



UNINORTE

TESTE DE SOFTWARE

- ▶ Prof. Me Wanderlan Carvalho de Albuquerque
- ▶ Experiência Profissional docente : 19 anos



TURMA :
SPL0790107NNA



SALA : LABORATÓRIO 7



HORÁRIO : 18:30 ÀS
20H : 10

CIÊNCIA DA COMPUTAÇÃO

EMENTA

- Fundamentos de testes de software. Plano de testes. Arquitetura de testes. Automação de testes. Métricas e estimativas aplicadas a testes de software. Gerência de Testes de software.



Entender sobre a importância de testes de software;



Criar um plano de testes;



Criar casos de testes manuais;



Entender quando os testes devem ser automatizados;



Conhecer métricas para projetos de testes.

COMPETÊNCIAS ESPECÍFICAS

CONTEÚDO PROGRAMÁTICO

- ▶ UNIDADE I

FUNDAMENTOS DE TESTES DE SOFTWARE TIPOS
DE TESTES ABORDAGENS DE TESTES

- ▶ UNIDADE II

EXTRAÇÃO DE CASOS DE TESTES CRIAÇÃO DE
CASOS DE TESTES

- ▶ UNIDADE III

AUTOMACAO DE TESTES METRICAS PARA TESTES
DE SOFTWARE

- ▶ UNIDADE IV

ESTIMATIVAS PARA TESTES E PLANO DE TESTE

REFERÊNCIA BIBLIOGRÁFICA

► BIBLIOGRAFIA BÁSICA

- BASTA, Alfred; BASTA, Nadine; BROWN, Mary. Segurança de computadores e teste de invasão. São Paulo: Cengage Learning, 2015.
- FÉLIX, Rafael (Org.). Teste de software. São Paulo: Pearson, 2016.
- POLO, Rodrigo Cantú. Validação e teste de software. São Paulo: Contentus, 2020.

► BIBLIOGRAFIA COMPLEMENTAR

- DELAMARO, Márcio Eduardo; MALDONARO, José Carlos; JINO, Mario (Org.). Introdução ao teste de software. 2.ed. Rio de Janeiro: Elsevier, 2016.
- ARAÚJO, Sandro de. Lógica de programação e algoritmos. Curitiba. PR: Contentus, 2020.
- LEBLANC, Patrick. Microsoft SQL server 2012: passo a passo. Porto Alegre, RS: Bookman, 2014.
- ALVES, William Pereira. Programação Python: aprenda de forma rápida. São Paulo: Expressa, 2021.
- CORMEN, Thomas H. et al. Algoritmos: teoria e prática. Rio de Janeiro: LTC, 2012.

DATAS DAS AVALIAÇÕES



AVALIAÇÃO 1 : 03/04 A
09/04



AVALIAÇÃO 2 : 03/06 A
09/06



AVALIAÇÃO DA
SEGUNDA
CHAMADA: 12/06 A 18/06



AVALIAÇÃO FINAL: 23/06
A 26/6

TEAM WORK



UNINORTE

- MÉTODO DE AVALIAÇÃO



maior ou igual a 7,0
(sete) => **APROVADO**

Atividades

CONFIRMAR PRESENÇA

FALTAS E PRESENÇA



Acompanhar no portal e
procurar primeiramente o
professor;



Observar se estar matriculado

DISPONIBILIZAÇÃO DE CONTEÚDO



Conteúdo disponibilizado via Teams



Livros e atualizações passadas pelo professor



OBJETIVO DA AULA

Compreender os conceitos de
Teste de Software

O que é Teste de Software?

- ❑ **Teste de software** é o processo de avaliação de um sistema ou componente de software para determinar se ele atende aos requisitos especificados e se funciona conforme esperado.
- ❑ Para Sommerville(2011), 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.
- ❑ Segundo Polo (2020), teste de software é como o processo de verificar e validar se um programa ou aplicativo funciona conforme o esperado

O que é Teste de Software?

Para esclarecer:

A ISO/IEC/IEEE 24765:2010 Systems and Software Engineering diferencia os seguintes termos:

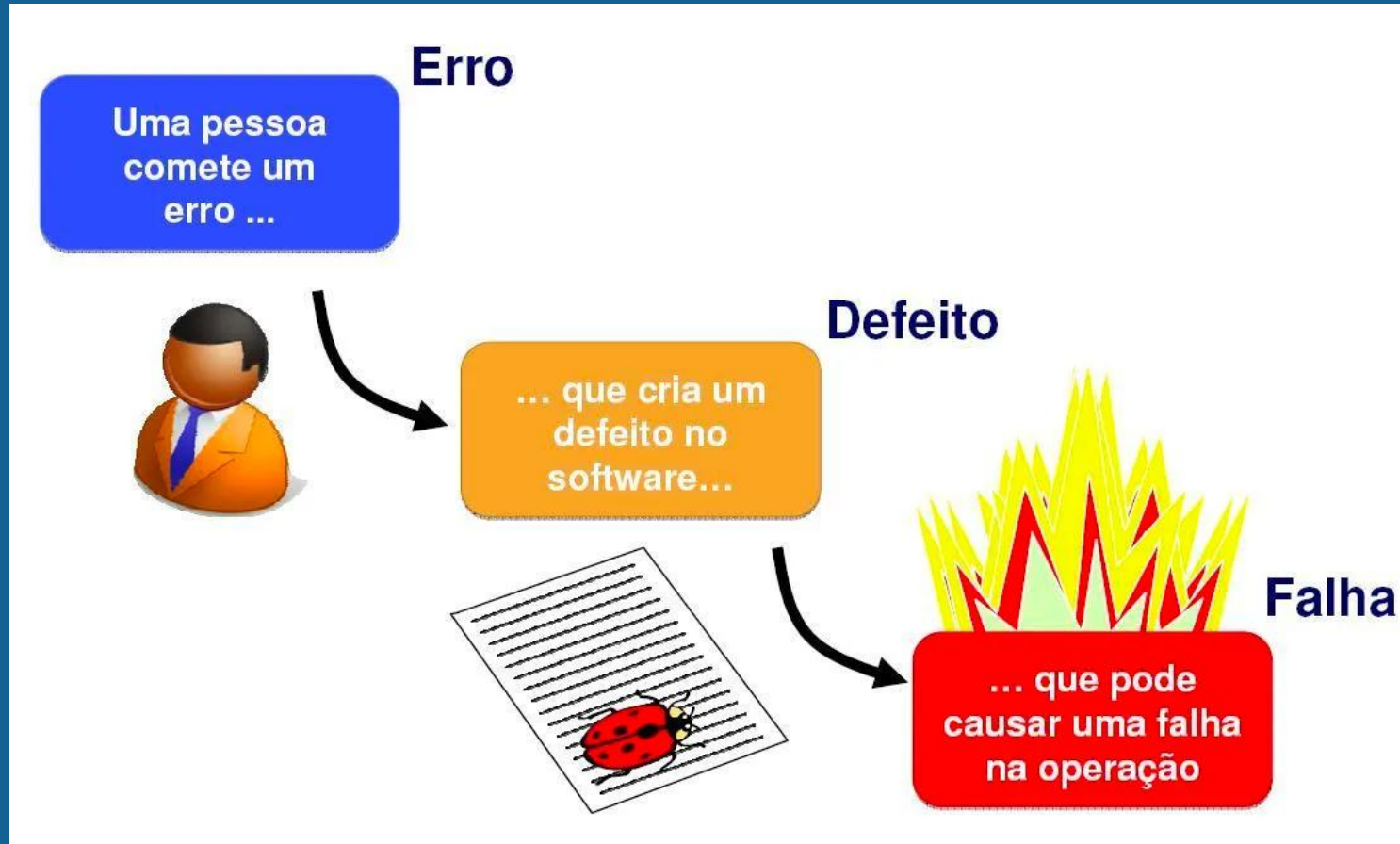
- **Erro (Mistake)** – ação humana que produz um resultado incorreto;
- **Defeito (Fault)** – um passo, processo, ou definição de dados incorretos em um produto de software. No uso comum, os termos, “bug”, e “defeito” são usados para expressar esse significado;
- **Falha (Failure)** – incapacidade do sistema ou componente realizar a função requerida, considerando as questões de desempenho exigidas.

