Resumo do Curso: Início Rápido em Teste e QA

Seção 01:

Integrante	Anotações feitas	Seção correspondente
@Agemilson Abreu	**Carreira em Teste de Software (QA)** A carreira de Teste de Software (QA) é uma excelente opção profissional, sustentada por vagas abundantes, boa remuneração, oportunidades de crescimento e contínua expansão do mercado. • **Mercado de Trabalho:** O mercado está em alta, com muitas vagas globalmente em diversos setores (financeiro, saúde, etc.). É crucial procurar por termos variados como "teste", "QA" e "quali". • **Remuneração e Contratação:** Oferece salários competitivos que aumentam com a especialização, com contratações via CLT, PJ, freelancer ou plataformas de crowdtesting. • **Crescimento e Oportunidades:** Proporciona uma clara progressão de carreira (Júnior a Sênior e liderança), com muitas oportunidades internacionais e a possibilidade de empreender. • **Desafios e Exigências:** Os desafios incluem a pressão por prazos e a necessidade crítica de **aprendizado contínuo** para acompanhar a rápida evolução tecnológica. **Por que o Dev não testa?**	Seção 01

A função do QA é essencial, mesmo que todos na equipe devam testar. A necessidade de um especialista se deve a três pontos principais:

- **Viés de Confirmação:** Desenvolvedores possuem um "viés de confirmação", uma tendência psicológica que dificulta a identificação de falhas no próprio trabalho.
- **Qualidade é um Esforço Coletivo:** A qualidade é uma responsabilidade de toda a equipe, onde cada membro contribui com sua perspectiva única.
- **O Valor do Especialista de QA:** O especialista de QA agrega valor por ter uma mentalidade e técnicas focadas em **investigar ativamente**, prevenir defeitos e analisar riscos, indo além da validação funcional.

Habilidades Pessoais do QA

As habilidades pessoais essenciais são a **motivação** e **persistência** para superar desafios; a **curiosidade** e o **gosto por aprender** em uma área que muda constantemente; **perfeccionismo** e **atenção aos detalhes** para identificar falhas; **resiliência** para se adaptar a mudanças com foco na solução; **organização** para gerenciar informações e tarefas; e, crucialmente, **autogerenciamento**, **proatividade** e uma **"atitude de dono"**, demonstrando um compromisso profundo com o sucesso do projeto.

Habilidades Interpessoais para o QA

1. **Fundamentos da Comunicação:** Envolve saber ouvir e adaptar a fala, ter uma linguagem corporal coerente, e principalmente, **ler e escrever bem** para

criar relatórios de defeitos claros e profissionais. O uso de diagramas também é um diferencial.

2. **O Papel Social e a Imagem do Testador:** O QA deve construir relacionamentos para não ser visto apenas como o "portador de más notícias". É preciso comunicar problemas de forma construtiva e elegante, criticando o produto e não a pessoa, agindo como um parceiro.
3. **Habilidades de Relacionamento e Influência:** Para ser eficaz, é preciso saber **negociar** buscando soluções ganha-ganha e ter **empatia**, a capacidade de se colocar no lugar dos outros (colegas, usuários) para entender suas necessidades e dores.

Trabalho em Equipe

1. **O Palco do Trabalho em Equipe: As Reuniões:** As reuniões ágeis são o principal palco para a colaboração. O QA atua como o **"advogado da qualidade"**, garantindo que ela não seja negligenciada. Sua participação ativa é vital em todas as cerimônias (Planning, Dailies, Review, Retrospectiva).
2. **A Dinâmica da Colaboração no Dia a Dia:** A colaboração diária se manifesta em ajudar e ser ajudado, dar e receber feedback de forma construtiva, e agir com transparência e humildade.
3. **Os Limites da Colaboração e as Consequências:** Colaborar não significa ser ingênuo. É preciso saber dizer "não" a colegas que não contribuem. Em times ágeis, a falta de colaboração pode ter consequências sérias, como ser desligado pelo próprio time.

Hard Skills

Representam o conhecimento técnico mínimo esperado, dividido em quatro áreas:

1. **Habilidades Gerais:** Conhecimentos essenciais para qualquer profissional, incluindo domínio de sistemas operacionais (Windows, Linux, Mac) e seus consoles, suítes de escritório (Excel avançado), comunicação profissional por e-mail, fluência em inglês e noções sólidas de segurança da informação.
2. **Conhecimento do Negócio:** É vital entender o setor de atuação da empresa (bancário, saúde, etc.), suas regras e leis. A melhor forma de aprender é "vivenciar" a realidade do usuário final.
3. **Conhecimento de Tecnologia:** Uma base sólida em TI é necessária, incluindo noções de programação, redes, infraestrutura (VMs, containers) e banco de dados (SQL).
4. **Conhecimento Específico de Teste de Software:** O núcleo da carreira. Inclui dominar o processo de teste (planejar, criar, executar), realizar testes manuais e automatizados, gerenciar defeitos com evidências claras e aplicar técnicas e estratégias para otimizar o trabalho.

Dívida Técnica

1. **A Dívida Técnica é Universal e Inevitável:** É normal ter lacunas de conhecimento ("dívida técnica"). O problema não é ter a dívida, mas a **inércia** de não se planejar para aprender.
2. **A Solução: Aprendizado Contínuo e Planejado:** A solução é criar um plano de estudos e, principalmente, executá-lo com disciplina. O

	aprendizado contínuo é um hábito chave para o sucesso. 3. **O Valor de Compartilhar o Conhecimento:** O conhecimento técnico fica obsoleto rapidamente. Por isso, o maior valor está em **compartilhá-lo**. Isso constrói sua credibilidade e reconhecimento como referência em qualidade, algo muito mais duradouro do que o domínio de uma ferramenta específica.	
@Eric Lima da Silva	**Visão Geral da Carreira e Mercado de Trabalho** • **Mercado:** Área de Teste e QA está aquecida, com muitas vagas no Brasil e no exterior. • **Salários:** Remuneração competitiva (Júnior: R$ 2.000, Pleno: R$ 5.000, Sênior: R$ 6.000+). • **Contratação:** Modelos flexíveis como CLT, PJ, freelancer e empreendedorismo. • **Desafios:** Lidar com alta pressão e a necessidade de aprendizado contínuo. • **Diversidade:** Setor reconhecido por ser inclusivo e com forte presença feminina. **Por que o Desenvolvedor não Testa Sozinho?** • **Viés de Confirmação:** É psicologicamente difícil encontrar erros no próprio trabalho. A tendência é confirmar que funciona, não procurar falhas. • **Habilidades Especializadas:** O profissional de QA possui técnicas, práticas e um mindset focado em avaliar riscos e encontrar defeitos de forma especializada. **Habilidades Essenciais do Profissional de Teste/QA** • **Habilidades Pessoais (Soft Skills):** ◦ **Motivação e Persistência:** Para superar desafios. ◦ **Curiosidade e Atenção aos Detalhes:** Para descobrir como o sistema pode falhar. ◦ **Resiliência e Foco na Solução:** Adaptar-se a mudanças e resolver problemas.	Seção 01

    - **Organização e Autogerenciamento:** Para gerenciar informações e ser proativo.
- **Habilidades Interpessoais:**
    - **Comunicação:** Saber ouvir, falar com clareza e adaptar a linguagem.
    - **Leitura e Escrita:** Interpretar documentos e redigir relatórios claros é fundamental para a credibilidade.
    - **Ser um "Pessimista Profissional":** Comunicar defeitos de forma construtiva, criticando o produto, não as pessoas.
    - **Negociação e Empatia:** Colocar-se no lugar do outro para colaborar e entender suas necessidades.

A Importância do Trabalho em Equipe

- **Participação Ativa:** Atuar como o "advogado da qualidade" em todas as reuniões.
- **Colaboração Real:** Ajudar colegas, saber receber ajuda e dar/receber feedback.
- **Transparência e Humildade:** Compartilhar informações e reconhecer a necessidade de aprendizado contínuo.
- **Saber Dizer "Não":** Colaborar não significa ser explorado; é preciso lidar com quem não contribui.
- **Responsabilidade Coletiva:** Em times ágeis, o sucesso e os problemas são de toda a equipe.

