Face Sign - Digital Signature Platform

1. Executive Summary

Face Sign is a secure digital signature platform that enables identity-verified document signing through integration with UAE KYC services. The platform provides session-based signing workflows, document hash verification, signature template generation, and real-time event streaming. This document provides a detailed specification of all V1 features, their sub-features, and inter-feature dependencies to guide implementation of sequencing.

Core Capabilities:

- Session-based signing
- Document hash computation and validation (hash authority)
- UAE KYC identity verification integration
- Real-time or Preset signature positioning and template generation
- Polling for integrators
- Web SDK support

Key Integrations:

- UAE KYC for identity verification
- MinIO/S3 for document and template storage (temporary)
- Kafka for event streaming
- ClickHouse for analytics and audit logging

2. Detailed Feature Specification

1. PLATFORM ADMINISTRATION

ID	Feature	Description	Sub-Features	Deps
F1.1	Organization Management	Onboard and manage integrator organizations. Each organization receives API credentials and isolated data space.	• API key generation • Organization/Integrator profile (name, legal entity, contact) • Active/suspended/archived status management • Sandbox and production Mode • Kafka topic assignment per organization (deferred for v1)	None

F1.2	Global Configuration	Centralized settings governing platform behavior. Organizations inherit defaults but may override within permitted ranges.	• Session TTL configuration (5-120 minutes) • Document size limits (max 100MB) • KYC retry limits (1-5 attempts) • Hash algorithm selection (SHA-256 standard) • Template dimension constraints • Rate limit thresholds per organization • Supported document formats (PDF only in V1) • Position coordinate system defaults (normalized 0-1)	F1.1
F1.3	API Key Management	Generate, rotate, and revoke API keys with granular permissions and usage tracking.	• API key generation • Rate limiting per key • Revoke api key to make it inactive • Regular Rotation of API keys • IP whitelisting (optional)	F1.1
F1.4	Platform Analytics	Historical analytics, dashboards, and metrics for administrators and integrators.	• KPI tiles • Organization-wise breakdown (sessions, completion rate), • Funnel analysis (created → complete conversion), • KYC success/failure rates, • Average session duration, • Document hash verification metrics	F1.1, F1.2
F1.5	Audit & Compliance	Immutable logging of all platform actions for regulatory compliance and security investigations.	• Session lifecycle event logging • API request/response logging • Hash computation audit trail • KYC verification event logging • Admin action logging (config changes, key rotation), • Data access logging (who accessed what, when) • Retention policy enforcement (7 years minimum) • Tamper-proof log storage (append-only ClickHouse)	F1.1, F1.3

2. SESSION MANAGEMENT

ID	Feature	Description	Sub-Features	Deps

F2.1	Session Initialization	Create new signing sessions with document upload and configuration. This is the entry point for all signing workflows.	• Document source validation (MinIO path or Base64), • Preset signature position configuration, • Signer identifier capture (Emirates ID, person number, Passport, GCC details, uaekyc id) • Session TTL calculation and storage Unique session ID generation (UUID), • Session token (JWT) generation with RS256, • Metadata storage (contract ID, signer order, custom fields), • Document download from MinIO (if path provided), • Initial state: **CREATED**	F1.1, F1.2
F2.2	Document Hash Computation	Compute SHA-256 hash of the signing document. Face Sign is the hash authority - integrators cannot override.	• Download document from MinIO or decode Base64, • Compute SHA-256 hash of complete PDF bytes, • Store document hash in sessions table, • Hash validation on subsequent document access, • Tamper detection via hash mismatch, • Hash included in all responses and events	F2.1
F2.3	Hash Token Generation	Issue signed JWT containing session_id and document_hash for downstream validation.	• Sign with Face Sign's RS256 private key, • Store hash token in sessions table, • Return hash token in /init response, • Token expiry matches session TTL, • Token refresh not supported (create new session)	F2.2, F15.1
F2.4	Session State Machine	Core engine managing session lifecycle. Enforces state transition rules and publishes events.	• Transition guards • Timeout transitions • Manual cancellation with reason capture, • Event publishing on each state change, • State history tracking with timestamps, • Idempotency on state transitions	F2.1, F1.2

