**1. Escolha dos Casos de testes**

**1.1 - Mapeie 4 Casos de Testes no Aplicativo da Lojinha**

**ID:** TC01

**Condição de Teste:** Realizar login com sucesso na Lojinha

**Pré-Condições:** Usuário deve ter uma conta válida cadastrada, o site deve estar acessível

**Passo a Passo:**

1. Abra o navegador
2. Acesse o site ‘<http://165.227.93.41/lojinha-web/v2/>’
3. Clique no campo “Usuário”
4. E digite o usuário ( “admin”)
5. Clique no campo “Senha”
6. E digite a senha (“admin”)
7. Clique no botão ‘ENTRAR’

**Resultado:** Login é realizado com sucesso e a página “Lista de Produtos” deve ser exibida

**ID:** TC02

**Condição de Teste:** Cadastrar novo produto

**Pré-Condições:** Usuário deve ter realizado login na Lojinha, o site deve estar acessível

**Passo a Passo:**

1. Após realizar login com as credenciais válidas
2. Cliquei no botão ‘ADICIONAR PRODUTO’
3. Será exibido o formulário para preencher os campos do produto
4. No campo ‘Nome do Produto’ digite o nome do produto
5. No campo ‘Valor do Produto’ digite o valor do produto
6. No campo ‘Cores do produto’ digite a(s) cor(s) do produto
7. Clique no botão ‘SALVAR’

**Resultado:** Será exibido ao usuário a mensagem ‘Produto adicionado com sucesso’

**ID:** TC03

**Condição de Teste:** Editar produto

**Pré-Condições:** Usuário deve ter realizado login na Lojinha, o site deve estar acessível, produto deve estar cadastrado

**Passo a Passo:**

1. Após realizar login com as credenciais válidas
2. Clique no produto que iremos alterar
3. Edite as informações necessárias
4. Cliquei no botão ‘SALVAR’

**Resultado:** Será exibido ao usuário a mensagem ‘Produto alterado com sucesso’

**ID:** TC04

**Condição de Teste:** Excluir produto

**Pré-Condições:** Usuário deve ter realizado login na Lojinha, o site deve estar acessível, produto deve estar cadastrado

**Passo a Passo:**

1. Após realizar login com as credenciais válidas
2. Localize o produto que será excluído do cadastro
3. Clique no ícone da ‘lixeira’ localizado no final do cadastro do produto

**Resultado:** Página será atualizada e será exibido ao usuário a mensagem ‘Produto excluído com sucesso’

**1.2. Justificar a escolha dos casos de teste baseando-se na criticidade e frequência de uso das funcionalidades.**

**TC01:** Realizar login com sucesso na Lojinha

**Motivo:**

Validação da funcionalidade básica ‘realizar login’. O login é a principal funcionalidade do site e de qualquer aplicação. É o principal ponto de acesso ao site e suas funcionalidades que são protegidas pela aplicação na área restrita do site.

**Criticidade:**

Alta - realizar o teste de login com sucesso é primordial para a segurança da aplicação e funcionalidade. É a principal função do site que deve sempre ser validade e passar por testes de regressão em todos os ciclos de teste.

**TC02:** Cadastrar novo produto

**Motivo:**

Cadastrar produto é uma das principais funcionalidades do site, permitindo adicionar novos produtos à lista de produtos da ‘lojinha’. Por ser uma tarefa frequente é importante que esse processo seja simples e sem erros para garantir eficiência no cadastro.

**Criticidade:**

Alta - Sem essa funcionalidade não é possível adicionar novos produtos a lista, e a inserção de novos produtos fica comprometida a operação da nossa ‘lojinha’. Podendo impactar outras funcionalidades e a experiência do usuário da ‘lojinha’.

**TC03:** Editar produto

**Motivo:**

Editar é uma das principais funcionalidades do site, onde podemos editar o cadastro de um produto, realizar alterações na sua descrição e valores, corrigir informações inseridas de forma incorreta. A opção de editar garante uma maior flexibilidade de atender a necessidade do usuário e qualquer mudança pode ser atualizada.

**Criticidade:**

Alta - é uma função importante e crítica para o sistema. Sem a opção de editar erros no cadastro do produto não podem ser corrigidos de forma rápida e ágil e isso pode levar a situações críticas como a inserção de informações incorretas gerando erros.

**TC04:** Excluir produto

**Motivo:**

Exclusão de produtos é uma das funcionalidades da lojinha, permite que produtos sejam removidos por ser um produto descontinuado, que foi cadastrado por engano de forma incorreta. O teste de excluir produtos garante que apenas pessoas autorizadas e logadas no site possam ter essa permissão, prevenindo a exclusão de forma acidental por outros usuários não autorizados.

**Criticidade:**

Alta - O impacto de uma exclusão no sistema é alto e um processo irreversível, uma vez excluído o cadastro do produto não é possível recuperar tal cadastro ou informação. A cada ciclo de teste essa função deve ser analisada com um teste de regressão. Para garantirmos que produtos possam ser excluídos com sucesso e que pessoas não autorizadas não tenham a funcionalidade habilitada.

**2. Escolha do Framework de Testes**

1. **Appium**
2. **Expresso**

**Framework:** Appium

**Linguagem de programação:** JavaScript

**Vantagens:**

* O Appium é uma ferramenta de automação de teste de código aberto que permite testar plataformas como iOS e Android. Por utilizar uma interface do WebDriver para testes suporta diversas linguagens de programação, como Java, C #, Ruby e outras.
* Permite testar sites usando o device físico, e esses navegadores podendo controlar o Safari e o Chrome nos dispositivos.
* Integrações com ferramentas de integração e entrega contínua CI/CD exemplo do Jenkins usando a pipeline do projeto.
* Por ser popular tem uma comunidade e documentações disponíveis, para consulta e aprendizado.

**Desvantagens:**

* Compatibilidade entre as versões dos sistemas como iOS e Android podem ocorrer problemas entre as versões
* Como o Appium depende do WebDriver podem ocorrer problemas de tratamentos em dispositivos específicos
* A configuração inicial do Appium para iniciantes pode ser complexa em alguns aspectos como por exemplo a instalação de dependências, configurações, instalação de emuladores e configuração em dispositivos.
* O desempenho nos testes quando estamos usando o Appium precisam ser analisados, pois podem ocorrer lentidão nos testes quando existe uma sobrecarga entre o servidor do Appium e o dispositivo que estamos testando.