Hard Skills

- **Habilidades Gerais:**
    - **Sistemas Operacionais:** Conhecimento de Windows, Linux e macOS, incluindo linha de comando.
    - **Aplicativos de Escritório:** Domínio de Excel (fórmulas, tabelas dinâmicas), Word e PowerPoint.

    - **Idiomas:** Inglês é fundamental, espanhol é um diferencial importante.
    - **Internet e Segurança:** Saber pesquisar, usar LinkedIn profissionalmente e ter boas práticas de segurança digital.
- **Conhecimento do Negócio:**
    - Estudar o setor de atuação da empresa (bancário, varejo, etc.).
    - Conhecer as leis e regulamentações aplicáveis.
    - Analisar concorrentes e vivenciar o uso real do produto.
- **Conhecimentos de Tecnologia:**
    - **Programação:** Noções de lógica são consideradas uma habilidade básica.
    - **Infraestrutura:** Entender conceitos de VMs, containers e métricas de desempenho.
    - **Banco de Dados:** Saber consultar bancos SQL e NoSQL para obter massa de teste.
- **Habilidades Específicas de Teste e QA:**
    - **Processo de Teste:** Dominar as fases de planejamento, modelagem, execução, etc.
    - **Execução:** Saber realizar testes manuais e, principalmente, criar automação de testes.
    - **Gestão de Defeitos:** Encontrar e registrar bugs de forma clara e com evidências.
    - **Testes de Confirmação e Regressão:** Validar correções e garantir que não surgiram novos problemas.
    - **Técnicas e Ferramentas:** Aplicar estratégias para otimizar testes e ter a capacidade de aprender novas ferramentas rapidamente.

Débito Técnico e Aprendizado Contínuo

- **Conceito:** "Débito Técnico" é o conjunto de conhecimentos que você ainda não adquiriu. O

	problema não é ter o débito, mas a inércia em não quitá-lo. • **Solução:** ◦ **Planeje:** Reserve um tempo semanal para estudar. ◦ **Aja:** Coloque o plano em prática. ◦ **Mantenha o Ritmo:** A consistência nos estudos é o mais importante. • **Credibilidade > Conhecimento:** O conhecimento técnico se desatualiza. Aplicá-lo e compartilhá-lo gera experiência e credibilidade, que são ativos mais duradouros. A mentalidade é: **“Devo, não nego, aprendo assim que puder.”** **Objetivos do Curso e Como Estudar** • **Objetivo:** Capacitar profissionais em todos os níveis com automação e processos modernos de QA. • **Como ter Sucesso:** Adotar uma mentalidade positiva, organizar um ambiente sem distrações, ser persistente e focar no objetivo final. **Um Mapa para o Futuro** **O Mundo se Transforma**: A tecnologia, os negócios, as profissões e as próprias pessoas estão em constante evolução. É inútil se apegar ao passado; o segredo é abraçar a mudança como parte da vida. **Seja o Café, não a Cenoura ou o Ovo**: Diante das dificuldades (a "água fervente"), você pode reagir de três formas: como a cenoura, que amolece e desiste; como o ovo, que endurece e se torna cético; ou como o café, que transforma o ambiente ao seu redor. A meta é ser um agente de mudança positiva. ☕	
@Lívia Barbosa	• Todo mundo deve ajudar a testar o sistema a fim de promover uma maior qualidade, entretanto, um QA possui habilidades diferentes de investigar e detectar defeitos - tomando ações para prevenir que os erros	Seção 01

aconteçam, visto que já identificam os padrões. Assim sendo, os outros profissionais utilizam o sistema e notam defeitos por meio da análise se o sistema está funcionando corretamente, enquanto o profissional QA busca ativamente os defeitos, analisando o que pode dar errado.

- É necessário conhecer o negócio que você testa, então é importante que o testador estude e conheça o funcionamento do ambiente do sistema a ser testado. Dessa forma, é relevante conhecer normas, modelos de trabalho, avaliações e certificações que fazem parte do negócio, a fim de entender o que a empresa deve seguir e o que pode implicar na sua penalização. Além disso, ter conhecimento a respeito de sistemas operacionais, ferramentas Office, idiomas (principalmente inglês, espanhol e mandarim - diferencial), lógica e linguagem de programação, infraestrutura.
- Vivenciar o ambiente de uso real do software por, pelo menos, 1 dia, pode proporcionar uma visão ampla dos problemas enfrentados e de como o sistema pode, de fato, auxiliar de maneira adequada na realização de tarefas.
- Uma pessoa da área da tecnologia deve estar constantemente estudando, buscando novas ferramentas e tecnologias emergentes.
- Um QA deve ter soft skills ligadas, principalmente, à comunicação e trabalho em equipe, uma vez que suas atividades se baseiam no planejamento, análise, modelagem e execução, relatando erros e inconsistências por meio da escrita e comunicação com a equipe. É importante ressaltar a relevância de características como motivação, resistência, capacidade de aprender rapidamente, curiosidade, detalhismo e perfeccionismo, as quais permitem que o profissional confie no seu próprio potencial e

	consiga aplicar suas habilidades de maneira minuciosa e pontual. • **Débito Técnico:** falta de habilidades e conhecimento de ferramentas para exercer uma profissão.	
@Mário Queiroz	• **Soft Skills** ◦ Motivação ◦ Persistência ◦ Autoconfiança ◦ Curiosidade ◦ Gosto por aprender ◦ Habilidade interpessoal / Comunicação ◦ Perfeccionismo ◦ Detalhista ◦ Resiliência ◦ Foco em solução ◦ Organização ◦ Gestão de tempo ◦ Autogerenciamento 1. **Automação de testes** • Conhecimento em ferramentas de automação como Selenium, Cypress, Playwright, etc. • Saber programar scripts de teste automatizado. 2. **Gestão de evidências** • Criar, capturar e organizar evidências (prints, logs, vídeos). • Saber gerar relatórios ou documentações de testes. 3. **Documentação de testes** • Escrever scripts de teste manuais ou automatizados. • Usar ferramentas de gerenciamento como TestRail, Zephyr, etc. 4. **Organização de arquivos**	Seção 01

- Trabalhar com pastas, versionamento de arquivos e histórico de mudanças.

5. **Comunicação com times técnicos**

- Saber ler e escrever tickets, bugs, e-mails técnicos.
- Participar de reuniões de planejamento (como plannings e reviews).

O Que Um Testador de Software Precisa Saber e Fazer?

a. **Planejamento de Testes**

Antes de qualquer clique, o testador precisa entender:

- Quais são os objetivos do sistema?
- Quais os riscos que precisam ser mitigados?
- Como organizar os testes para garantir que o projeto dê certo?
- **Análise e Modelagem dos Testes**

Aqui o foco é entender:

- Quais testes são mais importantes e viáveis com o tempo e recursos disponíveis?
- Quais variações e condições precisam ser verificadas?

Exemplo: Num quiosque de sorvete: vou testar casquinha? Sundae? Picolé? Quais sabores? Quais combinações são obrigatórias?

c. **Implementação dos Testes**

É a fase de **escrever os testes e preparar o ambiente**.

- Pode ser tudo junto com outras fases, especialmente com agilidade.
- Importante estar tudo pronto para execução.
- **Execução dos Testes**

O famoso “clicar no sistema”:

- **Manual:** você mesmo navega e interage com o software.
- **Automatizado:** você cria um robô (script) que testa por você.
- **Identificação de Defeitos**

Sim, você vai encontrar bugs. Muitos! Mais do que imagina. O segredo é:

- Ter um olhar crítico (mas não se tornar “o chato dos defeitos”).
- **Criticar o produto, não a pessoa.**
- **Comunicação dos Defeitos**

Você precisa:

- Explicar o que aconteceu.
- Mostrar como reproduzir o problema (passo a passo).
- Preferencialmente usar **prints ou vídeos**, para deixar bem claro.

Um bug intermitente é como ver um OVNI: se ninguém mais viu, precisa de prova!