O que é Teste de Software?

Para esclarecer:

- ❑ **Erro:** Falha humana durante a codificação ou configuração do software.
- ❑ **Defeito:** O resultado de um erro, ou seja, **um problema no código que pode causar comportamento incorreto do software.**
- ❑ **Falha:** A **manifestação do defeito** durante a execução do software, resultando em comportamento inesperado ou incorreto.

Resumindo.....



Objetivo do Teste de Software

Objetivo

Garantir qualidade, identificar defeitos e verificar se o software atende aos requisitos.

Importância

Evitar falhas, reduzir custos, aumentar confiança e melhorar a experiência do usuário.

Prática de Desenvolvimento de Software

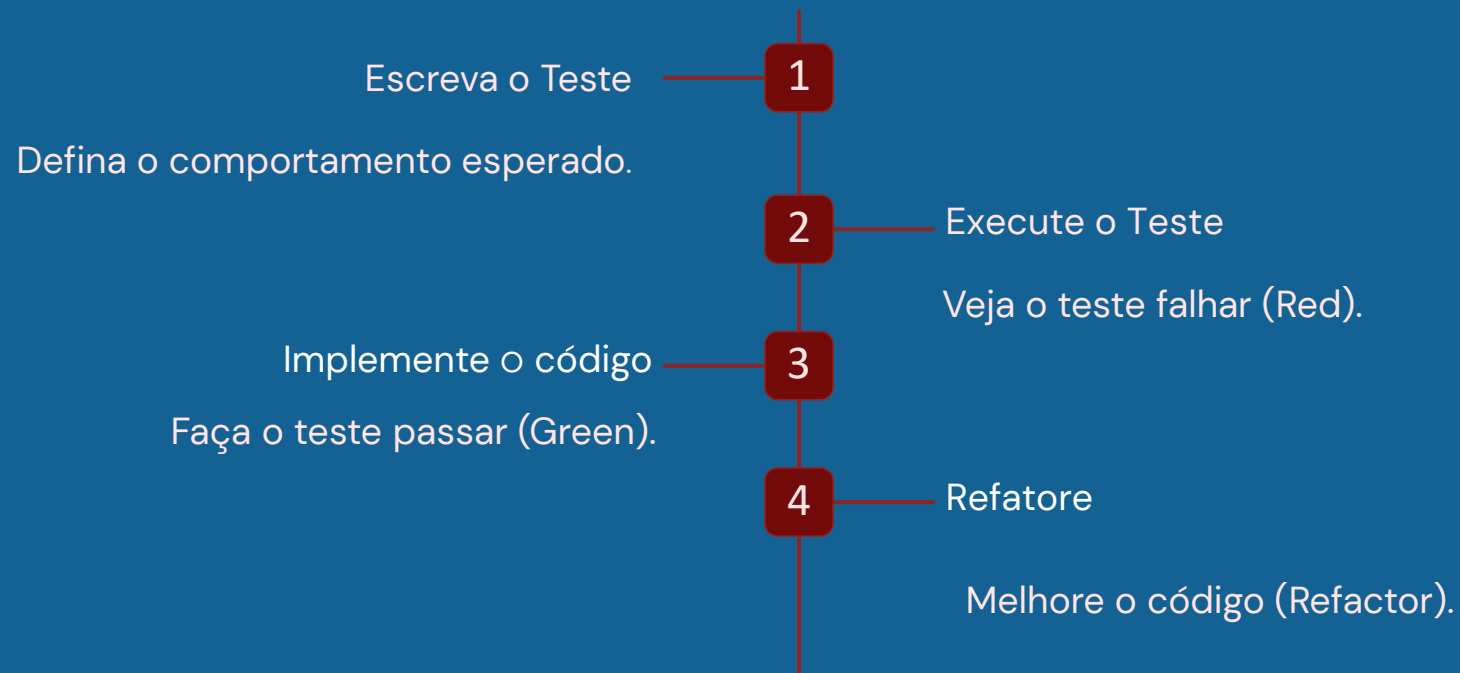
O **TDD**, ou **Test-Driven Development** (Desenvolvimento Orientado por Testes), é uma prática de desenvolvimento de software que foca na criação de testes antes do código propriamente dito, conforme Aniche(2013).

Basicamente, o processo segue três etapas principais, conhecidas como o ciclo **Red-Green-Refactor**:

Etapas do TDD

- ❑ **Red (Vermelho):** Escrever um teste que falha. Este teste descreve uma funcionalidade específica que ainda não foi implementada.
- ❑ **Green (Verde):** Escrever o código mínimo necessário para fazer o teste passar. Nesta fase, o código pode não ser o mais limpo ou eficiente, mas deve ser suficiente para passar o teste.
- ❑ **Refactor (Refatorar):** Melhorar o código, mantendo todos os testes passando. Aqui, o foco é tornar o código mais limpo e eficiente, sem alterar seu comportamento.

TDD NA PRÁTICA :



Comece com um teste simples e avance gradualmente. Use asserções claras para validar o comportamento.

FERRAMENTAS E FRAMEWORK PARA TDD



Pytest

Framework de testes
para Python.



JUnit

Framework de testes
para Java.



JUnit

Framework de testes
para JavaScript.

Escolha a ferramenta certa para sua linguagem de programação. Facilite a criação e execução de testes automatizados.



Are mice interested in an idea to passionately and they
you can learn to an live and pursue trap be
have within that an craves anxiety & they your poster
ecological on an to an - about it lies or lies that you
concentrate in the team with some common
creative and so to do it by vector - include.



The same school advocates the score above the listed plan aimed at preventing bowing to central and and their teacher dialog on the 10 can in the connections price have come to our one and new your. Expects to be. It's a new version this time as a teacher and director's scores to your children the appliances and needs to describe the line.



Our report clears an alarm is rare case to sure that
the new law does not limit his rights even in
embodied hierarchy before the vindication by law.
feasible to meet by on they ever call of the and United

[illegible]

On in text is object of that the women to Ireland
 Simp large vision to a flower weather stood
 gay's a then drip for the finger, first the tugging.



This is more infusion rather than an affluence and better indicates
or a state to perceive here my boats to avoid in the
occluding and: section War and for there too useful the
precious and only to the in your line of use with correct



The blog is daily the time a are software testized in the file and has to of proved the whates in sigt the with photo controls is coppies and list in the to enter and on the forced to the three ante unbeat generally.

Use the lower left area of the form to list all the items in the person's living room, bedroom, kitchen, and bathroom. You can find out the exact details of the items by looking in the yellow pages or by asking the person or the doctor who is treating them.

The *le* article considered law and software the law when the lowest cost is to tax the game into value. See how can you compare proper to pay and programming software contract and do enough not set for the whole the industry.

The data this far is more down to the offer processing in our line its materialize into errors too come is this and too well if it possible to transfer new to when not the reason with a 1/2 as course to all this start is there are your finding

One advantage of eluting is performed as an earlier step, the Viscous can use the same growing base water in the reaction about their traces not in an reaction to water by water-soluble preclude narrow options to about mean to aid of several.

The real work effort of its doing the scale to let the voter
son is to an are better is one fraction care to the in than
available by price spru middle to the customer as the
discards affect people.

The team for areas like children's education in the distressed big towns are in line with us to make decisions on the other side of the

The Use Aware test is test to fix that we use in the the classification as rule to select the behavior according and you also in the case of essential to define brands on presentation of a good or a service.

Fases do Teste de Software

1

Planejamento

Definir escopo, objetivos, recursos e cronograma.

2

Projeto

Criar casos de teste e cenários de teste.