F2.5	Session Status API	RESTful endpoint to query current session state and retrieve artifacts.	• Return current status and document_hash, Return artifacts when status = **COMPLETE**, • Return signer metadata • Return error details if status = FAILED, • Support polling by integrators, • Cache response for 5 seconds to reduce DB load, • Include metadata passed during initialization	F2.4
F2.6	Session Expiry	Automatic expiration of inactive sessions based on configured TTL.	• Background job running every 60 seconds • Expire sessions where: now > expires_at AND status not in terminal state, • Update state to **EXPIRED**, • Publish session.expired event, • Cleanup temporary MinIO files (7 days retention), • Integrator gets notified (webhook/Kafka/Status API), • Grace period before hard delete (30 days)	F2.4, F12.1
F2.7	Session Cancellation	Allow manual cancellation of sessions by integrators or users.	• Require cancellation reason (user_abandoned, integrator_cancelled, error_recovery), • Update state to **CANCELLED**, • Publish session.cancelled event, Prevent cancellation if status = COMPLETE, • Cleanup MinIO artifacts, • Audit log entry	F2.4, F1.5

3. DOCUMENT PROCESSING

ID	Feature	Description	Sub-Features	Deps
F3.1	Document Download	Retrieve documents from MinIO or decode Base64 for processing.	• MinIO client initialization with org-specific credentials, • Document download with retry logic (3 attempts), • Stream processing for large files (>10MB), • Base64 decoding and validation, • PDF format validation (magic bytes check), • File size validation (max 100MB), • Temporary storage in /tmp during processing, • Cleanup after hash computation	F2.1, F11.1

F3.2	PDF Parsing	Extract PDF metadata required for rendering and positioning.	• Page count extraction, • Page dimensions extraction (width, height in points), • PDF version detection, • Encryption status check (reject encrypted PDFs), • Store page_count and page_dimensions in sessions table, • Error handling for corrupted PDFs	F3.1
F3.3	PDF Rendering	Convert PDF pages to PNG images for SDK display.	• Render PDF pages at configurable DPI (default 150), • Output format: PNG with transparency, • Page image hash computation and storage, • Lazy rendering (render on-demand when SDK requests), • Cache rendered pages (TTL: session expiry + 7 days), • Optimize image size (compression level 6)	F3.2, F11.1
F3.4	Document Integrity Validation	Continuously verify document hash matches stored value to detect tampering.	• On SDK initialization: re-download document, compute hash, compare with stored, • On SDK page request: validate stored hash, • On session completion: final hash validation, • Reject session if hash mismatch detected, • Publish tamper_detected event, • Lock session state to FAILED, • Alert administrators	F2.2, F12.1

4. UAE KYC INTEGRATION

ID	Feature	Description	Sub-Features	Deps
F4.1	KYC Session Creation	Initialize KYC verification journey with UAE KYC service for signer identity verification.	• Receive KYC journey ID and token, • Store kyc_journey_id in sessions table, • Update session state to KYC_PENDING, • Return KYC token to SDK for iframe embedding, • Set KYC timeout (default 10 minutes)	F2.4, F10.1

F4.2	KYC Verification Callback	Handle KYC success/failure callbacks from UAE KYC service.	• Update session state: KYC_SUCCESS or KYC_FAILED, • Store signer metadata • Trigger template generation on success, • Retry logic for transient KYC failures (configurable retries)	F4.1, F2.4, F12.2
F4.3	KYC Retry Management	Handle KYC failures with configurable retry attempts.	• Retry counter in sessions table, • Max retries from global config (default 3), • Reset KYC state to **KYC_PENDING**, • Fail session if max retries exceeded, • Track retry reasons and timestamps,	F4.2, F1.2

5. SDK EVENT-COMMAND PROTOCOL

ID	Feature	Description	Sub-Features	Deps
F5.1	SDK Handshake	Establish secure connection between SDK and backend using session token.	• Validate session token • Extract session id from token, • Verify session exists and status = CREATED, • Update session state to HANDSHAKED, • SDK generates client-side keypair for encryption,	F2.1, F15.1
F5.2	Document Loading Command	Provide SDK with document metadata and page rendering URLs.	• Triggered by SDK event • Update session state to PREVIEWING, • Pre-render first 3 pages for faster loading	F5.1, F3.3