- **Confirmação e Regressão**
- **Reteste/Confirmação:** testar novamente o que foi corrigido.
- **Regressão:** testar tudo para garantir que nada mais quebrou com as mudanças.
- **Técnicas de Teste**

Ferramentas mentais para:

- Identificar os testes mais importantes.
- Cobrir cenários variados com inteligência.

Ex: análise de requisitos, diagramas, critérios de partição, tabelas de decisão, etc.

i. **Práticas e Metodologias**

Saber como o teste se encaixa em:

- **Metodologias ágeis:** Scrum, XP, Kanban, etc.

	◦ Práticas de equipe, colaboração, entrega contínua. **10. Estratégia de Testes** A arte de **priorizar, organizar e planejar os testes com inteligência.** ◦ Pode ser baseada em risco, requisitos, histórico de defeitos, entre outras. É como planejar um dia cheio de tarefas na cidade: com organização, você resolve tudo e ainda sobra tempo pra curtir. **11. Ferramentas de Teste** Não dá pra saber todas, mas: ◦ É essencial ter facilidade de aprender novas. ◦ Aprender conforme a demanda. • ***Débito técnico*** ◦ Sempre haverá algo que você ainda não sabe. ◦ Isso não é um problema — é natural. ◦ O problema é **não planejar como resolver esse débito**. • ***Aprendizado Contínuo*** ◦ A cada projeto ou empresa, novas ferramentas surgem. ◦ Conhecer uma ferramenta parecida acelera a curva de aprendizado.	
@Wesley Lima	• Áreas de atuação ◦ TI, Telecomunicação, Financeiro, Governo, Transporte, Saúde, Comércio, Indústria etc. • Hard e Soft Skills ◦ Soft Skills ▪ Pessoais: Motivação, Persistência, Curiosidade, Gosta de Aprender, Perfeccionista, Detalhista, Resiliência, Foco em Solução, Organização, Priorização, Autogerenciamento.	Seção 01

	▪ Interpessoais: Comunicação, Negociação, Empatia, Trabalho em Equipe. ◦ Hard Skills ▪ Gerais: Conhecimento de Sistemas Operacionais, Office, Idiomas, Internet e Segurança. ▪ Sobre o que se testa: Informações importantes como legislação, padrões, usos, costumes, regras e concorrentes. ▪ Sobre tecnologia: Lógica de programação, linguagens de programação, telecomunicação, infraestrutura. ▪ Sobre Teste e QA: Planejamento, Análise, Modelagem, Implementação, Execução (Manual, Automatizada), Técnicas de Teste, Práticas, Processos, Estratégias e Ferramentas.	

Seção 02:

Integrante	Anotações feitas	Seção correspondente
@Agemilson Abreu	**Norma ISO/IEC 25010 (SQuaRE)** A norma define a **qualidade de software** como um conceito multidimensional, não se limitando a um único fator. Ela serve como um padrão para facilitar a avaliação e o desenvolvimento de software, oferecendo um guia sobre as diversas características que compõem um produto de qualidade. **As 8 Características da Qualidade de Software** **1. Adequação Funcional** Avalia se o software cumpre seu propósito principal. É considerado um **teste de negócio (funcional)**.	Seção 02

- **Completude:** Executa todas as tarefas solicitadas?
- **Correção:** Produz os resultados corretos e precisos?
- **Apropriação:** Apresenta os resultados de forma adequada ao usuário e ao contexto?

2. Usabilidade

Mede a facilidade com que um software pode ser utilizado. É um **teste técnico (não funcional)**.

- **Reconhecibilidade:** O usuário reconhece os elementos e fluxos?
- **Aprendizabilidade:** O software ajuda o usuário a aprender a usá-lo?
- **Operabilidade:** As tarefas são realizadas de forma rápida e eficiente (poucos cliques)?
- **Proteção Contra Erros:** O sistema impede que o usuário cometa erros?
- **Estética:** A interface é visualmente agradável e ergonômica?
- **Acessibilidade:** Pode ser utilizado pelo maior número de pessoas possível?

3. Compatibilidade

Avalia se o software trabalha bem com outros sistemas.

- **Coexistência:** Consegue compartilhar recursos pacificamente, sem interferir ou sofrer interferência?
- **Interoperabilidade:** Consegue se comunicar e trocar dados com outros softwares?

4. Confiabilidade

Mede a capacidade do software de estar sempre disponível e funcionando quando o usuário precisa.

- **Maturidade:** Consegue prever e prevenir falhas antes que aconteçam?
- **Disponibilidade:** Permanece no ar e acessível durante o tempo esperado (uptime)?
- **Tolerância a Falhas:** Consegue lidar com uma falha em tempo real sem "quebrar" completamente?
- **Recuperabilidade:** Consegue se recuperar após uma falha total, restaurando dados e transações?

5. Eficiência de Desempenho

Avalia se o software é rápido, ágil e utiliza bem os recursos.

- **Comportamento em Relação ao Tempo:** O tempo de resposta é rápido o suficiente?
- **Otimização de Recursos:** Utiliza os recursos de hardware (CPU, memória) de forma inteligente?
- **Capacidade:** Suporta o volume esperado de usuários, especialmente em picos de demanda?

6. Manutenibilidade

Mede a facilidade de corrigir, melhorar ou adaptar o software.

- **Modularidade:** É dividido em partes independentes que podem ser alteradas sem afetar o todo?
- **Reusabilidade:** Seus componentes podem ser reaproveitados em outros contextos?
- **Analisabilidade:** O código-fonte é fácil de ler e entender?
- **Modificabilidade:** Pode ser alterado trocando componentes, sem precisar reescrever o código?
- **Testabilidade:** É fácil e viável de ser testado de forma objetiva e mensurável?

	7. Portabilidade Mede a capacidade do software de funcionar em múltiplos ambientes (sistemas operacionais, navegadores, dispositivos). • **Adaptabilidade:** Ajusta-se bem a diferentes ambientes, mantendo a consistência? • **Instalabilidade:** É fácil de instalar, configurar e desinstalar? • **Substituibilidade:** Consegue substituir uma versão anterior ou um concorrente de forma eficaz? **8. Segurança** Avalia a capacidade do software de proteger dados e sistemas contra acessos e manipulações indevidas. • **Confidencialidade:** A informação só é acessada por pessoas autorizadas? • **Integridade:** Os dados só são modificados por pessoas autorizadas, com registro das alterações? • **Não Repúdio:** É possível provar que uma pessoa realizou uma transação, sem que ela possa negar? • **Responsabilidade (Accountability):** Todas as ações dos usuários podem ser rastreadas? • **Autenticidade:** É possível garantir a identidade de um usuário ou a validade de uma transação?	
@Eric Lima da Silva	**Uma Breve História do Teste** • **Pioneiros (Séc. XIX - Década de 40):** ◦ **Babbage e Lovelace:** Babbage projetou a primeira máquina programável; Lovelace (primeira programadora) registrou o primeiro "defeito" conceitual. ◦ **Herman Holerith:** Fundador da IBM, deu origem ao conceito de "teste de caixa-branca". ◦ **Grace Hopper (1947):** Popularizou o termo "bug" ao encontrar uma mariposa em um computador.	Seção 02

- **Sistematização e Era Moderna:**
  - **Glenford Myers (1979):** Autor de "A Arte de Testar Software", definiu testes de caixa-branca/preta e a "Regra de 10" sobre o custo dos defeitos.
  - **Pioneiros no Brasil (Anos 2000):** Emerson Rios, Ricardo Cristália e Leonardo Molinari.
  - **Padronização Global:** Rex Black e o ISTQB unificaram termos e técnicas.
  - **Teste Ágil:** Lisa Crispin e Janet Gregory defenderam o papel do especialista em qualidade em equipes ágeis.

Importância do Teste e Danos dos Bugs

- **Empresas:** Bugs causam prejuízo financeiro, perda de confiança e danos à imagem.
- **Governos:** Falhas podem comprometer a segurança nacional e levar a decisões estratégicas erradas.
- **Pessoas:** Causam desde constrangimentos (cartão recusado) até risco de vida (acidentes aéreos).
- **Meio Ambiente:** Falhas em sistemas de alerta e desperdício de recursos podem causar danos ambientais.