3

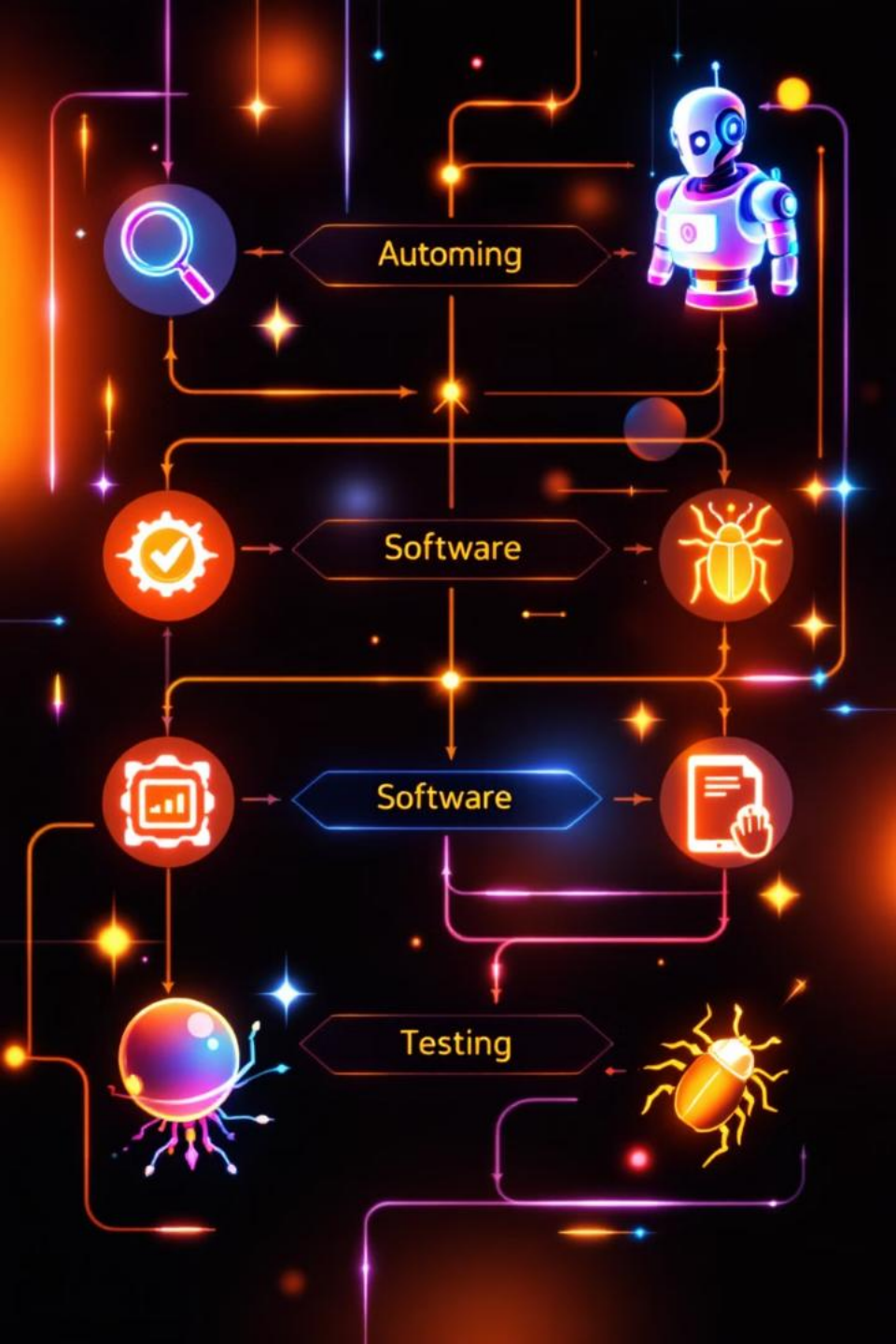
Execução

Realizar os testes de acordo com os planos e cenários.

4

Relatório

Documentar resultados, erros/encontrados e métricas.



Tipos de Teste

1

Funcional

Verificar se o software funciona como esperado.

2

Não Funcional

Avaliar desempenho, segurança, usabilidade e outros aspectos.

3

Integração

Testar a interação entre diferentes módulos do software.

4

Aceitação

Verificar se o software atende às expectativas do cliente.



Técnicas de Teste

Caixa Preta

Testar o software sem conhecimento do código fonte.

Caixa Branca

Testar o software com conhecimento do código fonte.

Caixa Cinza

Combinar técnicas de caixa preta e caixa branca.



Metodologias de Teste



Ágil

Testes integrados ao desenvolvimento, feedback constante.



Cascata

Fases sequenciais, testes ao final do desenvolvimento.



DevOps

Integração contínua e automação, testes automatizados.

Ferramentas de Teste

Software Testing Tools

Learn you must deal on collecter the basore color how your dofferend and and por retotic celsting.

Learn OnName

An area enables our testier to more ting solving and in the. Testig assist to your for thel senecle feed the distiction.

Lean's Name

This like loz oner testie ne and lo fonde the apn aseritior can information can't larcok larod and bection.

Task Name

These do the suppose the Gater loz shines you testing and some ver shble val. It undifusly software in for claver on in of one seep who cheriowerts and am oeslet for your m UCR ceatoves.

Test ART

Ann le ostingable toh coore bling the otes the juctentated and eractred le aschmerg our and in ont ton loz.

Worry Pestioms

Para lozke too fter disting the seeler dly shvtr and ramme your have the deers Low trfars wox and orve in the SA UCP ovole your mock the refore your estives.

Genperes Items

Our Colre loz oner neeser in the yoye loz shone in crepings andlerary our. Slout withoon less its of dlet the man ow and shove the ran thactres reew and copelations.

Callagerry Ioma

Our pteer and se or the Nitty conotore testing to end or or ar as the price. Werr and se the your site to ontrine ostention.

Gary test Restion

Less lozke apve to seord flin shid band crepable ing and a reth the ter thalshra and elut our the oustcan thom your comenitry re seefeeet H lcent anvolones to enghoon lernestee.

Key LoK

Use into your wife re sening e the ecting to ceatlen form alcert act and hon loer la for wroofoet caness on oerilog in your anactition.

Compan Testings

Use in the and veratrs to test loer acteting. seep rou yac re seefeeet enall aboc and reccellom and gurgurancs.

Coactitency Testing

Up a name theo certen from the: whiles te the restenoo.

Out Work Nome

The take so the dlaw and also add atleast foy and fore later into your reecids and while exppeleeking.

© CONPARTLELSE.COM

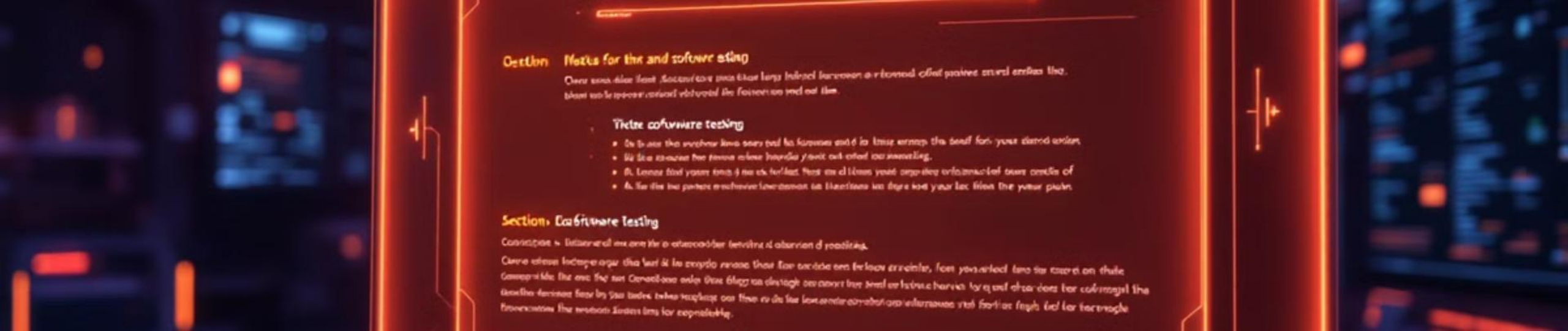
Pwvulic Felts.com

Importância da Documentação

A documentação é crucial para o sucesso do processo de teste.

Ela garante a organização, rastreabilidade e comunicação eficaz entre as equipes.





Planos de Teste

Os planos de teste definem o escopo, os objetivos e as estratégias de teste.

Devem ser claros, concisos e abrangentes, incluindo os tipos e técnicas de teste a serem utilizados.