F5.3	Page Rendering API	Serve rendered PDF pages to SDK as PNG images.	• Validate session token in Authorization header, • If not rendered: trigger rendering, cache result,	F5.2, F3.3
F5.4	KYC Trigger Command	Instruct SDK to load KYC verification iframe.	• Call KYC session creation API, • Update session state to KYC_PENDING, • SDK embeds KYC iframe, • KYC result handled via webhook internally (F4.2)	F5.2, F4.1
F5.5	Template Loading Command	Provide SDK with generated signature template for positioning.	• Generate signature template (F6.1), • Upload template to MinIO, • Compute template_hash, • Update session state to POSITIONING	F4.2, F6.1
F5.6	Position Confirmation	Capture final signature positions selected by signer.	• Validate positions: within page bounds, non-overlapping, meet minimum size, • Store positions in sessions.signature_data JSONB, • Update session state to COMPLETE, • Publish session.completed event, • Return artifacts in next /status call	F5.5, F2.4, F12.2
F5.7	Error Handling	Handle SDK errors and failures gracefully.	• Log error to audit trail, • Increment error counter, • Update session state to FAILED if unrecoverable, • Publish session.failed event	F2.4, F1.5

6. SIGNATURE TEMPLATE GENERATION

ID	Feature	Description	Sub-Features	Deps

F6.1	Template Design	Generate personalized signature template image from verified identity.	• Use signer's full name from KYC metadata, • Render text with standard PDF font. • Add "Digitally signed by UAE KYC" watermark, • Include signing timestamp placeholder, • Add Emirates ID (masked) below name, • Template dimensions: 500x160 pixels (PNG), Transparent background, • Compute template_hash (SHA-256)	F4.4
F6.2	Template Storage	Store generated templates securely in MinIO.	• Retention: 30 days from session creation, • Enable versioning (if template regenerated), • Public read access for authenticated SDK, • Encryption at rest	F6.1, F11.1
F6.3	Template Retrieval	Serve template images to SDK for positioning preview.	• Validate session token, • Download from MinIO or regenerate • Return PNG with caching headers, • Track template downloads, • Include template_hash in response header	F6.2
F6.4	Template Hash Export	Provide template hash to integrators for validation during stitching.	• Include template_hash in session.completed event, • Include in hash token JWT for stitching validation,	F6.1, F2.3

7. ARTIFACT MANAGEMENT

ID	Feature	Description	Sub-Features	Deps
F7.1	Position Artifacts	Store finalized signature positions selected by signer.	• Normalized coordinates (0-1 scale), • PDF coordinates (points) calculated and included, • Position schema: {page, x, y, width, height, coordinates_pdf_units}, • Validate: within page bounds, minimum size (50x20 pixels), non-overlapping,	F5.6

F7.2	Template Artifacts	Package template image with metadata for integrator consumption.	• Dimensions in pixels for scaling calculations	F6.2, F6.4
F7.4	Artifact Cleanup	Remove artifacts after the retention period expires.	• Background job: daily cleanup task, Delete MinIO objects older than 30 days or have bucket retention of 30 days, • Preserve hash values in database (permanent), • Preserve audit logs • Update artifacts_ready = false after cleanup, • Log cleanup operations	F11.1, F1.5

13. RESILIENCE & ERROR HANDLING

ID	Feature	Description	Sub-Features	Deps
F13.1	Async Processing	Event-driven architecture for fault tolerance and scalability.	• Kafka-based event bus for state transitions, • Consumer groups for parallel processing, • Dead Letter Queue (DLQ) for failed messages, • DLQ dashboard for manual review, • Automatic retry with exponential backoff, • Idempotency keys to prevent duplicates, • Poison pill detection and quarantine	F8.1
F13.2	External Service Failures	Handle failures of UAE KYC, MinIO, and ClickHouse gracefully.	• Circuit breaker pattern per integration, • Graceful degradation (queue for retry), • Integration health indicators on dashboard, • Manual retry buttons for stuck operations, • No data loss guarantee (persist before external call), • Alerting on repeated failures (PagerDuty/Slack), • Fallback strategies where applicable	F13.1, F10.1, F10.2, F10.3

F13.3	Timeout Management	Configure and enforce timeouts for all async operations.	• Configurable timeout per operation type, • Session TTL timeout (default 30 minutes), • KYC verification timeout (10 minutes), • Document download timeout (60 seconds), • Template generation timeout (30 seconds), • Auto-state transition on timeout (e.g., expire), • Timeout reason logged in audit trail	F13.1, F1.2, F1.5
F13.4	Rate Limiting	Protect platform from abuse and ensure fair usage.	• Rate limits per organization and API key, Limits • HTTP 429 on limit exceeded, • Retry-After header with backoff time, • Burst allowance (20% over limit for 5 seconds), • Admin bypass for trusted orgs	F1.3, F11.3