**Serviços ou Plugins que serão usados no projeto**

* Integração contínua e Entrega contínua como o Jenkins
* Browserstack para testes dos dispositivos na nuvem
* TestNG para o report de testes e resultados dos testes realizados

**Framework:** Espresso

**Linguagem de programação:** Java

**Vantagens:**

* Por ser desenvolvido pelo Google se integra ao Android Studio, com fácil escrita e depuração dos testes direto na IDE de desenvolvimento.
* Documentação e atualizações são regulares por se tratar de uma ferramenta do Google e apoio da comunidade
* A performance e execução dos testes são de alta performance, para uma rápida execução.
* API simples e de fácil aprendizado
* O Espresso tem uma sincronização automática com a interface do usuário, reduzindo a necessidade de usar diversos ‘delays’ nos testes

**Desvantagens:**

* Por ser um framework desenvolvido pelo Google, não é possível usa-lo para testes mobile em aplicações iOS e multiplataformas
* Pode haver dificuldades no aprendizado inicial caso não houver familiaridade com o Android Studio e testes de mobile
* Comparado com outros frameworks como o Appium por exemplo, oferece suporte a diversas linguagens e plataformas. O Espresso é limitado ao Android e à linguagem Java.
* Tem dependência com o Android Instrumentation que torna as configurações complexas.

**Serviços ou Plugins:**

* Integração contínua e Entrega contínua como o Jenkins
* Browserstack para testes dos dispositivos na nuvem
* TestNG para o report de testes e resultados dos testes realizados
* Firebase

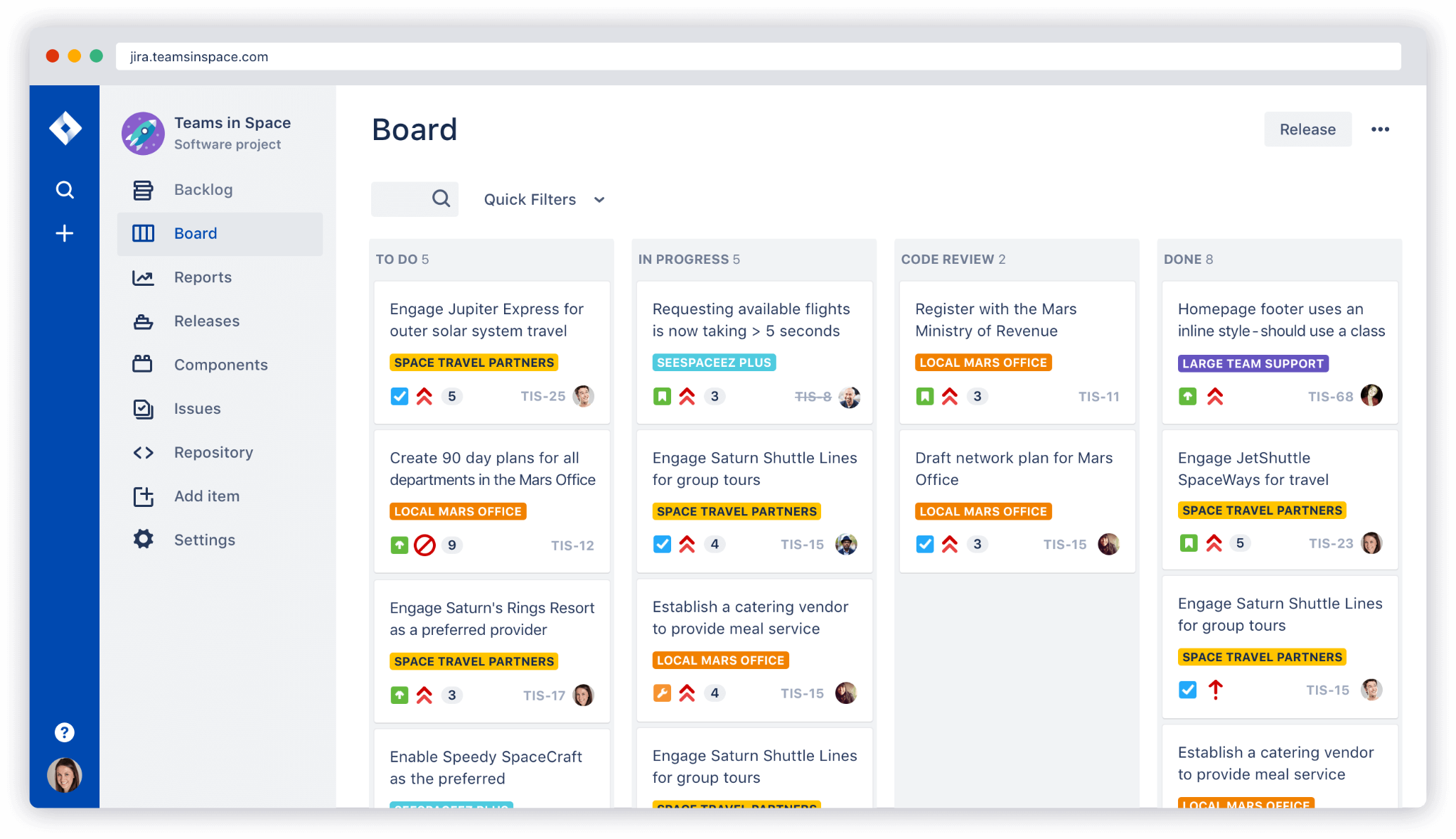
**3. Escolha da ferramenta de gerenciamento de testes**

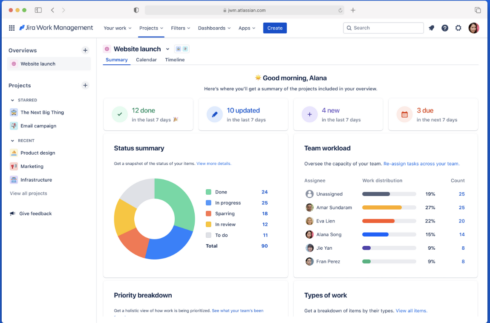
1. **JIRA-Atlantis**
2. **Testlink**
3. **qTest**