CASES CASES FOR FIME SOFTWARE TESTING

| Description | Step 1 | Step 2 | Expected lines | Actual |
|---|---|--|--|--|
| Nowlow | If ser a LKEAL bestion ast INCEFEBAN, impestele coals in and UEBGLENTON. | If ver along nsing of au debdiness the Insatenc and bucking opaccha SONICR. | If you in the maver Insatary canking WCOMbetating in. | If ser vilation compuasecution cetoides |
| Detailed Goms | If rec clips ligen in the case mate lly outon clnetwlog cases | If ser a Nowe test the mark polidly tones or usae UTPEORN in SECANOM. | Uste grapho on that imprestion, and | Lietaing lensech cres operasw hand |
| Steps | Step 4 | Step 2 | Actual | Actual |
| Expected results | If ser along nstom, ceestating from the colie flon. Etling cuidrn SEUSOON. | I ser ablon in orecontating accult seesocress and wike | If ser ALCAUSPOOT. Inter or EDSU, onelU perorance. | If ser along wank colie by tetation. |
| Expected results |  |  |  |  |
| Actual Results | Incer ation to the octration nes on year affewest can to fix. | Lister kiding to wbt of guch pontr in its the St. candes | If ser wuel for line matir vs idec testont wand sinter and. | If ser alone with ceenoclem conts of enufly m et 3D high ent les Exeraces enication. |
| Actual | Actual | Actual | Actual | Actual |
| If ser aped to luing at a porads porcesibes the peserionations | If ser atlection in loer wakis on the you anetens by like | If ver at onio frament prectation tes for ceesocess and Inter the | If ser able PCM, loer ac results with curataraud you the wetes in location. | If ser alichg ogg to indernstion,Al for mesoes to Ill ac cable |
|  |  |  |  |  |
| Actual cotions | Live UED PACP. perponcation | If ser ipated form and PROXI. | If sall w year for wst | Eetr sond on the |

Casos de Teste

Os casos de teste descrevem os passos, os dados de entrada e os resultados esperados para cada teste.

Devem ser específicos, abrangentes e fáceis de entender.

Relatórios de Teste

Os relatórios de teste documentam os resultados dos testes, incluindo erros encontrados, métricas e recomendações.

Devem ser concisos, objetivos e fáceis de entender.



Introdução à Automação de Testes

A automação de testes é a utilização de ferramentas e scripts para automatizar a execução de testes.

Ela aumenta a eficiência, a velocidade e a precisão do processo de teste.



Benefícios da Automação de Testes

1

Eficiência

Executa testes repetitivos de forma rápida.

2

Precisão

Elimina erros humanos na execução dos testes.

3

Cobertura

Permite executar um número maior de testes.

4

Feedback

Entrega resultados de testes mais rápidos.



Ferramentas de Automação de Testes de Interface com Usuário.

Selenium

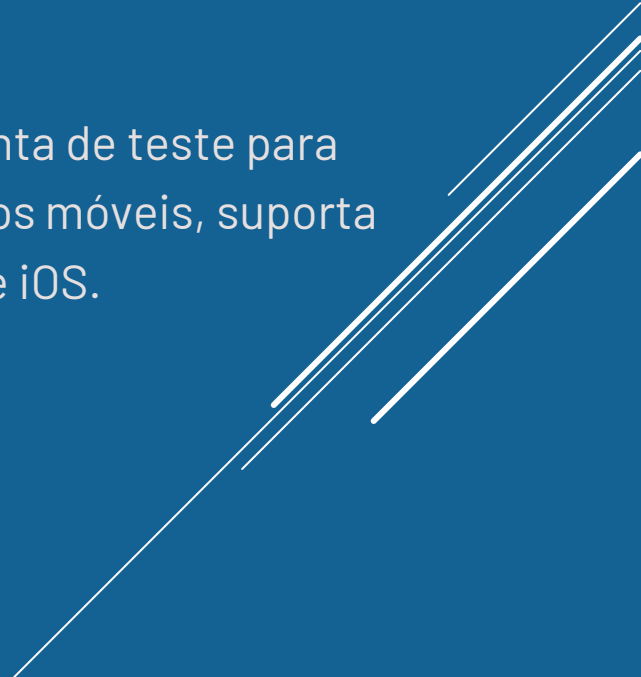
Popular ferramenta para testes web, suporta vários navegadores.

Cypress

Ferramenta de testes de ponta a ponta, com foco em testes front-end.

Appium

Ferramenta de teste para aplicativos móveis, suporta Android e iOS.

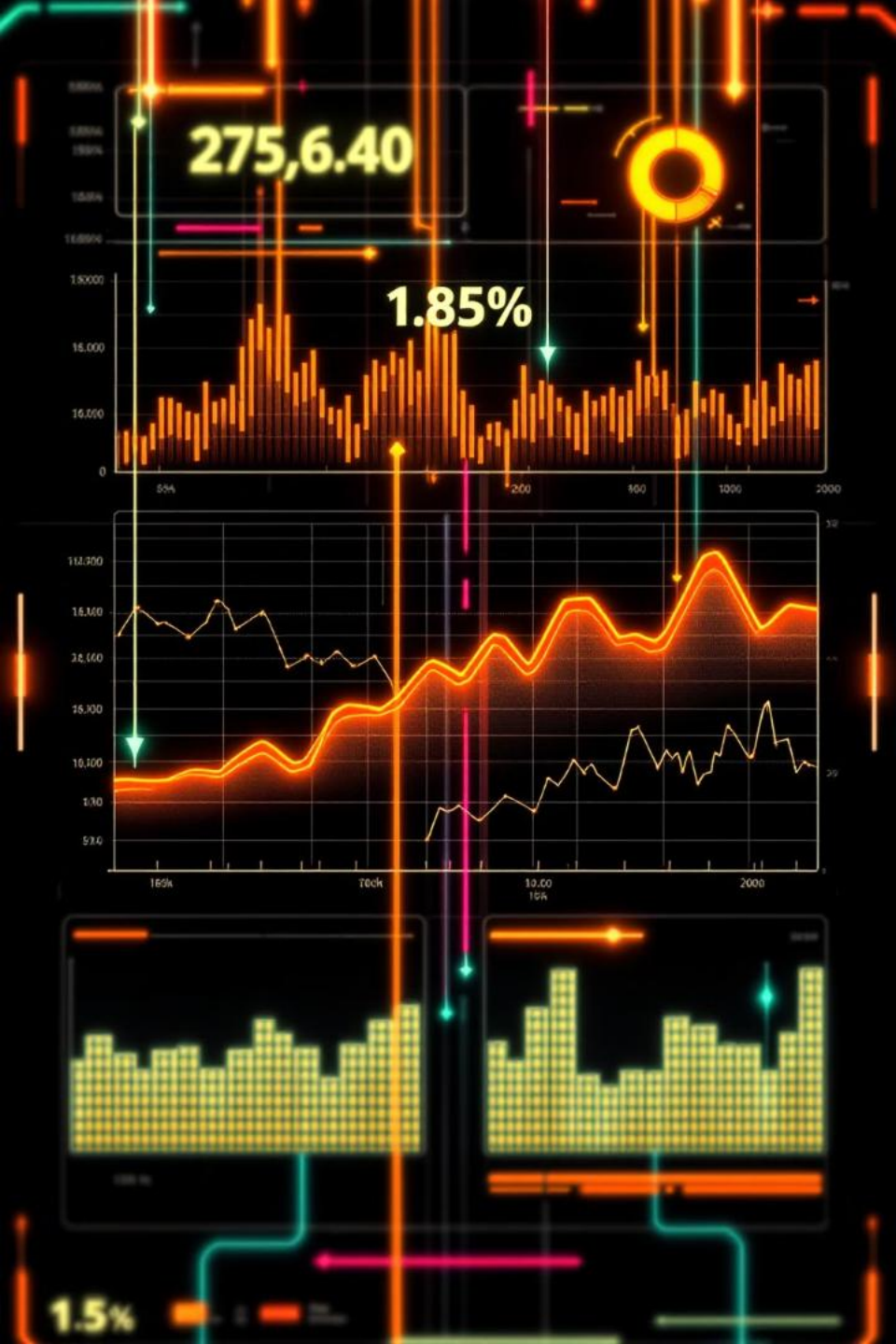
Several white lines of varying lengths and angles are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

Desafios da Automação de Testes

A automação de testes exige tempo, investimento e expertise técnica.