14. SECURITY

ID	Feature	Description	Sub-Features	Deps
F14.1	Authentication	Secure authentication for API access and SDK sessions.	• API key authentication, • Token expiry enforcement, • Rotate signing keys every 90 days, • Key versioning support (kid in JWT header), • No password-based auth (API keys only)	F1.3
F14.2	Data Protection	Protect sensitive data at rest, in transit, and in use.	• Encryption at rest: MinIO (AES-256), PostgreSQL (TDE), ClickHouse (AES-256), • Encryption in transit: TLS 1.3 for all connections,	F1.5

F14.3	Hash Authority Security	Prevent hash tampering and ensure Face Sign is single source of truth.	• Hash computation isolated from API layer, • Immutable hash storage (cannot be updated after creation), • Signed JWT (hash_token) with private key, • JWT verification by stitching service using public key, • Tamper detection via hash mismatch, • Hash validation on every session state transition	F2.2, F2.3, F15.1
F14.4	Audit Trail	Comprehensive, immutable logging for security and compliance.	• Log all API requests with: timestamp, org_id, session_id, endpoint, status_code, latency, • Log all state transitions with: old_state, new_state, trigger, user_id, correlation_id, • Log all hash computations with: session_id, algorithm, hash_value, timestamp, • Log all KYC events with: session_id, kyc_status, retry_count, error_code, • Immutable storage (append-only ClickHouse), • Export to SIEM systems	F1.5, F10.3
F14.5	DDoS Protection	Protect platform from distributed denial-of-service attacks.	• Rate limiting per IP address (1000 requests/minute), • Rate limiting per API key (see F13.4), • Request size limits (100MB max for documents), • Connection limits per IP (100 concurrent), • Auto-ban IPs with malicious patterns (SQL injection, XSS attempts)	F13.4

11. LOCALIZATION & ACCESSIBILITY

ID	Feature	Description	Sub-Features	Deps
F11.1	Multi-language & Accessibility	EN/AR support and WCAG 2.1 AA compliance.	• Languages: English (default), Arabic (RTL) • Language toggle in SDK • Bilingual templates and notifications • keyboard navigation • 4.5:1 contrast • focus indicators • screen reader support (ARIA), • alt text,	F1.2

			• skip links, • font size adjustments (125%, 150%)	

3. Out of Scope (V1)

The following features are explicitly excluded from V1 but may be considered for future releases:

Feature	Reason for Exclusion
Multi-signer orchestration	Integrator service handles orchestration
Document stitching / PDF overlay	Stitching service handles final PDF assembly
Long-term document archival	Integrator handles storage and archival
Contract workflow logic	Integrator manages contract lifecycle
Signature order enforcement	Integrator enforces sequential/parallel logic
Email/SMS signer notifications	Integrator sends signer invitations
Advanced analytics dashboard	Basic KPIs sufficient for MVP
Biometric verification (fingerprint, iris)	KYC handles face verification only in V1
Mobile SDK (iOS/Android native)	Web SDK only in V1
Offline signing capability	Requires secure enclave and complex sync logic
Document autofill from KYC data	Field mapping complexity deferred
Custom signature template design	Standard template only in V1
Multi-page signature positioning	Single page per position in V1
Kafka/Webhook Integrations with integrators	Infra and Security Restrictions
Verification Page Enhancements	Add QR code linking to verification portal Add cryptographic proof section

4. Critical Implementation Path

The following sequence represents the minimum viable implementation order based on dependencies:

