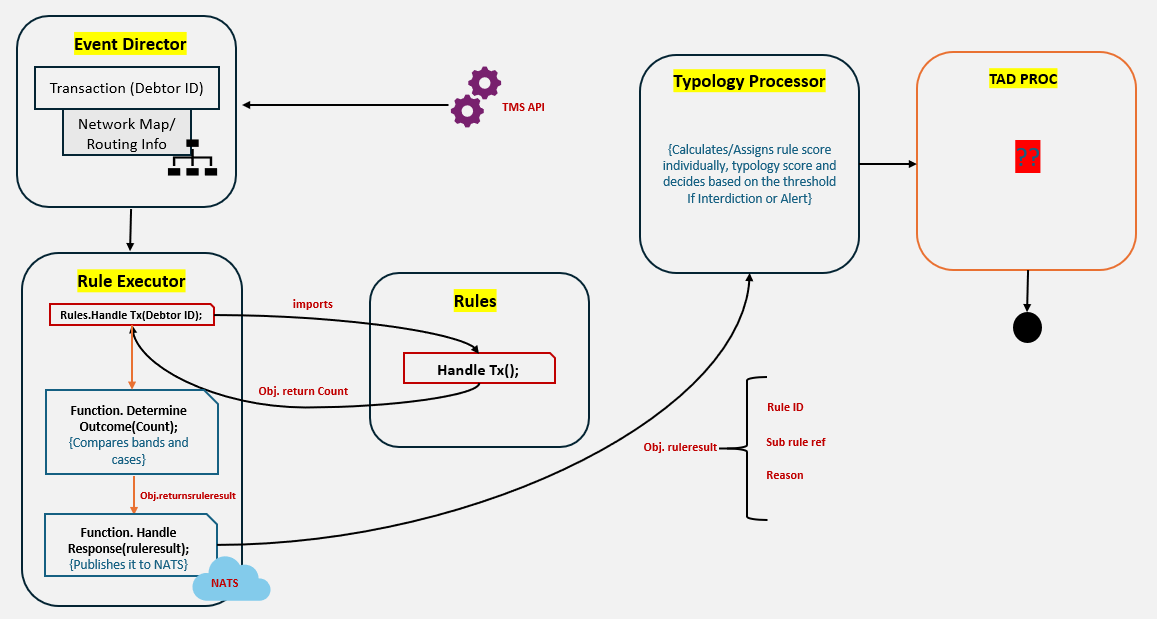
**BUSINESS REQUIREMENT SPECIFICATIONS DOCUMENT**

**MULTI TENANCY IN TAZAMA**

## Context



Flow of Events:

1. External messages (typically in JSON) are ingested into the TMS API for processing.
2. A Transaction Object consisting of filtered data elements is passed to the Event Director for routing. This will also contain a “tenant ID” to identify which tenant initiated the transaction. Tenant identification shall be handled through the **JWT token** used for authenticating API access. The tenant ID can be extracted from the token during request processing and should be added to the message payload as an attribute to enable downstream interpretation and routing.
3. The Event Director receives the Transaction Object, identifies applicable rule sets using its Network Map based on message type and tenant ID, and forwards the message with routing context to the Rule Executor.
4. The Rule Executor evaluates relevant rules on the Transaction Object by invoking HandleTransaction() for each rule, executing rule logic, and returning intermediate results; DetermineOutcome() function then interprets these results to produce a Rule Evaluation Object containing the rule ID, config version, sub-rule reference, and evaluation reason. HandleResponse() function in the Rule Executor collects Rule Evaluation Objects and forwards them to the Typology Processor for aggregation and scoring.
5. The Typology Processor aggregates individual rule evaluations into typology-level insights by computing rule scores, combining them into typology scores, and forwarding them to the TADProc for producing a final evaluation output for downstream actions like case creation or alerting.
6. The TADProc concludes the evaluation and combines all individual evaluation results from the Typology Processor and delivers the final aggregated alert outcome to the Case Management System and database.