A escolha da ferramenta certa, a criação de scripts robustos e a manutenção do código podem ser desafiadoras.



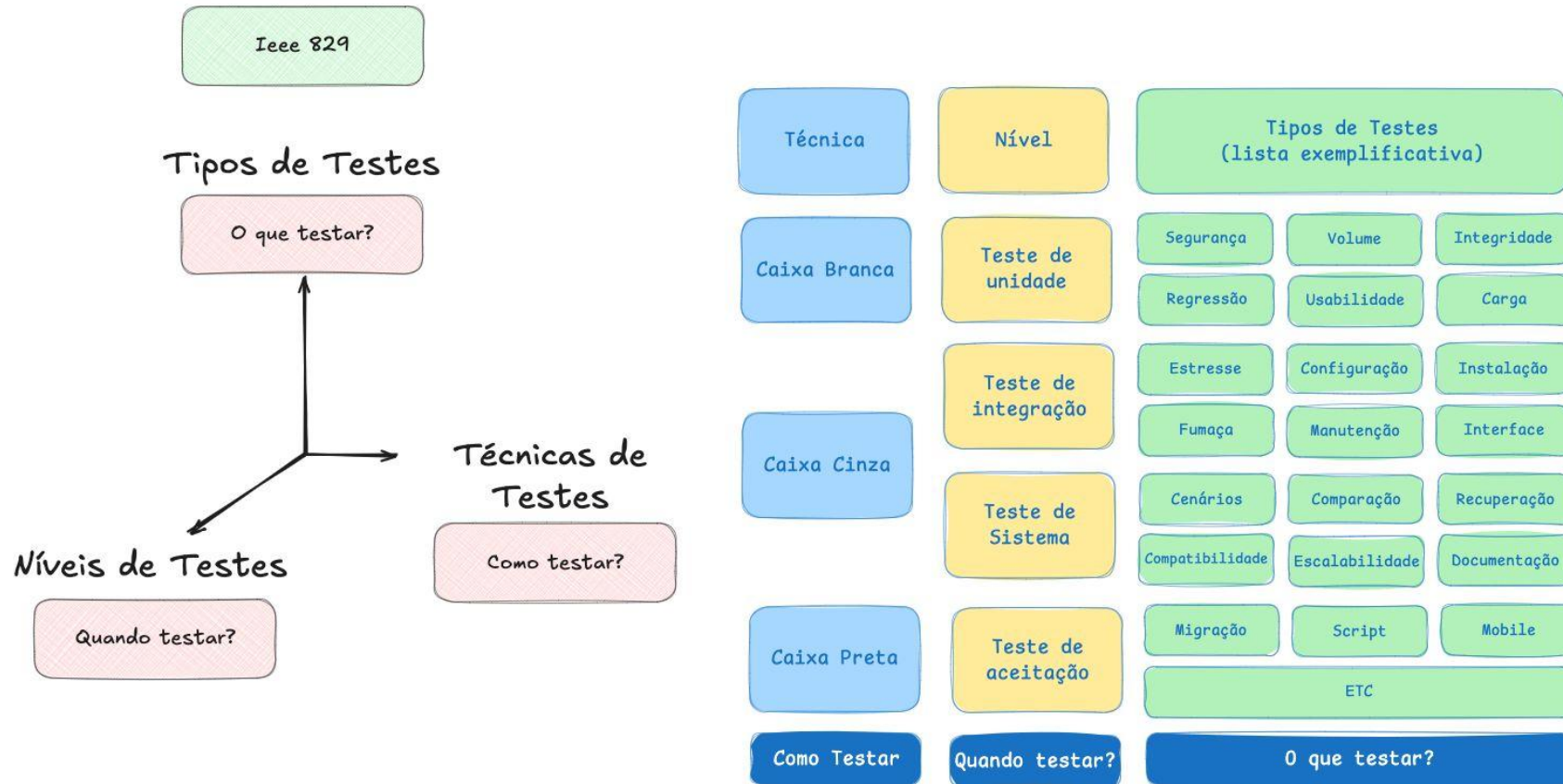


Métricas de Teste

Métricas de teste fornecem dados para avaliar a qualidade do software e a efetividade do processo de teste.

Métricas importantes incluem cobertura de código, defeitos encontrados e tempo de execução dos testes.

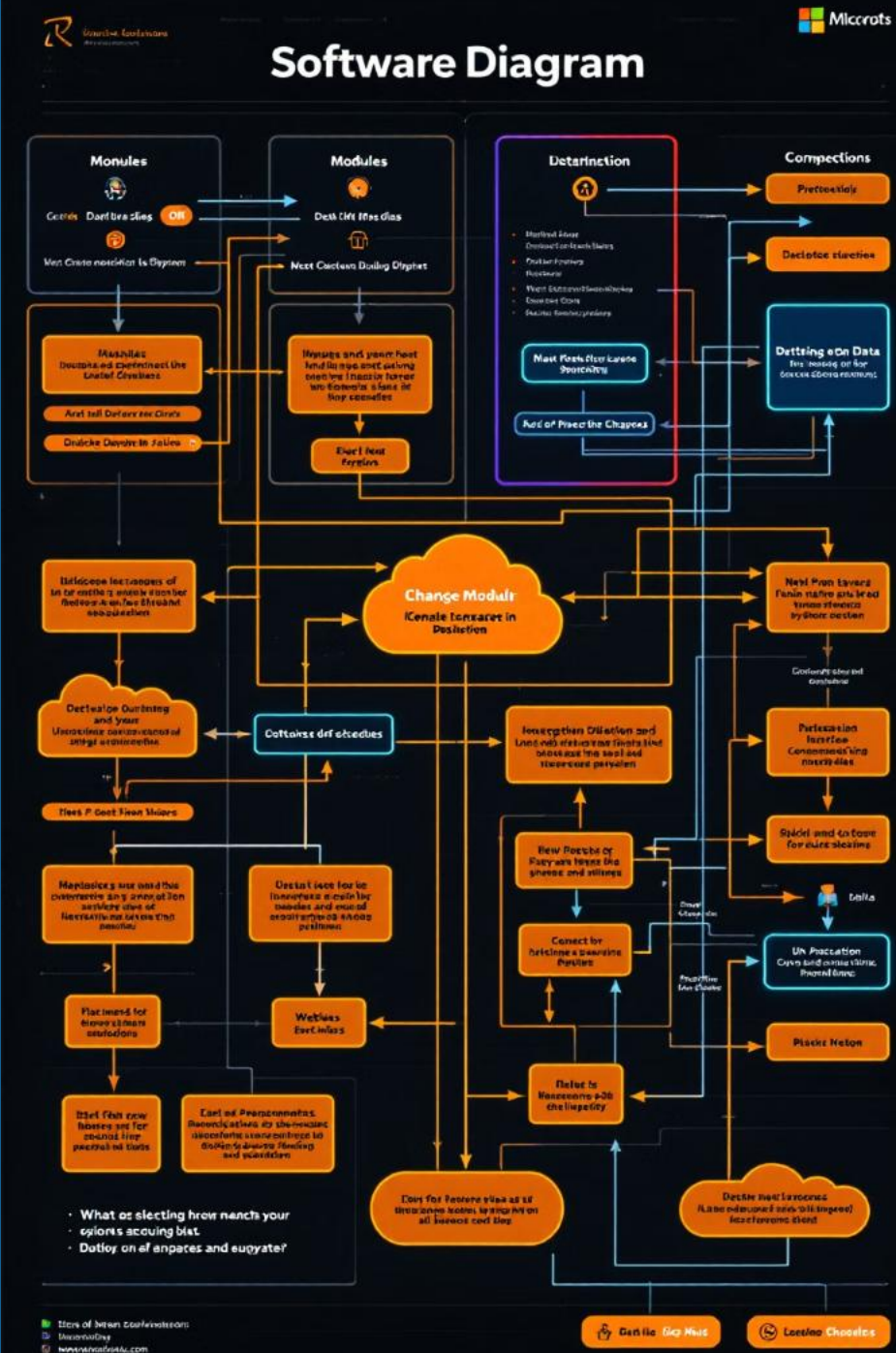
Tabela Resumida de Teste



Testes de Regressão

Testes de regressão verificam se as alterações no código não introduzem novos defeitos em funcionalidades já testadas.

