

Master Data Management Block (MDMB)

Business Requirements Document (BRD)

Version: 5.0
Date: Sep 1, 2026
Status: Finalized

Document Control

Version	Date	Author	Changes
5.0	Sep 1, 2026	Cross-Functional Team	Final Review

Table of Contents

1. Overview **3**
1.1 Purpose 3
1.2 Scope 4
1.1.1 In Scope 4
1.1.2 Out of Scope 5
1.3 Process Flow Diagram 5
1.4 Relationship to Other Building Blocks 6
1.5 Business Value 6
1.5.1 For Citizens 6
1.5.2 For Government Agencies (Service provider) 6
1.5.3 For the Agency Layer 7
2. Usage Scenarios **7**
2.1 Scenario A: A Field Already Held in SSOT 7
2.2 Scenario B: A Field Not Held in SSOT, Resolved via an MDMB Taxonomy 7
2.3 Example Scenario: Worked Walkthrough with Sample Data 8
2.4 Scenario C: Registering a New Golden Record, with Duplicate Detection 8
2.5 Scenario D: A Suspected Duplicate Is Found and Merged 9
2.6 Scenario E: Conflict Resolution Between Two Tenants 9
2.7 Scenario F: Search, Publication and Consumption 10
3. Functional Requirements **10**
3.1 Entity domains and field definition 10
3.2 Classification, hierarchy and code lists 11
3.3 Record lifecycle 11
3.4 Duplicate detection 12
3.5 Merge and unmerge 12
3.6 Conflict resolution 13
3.7 Multi tenancy, users and access 14
3.8 Search and retrieval 14
3.9 Publication and consumption 15
3.10 Change history and audit 15
4. Conceptual Data Model (Indicative) **15**
4.1 Golden Record 16
4.2 Field Definition 16
4.3 Hierarchy Relationship 16
4.4 Duplicate Review Task 17
4.5 Merge Event 17
4.6 Conflict Resolution Task 17

4.7 Tenant....18
4.8 User Account....18
4.9 Change History Entry....18
5. Integration with Other Building Blocks....19
6. Non-Functional Requirements....19
7. Roles and Governance....20
8. Business Rules....20
9. Glossary....21

1. Overview

1.1 Purpose

The Master Data Management Building Block (MDMB) holds a single, trusted, deduplicated golden record for the master data entities that more than one government agency needs. Agencies, services, and GA systems read that record from MDMB instead of maintaining their own copy of the same entity.

Today each source system keeps its own partial and inconsistent version of these entities, and the same entity is frequently duplicated across systems. There is no shared record that an agency can rely on when it needs to reference an organization, a location or a code value owned elsewhere.

MDMB is centralized. Any agency can register master data in it as and when required, subject to the stewardship controls. MDMB holds no personal data. Personal data remains in the system of the agency that collects it.

1.2 Scope

1.1.1 In Scope

- MDMB shall maintain golden records for the following entity domains (eg., Organization, Agency, Department, Location, Address), and any additional domains.
- The system shall allow a new field and its master data to be created as and when required. A field definition shall include at a minimum (e.g., its data type, length, and validation rules).
- The field and master data structure shall be dynamic, so that a new field can be introduced by configuration, without a code release.
- Data that requires classification, hierarchy, or taxonomy shall be mapped as a parent-child relationship. Examples are Dzongkhag, Gewog and Village, and Ministry, Department and Division.
- Master data held as a flat code list does not require classification, hierarchy, or taxonomy. Examples are unit (kg, metre, litre), marital status (married, divorced, single), and gender (male, female, other).
- The system shall check for duplicates when a new record is entered. A suspected duplicate shall be verified by a data steward.
- A confirmed duplicate shall be resolved by merging the records into a single golden record. The system shall record which records were merged and shall support reversal of a merge.
- Where two tenants submit different values for the same attribute of the same entity, the system shall resolve the conflict.
- Fields and master data shall be capable of being added and deleted. Deletion shall be a soft delete, with the record retained and marked inactive rather than removed.
- Master data submissions shall pass through a stewardship verification.

- The system shall support multiple tenants. Users shall be created per tenant, and access shall be role-based.
- A tenant user shall be able to view the master data belonging to that tenant, and shall be able to view published golden records shared across government.
- A tenant user shall be able to submit a new requirement for golden record data. Duplicates arising from such submissions shall be handled under points 7 and 8.
- The system shall provide search across master data.
- The audit log, like change history for every golden record, recording the user, the change, and the date and time, should be logged to the Journaling Block (JLB).
- Consuming agency systems, services, and GA systems shall obtain golden records from MDMB.
- The system shall allow a new tenant to be onboarded by the system administrator.

