Sumsub KYC Integration

Sumsub KYC/AML Integration

> Identity verification architecture for the Drop fintech platform using Sumsub as the KYC/AML provider. Covers webhook architecture, applicant lifecycle, document check types, risk level definitions, SDK integration for web and mobile, idempotency handling, and error recovery.

KYC Verification Flow

```
sequenceDiagram
    participant User as User (Web/Mobile)
    participant Client as Drop Client
    participant API as drop-api (Hono)
    participant DB as Database
    participant Sumsub as Sumsub API
    participant SumsubSDK as Sumsub SDK Widget

    Note over User,SumsubSDK: Step 1: Initiate KYC

    User->>Client: Navigate to KYC verification
    Client->>API: POST /v1/user/kyc/initiate
    API->>API: Verify auth (authMiddleware)
    API->>DB: Check users.kyc_status
    alt Already approved
        DB-->>API: kyc_status = 'approved'
        API-->>Client: { status: "approved" }
    else Needs verification
        API->>Sumsub: POST /resources/applicants<br/>{ externalUserId, email, levelName }
        Sumsub-->>API: { id: applicantId }
        API->>Sumsub: POST /resources/accessTokens<br/>{ userId, ttlInSecs: 3600 }
```

```
        Sumsub-->>API: { token: accessToken }
        API->>DB: UPDATE users SET kyc_status = 'pending'
        API->>DB: INSERT INTO screening_results
        API-->>Client: { status: "pending", redirectUrl, externalId }
    end

    Note over User,SumsubSDK: Step 2: Document Verification (in Sumsub SDK)

    Client->>SumsubSDK: Initialize SDK with accessToken
    User->>SumsubSDK: Upload ID document (passport/ID card)
    User->>SumsubSDK: Take selfie (liveness check)
    SumsubSDK->>Sumsub: Submit documents for review
    Sumsub->>Sumsub: AI + human review

    Note over User,SumsubSDK: Step 3: Webhook Callback

    Sumsub->>API: POST /v1/webhooks/sumsub<br/>{ type, applicantId, reviewResult }
    API->>API: Verify HMAC signature
    API->>API: Check idempotency (screening_results)
    API->>DB: UPDATE users SET kyc_status = 'approved'/'rejected'
    API->>DB: INSERT INTO screening_results
    API->>DB: INSERT INTO audit_log
    API-->>Sumsub: 200 OK
```

Applicant State Machine

```
stateDiagram-v2
    [*] --> init: POST /resources/applicants
    init --> pending: Documents submitted
    pending --> queued: Automated checks passed, queued for review
    queued --> completed: Review finished
    pending --> completed: Fast-track (low risk)
    completed --> [*]

    state completed {
        [*] --> GREEN: All checks passed
        [*] --> RED: Verification failed
```

```
    [*] --> RETRY: Resubmission requested
}

note right of init
    User created in Sumsub
    SDK widget opened
    No documents yet
end note

note right of pending
    Documents uploaded
    AI processing
end note

note right of queued
    AI checks passed
    Awaiting human review
    (if required by level)
end note

note left of GREEN
    kyc_status = 'approved'
    Full access granted
end note

note left of RED
    kyc_status = 'rejected'
    Financial operations blocked
    rejectLabels explains why
end note

note left of RETRY
    kyc_status = 'pending'
    User prompted to resubmit
end note
```

Status Mapping

Sumsub Status	Sumsub Review Answer	Drop `kyc_status`	User Impact

`init`	N/A	`pending`	Cannot perform financial operations
`pending`	N/A	`pending`	Cannot perform financial operations
`queued`	N/A	`pending`	Cannot perform financial operations
`completed`	`GREEN`	`approved`	Full access to remittance and QR payments
`completed`	`RED`	`rejected`	Blocked from financial operations, can appeal
`completed`	`RETRY`	`pending`	Prompted to resubmit documents
`onHold`	N/A	`pending`	Under manual review

Webhook Architecture

Webhook Endpoint

URL: `POST /v1/webhooks/sumsub` **Authentication:** HMAC-SHA256 signature verification

Webhook Events

Event Type	Trigger	Drop Action
`applicantCreated`	Applicant record created in Sumsub	Log to `audit_log`, no status change
`applicantPending`	Documents submitted, verification in progress	Update `kyc_status` to `pending` if not already
`applicantReviewed`	Verification completed (approved or rejected)	Update `kyc_status` based on `reviewResult.reviewAnswer`
`applicantOnHold`	Manual review required	Log to `audit_log`, status remains `pending`
`applicantActionPending`	Additional action required from user	Notify user via notifications table
`applicantReset`	Application reset for resubmission	Reset `kyc_status` to `pending`

