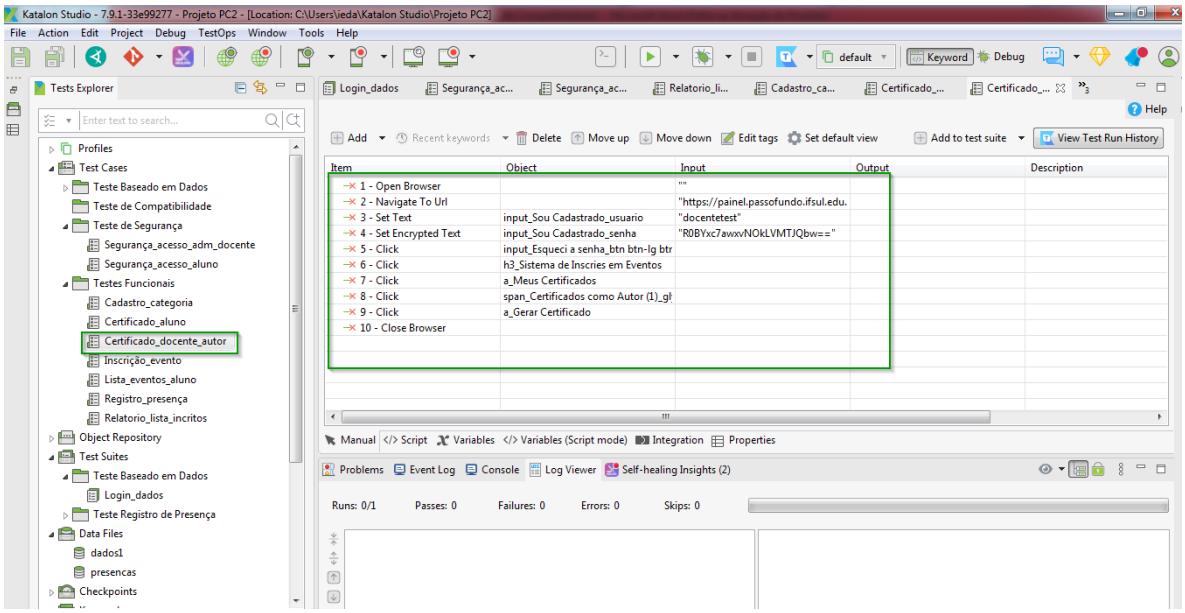
APÊNDICE A – Passo a passo caso de Cadastro Categoria



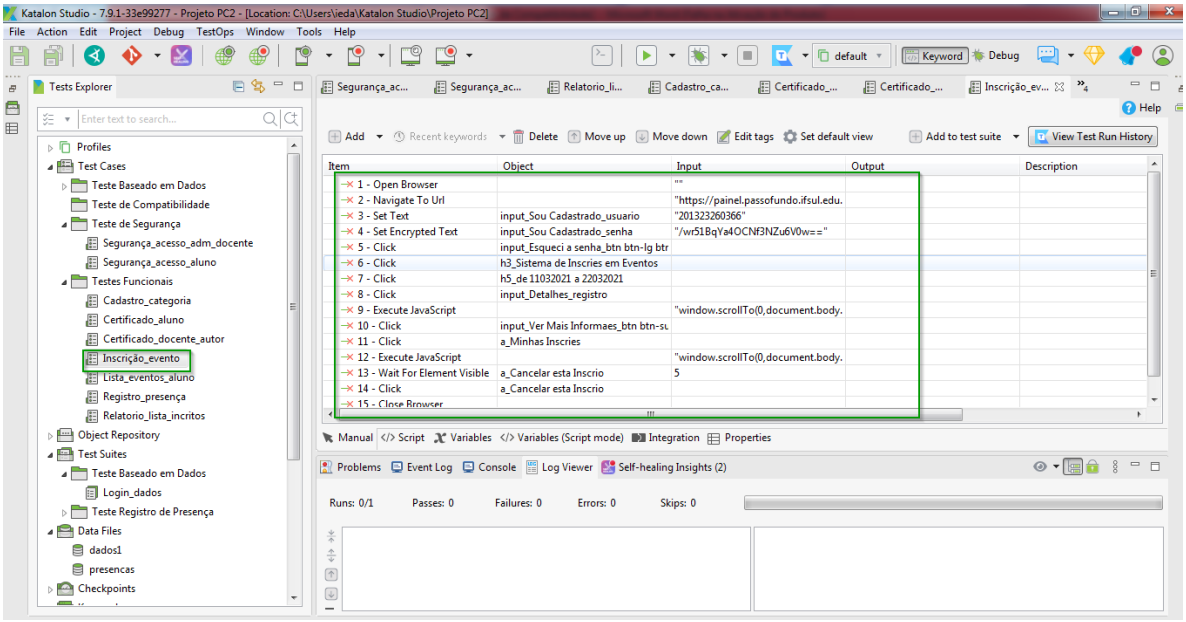
APÊNDICE B – Passo a passo caso de teste Certificado_Aluno.