1.1.2 Out of Scope

- Storage of personal data of any kind. Personal data remains in the system of the agency that collects it.
- Replacement of the authoritative registers that originate and legally own master data (e.g., Civil Registration, the Business Registry and the Land / Address registers). Those registers remain the legal owners of the data.

1.3 Process Flow Diagram

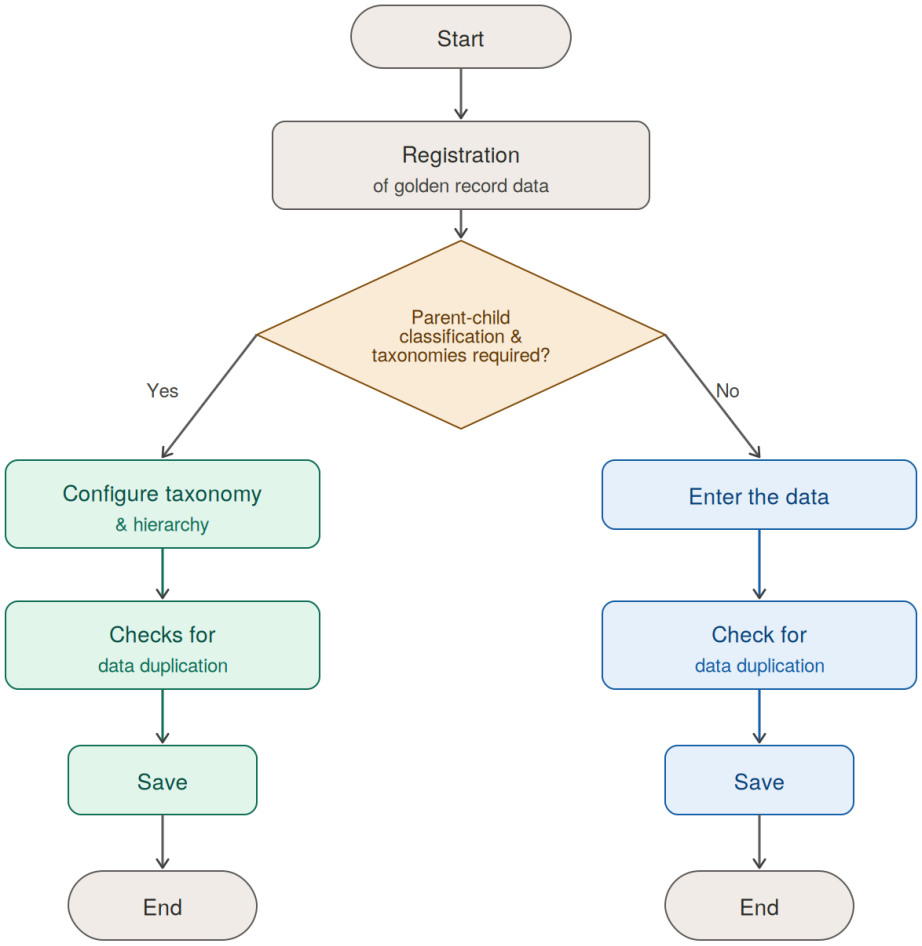

Figure 1: The Process flow of MDMB

1.4 Relationship to Other Building Blocks

MDMB sits at the end of a lookup chain and does not integrate with any of the following directly — every relationship below is mediated by NDE unless stated otherwise.

Building Block	Nature of Relationship	Description
NDE (National Data Exchange)	Transport	Carries every lookup request, lookup response, and Agency tenant-update request between MDMB and any consumer. MDMB has no other network-facing integration.
JLB (Journal / Audit Log Block)	Audit sink	Receives an immutable record of every lookup and Agency update MDMB processes.

Building Block	Nature of Relationship	Description
My Bhutan Portal	Fallback consumer (citizen-facing side)	Where the citizen avails the service. It queries MDMB, via NDE, only for attributes the owning Agency could not supply, following the Agency-first, MDMB-second, manual-entry-last chain.