Signature Verification

```
POST /v1/webhooks/sumsub
Headers:
  X-Payload-Digest: HMAC-SHA256(request_body, SUMSUB_SECRET_KEY)
  Content-Type: application/json
```

Verification logic:

1. Read raw request body
2. Compute `HMAC-SHA256(body, SUMSUB_SECRET_KEY)`
3. Compare with `X-Payload-Digest` header (timing-safe comparison)
4. Reject with 401 if signature mismatch

Retry Policy

Sumsub retries webhook delivery on non-2xx responses:

Attempt	Delay	Total Wait
1	Immediate	0s
2	30s	30s
3	2m	2m 30s
4	10m	12m 30s
5	30m	42m 30s
6	1h	1h 42m 30s
7	2h	3h 42m 30s
8 (final)	4h	7h 42m 30s

After 8 failed attempts, the webhook is marked as failed in Sumsub dashboard.

Idempotency Handling

Webhooks may be delivered multiple times. Drop handles this via the `screening_results` table:

1. On webhook receipt, check if `screening_results` already has an entry with matching `user_id` + `screening_type` + same `result`
2. If duplicate: return 200 OK immediately (acknowledge but don't process)
3. If new or changed result: process the update, insert new `screening_results` record

Database Records Created Per Webhook

Table	Record	Purpose
`users`	UPDATE `kyc_status`	Primary KYC status
`screening_results`	INSERT	Detailed verification result with timestamp
`audit_log`	INSERT	Immutable audit trail
`notifications`	INSERT (if status changed)	User notification

Document Check Types

Verification Level: `basic-kyc-level`

Check	Type	Description	Required
Document authenticity	`IDENTITY`	Verify ID document is genuine (not altered/forged)	Yes
Liveness check	`SELFIE`	Verify real person (not photo/video)	Yes
Face match	`SELFIE`	Match selfie to document photo	Yes
Data extraction	`IDENTITY`	OCR extraction of name, DOB, document number	Yes
Sanctions screening	`SCREENING`	Check against OFAC, UN, EU sanctions lists	Yes
PEP screening	`SCREENING`	Politically Exposed Persons database check	Yes
Adverse media	`SCREENING`	News and media screening for negative coverage	Optional

Accepted Document Types

Document Type	Sumsub Code	Countries
Passport	`PASSPORT`	All
National ID card	`ID_CARD`	EEA countries
Driver's license	`DRIVERS`	Norway, Sweden, Denmark, Finland
Residence permit	`RESIDENCE_PERMIT`	Norway (non-citizen residents)

Risk Level Definitions

Risk Level Matrix

Risk Level	Score Range	Criteria	Drop Action
Low	0-30	All checks passed, no PEP/sanctions match, standard country	`kyc_status = 'approved'`, standard transaction limits
Medium	31-60	Minor issues (document quality, partial name match), non-sanctioned PEP	`kyc_status = 'approved'`, enhanced monitoring via `aml_alerts`, lower initial limits
High	61-80	Potential PEP match, high-risk country, document concerns	`kyc_status = 'pending'`, manual review required, flag in `screening_results`
Critical	81-100	Sanctions match, confirmed fraud indicators, age verification failure	`kyc_status = 'rejected'`, immediate block, create `aml_alerts` record, file potential STR

Risk Assessment Factors

Factor	Weight	Source
Document authenticity score	30%	Sumsub AI analysis
Liveness/face match score	20%	Sumsub biometric analysis
Country risk (origin/destination)	20%	Drop internal risk scoring
PEP/sanctions screening	20%	Sumsub screening databases
Transaction pattern analysis	10%	Drop `aml_alerts` historical data

High-Risk Countries for Drop

Based on remittance corridor analysis and FATF grey/black lists:

Category	Countries	Additional Controls
Standard	Norway, Sweden, Denmark, Finland, EU/EEA	Standard KYC
Enhanced due diligence	Turkey, Pakistan	Enhanced monitoring, lower limits

Category	Countries	Additional Controls
Restricted	FATF grey list countries	Manual review required
Blocked	OFAC/UN sanctioned countries	Automatic rejection

SDK Integration

Web Integration (WebSDK)

```
// Initialize Sumsub WebSDK in drop-web
import snsWebSdk from '@sumsub/websdk';

function launchSumsubWidget(accessToken: string, applicantId: string) {
  const snsWebSdkInstance = snsWebSdk.init(accessToken, () => {
    // Token expiry handler — request new token from API
    return fetch('/v1/user/kyc/refresh-token')
      .then(res => res.json())
      .then(data => data.token);
  })
  .withConf({
    lang: 'nb',  // Norwegian
    theme: 'dark',
    uiConf: {
      customCssStr: ':root { --primary-color: #0B6E35; }',  // Drop brand green
    },
  })
  .withOptions({
    addViewportTag: false,
    adaptIframeHeight: true,
  })
  .on('onError', (error) => {
    captureError(error, { tags: { component: 'sumsub-websdk' } });
  })
  .on('onApplicantStatusChanged', (payload) => {
    if (payload.reviewResult?.reviewAnswer === 'GREEN') {
      // Redirect to dashboard
      window.location.href = '/dashboard';
```

```
    }
  })
  .build();

  snsWebSdkInstance.launch('#sumsub-container');
}
```