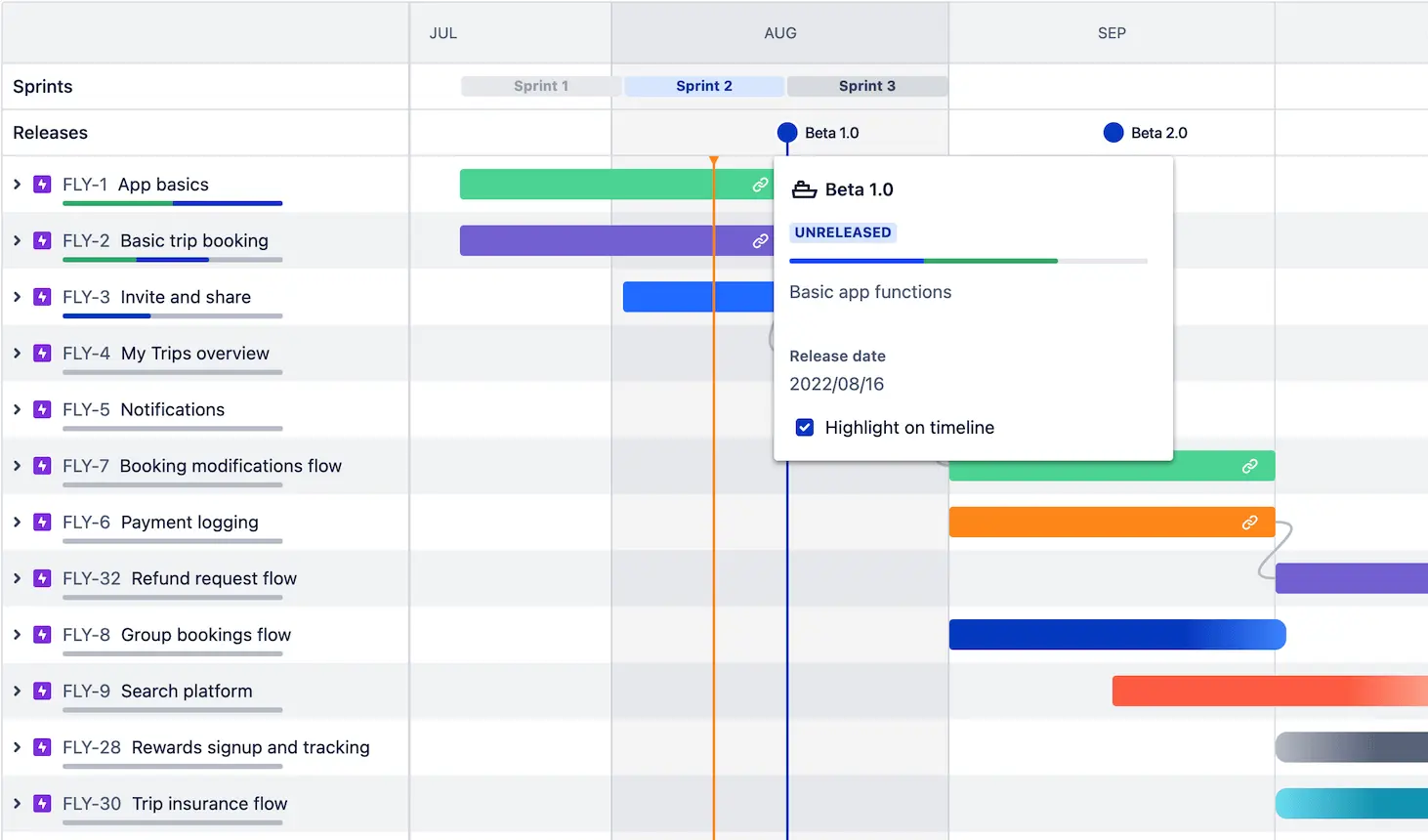
**Ferramenta:** Jira Atlatis

**Vantagens:**

* Permite o rastreamento de defeitos, requisitos e casos de testes facilitando a gestão de testes
* É possível personalizar dashboards, criação de filtros, dando uma visão clara para o time na tomada de decisão, facilitando a colaboração
* Relatórios podem trazer de forma detalhada para análise da cobertura de testes, progresso e desempenho do time
* É possível criar automações que podem ser integradas com Selenium, TestNG e outras ferramentas
* Plugins de integração com outras ferramentas
* Pode ser escalável conforme o crescimento de time
* Uso de inteligência artificial IA, na criação de bugs e refinamento de casos de testes







**Desvantagens:**

* Configuração pode ser custosa e requer conhecimento
* Custos de licenciamento elevados para pequenas empresas e times e até para grandes organizações
* Requer conexão com a internet para se acessar tickets não sendo possível o acesso offline
* Em projetos grandes/organizações onde diversos times usam o tempo todo, pode ter uma performance lenta e apresentar lentidão em algumas funcionalidades como extração de relatórios e dashboards com muitas informações