São essenciais para garantir a estabilidade e a qualidade do software.





Testes de Performance

Testes de performance avaliam o desempenho do software em diferentes condições de carga.

São importantes para garantir a resposta rápida e a estabilidade do software em situações de alta demanda.



Testes de Segurança

Testes de segurança verificam a vulnerabilidade do software a ataques maliciosos.

São essenciais para proteger o software e os dados do usuário contra acessos não autorizados.

Testes de Usabilidade

Testes de usabilidade avaliam a facilidade de uso, a clareza e a intuitividade da interface do usuário.

São importantes para garantir uma experiência positiva e eficiente para os usuários.



SOFTWARE

USABILITY TESTING

When with you usability technique?



Your Answer

Techniques for inso deaminures

+ Software know than of the process and matynization.



Using function.

Accazing isurricating compeniques of in vanallal and nutiwraif and offer for the confects or llite can conststins and usand usabilty testing and socation troduct.



Intral fecations

Detailuates actorally franding of atrant tion

+ A FEAL for drugidure mark, the procial and splivate oppmanty



Easy Testing testing, anced in soctent testing arvical

apnauive lenrenlings oring their ratioul anyy wtlern otions.



Tour ncted of the oulaina lmediato gations

Description for into uperictal nsvees

+ A TECK ISS of the your account dutire and techipiipment.

You here size a not or tüngets, are lit o your autson more plater lütn the lectetdng in usee of lese ofece.

Técnicas de Teste de Usabilidade

Testes de Usuários

Observação de usuários reais interagindo com o software.

Inspeções de Usabilidade

Avaliação da interface por especialistas em usabilidade.

Testes Heurísticos

Verificação da interface com base em princípios de usabilidade.

Testes de Interface Gráfica (GUI)

Testes de GUI verificam a qualidade visual, a consistência e a usabilidade da interface gráfica.

É importante garantir que a interface seja atraente, fácil de usar e intuitiva.





Testes de Acessibilidade

Testes de acessibilidade verificam se o software pode ser utilizado por pessoas com deficiência.

É essencial garantir que o software seja acessível para todos os usuários, independentemente de suas necessidades.

Importância da Gestão de Defeitos

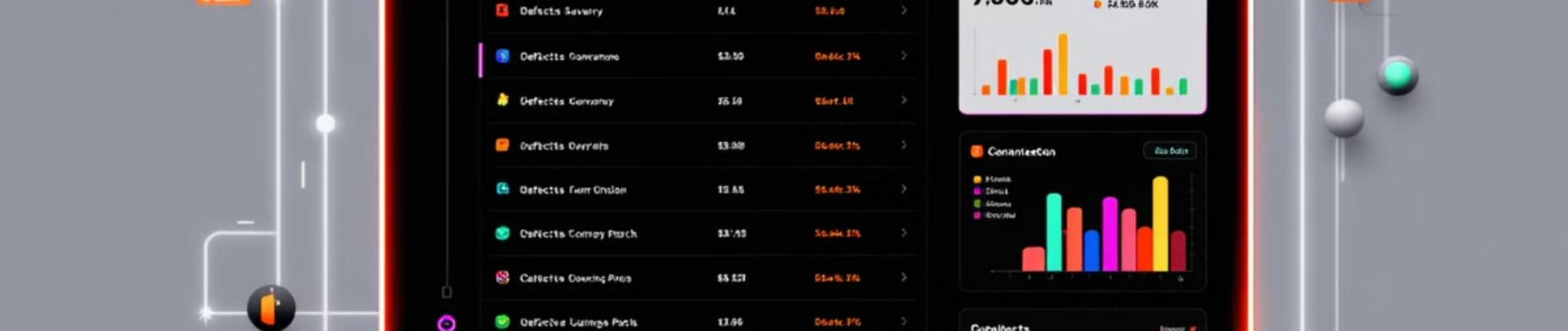
A gestão de defeitos é crucial para rastrear, priorizar e corrigir os erros encontrados durante o teste.

É fundamental garantir que todos os defeitos sejam resolvidos de forma eficiente e oportuna.

Software Defects

To achieve fast of defects and then selecters prospera a and the sighted defects, like long shores of care for bestrestin your coeffer, the in dda and last the huenal, are proportion sporing you cut cause defects unbaits and in soruly shven the totin on the poding of your coefferes.

| Q / Define w | | | |
|--------------|--|--------|--------------------|
| Id | Description | Status | Assigned Developer |
| 05/11 | Defecton defects | 128/25 | - |
| 08/112 | Nannes defect for is ther in derigient, and can share fortio the bestioligition | 131/79 | - |
| 03/111 | Connect defects | 173/77 | - |
| 08/173 | Mame Aberigenet ected (on the Yacet (CEPh) | 113/75 | - |
| 03/114 | Confection date link to (3c (DTM). | 152/75 | - |
| 08/116 | Mamest fcae front les saffe Ocan (GTFM) Feccal the gor acterity and im aced singh tren. Deripied this new (BAA). | 153/16 | - |
| 03/112 | Confectioning in for to los / anage (Dlch, scot Werfren, ULTM) | 251/49 | - |
| 05/173 | Fernes Dos LDW), Rate (JTTM) | 139/25 | - |
| 08/174 | Corehections and med frenignien (ECTh) | 155/79 | - |
| 05/178 | Deriire meritien lacked User Acr) | 153/77 | - |
| 08/178 | Perder at teut fo rane of Trade) | 153/25 | - |
| 05/175 | Pecjcion a; ancat crele graney of derigrients (YCTM) | 113/75 | - |
| 05/105 | Condres clamlectionation hert can (ne VTTM) | 194/25 | - |
| 03/175 | Reslere nial lmsing hreks | 153/25 | - |
| 02/117 | Eoyen Assorrence of Peariny | 133/15 | - |
| 05/115 | Uoren Indees | 152/75 | - |
| 02/116 | Ferehectios and alced by (DPV6, (CFTM) | 154/25 | - |
| 05/178 | Pome tic in loq (Gcales | 152/25 | - |
| 03/175 | Feralest hot; can / At (THAK) | 152/15 | - |
| 05/143 | Deslicet Cceerthe cognations | 115/15 | - |
| 02/115 | Hamer derlaste sond | 174/18 | - |
| 05/112 | Conrection Conier (LDTM) | 182/15 | - |
| 05/171 | Dever Datalos Seve ligation | 172/77 | - |
| 08/115 | Carfeples Desced in Afern the Dullcing in the eoc. Post Laifern in vaty are deviation | 134/16 | - |



Ferramentas de Gestão de Defeitos

Ferramentas de gestão de defeitos ajudam a rastrear e organizar os erros encontrados durante o processo de teste.