Os 7 Fundamentos do Teste (ISTQB)

1. **Teste Demonstra a Presença de Defeitos:** O teste encontra defeitos, mas nunca pode provar que não existem mais. O objetivo é reduzir o risco a um nível aceitável.
2. **Teste Exaustivo Não é Possível:** É inviável testar todas as combinações. A solução é priorizar testes com base no risco.
3. **Teste Antecipado:** Começar a testar o mais cedo possível no ciclo de vida do projeto, pois corrigir defeitos tardiamente custa exponencialmente mais caro ("Regra de 10 de Myers").

4. **Agrupamento de Defeitos:** Os bugs tendem a se concentrar em poucos módulos complexos. O testador deve focar os esforços nessas áreas de risco.
5. **Paradoxo do Pesticida:** Repetir os mesmos casos de teste se torna ineficaz para encontrar novos bugs. É preciso revisar e criar novos testes constantemente.
6. **Teste Depende de Contexto:** Não existe uma abordagem única. A estratégia de teste deve ser adaptada aos riscos específicos de cada sistema (ex: um app de banco exige mais testes que um site simples).
7. **A Ilusão da Ausência de Erros:** Um software sem bugs, mas que não atende às necessidades do usuário, é um fracasso. A utilidade e o valor para o cliente são mais importantes que a perfeição técnica.

Diferença entre Teste e QA

- **Teste (Testing):** Focado no **PRODUTO**. Atividade de **detecção** para encontrar defeitos.
- **QA (Garantia da Qualidade):** Focado no **PROCESSO**. Atividade de **prevenção** para melhorar a forma como o software é feito e evitar que defeitos ocorram.

Erro, Defeito, Ocorrência e Falha

- **Erro:** Ação humana que causa um problema.
- **Defeito (Bug):** O erro presente no código ou documento.
- **Ocorrência:** Um comportamento suspeito reportado pelo testador para análise.
- **Falha:** A manifestação visível do defeito quando o código é executado.

As 8 Características de Qualidade (IEC/ISO 25010)

1. **Adequação Funcional:** O software faz o que deveria fazer corretamente.

2. **Usabilidade:** É fácil de reconhecer, aprender, usar e acessível.
3. **Compatibilidade:** Funciona bem junto com outros softwares (coexistência) e se comunica com outros sistemas (interoperabilidade).
4. **Confiança:** É maduro (previne falhas), disponível (sempre no ar), tolerante a falhas e recuperável.
5. **Eficiência no Desempenho:** É rápido (tempo de resposta), otimiza recursos (CPU/memória) e suporta a carga de usuários (capacidade).
6. **Manutenibilidade:** É fácil de analisar, modificar, reutilizar e testar.
7. **Portabilidade:** Adapta-se a diferentes ambientes (celular, PC), é fácil de instalar e pode substituir outras versões.
8. **Segurança:** Garante confidencialidade, integridade, autenticidade e rastreabilidade das ações.

Testes Manuais vs. Testes Automatizados

- **Profissional Ideal:** É multidisciplinar, sabendo quando usar cada abordagem.
- **Papel da Automação:** Essencial para testes de regressão em larga escala e integração contínua (CI), garantindo que novas alterações não quebrem o que já funcionava.
- **Complementaridade:** A automação libera o testador de tarefas repetitivas para que ele possa focar em testes exploratórios e manuais em novas funcionalidades.

Testes Tradicionais vs. Testes Ágeis

- **Teste Tradicional:** Ocorre como uma fase isolada no final do desenvolvimento.
- **Teste Ágil:** É uma atividade contínua, integrada ao desenvolvimento desde o início.

	• **Testador Ágil:** Atua como "embaixador da qualidade", participando ativamente e promovendo a qualidade como responsabilidade de toda a equipe.	
@Lívia Barbosa	A aparição de bugs no sistema pode impactar negativamente diversas áreas, causando danos como: • atrasos nas entregas de software de qualidade funcionando, o que gera uma perda de confiança das empresas/organizações que necessitam de um sistema adequado. • prejuízo financeiro, visto que são gerados muitos custos em correção de bugs e novos testes • prejuízo de imagem, dado que o responsável pela tarefa acaba sendo associado a entregas de baixa qualidade. • invasão de pessoas não autorizadas. • vazamento de informações sigilosas. • em caso de Governos, decisões estratégicas incorretas. • risco de vida e acidentes, uma vez que sistemas para auxílio de atividades de alto risco ou de alta confidencialidade devem ser, obrigatoriamente, confiáveis. Para exemplificar, pode-se citar um sistema para controle de aviões: se o software mostrar a rota de um determinado avião de modo incorreto, o risco de um acidente grave ocorrer é iminente, já que o avião em questão pode acabar entrando na rota de outro e causando uma colisão. • desperdício de recursos e poluição, tendo em vista softwares que não desligam, softwares que calculam errado a quantia de recursos necessários e acabam gastando mais do que o devido ao produzir um determinado produto, etc. **ISTQB -** entidade com sede na Bélgica que criou sete fundamentos do teste, caracterizados por conceitos e	Seção 02

ideias que estabelecem o que o teste consegue ou não fazer:

- **Teste demonstra a presença de defeitos, mas nunca a sua ausência:** o teste pode demonstrar a presença de defeitos, mas não garante que eles não existem, o que significa que podem ser realizados diversos testes com conhecimentos de alto nível aplicados e, mesmo assim, pode ser que alguns defeitos ainda passem despercebidos. Assim sendo, o profissional QA deve ter sempre uma desconfiança de que existem outros bugs, uma vez que sua meta é promover um sistema perfeito. Em casos de sistemas de alto risco, os testes devem ser exaustivos, pois o software precisa funcionar perfeitamente para evitar erros, Já em casos de softwares comerciais, o limite de testes deve ser alinhado com o limite de custos da empresa.
- **Teste exaustivo não é possível:** testar tudo não é viável, exceto para casos triviais. Tendo isso em vista, é necessário levar em conta riscos e prioridades - com base, principalmente, no que se usa mais e o que é mais importante para a empresa - para dar foco aos esforços de teste.
- **Teste antecipado:** quanto mais breve a atividade de teste começar, mais retorno vai ser recebido. Se um bug não for reportado e corrigido logo no início do processo, este vai acarretar em outros, o que vai resultar em gastos que poderiam ser evitados anteriormente.
- **Agrupamento de defeitos:** os bugs não são distribuídos de maneira homogênea pelo sistema, o que significa que alguns módulos possuem mais defeitos que outros. Assim sendo, é interessante procurar defeitos em lugares que já apresentaram bugs, lugares com maior índice de reclamações de

clientes e em locais de alto risco - os quais podem gerar grandes consequências e prejuízos.

- **Paradoxo do pesticida:** realizar os mesmos testes sempre faz com que eles percam a eficácia, o adequado é atualizar os casos de teste conforme o software evolui.
- **Teste depende do contexto:** os testes são realizados de acordo com o seu contexto, ou seja, quanto maior o risco, mais testes.
- **A ilusão da ausência de erros:** mesmo que os bugs sejam encontrados e consertados, é importante que o sistema atenda, efetivamente, às expectativas e necessidades do usuário.

Erro: engano cometido por uma pessoa que só pode ser reconhecido por quem o cometeu.
Defeito: problema encontrado no trabalho de outra pessoa.
Falha: quando um defeito é executado e causa um problema visível no sistema.

IEC/ISO 25010:

Verificadas em Testes Funcionais:

- **Adequação funcional (anteriormente chamada de Funcionalidade):** característica referente a uma funcionalidade adequada ao que foi pedido, ou seja, aquela que cumpre o seu propósito. Essas características representam uma parte de negócio, não um aspecto técnico.
  - Analisa a **completude**, **correção** e **apropriação** - se as informações são exibidas de maneira adequada - as funcionalidades de um software.

Verificadas em Testes Não-Funcionais:

- **Usabilidade:** característica referente à facilidade com que o usuário tem de utilizar o sistema, sem que este necessite de manual.

