

TECHNICAL

API Integration Best Practices for Financial Compliance

A technical guide to integrating sanctions screening, PEP checks, and adverse media APIs into your platform — architecture patterns, error handling, performance optimisation, and security.

Published: February 2026 | 26 pages | PrimeGuardia Research

Table of Contents

1	
.	Why API-First Compliance Matters
2	
.	Authentication and Security
3	
.	Request Design and Query Optimisation
4	
.	Response Structure and Match Scoring
5	
.	Error Handling and Resilience Patterns
6	
.	Batch Processing and Asynchronous Screening
7	
.	Webhooks and Real-Time List Change Notifications
8	
.	Performance, Latency, and SLA Considerations
9	
.	Audit Logging and Compliance Trail
10	
.	Testing Strategies and Sandbox Environment
11	
.	Production Deployment Checklist

1. Why API-First Compliance Matters

Modern compliance operations demand real-time decisions. A customer completes an onboarding form in under three minutes — your sanctions and PEP screening must complete in under one second to avoid impacting conversion rates. A payment is initiated — it must be screened before authorisation, not after settlement. These operational realities make API integration the only viable architecture for compliance at scale.

API-first compliance also enables a consistent, auditable, and version-controlled approach to screening. Rather than relying on manual processes, file-based batch jobs, or embedded lists that rapidly become stale, API integration ensures that every screening decision uses the most current list data, applies consistent logic, and generates a standardised audit record.

<200ms	99.99%	10,000	15 min
Target p99 API response time	Recommended API uptime SLA	Batch records per API call	Max acceptable list update lag

2. Authentication and Security

2.1 API Key Management

All API authentication should use dedicated API keys, never embedded in client-side code or committed to source control repositories. Keys

should be stored in environment variables or a secrets management system (AWS Secrets Manager, HashiCorp Vault, Azure Key Vault). Rotate keys quarterly at minimum and immediately upon any suspected compromise.

2.2 Transport Security

All API traffic must use TLS 1.2 or higher. Validate server certificates in all environments, including development. Never disable certificate validation — even temporarily. Use HTTP Strict Transport Security (HSTS) headers. Consider mutual TLS (mTLS) for the highest-security integration patterns.

2.3 Data Minimisation

Send only the data fields required for an accurate match. For a person check, this typically means: full name (or component parts: first, middle, last), date of birth, nationality, and ID numbers if available. Avoid sending more personal data than necessary — this reduces data protection risk and keeps requests lean for lower latency.

■ **Security Tip: Never log raw API request bodies that contain PII in application logs. Log only the screening request ID and response metadata. Full request/response logging should only occur in encrypted, access-controlled audit systems.**

3. Request Design and Query Optimisation

3.1 Providing Structured Name Data

Where possible, send name components separately (`given_name`, `family_name`, `middle_name`) rather than a single `full_name` string. This allows the screening engine to apply smarter matching logic, particularly for cultures with different name ordering conventions. When only a single string is available, indicate whether it is in Western (given-family) or Eastern (family-given) order.

3.2 Enriching Requests with Supporting Data

Including additional data fields in screening requests significantly improves match precision and reduces false positives. Each additional confirmed attribute that does not match a list entry's data allows the score for that candidate to be reduced. Most valuable enrichment fields (in order of impact): date of birth, nationality, country of residence, company registration number (for entity checks), and passport/ID number.

3.3 Selecting the Right Fuzzy Threshold

Fuzzy matching thresholds control the sensitivity/specificity tradeoff. A threshold of 100% (exact match only) will miss virtually all real sanctions hits — sanctioned individuals routinely use aliases, transliteration variants, and name reversals. Thresholds below 70% generate excessive false positives. Most institutions operate between 75% and 90%, with the optimal setting varying by customer risk profile and name origin.

4. Response Structure and Match Scoring

A well-designed screening API response provides everything needed to make and document a compliance decision without further manual research. The response should include: a unique `request_id` for audit correlation; a match array containing each candidate match; per-match

fields including `match_score` (0–100), `matched_name`, `list_source`, `designation_date`, and available profile data; and a clear `is_match` boolean at the appropriate threshold.

5. Error Handling and Resilience

Compliance APIs are business-critical infrastructure. Your integration must handle failures gracefully without compromising compliance obligations. Key patterns: implement exponential backoff with jitter for 429 (rate limited) and 5xx responses; use circuit breakers to prevent cascade failures; maintain a local fallback queue for deferred processing if the API is temporarily unavailable; and never silently fail — a failed screening check must trigger a manual review workflow.

Define and document your organisation's policy for API unavailability: should onboarding be halted? Should transactions be queued pending screening? What is the maximum acceptable delay before a human decision is required? These are compliance and business continuity decisions that must be made before production deployment.

9. Audit Logging and Compliance Trail

Regulators may request evidence of screening decisions made months or years in the past. Your audit trail must be immutable, comprehensive, and queryable. For each screening event, store: timestamp (UTC, millisecond precision); screening request payload (hash, not PII in cleartext); API response summary (match count, highest score, decision); analyst review decision if applicable; and the list versions active at the time of the check.

11. Production Deployment Checklist

- API keys stored in secrets manager, never in code or config files
- TLS 1.2+ enforced and certificate validation enabled in all environments
- Fuzzy threshold calibrated based on risk appetite and tested against historical data
- Retry logic with exponential backoff and circuit breaker implemented
- API unavailability fallback procedure documented and tested
- Audit log capturing all screening events with immutable storage
- Batch screening scheduled for full customer base re-screening
- Webhook endpoint configured for real-time list change notifications
- Performance tested at 2x expected production load
- Sandbox testing completed with all relevant test cases
- Runbook documented for on-call engineer response to API incidents
- Data retention policy applied to screening logs

PrimeGuardia's API is available in sandbox and production environments with comprehensive documentation, code samples in Python, Node.js, Go, Java, and PHP, and dedicated technical support. Start at primeguardia.com/docs.