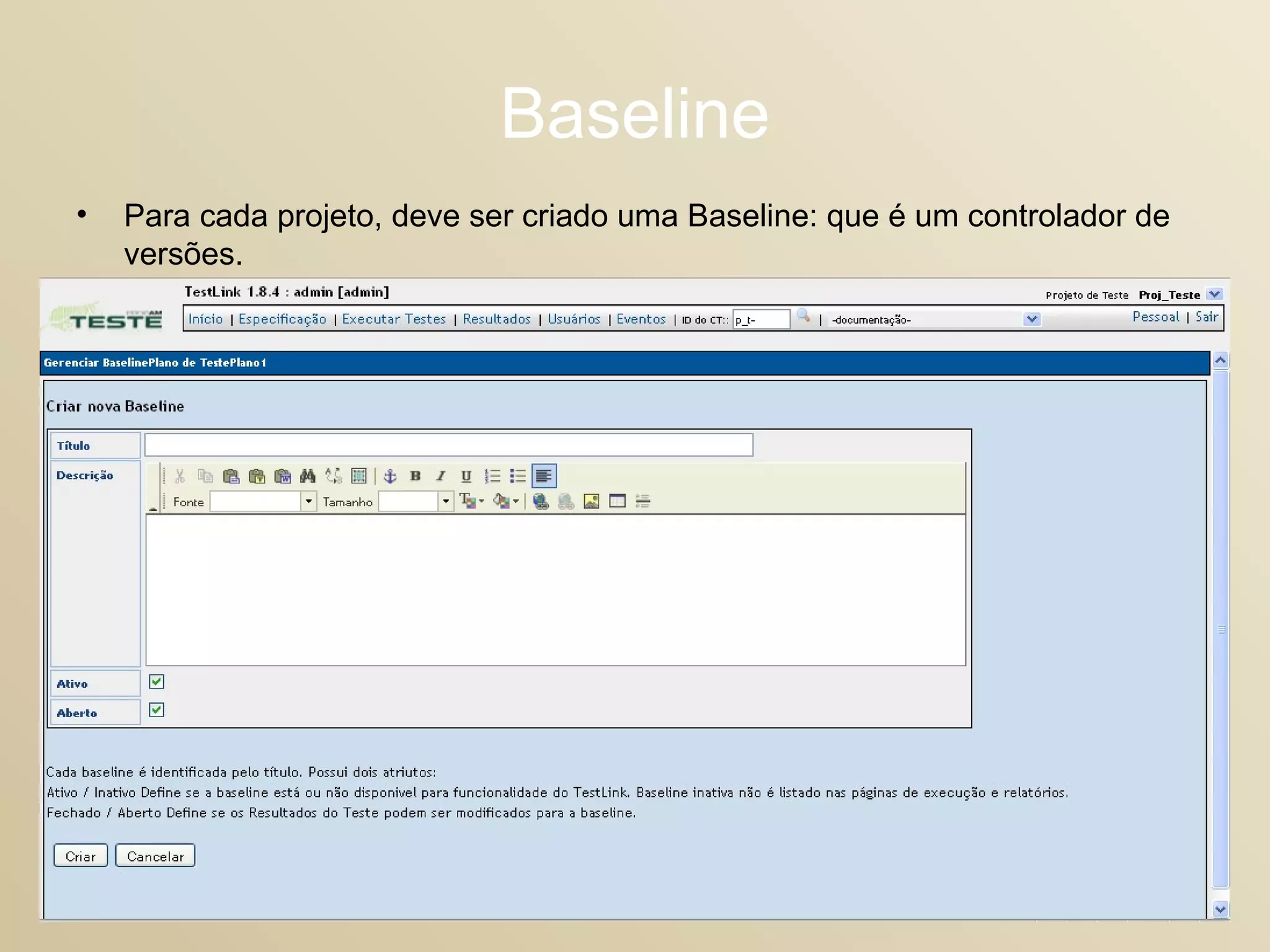
**Ferramenta:** TestLink

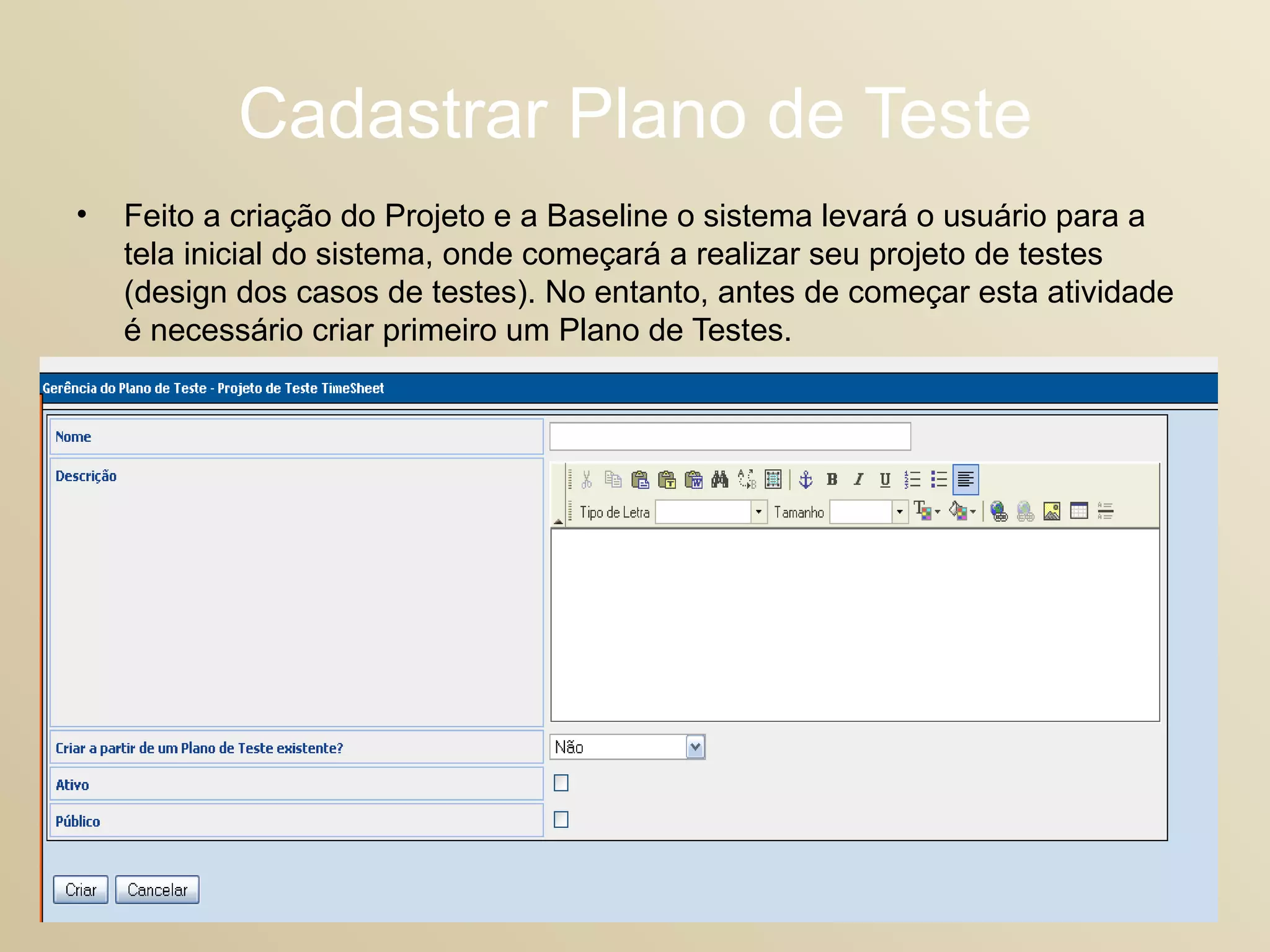
**Vantagens:**

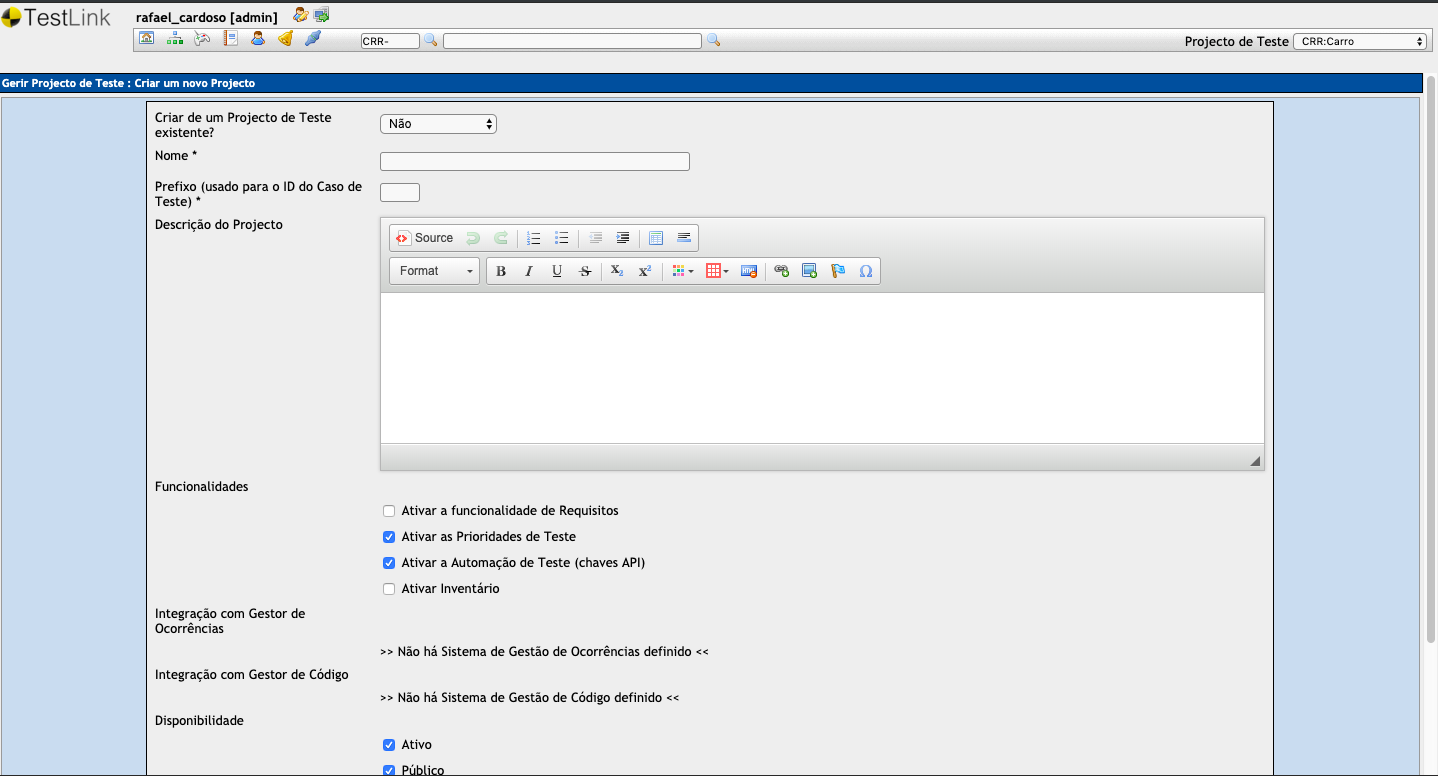
* TestLink é uma ferramenta open source que seu foco é o gerenciamento de testes e casos de testes e pode ser customizado
* É possível se manter todo o histórico dos casos de testes criados, dando a visibilidade do rastreio
* Integração com o Jira e Mantis
* Relatórios podem ser customizado para a extração de métricas e análise de melhorias
* Tem suporte para automação de testes manuais e automatizados
* Colaborações entre diversos membros do tipo
* Pode ser instalado localmente e acesso dentro de uma rede
* Funcionalidades robustas para rastreamento de requisitos e organização de casos de teste

**Desvantagens:**

* Interface é antiga e desatualizada o que pode gerar estranheza por ser pouco intuitiva
* Por ser uma ferramenta open source o suporte é limitado o que pode prejudicar caso houver problemas com a ferramenta
* Manutenção e novas atualizações não são constantes
* Em grandes projeto pode ocorrer lentidão quando se tem um número elevado de casos de testes
* Instalação complexa já que exige conhecimento técnico



