Daily Board #03 - KYC (Know Your Customer)

NotusLabs DX Research - Relatório Completo

Data: 02/10/2025

Sessão de Teste

1. Qual é o objetivo desta sessão?

Validar a funcionalidade de **KYC (Know Your Customer)** da API NotusLabs, testando:

- Criação de sessões de verificação individual
- Consulta de status das sessões
- Processamento de verificações
- Validações de campos obrigatórios
- Tratamento de erros (sessões inexistentes, campos faltando)
- Fluxo completo de verificação de identidade

---

2. Qual abordagem você vai usar?

Ferramenta: Postman para testes de API REST

Metodologia:

1. Testar criação de sessão KYC (happy path)
2. Consultar status da sessão criada
3. Tentar processar verificação (sem documentos para validar comportamento)
4. Consultar status após processamento
5. Testar casos de erro (campos faltando, sessão inexistente)
6. Documentar fluxo completo e identificar problemas de DX

Endpoints testados:

- POST /kyc/individual-verification-sessions/standard
- GET /kyc/individual-verification-sessions/standard/{sessionId}
- POST /kyc/individual-verification-sessions/standard/{sessionId}/process

---

3. Há algo que precisa ser configurado antes de começar?

Configurações realizadas:

- API Key configurada no Postman
- Variáveis de ambiente atualizadas
- Documentação de KYC consultada

- Campos obrigatórios identificados
- Pasta "KYC" criada no Postman

Informações da documentação:

- 3 endpoints disponíveis
- Campos obrigatórios identificados (firstName, lastName, birthDate, etc)
- Tipos de documentos aceitos: PASSPORT, DRIVERS_LICENSE, IDENTITY_CARD
- Upload de documentos via URLs pré-assinadas (S3)

Contexto:

- Sessions anteriores (Smart Wallet) já completadas
- 8 problemas de DX já identificados nas sessões anteriores

4. Você conseguiu atingir o objetivo da sessão?

- Sim

Detalhamento:

Consegui testar todos os 3 endpoints de KYC e validar o fluxo completo de verificação:

- Criação de sessão funcionou perfeitamente
- Consulta de status operacional
- Processamento testado (comportamento assíncrono validado)
- Casos de erro testados com sucesso

Estatísticas:

- Total de testes: 6
- Testes de sucesso: 4
- Testes de erro/validação: 2
- Novos problemas identificados: 1
- Observações sobre UX: 1

Limitações:

- Não foi possível testar upload real de documentos (requer arquivos de imagem)
- Não foi possível alcançar status APPROVED (requer documentos válidos)
- Teste limitado ao fluxo de validação e estrutura da API

5. Problemas encontrados

Problema #9: Documentação incorreta sobre campos opcionais de KYC

Severidade: Médio

Descrição: a documentação da API indica que diversos campos são opcionais (alguns marcados com `?`, outros sem marcação clara), mas na prática a API os trata como obrigatórios, retornando erro 400 quando ausentes.

Campos com inconsistência:

Documentação indica como opcional:

> - `nationality?` (marcado com `?`)

Documentação não marca claramente, mas API trata como obrigatório:

> - `livenessRequired` (boolean)
> - `email` (string)
> - `address` (string)
> - `city` (string)
> - `state` (string)
> - `postalCode` (string)

Como reproduzir:

```
POST /kyc/individual-verification-sessions/standard

Body (seguindo documentação - campos "opcionais" omitidos):
{
  "firstName": "João",
  "lastName": "Silva",
  "birthDate": "1990-01-15",
  "documentCategory": "IDENTITY_CARD",
  "documentCountry": "BRAZIL",
  "documentId": "123456789"
}
```

Response:

```
Status: 400 Bad Request
{
    "message": "Bad Request",
    "id": "BAD_REQUEST",
    "errors": [
        {"path": ["firstName"], "message": "Required"},
        {"path": ["lastName"], "message": "Required"},
        {"path": ["livenessRequired"], "message": "Required"},
        {"path": ["email"], "message": "Required"},
        {"path": ["address"], "message": "Required"},
        {"path": ["city"], "message": "Required"},
        {"path": ["state"], "message": "Required"},
        {"path": ["postalCode"], "message": "Required"}
    ]
}
```