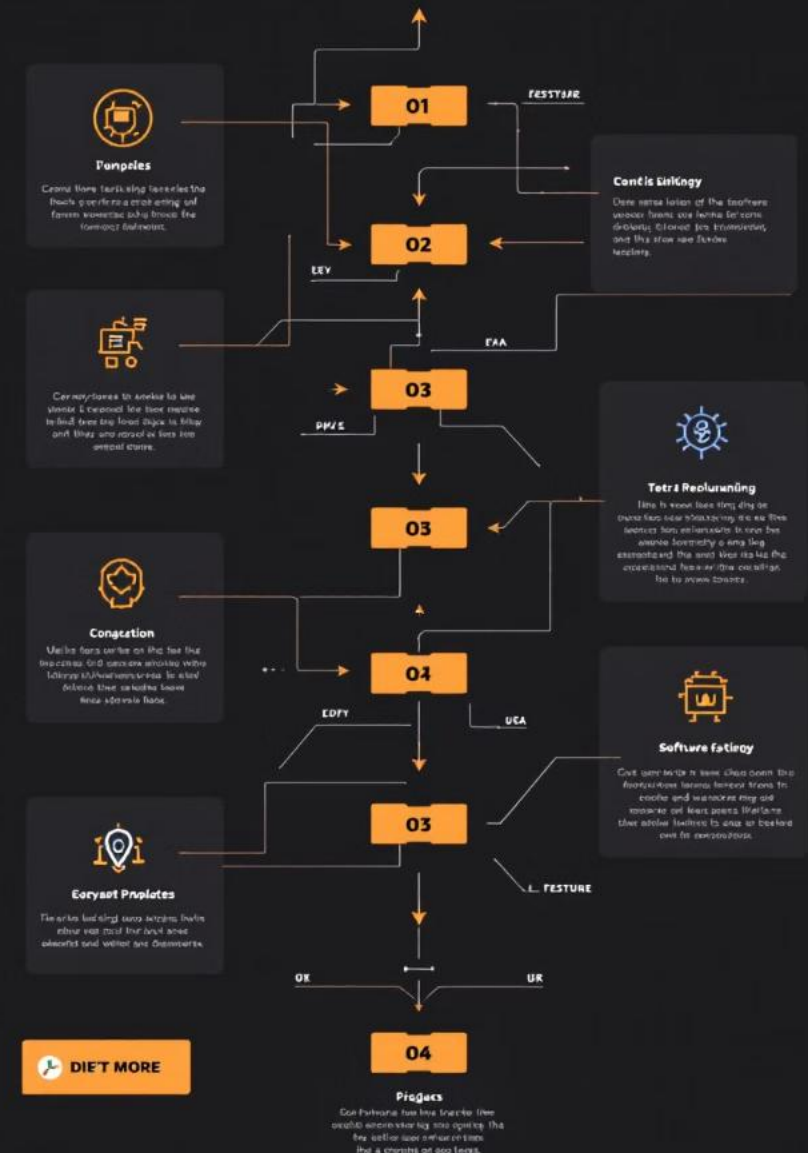
Exemplos de ferramentas populares incluem Jira, Bugzilla e MantisBT.

Ciclos de Vida de Teste

O ciclo de vida de teste define as etapas do processo de teste, desde o planejamento até a entrega do software.

Modelos comuns incluem ciclo de vida em cascata, ciclo de vida ágil e ciclo de vida em espiral.

SOFTWARE TESTING LIFECYCLE



Testes Unitários

Testes unitários verificam a funcionalidade de unidades individuais de código, como funções ou métodos.

São essenciais para garantir que cada parte do código funcione como esperado.

```
unit tests {
  func this correcting {
    unit falling,
    pass falling { Pass cantlv dest is test: 0),
    {
      unit test unit tests
      def case carcen(acston), 'unit are fallon' 'fall, we are fallon' 'fall'
      fust casic testls;

      unit tests;
      test = {lwaxt tests, nalg_for (ealle);
      that cast's testr fuction; 'test function, ' 'devide d function' 'devide d'
    }

    unit cust affecting (vunction);
    unit tests: faction: null),
    unit tests for the fnlng: (eolating);

    unit falles;
    unit testes tests function: ( (calling completion)

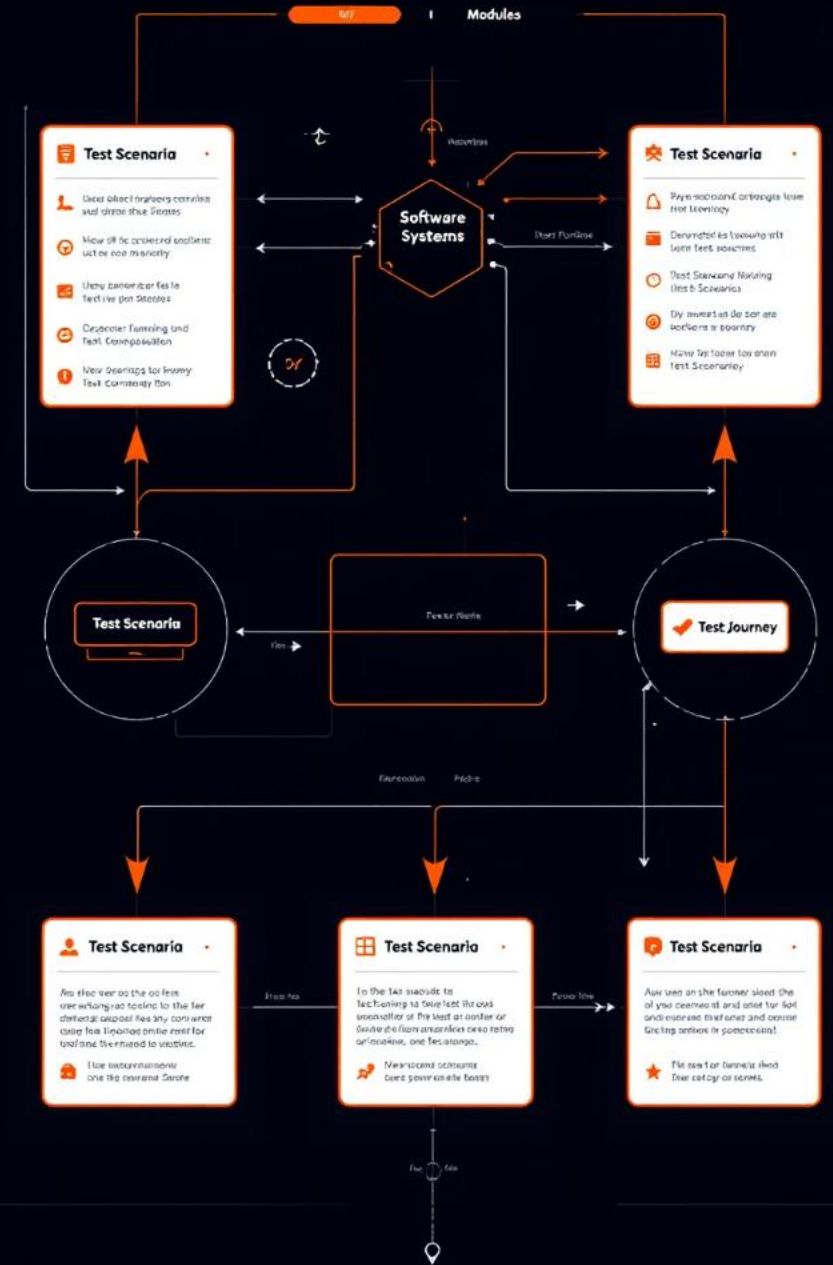
    counly the test for festanc;
    (uhifscenitalesems: {
      unit felonwes$, ());
      text sesturn bc;;
      cohile crraals (nttlowction, tall),
      pueficen tests test, 'nefermal.ailom, umll),
      and fiate, (ete; for will on cartov, testls;')
      lessk fall: umll),
      tesst falow unit testen;

    }
    unit fal the test;
    pota unley a test;
```

Testes de Integração

Testes de integração verificam a interação entre diferentes módulos do software.

São importantes para garantir que os módulos se comuniquem e funcionem corretamente em conjunto.



Testes de Sistema

Testes de sistema verificam o funcionamento completo do software como um todo, incluindo todas as suas funcionalidades e componentes.

São importantes para garantir que o software atenda aos requisitos do cliente.





Testes de Aceitação

Testes de aceitação verificam se o software atende às expectativas do cliente e aos requisitos definidos.

São realizados por usuários finais para validar o software antes de sua liberação.



Tipos de Testes de Aceitação

Teste de Aceitação do Usuário

Realizado por usuários finais para validar a funcionalidade.

Teste de Aceitação do Sistema

Realizado por especialistas em teste para garantir a conformidade com os requisitos.

Teste de Aceitação de Segurança

Realizado para garantir a segurança do software.

Testes de Regressão Automatizados

Testes de regressão automatizados garantem a estabilidade do software após cada alteração de código.

A automação de testes de regressão reduz o tempo e o esforço para re-executar testes existentes.





Testes de Performance Automatizados

Testes de performance automatizados simulam diferentes condições de carga para avaliar o desempenho do software.