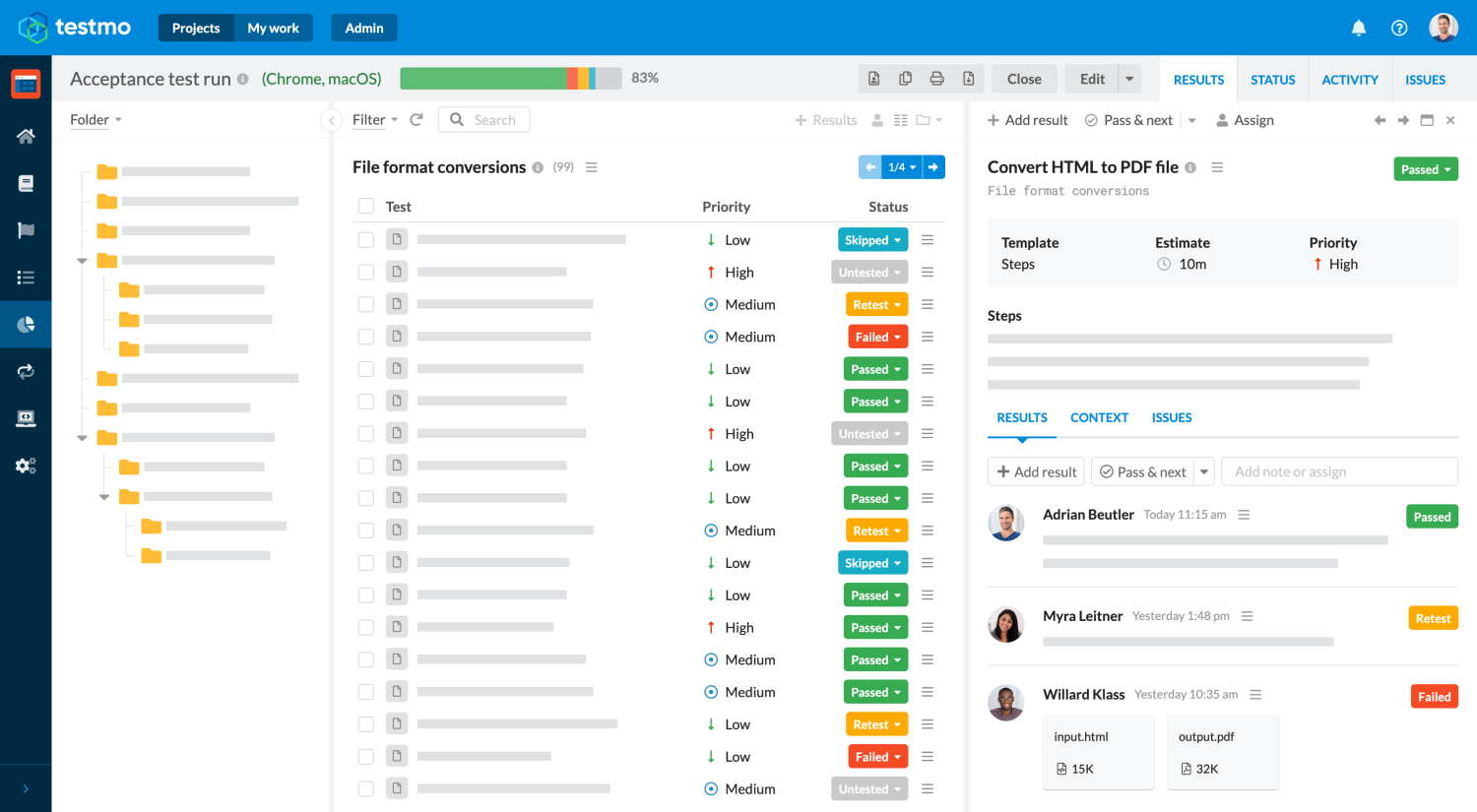




**Ferramenta:** qTest

**Vantagens:**

* Interface simples e fácil entendimento
* Dashboards mostra uma visão da análise do progresso de testes
* É possível integrar Jira e Jenkins
* Rastreio de casos de testes e seus bugs podem ser feitos vinculando as informações
* É possível criar relatórios e customizar para monitorar o progresso nos testes
* Colaboração entre as equipes de testes
* Pode ser escalável conforme o aumento do projeto e suas equipes





**Desvantagens:**

* Custo elevado para pequenas e grandes equipes
* Recursos adicionais podem requerer outros custos não planejados
* Por ser uma ferramenta de pouco conhecimento e familiaridade pode haver custo e tempo para o treinamento
* Requer acesso à internet
* Em grandes projeto pode ocorrer lentidão e pode ocorrer dependência de suporte

**Ferramenta escolhida:** TestLink

**Motivo:**

TestLink devido ao custo ser baixo e por se tratar de uma ferramenta open source.

Existe uma preocupação inicial quanto ao custo do projeto e por ser uma ferramenta com todas as opções e solução básica para o gerenciamento de testes, optamos pelo uso do TestLink.

É possível se criar e customizar dashboards e relatórios para o acompanhamento de testes, progresso e entregas.

Funcionalidades robustas para rastreamento de requisitos e organização de casos de teste, já que caso de testes e possíveis problemas ficam registrados em uma espécie de histórias.

Pensando em um futuro, caso houver a necessidade de se integrar ao Jira, o TestLink oferece essa integração.