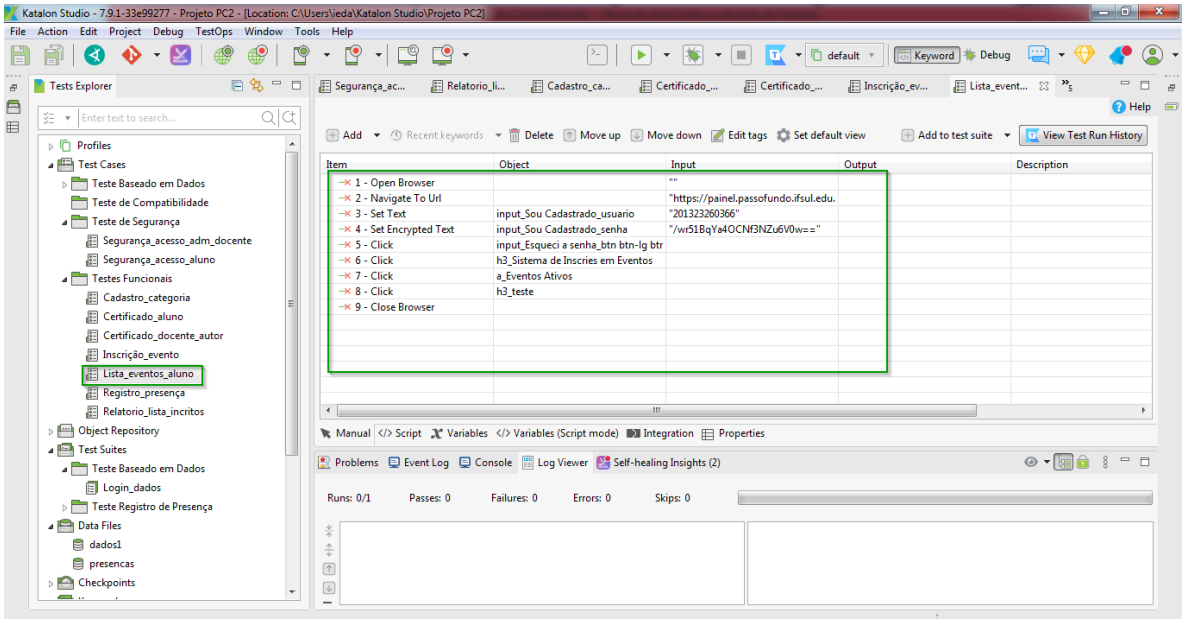
APÊNDICE C – Passo a passo caso de teste Certificado_Docente_autor.



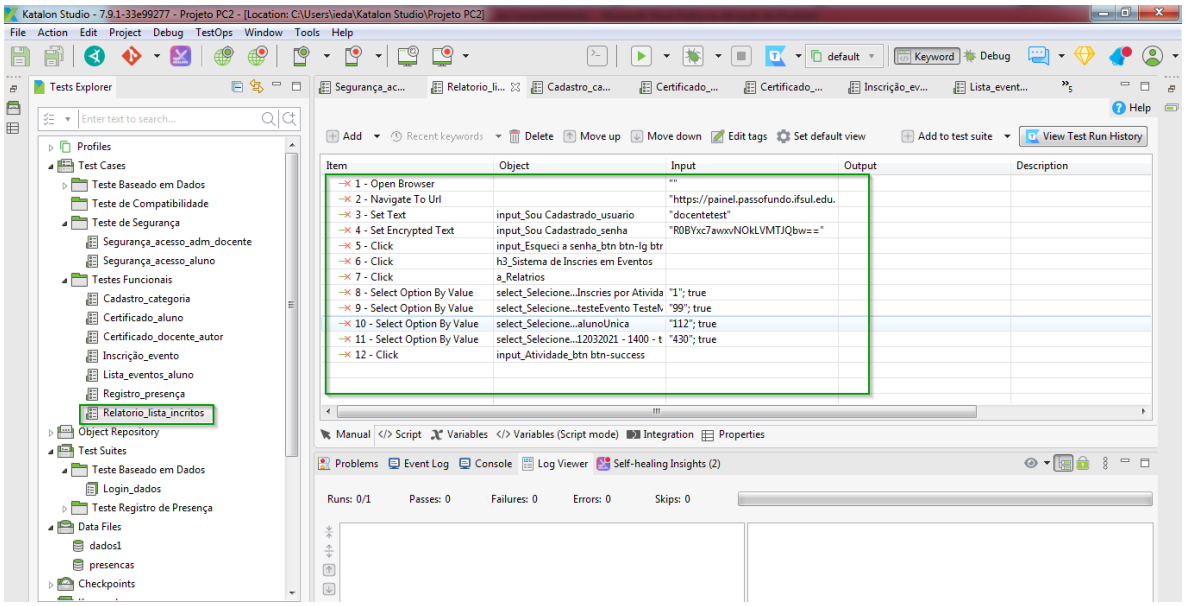
APÊNDICE D – Passo a passo caso de teste Inscrição Evento.