Impacto na DX:

> - Desenvolvedor seguindo documentação receberá erro inesperado
> - Precisa descobrir campos verdadeiramente obrigatórios por tentativa e erro
> - Aumenta tempo de integração
> - Causa frustração e desconfiança na documentação

- Múltiplas tentativas necessárias para descobrir todos os campos

Pontos positivos da validação:

- Mensagens de erro claras
- Lista todos os campos faltando de uma vez (não um por vez)
- Pattern de erro consistente

Sugestão de melhoria:

1. Atualizar documentação marcando todos campos obrigatórios corretamente
2. OU tornar campos realmente opcionais na API (se fizer sentido do ponto de vista de negócio)
3. Adicionar nota explicando quais campos são obrigatórios para compliance

---

Observação sobre Teste #16/17: Comportamento do processamento assíncrono

Tipo: observação de UX (não é bug, mas pode melhorar)

Descrição: o endpoint POST /process retorna status 204 No Content mesmo quando não há documentos enviados, sugerindo sucesso. Porém, o processamento efetivamente falha de forma assíncrona.

Fluxo observado:

Passo 1 - Processar sem documentos:

```
POST /kyc/.../process
Response: 204 No Content (sugere sucesso)
```

Passo 2 - Consultar status:

```
GET /kyc/.../standard/{sessionId}
Response:
{
    "session": {
        "status": "FAILED",
        "updatedAt": "2025-10-02T06:06:07.357Z"
    }
}
```

Comportamento:

- Status 204 normalmente indica "operação bem-sucedida, sem conteúdo para retornar"
- Mas verificação KYC falhou por ausência de documentos
- Desenvolvedor pode assumir que processamento foi concluído com sucesso
- Necessário fazer segunda requisição GET para descobrir falha

Por que pode causar confusão:

- HTTP 204 é usado para indicar sucesso
- Não há indicação imediata de que algo deu errado
- Processamento assíncrono não é explicitamente documentado
- Sem mensagem sobre o motivo da falha no response

Não é necessariamente um bug porque:

- Processamento assíncrono é uma escolha arquitetural válida
- 204 indica que requisição foi aceita (não que verificação passou)
- Status FAILED aparece corretamente na consulta posterior

Sugestão de melhoria:

1. Validar presença de documentos ANTES de aceitar processamento
   - Retornar 400 se documentos obrigatórios não foram enviados
2. OU documentar claramente o comportamento assíncrono
3. OU adicionar campo "failureReason" no GET response:

```
{
    "session": {
        "status": "FAILED",
        "failureReason": "DOCUMENTS_NOT_UPLOADED",
        "failureDetails": "Front and back document images are required"
    }
}
```

6. Observações adicionais

Testes Executados - Detalhamento

Teste #14: Create KYC Verification Session

- Endpoint: POST /kyc/individual-verification-sessions/standard
- Status: Sucesso completo
- Session ID gerado: `a7fd2ad4-12b4-4464-bbca-0fbbafd7ef98`
- Response inclui URLs pré-assinadas para upload S3
- Tempo de expiração das URLs: 10 minutos

Teste #15: Get KYC Session Status (inicial)

- Endpoint: GET /kyc/.../standard/{sessionId}
- Status: PENDING (correto - documentos não enviados)
- Retorna dados completos da sessão

Teste #16: Process KYC Session (sem documentos)

- Endpoint: POST /kyc/.../standard/{sessionId}/process
- Response: 204 No Content
- Operação aceita mas verificação falhará

Teste #17: Get KYC Session Status (após processar)

- Status mudou: PENDING → FAILED
- Campo updatedAt preenchido

- Confirma processamento assíncrono

Teste #18: Create KYC Session - Missing Fields

- Validação funciona corretamente
- Revelou inconsistência na documentação
- Lista todos campos faltando de uma vez