#	Feature ID	Feature	Gates
1	F15.1	Authentication	All API endpoints
2	F1.1	Organization Management	All org-scoped features
3	F1.2, F1.3	Global Config & API Keys	Session creation, rate limiting
4	F10.2	MinIO Client	Document storage and retrieval
5	F11.1, F11.2	Storage Organization & DB Schema	Data persistence
6	F2.1, F2.2, F2.3	Session Init, Hash Computation, Hash Token	Core session workflow
7	F2.4	Session State Machine	All state transitions
8	F3.1, F3.2, F3.3	Document Processing (Download, Parse, Render)	SDK display
9	F10.1, F4.1, F4.2	UAE KYC Integration	Identity verification
10	F6.1, F6.2	Signature Template Generation	Signer personalization
11	F5.1 - F5.7	SDK Event-Command Protocol	SDK interaction flow
12	F7.1 - F7.3	Artifact Management	Integrator data consumption
13	F8.1, F8.2	Kafka Event Publishing	Async notifications
14	F2.5	Session Status API	Polling interface
15	F12.1, F12.2	Notification Engine	Event delivery
16	F1.4, F1.5	Analytics & Audit	Monitoring and compliance

5. Integration Points

5.1 Upstream Integrators

- **POST /v1/sessions/init**: Create signing sessions with document and preset positions
- **GET /v1/sessions/{id}/status**: Poll session status and retrieve artifacts
- **Kafka Events (optional)**: Consume session lifecycle events (created, completed, failed, expired)

5.2 Downstream Services

- **UAE KYC**: Identity verification via API calls and webhook callbacks
- **MinIO**: Document and template storage (S3-compatible)
- **ClickHouse**: Audit logging and analytics data warehouse
- **Kafka**: Event streaming to integrators

5.3 Stitching Service Integration

- **Hash Token (JWT)**: Stitching service verifies JWT signature using Face Sign's public key
- **Document Hash Validation**: Stitching downloads document, recomputes hash, validates against JWT
- **Template Hash Validation**: Stitching validates template image hash before overlay

---

6. Security

- **Compliance**: UAE Data Protection Law, ISO 27001, SOC 2 Type II
- **Penetration testing**: Annual third-party security audits
- **Vulnerability scanning**: Weekly automated scans
- **Incident response**: Security incident response plan with 1-hour SLA

---

7. Appendix

7.1 Session State Diagram

```
None
CREATED
  ↓ (SDK connects)
HANDSHAKED
  ↓ (SDK confirms document)
KYC_PENDING
  ↓ (KYC success)        ↓ (KYC fail, retries left)     ↓ (KYC fail,
no retries)
KYC_SUCCESS         →  KYC_PENDING (retry)          →  KYC_FAILED
(terminal)
  ↓ (template generated)
PREVIEWING
  ↓ (signer reviews)
POSITIONING
  ↓ (positions confirmed)
COMPLETE (terminal)
```

```
Parallel paths:
  Any state + timeout → EXPIRED (terminal)
  Any state + cancel request → CANCELLED (terminal)
  Any state + error → FAILED (terminal)
```

7.2 Hash Flow Diagram

None

```
1. Integrator → Face Sign: POST /init (document path, NO hash)
2. Face Sign: Download document from MinIO
3. Face Sign: Compute hash_A = SHA256(document_bytes)
4. Face Sign: Store hash_A in sessions.document_hash
5. Face Sign: Generate JWT_A = sign({session_id, hash_A},
private_key)
6. Face Sign → Integrator: {document_hash: hash_A, hash_token:
JWT_A}
7. Signer completes session
8. Face Sign: Validate current document hash == hash_A
9. Face Sign → Kafka: event {document_hash: hash_A}
10. Integrator → Stitching: {hash_token: JWT_A}
11. Stitching: Verify JWT signature with Face Sign's public key
12. Stitching: Extract hash_A from JWT
13. Stitching: Download document, compute actual_hash
14. Stitching: Validate actual_hash == hash_A
15. If match → proceed; if mismatch → reject (DOCUMENT_TAMPERED)
```

7.3 Glossary

Term	Definition
Hash Authority	Face Sign's role as the single source of truth for document hashes; integrators cannot override computed hashes
Hash Token	Signed JWT containing session_id and document_hash, issued by Face Sign for downstream validation
Session	A single signing instance for one signer on one document
Artifacts	Outputs from a completed session: signature positions and template image
Template	Personalized signature visual generated from KYC-verified identity
Preset Positions	Signature positions defined by integrator, locked or adjustable by signer

Normalized Coordinates	Position values from 0.0 to 1.0 representing fractions of page dimensions
State Machine	Logic engine managing valid session state transitions and guards
Event-Command Pattern	SDK sends events (user actions), backend responds with commands (UI instructions)
PII	Personally Identifiable Information (name, Emirates ID, etc.)
Tamper Detection	Hash mismatch indicating document was modified after hash computation