    - **Reconhecibilidade:** facilidade do usuário ao reconhecer elementos da tela e comportamentos.
    - **Aprendizibilidade:** capacidade do software de ensinar o usuário - como ícones de interrogação que informam para que um determinado campo serve.
    - **Operabilidade:** facilidade de operação e navegação no sistema - como menos cliques para realizar uma operação.
    - **Proteção contra erro do usuário:** sistemas com meios de não permitir o usuário cometa determinados erros - como uma lista para selecionar estados, evitando que a pessoa insira um nome incorreto.
    - **Estética (da interface do usuário)**
    - **Acessibilidade:** facilidade de acesso de todas as pessoas ao software.
- **Compatibilidade:** característica referente à capacidade de um software ser compatível com outros softwares.
    - **Coexistência:** facilidade de coexistir com outros sistemas.
    - **Interoperabilidade:** facilidade de comunicação entre os sistemas.
- **Confiança ou Confiabilidade:** característica referente a um software que está sempre disponível para o usuário.
    - **Maturidade:** capacidade de perceber e prevenir a falha antes que ela aconteça - aviso de que uma coisa vai dar errado.
    - **Disponibilidade:** capacidade do sistema se manter à disposição de usuários e sistemas.
    - **Tolerância a falhas:** capacidade de perceber e compensar as falhas em tempo real.

    - **Recuperabilidade:** capacidade de se recuperar de falhas e travamentos.
- **Eficiência (de desempenho):** característica referente à capacidade de um software funcionar rapidamente.
    - **Comportamento em relação ao tempo:** performance em si do software.
    - **Utilização de recursos:** observar como o usuário utiliza os recursos disponíveis - é melhor um software que usa muitos recursos do que um software que possui recursos mas não os utiliza.
    - **Capacidade:** capacidade do software de atender a transações e usuários.
- **Manutenibilidade:** característica referente à facilidade de submeter um software à manutenção.
    - **Modularidade:** software dividido em módulos/partes.
    - **Reusabilidade:** facilidade do software em ser reutilizado em outros lugares.
    - **Analisabilidade:** facilidade de analisar o programa, se o código é fácil de compreender.
    - **Modificabilidade:** facilidade em modificar o software, se é possível mudar os componentes de maneira simples.
    - **Testabilidade:** facilidade de testar um software - questionar a testabilidade na hora que receber a documentação para evitar futuros problemas.
- **Portabilidade:** característica referente à capacidade de um software de funcionar em vários sistemas operacionais, navegadores e dispositivos.
    - **Adaptabilidade:** facilidade do software funcionar em um ambiente novo, de se adaptar.
    - **Instalabilidade:** facilidade de instalar ou desinstalar uma aplicação.
    - **Substituibilidade:** facilidade de substituir um software por um novo.

	• **Segurança:** característica referente à capacidade de um software não ser invadido e/ou manipulado.
	  ◦ **Confidencialidade:** capacidade de que apenas quem criou ou quem tem uma hierarquia maior tenha acesso a uma determinada informação.
	  ◦ **Integridade:** capacidade de que apenas pessoas autorizadas podem modificar determinadas informações e, se houver alguma mudança, esta deve ser registrada.
	  ◦ **Não repúdio:** garantia que a pessoa que está fazendo uma transação é realmente um usuário específico do sistema.
	  ◦ **Responsabilidade:** garantir que uma pessoa que utilizou o software fez uma determinada ação de uma forma - por meio de um log, por exemplo.
	  ◦ **Autenticidade:** garantia das ações realizadas no sistema.	

@Mário Queiroz	**Por que testar?** Bugs causam prejuízos reais: • **Financeiros**: Custos com correções, indenizações, perda de negócios. • **Imagem**: Danos à reputação (recuperação lenta ou irreversível). • **Operacionais**: Atrasos, insatisfação do cliente, perda de confiança. • **Sociais/Governamentais**: Vazamento de dados sigilosos, decisões públicas erradas, riscos à segurança nacional. --- Conceitos • Teste e QA **Teste** é focado no **produto**: o testador verifica se o software funciona conforme o esperado, de acordo	Seção 02

com requisitos, documentação e necessidades do cliente. Pode ser feito manualmente ou com automação, mas sempre tem como objetivo identificar problemas no produto para que sejam corrigidos.

- **Garantia da Qualidade (QA)** é focada no **processo**: o profissional de QA trabalha para melhorar o processo de desenvolvimento e testes, visando evitar que erros aconteçam novamente. Ele analisa lições aprendidas, causas raízes dos problemas e busca aprimorar práticas, reduzir custos, aumentar eficiência e qualidade futura do produto.
- São **duas funções diferentes**, mas que se complementam. O teste é uma ferramenta para medir a qualidade do produto, enquanto o QA atua preventivamente para melhorar o processo e a qualidade geral.
- No mercado, ainda há muita confusão entre os papéis. Muitas vezes, testadores são chamados de QA, mas o trabalho de QA vai além de encontrar defeitos — envolve análises, treinamentos, recomendações e melhoria contínua.
- O objetivo do QA é criar um ambiente que previna erros e melhore o produto a longo prazo, não só detectar problemas pontuais.
- Erro, defeito e falha

  Erro: É a ação humana que gera um problema, como um engano no código ou documentação. Só quem comete o erro pode reconhecê-lo — é um engano próprio.
- **Defeito:** Quando alguém encontra um problema no trabalho de outra pessoa, chama-se defeito (ou bug). É o erro de outro identificado por alguém. Mas chamar direto de defeito pode gerar atrito.

- **Incidente/Ocorrência:** Um termo usado para suavizar a comunicação. É uma dúvida ou suspeita levantada sobre algo que pode estar errado, sem acusar ninguém. A ideia é incentivar a análise conjunta para confirmar se é defeito ou não.
- **Falha:** Quando um defeito é executado e causa um problema visível no sistema em funcionamento. Ou seja, o bug "explode" quando o software está rodando.

---

7 Fundamentos do ISTQB

1. **Testes mostram a presença de defeitos, nunca sua ausência**:
   - Software é complexo; testes não garantem 100% de ausência de bugs.
   - *Por quê?* Corrigir um bug pode gerar novos, e testar tudo é inviável.
2. **Teste exaustivo é impossível**:
   - Priorize com base em **risco** e **frequência de uso** (ex: funcionalidades críticas primeiro).
3. **Teste antecipadamente**:
   - Quanto mais cedo, mais barato é corrigir (*bug = barata: multiplica-se se ignorado*).
4. **Agrupamento de defeitos**:
   - Bugs se concentram em módulos complexos, instáveis ou pouco testados.
   - Foque em "ninhos de bugs" (histórico de falhas + análise de risco).
5. **Paradoxo do pesticida**:
   - Testes repetitivos perdem eficácia. Atualize-os conforme o software evolui.
6. **Teste depende do contexto**:
   - Mais risco = mais testes (ex: software médico vs. quiosque de shopping).
   - *Regra:* "Sem riscos, sem testes".

7. **Ilusão da ausência de erros**:
    - Software "perfeito" tecnicamente pode não atender às necessidades do cliente.
    - Foque em **validação** (atende o objetivo?) além de **verificação** (funciona?).

ISO 25010: Características de Qualidade

1. **Funcionalidade**:
    - Completude, correção e adequação ao usuário.
2. **Usabilidade**:
    - Reconhecibilidade, aprendizagem, operação, proteção contra erros, estética e acessibilidade.
3. **Confiança**:
    - Maturidade(prevenção de problemas), tolerância a falhas, disponibilidade, recuperação.
4. **Eficiência**:
    - Desempenho, otimização de recursos, capacidade.
5. **Segurança**:
    - Confidencialidade, integridade, não repúdio, responsabilidade, autenticidade.
6. **Compatibilidade**:
    - Coexistência com outros sistemas e interoperabilidade.
7. **Manutenibilidade**:
    - Facilidade de ajustes e testes (modularidade, reusabilidade, analisabilidade, modificabilidade, testabilidade).
8. **Portabilidade**:
    - Adaptação a diferentes ambientes (SOs, dispositivos).