1.5 Business Value

1.5.1 For Citizens

Value Dimension	Description
Tell Us Once	Information already captured by an Agency is reused automatically; the citizen is not asked to supply it again.
One Trusted Version of Their Data	Duplicate detection and steward-managed merges (Capability 3.4, 3.5) mean a citizen's information is never fragmented across multiple, conflicting golden records.
No Mismatched or Outdated Information	Referential integrity, lifecycle management, and audit trails mean a service always reflects the current, correct golden record — never a stale or orphaned one.
Fewer Manual Entry Errors	Fields backed by validated, governed master data reduce typing and the transcription errors that come with it.
Faster Service Completion	Less time is spent typing, searching for, or correcting duplicate or mismatched information.

1.5.2 For Government Agencies (Service provider)

Value Dimension	Description
One Place to Maintain Data	Each Agency maintains its own golden records once, in MDMB, instead of duplicating the same data across every consuming system.
Reduced Duplicate Records	Automatic duplicate checking at the point of submission, followed by steward-managed merge (Capability 3.4, 3.5), catches likely duplicates before they become two competing golden records.
Clear Ownership and Accountability	Multi-tenancy and role-based access (Capability 3.7) mean every contribution, correction, merge, and conflict resolution is traceable to the tenant and user responsible for it.

1.5.3 For the Agency Layer

Value Dimension	Description
Governed Change Control	Every merge, conflict resolution, and lifecycle change is logged, reviewable, and reversible, giving central oversight full visibility into how golden records evolve.
Single Version of Truth Across Government	A golden record published from MDMB — already deduplicated and conflict-resolved — is the one authoritative reference every consuming system can rely on, regardless of which tenant originated it.

2. Usage Scenarios

2.1 Scenario A: A Field Already Held in SSOT

1. A citizen enters their identifier to avail a service on a My Bhutan Portal form. The form requires several fields, including one that SSOT already holds a value for.
2. My Bhutan Portal, via NDE, asks the citizen's SSOT for that field's value.
3. SSOT holds this value for the citizen and returns it.
4. My Bhutan Portal pre-fills the field directly from SSOT's response. MDMB is not queried for this field at all, because SSOT already had a value.
5. If the citizen ever needs to correct this information, that correction is made in SSOT, not in MDMB.

2.2 Scenario B: A Field Not Held in SSOT, Resolved via an MDMB Taxonomy

1. The same form also requires a field that maps to a hierarchical taxonomy MDMB maintains.
2. My Bhutan Portal, via NDE, asks SSOT for this value first. SSOT does not hold it for this citizen.
3. Because this field maps to a taxonomy MDMB maintains, My Bhutan Portal asks MDMB, via NDE, for the current list of valid top-level options.
4. MDMB returns the full top-level list from the owning Agency's tenant partition.
5. The citizen selects a top-level option. My Bhutan Portal then asks MDMB again for the child-level options filtered by that selection, and the citizen selects from that filtered list.
6. MDMB has not received or stored anything about this citizen; it only answered "what are the valid options" at each level. Both the SSOT lookup and the MDMB lookups are logged for audit.

2.3 Example Scenario: Worked Walkthrough with Sample Data

This section walks through a single, concrete example end to end, using sample data, combining both routes in one form.

Context

A citizen enters their identifier on a My Bhutan Portal form that requires Field A (already held in SSOT) and Field B (not held in SSOT, and resolved via a hierarchical MDMB taxonomy).

Step-by-Step Resolution

Field	Step 1: SSOT	Step 2: MDMB	Result
Field A	Held by SSOT	— (not needed)	Pre-filled directly from SSOT
Field B (parent level)	Not held by SSOT	Full top-level list returned	Citizen selects a top-level option
Field B (child level)	— (not applicable)	Filtered child-level list returned, based on the selected parent	Citizen selects from the filtered list

What Happens Next

- Field A appears on the form automatically, exactly as SSOT holds it the citizen does not type it.
- Field B is presented as a two-step dropdown, since SSOT has no value: the citizen selects a top-level option, then a child-level option filtered by that choice.
- All lookups are logged for audit of the SSOT lookup and each MDMB lookup is recorded separately.

This example shows both steps of the model working together on a single form: SSOT answers what it already holds, and MDMB supplies the taxonomy only when SSOT has nothing to offer.

2.4 Scenario C: Registering a New Golden Record, with Duplicate Detection

1. An Agency user submits a request to register a new golden record within an entity domain (e.g. Organization, Agency, Department, Location, or Address) the tenant is permitted to contribute to.
2. The system determines whether this domain requires parent-child classification and taxonomies.
3. If classification is required, the user configures the taxonomy and hierarchy placement for the new record (Capability 3.2). If not, the user enters the record's data directly, with no hierarchy assignment needed.
4. Before the record is saved, the system automatically runs a duplicate check against existing golden records in the same domain, using the configured matching rules (Capability 3.4).