Teste #19: Get KYC Session - Not Found

- Status 404 apropriado
- Mensagem clara
- Pattern consistente

Pontos Positivos da API

1. **Criação de Sessão:**
   - Processo simples e direto
   - Response completo com todas informações necessárias
   - URLs pré-assinadas para upload direto no S3 (eficiente)
   - Separação clara entre front e back do documento
2. **Consulta de Status:**
   - Endpoint simples (GET com sessionId)
   - Response limpo com status claro
   - Campos bem estruturados
3. **Validações:**
   - Validação robusta de campos
   - Mensagens de erro estruturadas
   - Lista todos problemas de uma vez (boa UX)
4. **Segurança:**
   - Upload via URLs pré-assinadas (não expõe credenciais)
   - Limite de tamanho de arquivo: 25MB
   - Expiração de URLs (10 minutos)
   - Autenticação via API Key
5. **Padrões:**
   - Status HTTP corretos (200, 204, 400, 404)
   - Pattern de erro consistente (message, id, errors/traceId)
   - IDs específicos para cada tipo de erro

Fluxo KYC Identificado

Fluxo completo esperado:

1. **Criar Sessão:**
   - POST /kyc/.../standard
   - Recebe sessionId + URLs de upload
2. **Upload de Documentos:**
   - Usar URLs fornecidas para fazer POST multipart para S3
   - Enviar frente e verso do documento
   - (Não testado - requer arquivos de imagem)
3. **Processar Verificação:**
   - POST /kyc/.../standard/{sessionId}/process
   - Retorna 204 (requisição aceita)
   - Processamento acontece assíncrono
4. **Consultar Resultado:**
   - GET /kyc/.../standard/{sessionId}
   - Status possíveis: PENDING, PROCESSING(?), APPROVED, FAILED
   - IndividualId preenchido quando aprovado

Sugestões de Melhoria

Alta Prioridade:

1. **Corrigir documentação de campos obrigatórios**
   - Marcar claramente todos campos obrigatórios
   - Remover ? de campos que são obrigatórios
   - Ou tornar campos realmente opcionais se possível
2. **Melhorar feedback de falhas**
   - Adicionar campo "failureReason" no GET response
   - Documentar possíveis valores de status
   - Explicar o que causou cada falha

Média Prioridade:

3. **Documentar processamento assíncrono**
   - Deixar claro que 204 significa "aceito" não "concluído"
   - Explicar tempo esperado de processamento
   - Documentar quando consultar status novamente
4. **Validação prévia de documentos**
   - Validar se documentos foram enviados antes de processar
   - Retornar erro específico se documentos faltam
5. **Melhorar exemplos na documentação**
   - Adicionar exemplo completo de upload de documentos
   - Mostrar fluxo end-to-end

- Incluir exemplos de responses para cada status

Baixa Prioridade:

6. **Webhook para notificações**
   - Notificar quando processamento completar
   - Evita necessidade de polling

Dúvidas Técnicas

1. **Upload de Documentos:**
   - Qual formato exato usar no POST para S3?
   - Multipart form-data com quais campos exatos?
   - Há validação de tipo de arquivo (JPEG, PNG)?
   - Há validação de tamanho de imagem?
2. **Status da Sessão:**
   - Quais são TODOS os status possíveis? (PENDING, PROCESSING, APPROVED, FAILED, outros?)
   - Há status intermediários?
   - Quanto tempo leva processamento típico?
3. **Liveness:**
   - Como funciona quando livenessRequired=true?
   - URLs adicionais são fornecidas?
   - Que tipo de validação é feita?
4. **IndividualId:**
   - Quando é gerado?
   - Para que serve?
   - Como usar em outras operações?
5. **Reprocessamento:**
   - Pode reprocessar uma sessão FAILED?
   - Precisa criar nova sessão?
   - Como fazer upload de novos documentos?

Comparação com Sessões Anteriores

Daily Board #01 (Smart Wallet - Base):

- 8 testes executados
- 4 problemas identificados

Daily Board #02 (Smart Wallet - Complementar):

- 5 testes executados
- 3 novos problemas + 1 confirmado

Daily Board #03 (KYC):

- 6 testes executados
- 1 novo problema + 1 observação
- Documentação é o principal ponto de atenção

Total acumulado:

- 19 testes executados
- 9 problemas identificados
- 1 observação importante de UX