Teste Manual vs. Automação

- **Manual**: Essencial para testes exploratórios e validação de correções.

	• **Automação**: Crucial para regressão, velocidade e integração contínua. > **Sinergia**: Combine ambos para cobertura máxima e eficiência. --- **Conclusão** • **Testar é prevenir**: Reduz riscos, mas não elimina todos os bugs. • **Qualidade é responsabilidade de todos**: Desenvolvedores, testadores e gestores. • **Contexto define esforço**: Software crítico exige rigor; aplicações simples priorizam custo-benefício. > *"Testadores são guardiões da qualidade: buscam defeitos incansavelmente, mas sabem que a perfeição é inatingível."*	

@Wesley Lima	• História do teste
	  ◦ Primeiro registro de defeito: Ada Lovelace
	  ◦ Origem do termo “teste de caixa branca”: Herman Hollerith
	  ◦ Cunhou o termo “bug”: Grace Hopper
	  ◦ Criou a “bíblia do teste”: Glenford Myers
	  ◦ Profissionais importantes no Brasil: Emerson Rios, Ricardo Cristália, Leonardo Molinari
	• Danos dos Bugs
	  ◦ Empresa/Organizações:
	  ▪ Atrasos
	  ▪ Perda de Confiança
	  ▪ Vendas
	  ◦ Pessoas
	  ▪ Constrangimento
	  ▪ Perda ou Supressão de Direitos
		Seção 02

        - Risco de vida e acidentes
    - Governos
        - Vulnerabilidade de Informações
        - Decisões estratégicas incorretas
        - Derrotas Militares
    - Meio Ambiente
        - Alertas Atrasados
        - Desperdício de Recursos
        - Poluição
- 7 fundamentos do teste (ISTQB)
    - O teste mostra a presença de defeitos, mas nunca sua ausência
        - Não há como garantir um software 100% sem defeitos.
        - Correções podem gerar novos defeitos
        - O ponto é resolver os problemas críticos e mais óbvios como prioridade.
    - Teste exaustivo não é possível
        - Testar absolutamente tudo é inviável
        - Riscos e prioridades são levadas em consideração para a seleção dos testes
    - Teste antecipado
        - Começar os testes o quanto antes no processo de desenvolvimento, com objetivos definidos
        - Quanto mais cedo o defeito for achado, mais barato será sua correção
        - Regra 10 de Myers
    - Agrupamento de Defeitos
        - Bugs se distribuem de forma heterogênea
        - Alguns módulos têm mais defeitos que outros
        - Achando os módulos problemáticos, se encontram a maioria dos defeitos (ninho)
    - Paradoxo do Pesticida

        - testes específicos podem não ter mais resultado após algum tempo.
        - Isso não indica que não há bugs, e sim que os testes devem ser revisados e atualizados
    - Testes dependem do contexto
        - Testes dependem de contextos de: uso, riscos, impactos, etc.
        - Foco no cliente
    - Ilusão da ausência de erro
        - Encontrar e consertar defeitos é inefetivo se o software não atende as expectativas e necessidades do cliente (e do usuário).
- Diferença entre teste e QA
    - Teste foca em produto
    - Teste é uma ferramenta
    - QA foca em processo
        - Retrospectivas para melhorar processos
        - Prevenção para qualidade
        - Monitoramento
        - Recomendações de boas práticas
- Erro vs Defeito vs Falha
    - Erro
        - Enganos cometidos pela pessoa
        - Cometidos durante o desenvolvimento (pelo programador), ou durante o uso (pelo usuário), por exemplo.
        - Identificado por quem cometeu
    - Defeito
        - Engados cometidos por outra pessoa, identificados por outra
    - Falha
        - Instância do problema, onde se deve averiguar a causa e o efeito.
- Testes IEC/ISO 25010

- Adequação funcional (AF)
  - Funcionalidade
    - Completude funcional
    - Correctness funcional
    - Apropriado como funcionalidade
  - Testes de negócio
  - Como o cliente quer a funcionalidade
- Usabilidade (U)
  - Focado no usuário e interface.
  - Reconhecer
  - Aprender
  - Operar
  - Proteção contra erros do usuário
  - Estética (UI)
  - Acessibilidade
- Compatibilidade (C)
  - Relação com outros softwares
  - Coexistência
  - Interoperabilidade
- Confiabilidade (C)
  - Maturidade
  - Disponibilidade
  - Tolerância a falha
  - Recuperabilidade
- Eficiência no desempenho (E)
  - Comportamento em relação ao tempo
  - Utilização de Recursos
  - Capacidade
- Manutenibilidade (M)
  - Modularidade (organizado em módulos)
  - Reusabilidade
  - Analisabilidade
  - Modificabilidade

	▪ Testabilidade ◦ Portabilidade (P) ▪ Adaptabilidade ▪ Instalabilidade ▪ Substituibilidade ◦ Segurança (S) ▪ Confidencialidade ▪ Integridade ▪ Não repúdio (garantia de uso) ▪ Responsabilidade (Auditável) ▪ Autenticidade (Autenticação	

Seção 03:

Integrante	Anotações feitas	Seção correspondente
@Agemilson Abreu	**1. Lidando com a Pressão e Autogerenciamento** Para prosperar no ambiente de alta **pressão** dos projetos de tecnologia, o profissional de testes deve adotar uma postura estratégica e praticar um rigoroso **autogerenciamento**. • **Papel Estratégico:** Atuar como o **"conselheiro da qualidade"**, educando a equipe, comunicando riscos de forma clara para todos os stakeholders (gestores, POs, desenvolvedores) e construindo credibilidade. • **Mentalidade Profissional:** Ser **"comprometido"**, com uma "visão de dono", em vez de apenas "envolvido". Isso significa ser proativo e focado no sucesso do projeto, não apenas em cumprir o mínimo.	Seção 03

- **Ferramenta Prática (Autogerenciamento):** Gerenciar de forma eficaz seus recursos pessoais:
  - **Tempo:** Usá-lo com sabedoria, delegando tarefas de baixo valor.
  - **Energia:** Conhecer seu próprio ritmo e usá-lo a seu favor.
  - **Tarefas:** Priorizar usando métodos como a matriz GUT (Gravidade, Urgência, Tendência).
  - **Compromissos:** Cuidar primeiro de si mesmo para poder cuidar melhor dos outros.

---

2. Guia de Comunicação e Negociação

A comunicação eficaz e a negociação justa são pilares para construir relacionamentos de trabalho sólidos e produtivos.

- **A Complexidade da Comunicação:** O sucesso de uma mensagem depende menos das **palavras (10%)** e muito mais do **tom de voz (35%)** e da **linguagem corporal (55%)**. É vital adaptar a comunicação (formal/informal) ao contexto e ao público, e praticar a escuta ativa.
- **A Arte da Negociação:** A abordagem ideal é sempre o **"Ganha-Ganha" (Win-Win)**, onde ambas as partes saem satisfeitas. Negociações "Ganha-Perde" criam ressentimento e prejudicam relações a longo prazo, enquanto "Perde-Perde" é um fracasso para todos. Um bom negócio é aquele que beneficia ambos os lados.

