Responsabilidades QA

1. Contexto e Objetivo

Este documento define, de forma clara e objetiva, as responsabilidades de Quality Assurance (QA) no projeto BIUD V3 / ESG Sebrae. O objetivo é estabelecer o escopo de atuação do QA em todas as fases do ciclo de desenvolvimento, desde o refinamento até o deploy, promovendo alinhamento com o time de desenvolvimento, PO e stakeholders.

Problemas que este documento busca evitar:

- Expectativas desalinhadas sobre o que o QA deve validar e em qual momento.
- Sobrecarga do QA com atividades fora do escopo (ex.: correção de bugs, deploy manual).
- Gaps de qualidade por falta de definição de “quem valida o quê”.
- Conflitos sobre prazos e entregas (ex.: “o QA não testou a tempo”).
- Baixa participação do QA em momentos estratégicos (refinamento, planejamento, discussões de solução).

2. Princípios de Atuação do QA

- Shift-Left: testar e questionar o mais cedo possível (principalmente no refinamento).
- Qualidade como responsabilidade compartilhada: QA facilita e valida; o time constrói qualidade.
- Teste orientado a risco: foco adicional em “caminho crítico” (pagamento, segurança, banco de dados, integrações).
- Transparência: status, riscos, evidências e bugs visíveis para o time.
- Reprodutibilidade: todo bug deve ter passos claros e evidência. Todo bug relevante deve estar ligado a um ciclo/release e a um caso de teste quando aplicável.
- Acompanhamento de dados quantitativos e qualitativos de bugs em produção.

3. Responsabilidades do QA por Fase

3.1. Refinamento (Antes do Desenvolvimento)

- Participar das reuniões de refinamento de backlog.
- Validar se os cards atendem ao DoR (história no formato padrão; critérios de aceite claros e testáveis; dependências identificadas; priorização definida; anexos/protótipos quando aplicável).
- Questionar critérios de aceite ambíguos ou não testáveis.
- Identificar riscos de qualidade e dependências técnicas.
- Sugerir cenários alternativos, edge cases e cenários de regressão.
- Antecipar necessidade de massa de dados, ambientes e acessos para teste.

3.2. Planejamento (Planning / PB Meetings)

- Levantar dúvidas e riscos sobre funcionalidades e integrações.
- Ajudar o time a explicitar critérios de aceite e escopo de testes.
- Estimar esforço de testes (inclui entendimento de cenários, dados e preparação de ambiente).

3.3. Durante o Desenvolvimento

- Acompanhar progresso das tarefas no gerenciador (ClickUp) e sinalizar riscos/bloqueios.
- Trabalhar próximo ao dev para validar entendimento do Acceptance Criteria ("devbox"/pareamento quando necessário).
- Criar/atualizar casos de teste (manuais e/ou automatizados) conforme o documento/escopo da sprint.
- Preparar massa de dados e ambientes de teste.
- Validar se testes unitários foram executados pelos devs quando aplicável (o QA não substitui o dev em testes unitários).

3.4. Fase de Testes (QA Ativo)

- Executar testes funcionais (happy path e cenários alternativos).
- Executar testes de regressão em funcionalidades impactadas.
- Executar testes de integração (APIs, banco de dados, serviços), quando aplicável.
- Validar responsividade e compatibilidade (browsers/dispositivos) quando aplicável.
- Executar testes básicos de segurança quando aplicável (ex.: validações de acesso, input, dados sensíveis).
- Reportar bugs com informações corretas, evidências e severidade/prioridade.
- Re-testar correções e aprovar/reprovar entrega conforme critérios de aceite.

3.5. Homologação e Deploy

- Validar o deploy em ambiente de homologação (pré-condição para testes finais).
- Executar smoke tests pós-deploy.
- Apoiar o PO na validação final e na tomada de decisão de go/no-go.
- Documentar evidências de testes (links, prints, vídeos curtos quando necessário).
- Bloquear go-live quando houver bugs críticos/bloqueadores (comunicando riscos e impacto).

3.6. Pós-Deploy (Produção)

- Apoiar validação de hotfixes em produção (quando aplicável).
- Apoiar monitoramento de incidentes/bugs reportados por usuários (triagem e reprodução).
- Coletar feedback para retroalimentar melhorias de qualidade e prevenção de regressões.
- Atualizar documentação técnica de testes quando relevante.

3.7. Melhoria Contínua

- Propor automação de testes repetitivos e regressões frequentes.
- Manter e evoluir suíte automatizada (quando existir), em conjunto com o time técnico para garantir viabilidade.

- Gerar e acompanhar métricas de qualidade (ex.: bugs por sprint, reabertura, vazamento para produção).
- Participar de retrospectivas com foco em ações de qualidade e prevenção.
- Revisar e atualizar padrões de QA (DoR/DoD, template de bug, checklists).

4. O que NÃO é responsabilidade do QA

- Corrigir bugs (responsabilidade do Dev).
- Escrever código de produção (exceto scripts/infra de automação de testes, quando alinhado).
- Definir prioridades de backlog (responsabilidade do PO).
- Executar deploy em produção (responsabilidade de DevOps/Infra; QA valida em homologação e apoia go/no-go).
- Substituir testes unitários do time de desenvolvimento.
- Aprovar cards que não atendem ao DoR/DoD por pressão de prazo.

5. Fluxo Mínimo de QA (Operacional)

Fluxo recomendado por história/tarefa:

1. Refinamento: validar DoR + critérios de aceite testáveis + riscos e dependências.
2. Planejamento: alinhar escopo de testes e necessidades (dados/ambiente).
3. Durante o desenvolvimento: acompanhar, revisar testabilidade quando necessário e preparar CTs/massa.
4. Deploy em homologação: confirmar versão/correção correta para o ciclo/release.
5. Execução de testes: funcional + regressão + integrações conforme risco.
6. Registro de bugs: com severidade, prioridade, evidência e rastreabilidade (CT/ciclo).
7. Re-teste: validar correções, evitar reabertura por falta de evidência.
8. Aprovação: liberar ou bloquear conforme critérios e riscos.

Observação: este documento é vivo e deve ser revisado conforme evolução do produto e do processo.