Axra Compliance: PII Handling & GDPR Data Deletion

Author: Lekan Adejumo (Founder & CEO, Axra Holdings LLC).

1. PII Data Classification

Category A: High Sensitivity (Financial)

Model	Field(s)	Description
Wallet	accountNumber, routingNumber, iban	Virtual bank account identifiers
Beneficiary	accountNumber, routingNumber, iban, swiftBic, walletAddress	Recipient payment details
Transfer	amount, fee, totalAmount, destinationAmount	Transaction amounts
User	pinHash	Hashed transaction PIN

Controls: Never logged. Excluded from API responses via user.service.ts field stripping. Stored encrypted at rest (database-level encryption recommended).

Category B: High Sensitivity (Identity)

Model	Field(s)	Description
User	email, firstName, lastName, phoneNumber	Core identity
User	passwordHash, emailVerifyToken, passwordResetToken	Authentication credentials
KycRecord	status, level, rejectionReason	Identity verification status
Session	ipAddress, userAgent	Device fingerprints
AuditLog	ipAddress, userAgent	Activity tracking

Controls: Passwords and PINs stored as bcrypt hashes (cost factor 12). Tokens are single-use and time-limited. IP addresses logged for security auditing only.

Category C: Medium Sensitivity (Behavioral)

Model	Field(s)	Description
User	memory (JSON)	AI-saved facts about the user
User	preferences (JSON)	User preferences
Message	content	Chat message content (may contain

PII) | | Conversation | userId, channel | Conversation metadata | |
Notification | title, body | May reference transactions |

Controls: Message content retained for 90 days (inactive conversations). AI memory is user-controlled. Notifications do not contain full account numbers.

Category D: Low Sensitivity (Linking)

Model	Field(s)	Description
ChannelMapping	channelUserId, channelUsername	External platform IDs
Beneficiary	fullName, nickname, country	Recipient display info
BillPayment	Beneficiary reference	Recurring payment metadata

Data Flow Summary

```
User Input â   API (HTTPS/TLS) â   NestJS Backend â   PostgreSQL (encrypted at rest)
â
Redis (TTL-bound, in-memory)
â
Bridge/YellowCard APIs (external processors)
```

- All external API calls use HTTPS.
- Redis stores only ephemeral data (rate limits, session dedup, onboarding state) with explicit TTLs.
- No PII is written to application logs. Structured logs contain only userId (UUID), never names/emails/account numbers.

---

2. GDPR Data Deletion Process

Right to Erasure (Article 17)

When a user requests account deletion, all personal data must be removed except where retention is legally required (e.g., financial transaction records for anti-money laundering compliance).

Deletion Scope

Fully Deleted:

Model	Rationale
Session	No retention requirement
Notification	No retention requirement
ChannelMapping	No retention requirement
Message	No retention requirement
Conversation	No retention requirement
User.memory	No retention requirement
User.preferences	No retention requirement

Anonymized (retained for compliance):

Model	Rationale	Anonymization
Transfer	AML/financial regulation (5-7 years)	Replace userId with DELETEDUSER, preserve amounts and timestamps
LedgerEntry	Double-entry accounting integrity	Preserve entries, wallet reference anonymized
AuditLog	Security audit trail (5-7 years)	Replace userId with DELETEDUSER, preserve action and timestamp
KycRecord	Regulatory requirement	Anonymize, preserve verification dates
BillPayment	Financial record	Cancel active payments, anonymize completed ones

Cascade Deleted (dependent data):

Model	Depends On
TransferStatusChange	Preserved with anonymized Transfer
LedgerEntry	Preserved with anonymized Wallet

Deletion Order

Execute in this order to respect foreign key constraints:

```
1. Cancel active BillPayments
2. Delete Messages â   Delete Conversations
3. Delete Notifications
4. Delete Sessions
5. Delete ChannelMappings
6. Anonymize Transfers + TransferStatusChanges (set userId = ’DELETED_USER’)
7. Anonymize LedgerEntries
8. Delete Wallets (after ledger anonymization)
9. Delete Beneficiaries
10. Anonymize AuditLogs (set userId = ’DELETED_USER’)
11. Anonymize KycRecords
12. Delete User record
```

Implementation

Deletion should be triggered by an admin endpoint or internal tool, not directly by the user. The process should:

1. Verify the request is authenticated and authorized
2. Send a confirmation email to the user’s registered email
3. Apply a 30-day cooling-off period before execution
4. Execute the deletion order above within a database transaction
5. Log the deletion event in AuditLog (with userId = ’DELETED_USER’)
6. Revoke all API keys and external service linkages (Bridge customer, channel mappings)
7. Send final confirmation email

Data Processor Obligations

External processors that hold user data:

Processor	Data Held	Deletion Method
Bridge.xyz	Customer profile, bank accounts, transfers	Bridge API: delete customer endpoint
YellowCard	Payment records	Contact YellowCard support
Anthropic	None (stateless API, no data retention)	N/A
Telegram/WhatsApp	Message history	User deletes chat; bot cannot delete remotely

Retention Schedule

Data Type	Retention Period	Legal Basis
Financial transactions	7 years	AML/CFT regulations
Audit logs	7 years	Security compliance
KYC records	5 years after account closure	KYC regulations
Chat messages	90 days (inactive)	Legitimate interest
Sessions/tokens	Cleaned daily	No retention required