## Scope of Requirements

| **#** | **Requirement** |
| --- | --- |
| 1. **API Access** | |
| A.1 | Every API call shall carry tenant ID forward into the internal processing layers like Event Director, Rule Processors, and Data Storage. |
| A.2 | Tenant-specific user administration (Role-Based Access Control) shall be enabled. Tenants can manage their own users, but the internal integrator retains ultimate control. |
| A.3 | Extend KeyCloak-based authentication for Tenant Admin APIs as well (for accessing certain parts of the back-office). |
| A.4 | All incoming messages must be tagged with the Tenant ID when the message is written to the database. |
| 1. **Keycloak** | |
| B.1 | Tenant Groups should be centrally defined and administered within Keycloak to ensure consistent group referencing across the broader ecosystem, allowing shared group identifiers to be leveraged across different components (e.g. Configuration Management, Case Management, Transaction Monitoring, and Reporting). |
| 1. **Event Director** | |
| C.1 | Event Director shall read the tenant ID from transaction object and route the transaction to tenant-specific rules (as configured in the network map). |
| C.2 | The network map must be extended to include the Tenant ID at the root of the network map object to route a transaction according to Tenant ID first, Transaction Type second, Typology third, and Rule last. |
| C.3 | The Event Director must be able to group a number of Tenant IDs together to represent a group of Tenants who have a data sharing agreement in place. |
| 1. **Rule Processors** | |
| D.1 | Rule Processors to cache multiple tenant-specific rule configurations simultaneously and ensure isolation of tenant-segregated data. |
| 1. **Typology Processor** | |
| E.1 | Similar to rule processors, typology processors shall cache multiple typology configurations for different tenants. In the future, NATS routing will be set dynamically based on the Tenant ID from the network map, replacing the current static environment variable setup. |
| 1. **Relay Service** | |
| F.1 | When Tazama accepts transactions from tenants directly, a tenant-specific relay service for each tenant would be deployed. |
| F.2 | Relay service for case/alert creation shall also be tenant specific, using designated NATS endpoints. |
| F.3 | The Typology Processor shall generate interdictions to the tenant-specific relay service. |
| F.4 | For notifying the specific tenant of an interdiction, notification shall be routed to the tenant through dedicated relay services per tenant. |
| F.5 | The TADProc also generates investigation alerts to a tenant-specific case management system. |
| 1. **Data Storage** | |
| G.1 | All data indices where ID is present shall include Tenant ID (transactions, evaluation reports, user roles). Tenants shall only be able to access the data tagged with their tenant ID. |
| G.2 | There shall be separate collections for each message for each tenant in ArangoDB (e.g., TransactionHistoryPacs002\_t1) |
| G.3 | For storing enrichment data, there shall be a separate relational database schema containing separate collections for storing data of each tenant. |
| G.4 | The evaluation results schema shall also have separate collections for each tenant. |
| G.5 | Any tenant-specific data that has been propagated to the data warehouse must be accessible by that tenant only. |
| 1. **Case Management** | |
| H.1 | Cases created shall be segregated per tenant. |
| H.2 | For data sharing between tenants during a case investigation, evaluation reports may be shared. |
| H.3 | Data available to an investigator during the investigation of a case must be segregated by tenant. |
| H.4 | Data shared between investigators from different tenants must be shared by extract and transmission and not by access to the source data directly. |
| 1. **Configuration Management** | |
| I.1 | Rule and typology configurations will differ from tenant to tenant (e.g., breaching thresholds, additional field checks etc.) |
| I.2 | Tenants shall be able to personalize their own configurations to achieve detection of fraud according to their standards. Any customizations made to the configurations must be checked first by the internal integrator and then published. |
| I.3 | The actions tenants can perform are:   1. User Management:  * Create, update, delete, or disable users within the tenant’s scope. * Assign roles and permissions to users (e.g., Analyst, Supervisor, Admin).  1. Rule and Configuration Management:  * Modify fraud detection rules, thresholds, or typologies specific to that tenant. * Enable tenants to calibrate detection logic based on their risk appetite.  1. Notification Preferences:  * Set up where and how alerts are sent (e.g., email, webhook, SMS). * Customize notification workflows for specific scenarios.  1. Data Access Controls:  * Define which roles can access specific datasets (e.g., transaction history, case logs). * Adjust data-sharing agreements (e.g., opting in/out of shared data pools).  1. Case Management Settings:  * Customize case assignment rules, investigation workflows, or escalation processes.  1. Audit and Logging Access:  * Retrieve audit logs of admin actions within the tenant.   Ensure accountability and support compliance audits. |