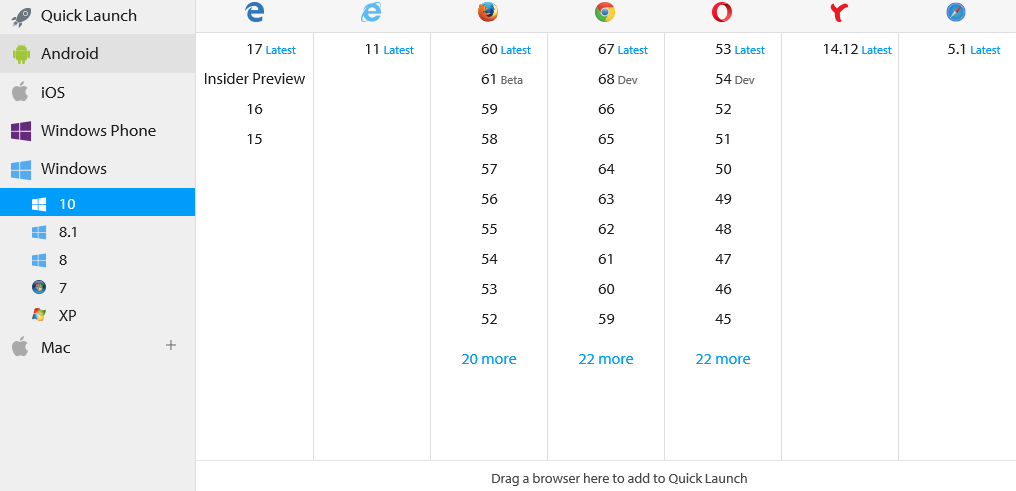
**4. Escolha do Device Cloud**

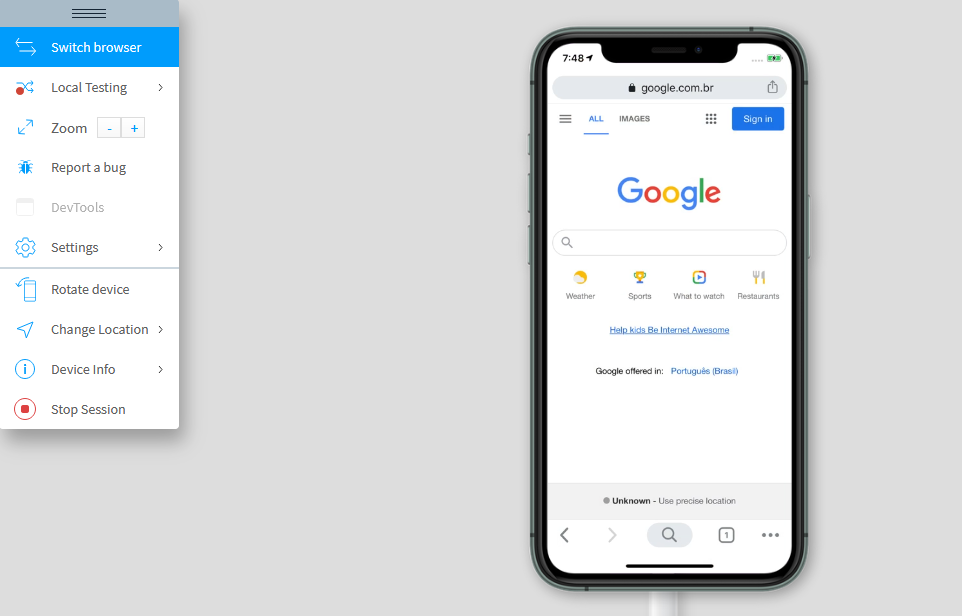
1. **BrowserStack**
2. **SauceLabs**

**Device Cloud:** BrowserStack

**Vantagens:**

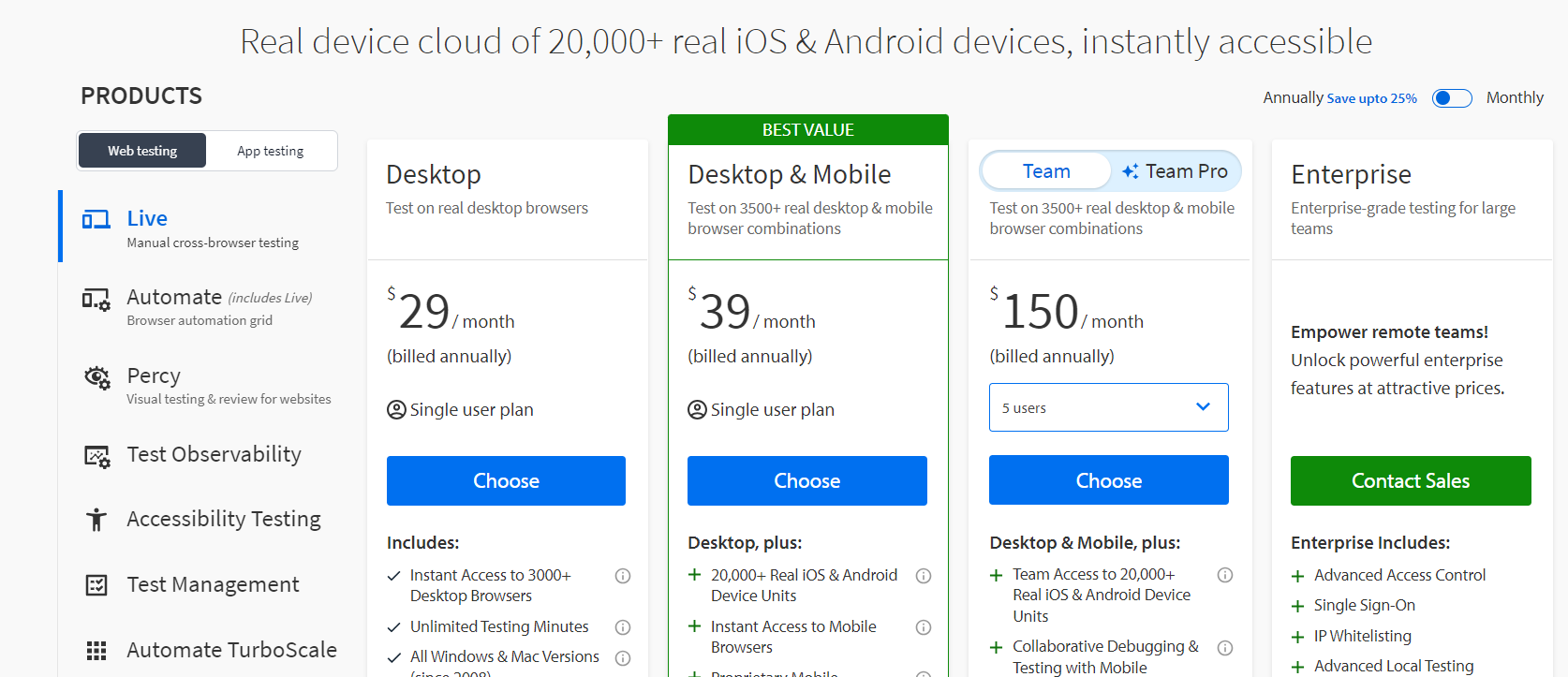
* Ampla variedades de dispositivos como smartphones, tablets e computadores
* Disponível diversos browsers para os testes onde podemos garantir a qualidade dos testes
* Versões antigas de sistemas operacionais
* É possível automatizar os testes usando Appium, Cypress e outros frameworks
* Integração com a pipeline





**Desvantagens:**

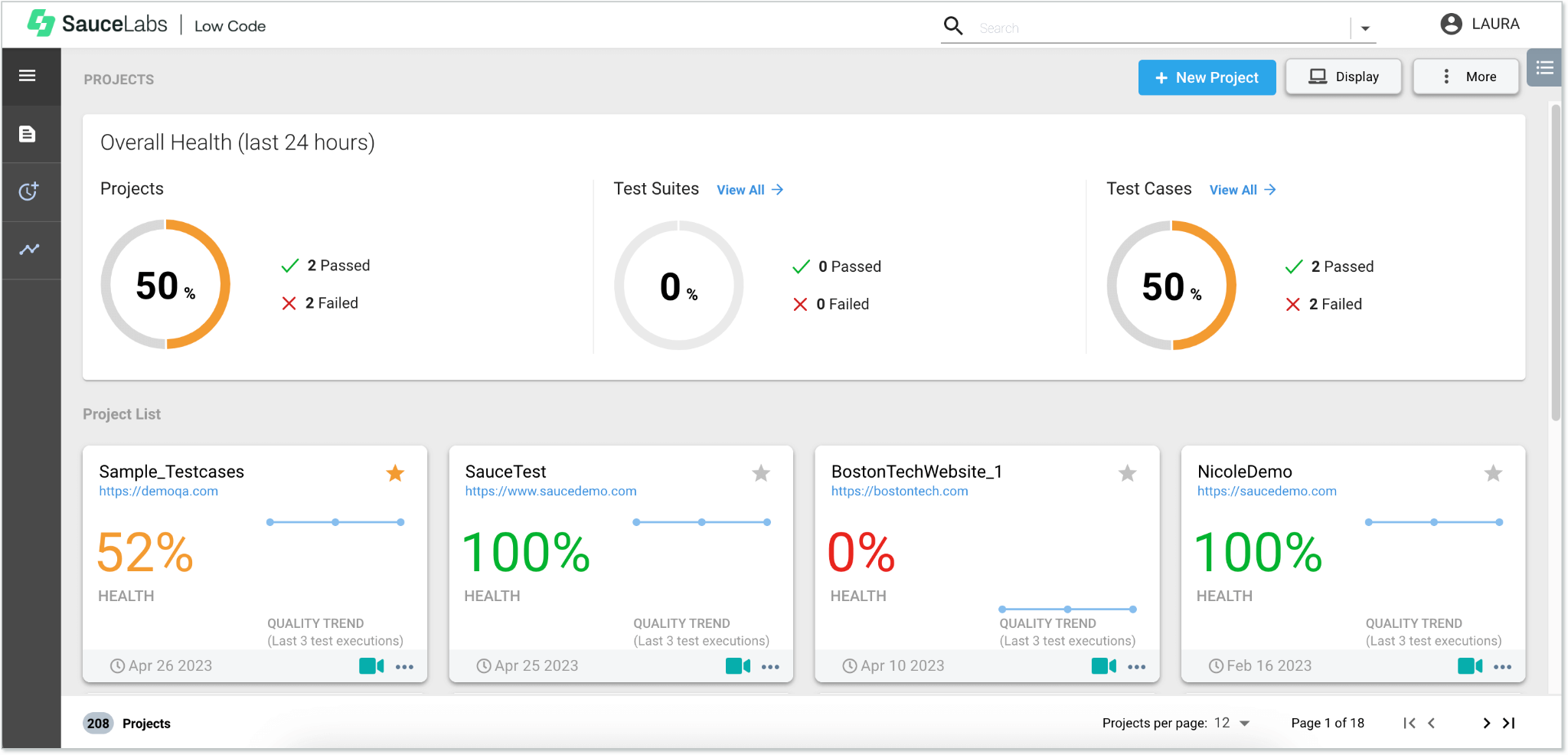
* Custo pode ser muito alto já que é uma assinatura por usuário e podem aumentar com a necessidade de mais dispositivos
* É necessário acesso a internet e isso pode levar a um desempenho lento
* Configurações e alterações nas configurações dos dispositivos não é permitido