---

	3. Produtividade Pessoal e de Equipe A produtividade é construída através de hábitos individuais e de um ritmo de trabalho sustentável para a equipe, utilizando ferramentas práticas para otimizar o fluxo. • **Produtividade Pessoal:** Baseia-se em cultivar **hábitos saudáveis** (exercícios, sono, boa alimentação), focar no presente, aprender com os erros e cercar-se de pessoas positivas. • **Produtividade da Equipe (Fluxo Contínuo):** Uma equipe deve trabalhar em um ritmo de **"maratona", não de "sprint"**. Um ritmo sustentável e sem pressão excessiva evita o esgotamento e, paradoxalmente, aumenta a produtividade real a longo prazo. • **Ferramentas Práticas:** ◦ **Kanban:** Método visual que otimiza o fluxo de trabalho ao limitar as tarefas em progresso (WIP Limit), evitando "engarrafamentos" e focando na conclusão. ◦ **Técnica Pomodoro:** Combate as interrupções dividindo o trabalho em blocos de **25 minutos de foco total**, seguidos por pequenas pausas. Isso torna grandes tarefas gerenciáveis e aproveita pequenos intervalos de tempo.	
@Eric Lima da Silva	https://weslima.atlassian.net/wiki/spaces/PSB/pages/edit-v2/2687053#Se%C3%A7%C3%A3o-3%3A-Implementa%C3%A7%C3%A3o-e-Pr%C3%B3ximos-Passos Can't find link **Pressão Organizacional e o Papel do Testador** • **Fontes de Pressão:** O testador enfrenta pressão de diversas áreas: ◦ **Desenvolvedores:** Pressionados por prazos. ◦ **Product Owner (PO):** Ansioso para mostrar progresso.	Seção 03

    - **Scrum Master:** Focado em produtividade.
    - **Gestores:** Precisam de informações sobre qualidade e prazos.
    - **Cliente e Usuário:** Querem que os objetivos de negócio e a usabilidade sejam atendidos.
- **O Testador como "Conselheiro da Qualidade":** A função principal do testador é orientar a equipe sobre qualidade.
    - **Educar a Equipe:** Promover uma cultura de qualidade coletiva.
    - **Comunicar Riscos:** Informar sobre o que não foi testado e os bugs existentes.
    - **Lembrar do Impacto:** Conscientizar sobre as consequências dos defeitos (financeiras, de imagem, etc.).
    - **Aconselhar, Não Decidir:** Fornecer dados para que a equipe e o cliente tomem a decisão final sobre o lançamento.

Comprometido vs. Envolvido

- **Profissional "Envolvido" (Passivo):**
    - Faz apenas o mínimo necessário.
    - Usa desculpas como "fiz a minha parte" e não tem visão de equipe.
    - Espera que as coisas aconteçam sem agir proativamente.
- **Profissional "Comprometido" (Ativo):**
    - Gosta do que faz, gerando um ciclo positivo de motivação e reconhecimento.
    - Busca meritocracia e agrega valor à empresa.
    - Tem "visão de dono": é proativo para melhorar processos e economizar recursos.
    - É um eterno estudante, aproveitando o trabalho para aprender.

- **Filosofia "Ser para Ter":** Para obter uma posição (como líder ou testador), primeiro aja como se já a tivesse. A promoção é o reconhecimento do valor que você já entrega.

Autogerenciamento

- **Gestão do Tempo:** Priorize tarefas importantes, equilibre as áreas da vida e delegue quando possível.
- **Gestão da Energia:** Identifique seus horários de pico de produtividade e use-os para as tarefas mais difíceis.
- **Gestão de Recursos:** Seja criativo para obter o que precisa (livros, ferramentas) em vez de usar a falta deles como desculpa.
- **Gestão de Tarefas:** Siga um processo:
  a. **Identificar:** Liste tudo o que precisa ser feito.
  b. **Classificar:** Priorize usando métodos como a Matriz GUT.
  c. **Negar ou Delegar:** Diga "não" a tarefas de baixo impacto.
  d. **Sequenciar:** Organize as tarefas na ordem mais eficiente.
  e. **Medir:** Monitore o tempo gasto para melhorar estimativas futuras.
- **Gestão de Restrições:** Reconheça e trabalhe dentro dos limites que não podem ser alterados (prazos, regras).
- **Gestão de Compromissos:** Honre suas promessas, seguindo a hierarquia: 1º) você mesmo, 2º) família/amigos, 3º) cliente/empresa.

Comunicação Verbal e Não Verbal

- **Comunicação Verbal:**
  - **Escrita:** Exige clareza e boa redação, pois falta o contexto da linguagem corporal.

    - **Oral:** A habilidade mais importante é saber ouvir para entender o outro.
- **Comunicação Não Verbal (a mais impactante):**
    - **Símbolos e Cores:** O uso de cores, ícones e formatação (ex: letras maiúsculas) afeta a percepção.
    - **Aparência:** A primeira impressão é formada em segundos e é influenciada pela vestimenta e ambiente.
    - **Movimentos (Cinestésica):** Gestos, expressões faciais e postura devem estar alinhados com a mensagem.
    - **Para-linguagem:** O tom, o ritmo e as pausas da voz podem reforçar ou contradizer as palavras.
    - **Proxêmica:** O uso do espaço (distância entre pessoas, organização do ambiente) comunica profissionalismo.

Negociação

- **Relação Ganha-Perde:** Uma parte vence à custa da outra. Não é sustentável a longo prazo, pois gera ressentimento.
- **Relação Perde-Perde:** O pior cenário, onde ninguém atinge seus objetivos e ambos saem frustrados.
- **Relação Ganha-Ganha (O Modelo Ideal):**
    - Ambas as partes saem satisfeitas, chegando a um acordo justo.
    - Baseia-se no benefício mútuo e constrói relações de confiança e de longo prazo.
    - A chave é entender os objetivos e limites da outra parte para encontrar uma solução boa para todos.

Produtividade

- **Viver o Hoje:** Concentre-se no presente, pois é o único momento que você pode controlar. Suas

ações de hoje constroem o seu futuro.

- **Se Exercitar:** A saúde física é fundamental para o desempenho mental. Exercícios melhoram o sono, a memória, o raciocínio e aliviam o estresse.
- **Cultivar Rituais Saudáveis:** Adote hábitos como alimentação correta, sono de qualidade (pelo menos 7 horas) e boa hidratação para manter a energia.
- **Afastar-se de Pessoas Negativas:** O ambiente social impacta sua motivação. Afaste-se de quem te coloca para baixo e conecte-se com pessoas inspiradoras.
- **Aprender com os Erros:** Encare o erro como parte essencial do aprendizado. Analise suas falhas para aplicar a melhoria contínua.
- **Formalizar Seus Sonhos:** Torne seus objetivos públicos (escrevendo, postando). Isso cria um compromisso social e permite que outros te ajudem.
- **Ser Grato:** Praticar a gratidão pelo que você já tem acalma a mente, reduz o estresse e cria uma base segura para seguir em frente.
- **Curtir a Jornada:** A felicidade não está apenas no destino, mas no processo. Encontre satisfação no percurso e valorize quem te acompanha.
- **Focar na Solução:** Não se prenda aos problemas. Concentre sua energia em encontrar soluções, adotando a mentalidade de que quase tudo pode ser resolvido.
- **Conclusão:** O objetivo final é ser melhor hoje do que o seu "eu" de ontem.

Fluxo Contínuo

- **O Problema (Ritmo Insustentável):** Trabalhar constantemente em ritmo de urgência ("correr para

pegar o ônibus") leva ao esgotamento (*burnout*) e é sinal de mau planejamento.

- **A Solução (Ritmo Sustentável):**
  a. **Aceitar a Capacidade Real:** Ajuste o planejamento ao que a equipe realmente consegue entregar, reduzindo a pressão e a frustração.
  b. **Treinar para Aumentar a Capacidade:** Trabalhando em um ritmo sustentável, a equipe ganha confiança e eficiência, aumentando sua produtividade naturalmente com o tempo.
- **O Paradoxo (Ir mais Devagar para Chegar mais Rápido):** Um ritmo de trabalho mais lento e constante reduz erros graves (que paralisam o projeto), tornando o fluxo geral mais eficiente e rápido no final.