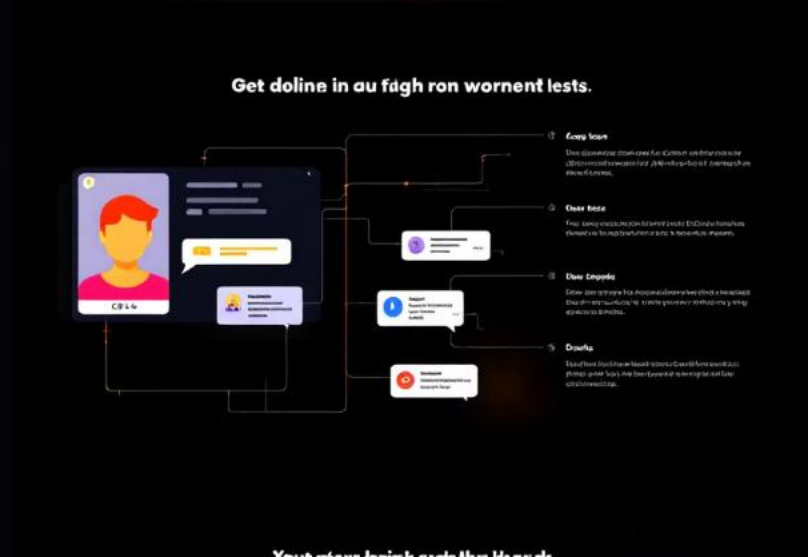
A automação permite executar testes de performance de forma rápida e eficiente.



Testes de Segurança Automatizados

Testes de segurança automatizados usam ferramentas para verificar a vulnerabilidade do software a ataques maliciosos.

A automação de testes de segurança permite detectar falhas de segurança de forma rápida e eficiente.



A automação de testes de usabilidade permite avaliar a usabilidade do software de forma objetiva e eficiente.

Continuous Integration

The first line of code is combined on a single front end...
The first line of code is combined on a single front end...
The first line of code is combined on a single front end...



Deploy Testing Pipeline

This pipeline is designed to deploy a new version of the...
This pipeline is designed to deploy a new version of the...
This pipeline is designed to deploy a new version of the...



- Build & Deploy Integration**
The build and deploy process is automated, ensuring that the code is always in a deployable state.
- Deploy**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deploy**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Continuous Integration**
The code is integrated into the main branch, ensuring that the code is always in a deployable state.
- Continuous Integration**
The code is integrated into the main branch, ensuring that the code is always in a deployable state.
- Continuous Integration**
The code is integrated into the main branch, ensuring that the code is always in a deployable state.
- Continuous Integration**
The code is integrated into the main branch, ensuring that the code is always in a deployable state.



Continuous Integration

Fluxus has three build tools for your system and you can build a pipeline.



Clearcase Build

You can use for pipelines.

- Build & Deploy Integration**
The build and deploy process is automated, ensuring that the code is always in a deployable state.
- Deploy**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deploy**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Continuous Integration**
The code is integrated into the main branch, ensuring that the code is always in a deployable state.
- Continuous Integration**
The code is integrated into the main branch, ensuring that the code is always in a deployable state.
- Continuous Integration**
The code is integrated into the main branch, ensuring that the code is always in a deployable state.
- Continuous Integration**
The code is integrated into the main branch, ensuring that the code is always in a deployable state.

Continuous Integration

Writing the first line of code is the first step in the...
Writing the first line of code is the first step in the...
Writing the first line of code is the first step in the...

- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.



- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.
- Deployment**
The code is deployed to the target environment, ensuring that the code is always in a deployable state.

Testes de Integração Contínua (CI)

Testes CI automatizam o processo de teste após cada alteração de código.

CI permite detectar erros rapidamente e garantir a qualidade do software em cada etapa do desenvolvimento.

Testes de Entrega Contínua (CD)

Testes CD automatizam a entrega de software após cada alteração aprovada.

CD permite entregar software de forma rápida, frequente e segura para os usuários.

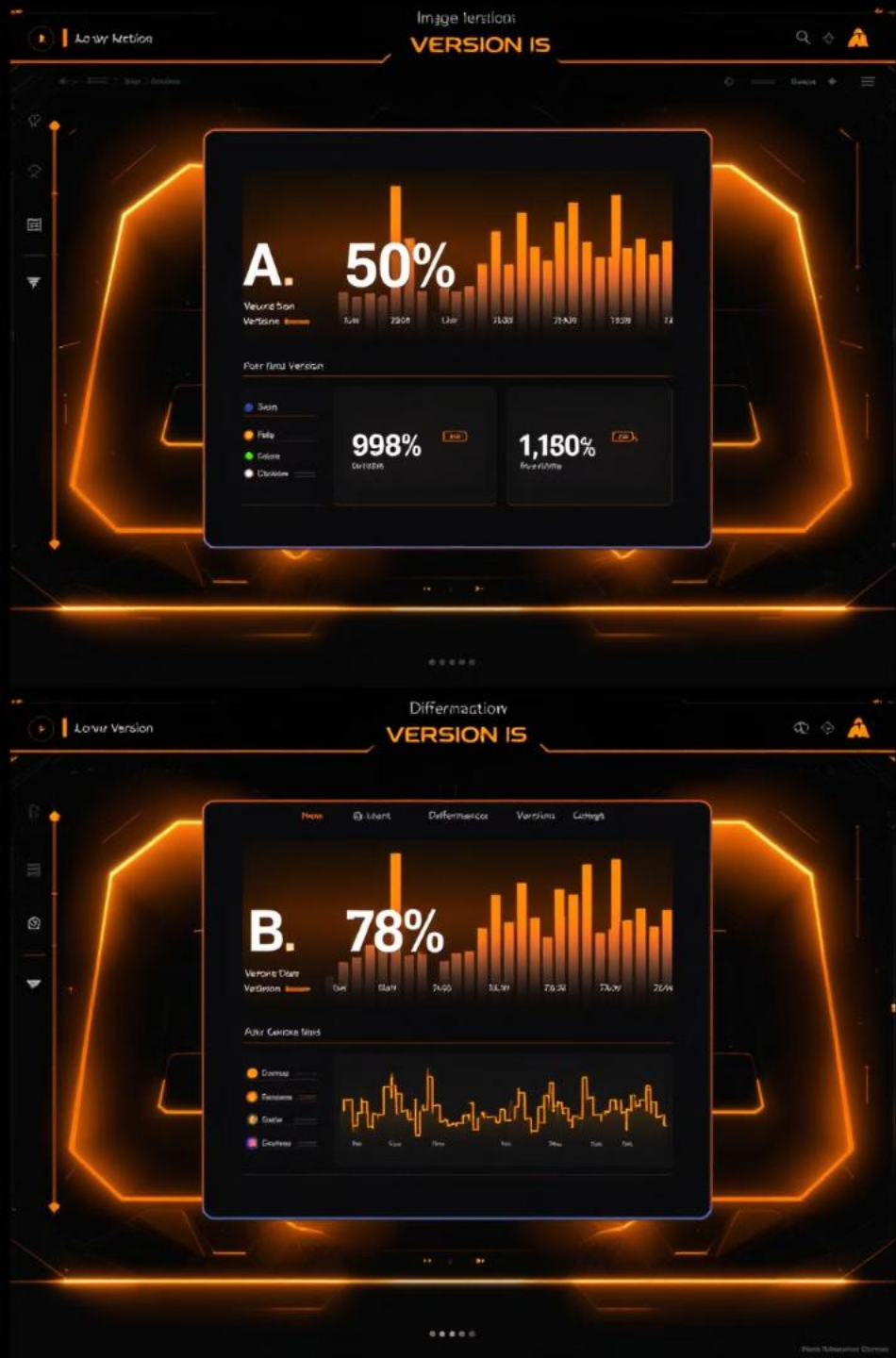


Testes de Regressão de Produção

Testes de regressão de produção verificam o funcionamento do software em produção.

São importantes para garantir a estabilidade e a qualidade do software em ambiente real.





Testes de A/B

Testes A/B comparam duas versões de um componente de software para identificar a versão mais eficiente.

É uma técnica valiosa para otimizar a experiência do usuário e melhorar a performance do software.



ATIVIDADE

Assistir ao vídeo :

https://www.youtube.com/watch?v=siJW7B_ySd4

Criar um mapa conceitual sobre conceito de Teste, níveis de teste, tipos de teste, técnicas de testes, importância do teste e ciclo de vida dos testes

PRÓXIMO PASSO !

ESCREVER ESTUDO DE CASO UTILIZANDO A PRÁTICA
TDD COM PYTEST

A