APÊNDICE E – Passo a passo caso de teste Lista_eventos_aluno.



APÊNDICE F – Passo a passo caso de teste Relatorio_lista_inscritos.



APÊNDICE G – Documento de Plano de Testes

Plano de Testes

Sistema de Eventos- IFSUL

Professor:

Andre Fernando Rollwagen

Equipe:

Ieda Rosana Kolling Wiest

Maio / 2021

Histórico de Revisões

Data	Versão	Descrição	Autor
23/fev/2021	1.0	Definição inicial	Ieda
09/abr/2021	1.1	Alteração Casos de Teste	Ieda
25/abr\2021	1.2	Finalização Documento	Ieda

Índice

1. Introdução

1.1 Objetivos

Esse documento do Plano de Testes compõe-se dos seguintes objetivos:

- Identificar informações do sistema de eventos do IFSUL câmpus Passo Fundo existente e os componentes de software que devem ser testados;
- Listar os Requisitos a Testar recomendados (alto nível);
- Recomendar e descrever as estratégias de teste a serem empregadas;
- Identificar os recursos necessários;
- Listar os elementos resultantes do projeto de testes.

1.2 O Sistema de Eventos

O Sistema de Inscrições em Eventos {Sys Eve}, foi desenvolvido com o propósito de auxiliar nas Inscrições dos eventos e suas atividades, sendo também utilizado para otimizar a geração de certificados dos participantes, em relação aos eventos sediados pelo Instituto Federal Sul-Rio-Grandense - Campus Passo Fundo.

Para acessa-lo será necessário realizar um novo cadastro ou realizar login com usuário e senha de aluno ou servidor. Este acesso será ao painel de sistemas onde cada usuário terá previamente configurado suas permissões de acesso e a partir disso será exibida a lista de todos os sistemas permitidos.

Os eventos podem ser classificados em dois tipos: Interno - disponível para alunos e servidores, e externo - disponível para o público em geral. Os requisitos do sistema compõem:

- O cadastro do período de inscrição para cada evento.
- O registro de presenças nas atividades por código de inscrição ou nome.
- A geração dos certificados para os participantes com presença registrada.
- O cadastro de atividades relacionadas ao evento ocorrido, ex: palestras e oficinas ofertadas.
- A geração de relatórios de inscritos por atividade, em relação às presenças computadas de acordo com seus crachás, código de barra, e presença por dia e turno.

- A listagem de eventos ativos aos usuários.

- A permissão do cadastro dos usuários nos eventos e atividades.

1.3 Escopo

O sistema de eventos passará pelos seguintes testes: Teste funcional sendo executadas algumas funcionalidades principais e mais usadas, teste baseado em dados, teste de portabilidade e teste de segurança onde todos eles, após a primeira execução manual ficaram automatizados na ferramenta para serem executados em testes futuros ou testes de regressão, não sendo necessário repeti-los de forma manual. Os testes funcionais vão lidar com a qualidade funcional, teste baseado em dados com o controle de acesso, ou seja, login com vários tipos de usuários e teste de portalibilidade com o comportamento de funcionalidades em diferentes navegadores, avaliando a interface gráfica. E o teste de segurança irá verificar a proteção do sistema contra invasões ou acesso não autorizado a informações que não se referem ao perfil do usário.

Para a execução dos testes será utilizada apenas uma máquina, onde está instalada a ferramenta Katalon Studio, esta que, será utilizada para execução manual e posterior automatização, bem como a execução em diferentes navegadores, a fim de garantir a previsibilidade de performance e compatibilidade.

Requisitos a Testar

A lista abaixo identifica aqueles itens – use cases, requisitos funcionais – que foram identificados como alvos de teste. Essa lista representa o que será testado.

1.4 Teste baseado em Dados

- Realizar teste de login com arquivo de dados configurando a vinculação de variável para ler os valores dos dados de teste como os tipos de dados pretendidos, executando um caso de teste de login várias vezes através de um arquivo de dados.