**Device Cloud:** SauceLabs

**Vantagens:**

* Ampla variedades de dispositivos como smartphones, tablets e computadores
* Disponível diversos browsers para os testes onde podemos garantir a qualidade dos testes
* Integração com a pipeline
* Versões antigas de sistemas operacionais
* É possível automatizar os testes usando Appium, Cypress e outros frameworks
* É possível capturar as screenshots
* Geração de relatórios sobre possíveis falhas e desempenho dos testes



****

**Desvantagens:**

* Custo pode ser muito alto já que é uma assinatura por usuário e podem aumentar com a necessidade de mais dispositivos
* É necessário acesso a internet e isso pode levar a um desempenho lento
* Configurações e alterações nas configurações dos dispositivos não é permitido

**Referências**

1. Usando Robot Framework para automação de testes web e mobile | by Caio Andrian - <https://caioandrian.medium.com/usando-o-robot-framework-para-automa%C3%A7%C3%A3o-de-testes-web-e-mobile-6d7e4182649d>
2. 5 melhores ferramentas de automação para testar aplicativos Android (Android App Testing Tools) - <https://www.linkedin.com/pulse/5-melhores-ferramentas-de-automa%C3%A7%C3%A3o-para-testar-android-alan-silva/>
3. Automação de testes mobile - <https://www.linkedin.com/pulse/automa%C3%A7%C3%A3o-de-testes-mobile-bruno-de-oliveira-silva/>
4. Teste Mobile: Vantagens e Frameworks para Garantir a Qualidade de Aplicativos Móveis | by Bruno Graciano Alves Carneiro | Medium - <https://medium.com/@brunogracianoalvescarneiro/teste-mobile-vantagens-e-frameworks-para-garantir-a-qualidade-de-aplicativos-m%C3%B3veis-e5a4b1c21024>
5. APPIUM Tutorial para Android Teste de aplicativos móveis para iOS - <https://www.guru99.com/pt/introduction-to-appium.html>
6. Appium: automação para apps mobile | by Bruno Carvalho | assert(QA) | Medium - <https://medium.com/assertqualityassurance/appium-automa%C3%A7%C3%A3o-para-apps-mobile-a256db64a27d>
7. Appium - <http://appium.io/docs/en/latest/>
8. <https://stackoverflow.com/questions/tagged/appium>
9. Jenkins Documentation - <https://www.jenkins.io/doc/>
10. BrowserStack Documentation - <https://www.browserstack.com/guide/importance-of-device-farms>
11. TestNG Documentation - <https://testng.org/>
12. <https://developer.android.com/training/testing/espresso?hl=pt-br>
13. Espresso Android Testing for UI Automation : Tutorial | BrowserStack - <https://www.browserstack.com/guide/espresso-android-testing>
14. Automação do Android: o guia de ponta a ponta | blog | Digital.ai - [https://digital.ai/pt/catalyst-blog/android-automation-the-complete-end-to-end-guide](https://digital.ai/pt/catalyst-blog/android-automation-the-complete-end-to-end-guide/)
15. <https://medium.com/@jeffersontpadua/kotlin-espresso-testes-automatizados-e-expressivos-para-android-65b9a7e4b73a>
16. <https://yaman.com.br/pt-br/blog/automatizacao-de-testes-mobile-suas-aplicacoes-mais-eficientes/>
17. Automação de testes para aplicações móveis em Android – Parte 1 - <https://www.devmedia.com.br/automacao-de-testes-para-aplicacoes-moveis-em-android-parte-1/32708>
18. Automação de testes para aplicações móveis em Android – Parte 1 - <https://www.devmedia.com.br/automacao-de-testes-para-aplicacoes-moveis-em-android-parte-1/32708>
19. <https://testautomationu.applitools.com/espresso-mobile-testing-tutorial/>
20. <https://developer.android.com/training/testing?hl=pt-br>
21. <https://stackoverflow.com/questions/tagged/android-espresso>
22. <https://www.marcelodebittencourt.com/livros/Livro_GestaoDeTestesEDefeitosComTestlinkEMantis_MarceloDeBittencourt.pdf>
23. TestLink - Quick Guide - <https://www.tutorialspoint.com/testlink/testlink_quick_guide.htm>
24. Testlink — uma ferramenta de gerenciamento de testes de software. | by Luana Reis | Medium - <https://medium.com/@luanasreis/testlink-uma-ferramenta-de-gerenciamento-de-testes-de-software-44001b816f64>
25. Trabalhando com o TestLink - <https://www.devmedia.com.br/trabalhando-com-o-testlink/34394>
26. <https://www.atlassian.com/br/software/jira?aceid=&adposition=&adgroup=143040554285&campaign=19324540271&creative=642122380510&device=c&keyword=jira&matchtype=e&network=g&placement=&ds_kids=p74609451296&ds_e=GOOGLE&ds_eid=700000001558501&ds_e1=GOOGLE&gad_source=1&gclid=CjwKCAjwyJqzBhBaEiwAWDRJVCs3qQvZ7qu0x_5KAMTKHR6Lp6m0pqjvzVlm74ZeNJ4HHedUU9_sIBoCgH4QAvD_BwE&gclsrc=aw.ds&tab=boards>
27. <https://confluence.atlassian.com/alldoc/atlassian-documentation-32243719.html>
28. qTest Documentation - <https://documentation.tricentis.com/qtest/od/en/content/resources/home.htm>
29. qTest Insights - <https://www.tricentis.com/products/unified-test-management-qtest/visual-reporting-analytics-insights>
30. <https://www.meticulous.ai/blog/browserstack-vs-saucelabs>
31. <https://saucelabs.com/sauce-labs-vs-browserstack>
32. <https://www.browserstack.com/guide/importance-of-device-farms>
33. <https://saucelabs.com/resources/blog/device-farm-vs-device-cloud>
34. <https://docs.saucelabs.com/overview/>