Mobile Integration (React Native SDK)

```
// Initialize Sumsub React Native SDK in drop-mobile
import SNSMobileSDK from '@sumsub/react-native-mobilesdk-plugin';

async function launchSumsubMobile(accessToken: string) {
  const snsMobileSDK = SNSMobileSDK.init(accessToken, async () => {
    // Token expiry handler
    const response = await fetch(`${API_URL}/v1/user/kyc/refresh-token`, {
      headers: { Authorization: `Bearer ${authToken}` },
    });
    const data = await response.json();
    return data.token;
  })
  .withHandlers({
    onStatusChanged: (event) => {
      if (event.newStatus === 'Approved') {
        navigation.navigate('Dashboard');
      }
    },
    onError: (error) => {
      Sentry.captureException(error);
    },
  })
  .withLocale('nb')
  .withTheme({
    primaryColor: '#0B6E35',
  })
  .build();

  const result = await snsMobileSDK.launch();
  return result;
```

```
}
```

Error Recovery

Failure Scenarios and Recovery

Failure	Detection	Recovery
Sumsub API timeout (applicant creation)	30s timeout in `kyc.ts:44`	Return `rejected` with error message, user can retry
Sumsub API timeout (status check)	30s timeout in `kyc.ts:148`	Return `rejected`, poll again on next request
Access token generation failure	Non-200 response from `/resources/accessTokens`	Return `pending` with applicant ID (applicant created but no widget)
Webhook delivery failure	Sumsub retry (8 attempts over 7h)	Automatic retry by Sumsub
Webhook signature mismatch	401 response to webhook	Alert ops team, check `SUMSUB_SECRET_KEY` rotation
Database write failure on webhook	Transaction rollback	Return 500 to trigger Sumsub retry
Duplicate webhook	Idempotency check via `screening_results`	Acknowledge with 200, skip processing
SDK widget crash	`onError` callback	Capture in Sentry, show user-friendly error with retry option

Manual Recovery Procedures

If a user is stuck in `pending` status after Sumsub has completed review:

1. Check `screening_results` table for latest result
2. Query Sumsub API: `GET /resources/applicants/{externalUserId}/status`
3. If Sumsub shows `GREEN` but DB shows `pending`: manually update via admin endpoint
4. If Sumsub shows `RED`: communicate rejection reason to user

Environment Variables

Variable	Required	Description
`SUMSUB_API_URL`	Production	Sumsub API base URL (e.g., `https://api.sumsub.com`)

Variable	Required	Description
`SUMSUB_APP_TOKEN`	Production	Application token from Sumsub dashboard
`SUMSUB_SECRET_KEY`	Production	Secret key for HMAC signature verification
`SUMSUB_WEBHOOK_SECRET`	Production	Separate secret for webhook payload verification
`NEXT_PUBLIC_SERVICE_MODE`	All	When `mock`: uses auto-approve mock, no Sumsub calls

Demo Mode Behavior

Source: `lib/services/kyc.ts:26-28`

When `NEXT_PUBLIC_SERVICE_MODE=mock` (demo mode):

- `initiateKyc()` returns `{ status: "approved" }` immediately
- `checkKycStatus()` returns `{ status: "approved" }` immediately
- No Sumsub API calls are made
- Users created via BankID get `kyc_status = 'approved'` automatically (`bankid.ts:233`)

The mock Sumsub service (`mock-sumsub.ts`) provides a full simulation for UI development with applicant creation, document submission, and asynchronous verification (3-second delay, 90% approval rate).

Cross-References

- **KYC service:** `src/drop-app/src/lib/services/kyc.ts` — Production KYC initiation and status check
- **Mock Sumsub:** `src/drop-app/src/lib/services/mock-sumsub.ts` — Demo mode mock implementation
- **BankID integration:** AUTHENTICATION.md — BankID auto-approves KYC
- **Database schema:** DATABASE-SCHEMA.md — `screening_results`, `aml_alerts`, `users.kyc_status`
- **Compliance:** COMPLIANCE.md — AML/KYC regulatory requirements
- **Security:** SECURITY-ARCHITECTURE.md — Role-based access, KYC status enforcement

---

Revision #4
Created 2026-02-21 05:59:05 UTC by John
Updated 2026-05-23 10:57:09 UTC by John