1.5 Teste Funcional

1. Validar lista de eventos para aluno;
2. Validar download de certificado para aluno;
3. Validar emissão relatório lista de inscritos;
4. Validar download de certificado docente;
5. Cadastro categoria OK
6. Registro de Presença;
7. Inscrição em um evento.

1.6 Teste de compatibilidade

- Realizar teste funcional de 2 casos de teste de função a fim de validar sua com portabilidade em diferentes navegadores.

1.7 Teste de Segurança e de Controle de Acesso

Verificar que além do administrador e docente, ninguém mais tem acesso aos menus “Cadastros” e “Operacional”.

Verificar que os alunos apenas tem acesso ao Menu “Minha Inscrição”.

Verificar que os alunos do sistema podem acessar apenas as funcionalidades e dados associados ao seu próprio tipo de usuário.

Estratégia de Teste

1.8 Tipos de Teste

1.8.1 Teste baseado em Dados

Objetivo do Teste:	Garantir que os acessos de diferentes tipos de usuários através de dados oriundos de uma tabela funcionam apropriadamente e sem corrupção dos dados.
Técnica:	▪ Invocar cada método e processo de acesso a diferentes usuários automaticamente através de dados oriundos de uma tabela.
Critério de Finalização:	Todos os métodos e processos de acesso a dados externos funcionam como projetados e sem nenhuma corrupção de dados.
Considerações Especiais:	▪ O teste pode necessitar da criação de uma tabela com dados de usuário e senha dos usuários. ▪ Para um futuro teste ou exclusão de usuários será necessário fazer alterações no caso de teste ou na tabela de dados.

1.8.2 Teste de Função

Objetivo do Teste:	Garantir a funcionalidade apropriada do alvo do teste, incluindo navegação, entrada de dados e processamento.
Técnica:	Executar cada caso de uso, fluxo de caso de uso, usando dados válidos e inválidos, para verificar o seguinte: ▪ Os resultados esperados ocorrem quando dados válidos são usados ▪ As mensagens de erro ou aviso apropriadas são exibidas quando dados inválidos são usados.
Critério de Finalização:	▪ Todos os testes planejados foram executados. ▪ Todos os defeitos identificados foram tratados.
Considerações Especiais:	Nenhum

1.8.3 Teste de Compatibilidade

Objetivo do Teste:	Verificar o seguinte: ▪ Validar se a navegação e a execução do teste funcional dá suporte a diversos navegadores de internet (Google Chrome, Firefox, Internet Explorer).
Técnica:	Executar manualmente em um navegador e após selecionar o mesmo caso de teste para executá-lo automaticamente em outros navegadores.
Critério de Finalização:	É verificado que a execução em cada navegador permanece consistente e o fluxo é realizado da mesma maneira.
Considerações Especiais:	Nenhuma

1.8.4 Teste de Segurança e Controle de Acesso

Objetivo do Teste:	• Segurança do Nível de Aplicação: Verifique que um ator pode acessar apenas aquelas funções ou dados para os quais o seu tipo de usuário tem permissão. • Segurança do Nível de Sistema: Verifique que apenas aqueles atores com acesso ao sistema e aplicações têm permissão de acessá-los.
Técnica:	• Segurança do Nível de Aplicação: Identifique e liste cada tipo de usuário e as funções ou dados para os quais cada tipo tem permissão. • Crie testes para cada tipo de usuário e verifique cada permissão criando transações específicos para cada tipo de usuário. • Modifique o tipo de usuário e repita os testes para os mesmos usuários. Em cada caso, verifique que funções ou dados adicionais estão corretamente disponíveis ou negados. • Acesso de Nível de Sistema: Ver Considerações Especiais abaixo.
Critério de Finalização:	Para cada tipo de ator conhecido as funções ou dados apropriados estão disponíveis, e todas as transações funcionam como esperado e rodam nos Testes de Função anteriores.
Considerações Especiais:	O Acesso ao sistema deve ser revisado ou discutido com o administrador de rede ou de sistema apropriado. Esse teste pode não ser necessário já que ele pode ser uma função da administração da rede ou sistema.

Recursos

Essa seção apresenta os recursos recomendados para este projeto, suas principais responsabilidades, e seus conhecimentos ou conjunto de habilidades.

1.9 Sistema

A tabela seguinte expõe os recursos do sistema para o projeto de teste.

Recursos do Sistema
Navegadores disponíveis. Internet Explorer, Firefox, Chrome
Terminais e Programas 1 PC com acesso a internet Ferramenta Katalon Studio instalada

Cronograma

Milestone	Data de Início	Data de Término
Planejar Teste	08/02/2021	01/03/2021
Projetar Teste	01/03/2021	15/03/2021
Implementar Teste	15/03/2021	12/04/2021
Executar Teste	12/04/2021	26/04/2021
Avaliar Teste	26/04/2021	03/05/2021