Metodologia Kanban

- **O que é:** Uma metodologia visual (nascida na Toyota) para gerenciar e otimizar o fluxo de trabalho.
- **Componentes do Quadro:**
  - **Colunas:** Representam as etapas do processo (Ex: A Fazer, Em Construção, Teste, Pronto).
  - **Limites de WIP (Work in Progress):** Número máximo de tarefas permitidas em cada coluna de trabalho ativo para evitar gargalos e forçar a finalização das tarefas.
- **Duas Formas de Uso:**
  - **Kanban dentro de um Sprint:** A equipe trabalha para mover um "pacote" de tarefas para "Pronto" dentro de um ciclo fixo (ex: duas semanas).
  - **Fluxo de Entrega Contínua ("Kanban Puro"):** Não há ciclos fixos. Cada tarefa é entregue

	individualmente assim que fica pronta. Exige alta maturidade e automação. **Técnica Pomodoro** • **Como Funciona:** ◦ Trabalhe com foco total em uma única tarefa por **25 minutos**. ◦ Faça uma pausa curta de **5 minutos**. ◦ Após 4 ciclos ("pomodoros"), faça uma pausa longa de **15 a 30 minutos**. • **Aplicação Estratégica:** Quebre grandes tarefas em pequenos "pomodoros". Isso permite usar pequenas janelas de tempo ao longo do dia para progredir em grandes projetos. • **Impacto da Consistência:** A prática regular, mesmo que de poucos pomodoros por dia, cria um hábito de foco que gera centenas de horas de trabalho produtivo ao longo do ano, dando a você o controle do seu tempo.	
@Lívia Barbosa	**Comprometido x Envolvido:** Comprometido é aquele que se interessa pela sua função, que está sempre buscando aprimorar os seus conhecimentos e impactar positivamente o seu ambiente de trabalho. Já o envolvido é aquele que apenas vai conforte o fluxo, não se esforça para realizar as suas atividades e tem um pensamento individualista. **Autogerenciamento:** gestão de tempo, energia, recursos, tarefas, restrições e compromissos para promover uma maior organização pessoal e, consequentemente, profissional. **Negociação:** buscar um meio de que o resultado se torne favorável, mesmo que parcialmente, para todas as partes. **Produtividade:** Priorizar e dividir seu tempo para a realização das atividades diárias, sabendo quando	Seção 03

	realizar uma menor quantidade de atividades para manter a produtividade.	
@Mário Queiroz		Seção 03

---

1. O Problema: Ritmo de "Corrida para o Ônibus" vs. "Maratona"

- **O "Gás" Insustentável:** Assim como uma pessoa não consegue correr para casa na mesma velocidade de um pique para pegar o ônibus, uma equipe não pode trabalhar constantemente em ritmo de "emergência". Fazer isso é um sinal de **mau planejamento** e leva ao esgotamento (burnout).
- **O Ritmo da "Maratona":** O ideal é um **fluxo contínuo e sustentável**, um ritmo que a equipe consegue manter a longo prazo sem prejudicar a saúde dos membros ou a qualidade do produto.

---

2. O Paradoxo: "Ir Mais Devagar para Chegar Mais Rápido"

- **Analogia do Atleta:** Uma equipe que não está performando no nível desejado é como um atleta sedentário. Forçá-la a "correr a São Silvestre" (entregar em alta performance) só causará frustração e fracasso. O correto é **aceitar a capacidade atual** (ex: 15 pontos por sprint, em vez de 20) e treinar para melhorar gradualmente.
- **Analogia do Trânsito:** Reduzir o limite de velocidade em uma avenida (de 90 para 70 km/h)

pode **aumentar a velocidade média** do fluxo, pois diminui a frequência e a gravidade dos acidentes (paradas). Da mesma forma, uma equipe que aceita uma carga de trabalho um pouco menor pode reduzir erros e estresse, tornando-se, no fim, **mais produtiva e previsível**.

---

3. Kanban: A Ferramenta para Otimizar o Fluxo

Kanban é uma metodologia visual focada em otimizar o fluxo de trabalho e aumentar a eficiência.

- **O Quadro Kanban:** É a ferramenta central, que visualiza o trabalho em colunas. Uma estrutura comum inclui:
  a. **A Fazer (To Do):** Tarefas que ainda não foram iniciadas.
  b. **Em Construção (Doing):** Tarefas que estão sendo trabalhadas ativamente.
  c. **Teste (Testing):** Tarefas concluídas que estão em fase de verificação.
  d. **Pronto (Done):** Tarefas finalizadas e entregues.
- **Limites de Trabalho em Progresso (WIP - Work in Progress):** Este é o conceito mais importante do Kanban. Limita-se o número de tarefas que podem estar na coluna "Em Construção" ao mesmo tempo.
  - **Benefício:** Isso evita que a equipe comece muitas coisas sem terminar nada, forçando o foco na **conclusão de tarefas** e identificando gargalos no processo.
- **Entrega Contínua:** Em sua forma mais avançada, o Kanban permite que cada tarefa seja implantada assim que fica pronta, eliminando a necessidade de esperar pelo fim de um ciclo (sprint) para entregar

valor.

Técnica Pomodoro

O ciclo funciona da seguinte forma:

a. **Escolha uma tarefa** para ser realizada.
b. **Trabalhe com foco total** nela por **25 minutos** (isso é um "pomodoro"). Durante este período, nenhuma interrupção é permitida.
c. Ao final dos 25 minutos, faça uma **pausa curta de 5 minutos**. Use esse tempo para se levantar, beber água ou ir ao banheiro.
d. Repita o ciclo. Após completar **quatro pomodoros**, faça uma **pausa longa** (de 15 a 30 minutos).

A Chave do Sucesso: Dividir e Conquistar

O verdadeiro poder da técnica não está apenas no cronômetro, mas na mentalidade que ela promove:

- **Quebrar Grandes Tarefas:** Em vez de se intimidar com uma tarefa que levará "dois dias", você a quebra em blocos gerenciáveis. Uma tarefa de 16 horas, por exemplo, se transforma em **32 pomodoros**.
- **Aproveitar "Bolsões de Tempo":** Essa abordagem permite que você utilize pequenas janelas de tempo inesperadas durante o dia (um voo atrasado, a espera no consultório médico) para completar um ou mais pomodoros e avançar em grandes projetos.

Benefícios e Desafios

- **Benefícios:**
- **Aumento drástico da produtividade** ao forçar o foco.

	◦ **Sensação clara de progresso**, pois você pode medir seu dia em "pomodoros concluídos". ◦ **Melhor gestão do tempo** e combate à procrastinação. ◦ **Desafio:** ◦ A técnica exige **disciplina para não se deixar interromper** durante os 25 minutos de foco. É preciso aprender a ignorar notificações e, quando possível, comunicar aos colegas que você está em um bloco de trabalho focado. ***Um Começo Prático e o Poder do Hábito*** O texto sugere um desafio inicial: tente completar **15 pomodoros por semana**. Isso equivale a cerca de 1h30 de trabalho focado por dia. Embora pareça pouco, o efeito cumulativo é enorme: ◦ **15 pomodoros/semana** = 60 por mês = **720 por ano**. Essa consistência gera um crescimento profissional imenso em poucos meses, pois a maioria das pessoas não consegue manter esse nível de foco. **Em resumo, a Técnica Pomodoro é uma ferramenta para você tomar o controle do seu tempo, garantindo que, em vez de passar o dia reagindo a interrupções, você construa resultados concretos e progressivos.**	
@Wesley Lima	• Pressão organizacional ◦ Stakeholders do testador ▪ Desenvolvedor ▪ PO ▪ Scrum Master/Agile Coach ▪ Gestores (produto, projeto, teste, qualidade, desenvolvimento, infraestrutura, etc)	Seção 03

        - Clientes
        - Usuários
    - Você é o “conselheiro do Rei”
        - Tudo o que fazemos tem um nível de qualidade, mas talvez não seja o nível aceitável. A quaidade é subjetiva para quem não foca nela. Logo, o QA é a autoridade nisso.
        - Lembrar o time das consequências e das finalidades.
        - Não toma decisões, apenas aconselha e evidencia problemáticas.
- Comprometido x Envolvido
    - Envolvido: Indo com o fluxo, sem esforço.
    - Se esforça
- Autogerenciamento:
    - Seus Assets:
        - Tempo
        - Energia
        - Recursos
        - Tarefas
        - Restrições
        - Compromisso
- Comunicação verbal e não verbal
    - Influencia na mensagem:
        - Palavras → 10%
        - Tom de voz → 35%
        - Comportamento não verbal → 55%
    - Entender o nível de formalidade necessário para cada interação
    - Planejamento é importante para a fala
- Negociação
    - Prezar que o resultado seja favorável a todos.
- Produtividade

	◦ Saber focar e dividir seu tempo, mantendo o fluxo contínuo. Em momentos específicos, diminuição de ritmo para manter a produtividade.	