5. No matching record is found. The system saves the new record as a golden record and assigns it a permanent, system-generated identifier that is never reused.
6. The creation is written to the change history, recording the user, the tenant, and the timestamp (Capability 3.10).

2.5 Scenario D: A Suspected Duplicate Is Found and Merged

1. A different Agency user submits a new record that closely matches an existing golden record in the same domain (for example, the same organisation with a minor spelling difference).
2. The duplicate check (Capability 3.4) flags this as a suspected match below the exact-match threshold, rather than an automatic rejection.
3. The system places the submission in a pending state and raises a duplicate review task for a data steward. The record is not published while the task remains open.
4. The data steward opens the review screen, which presents the submitted record and the candidate match side by side, with the matching score and the differing attributes indicated.
5. The steward confirms the two records are duplicates, selecting the surviving record and the surviving value for each attribute where the records differ.
6. The system merges the records into the surviving golden record, maps the retired identifier to the survivor so that any consumer presenting it is redirected, and notifies consuming systems of the merge (Capability 3.5).
7. The full merge event, the source records, the surviving record, the attribute-level selections, the steward, and the timestamp, is recorded and remains reversible by an authorised role.

2.6 Scenario E: Conflict Resolution Between Two Tenants

1. Two different tenants each submit a different value for the same attribute of the same golden record at around the same time (for example, two Agencies submitting different addresses for the same organisation).
2. The system checks whether a source-of-truth precedence is configured for this entity domain and attribute (Capability 3.6). No precedence applies in this case.
3. Rather than silently overwriting one value with the other, the system raises a conflict resolution task, presenting both values, the submitting tenants, and their submission timestamps to the domain steward.
4. The domain steward reviews the conflict and manually resolves it, selecting which value should become the golden record’s published value.
5. The resolved value becomes the published golden record value; the non-surviving value is retained against its submitting tenant for reference, not discarded.
6. Both submitting tenants are notified of the resolution outcome, and the resolution is logged.

2.7 Scenario F: Search, Publication and Consumption

1. A user with read access searches across the entity domains they are permitted to see, filtering by domain and status (Capability 3.8).
2. The user opens a golden record from the results and reviews its attributes, its hierarchy path, its current status, and its change history.
3. Separately, an external consuming system, authenticated and authorised for this purpose, calls the publication API to retrieve golden records changed since its last successful call (Capability 3.9).
4. The consuming system is also subscribed to change notifications, so it is informed automatically the next time this record is amended, deactivated, merged, or has a merge reversed, without needing to poll.
5. Every API call is logged with the consumer, the endpoint, the parameters, and the timestamp, and is subject to rate limiting.

3. Functional Requirements

3.1 Entity domains and field definition

ID	Requirements
FR-1.1	The system shall maintain golden records for the entity domains Organization, Agency, Department, Location, and Address, and other required domains.
FR-1.2	The system shall allow a new entity domain to be defined by an authorised administrator without a code release.
FR-1.3	The system shall allow a new field to be added to an entity domain by configuration. A field definition shall carry at minimum a field name, data type, length, mandatory or optional flag, default value where applicable, and validation rules.
FR-1.4	Supported data types shall include text, number, date, boolean, and reference to another master data record or code list.
FR-1.5	The system shall apply field validation rules at the point of data entry and at the point of API submission, and shall reject values that fail validation with a message identifying the field and the rule breached.
FR-1.6	The system shall version field definitions, so that a change to a definition does not invalidate records captured under the earlier definition.
FR-1.7	The system shall allow a field to be deactivated. Deactivation shall be a soft delete: the field and the values already captured against it are retained and marked inactive.

FR-1.8	The system shall prevent deactivation of a field that is referenced by an active validation rule, hierarchy, or published API contract, and shall list the dependencies to the administrator.

3.2 Classification, hierarchy and code lists

ID	Requirements
FR-2.1	The system shall support parent and child relationships between master data records, so that classifications, hierarchies and taxonomies can be represented.
FR-2.2	The system shall support multi-level hierarchies of arbitrary depth, for example Dzongkhag to Gewog to Village, and Ministry to Department to Division.
FR-2.3	The system shall enforce referential integrity within a hierarchy. A child record shall not exist without a valid parent, and a parent shall not be deactivated while active children reference it.
FR-2.4	The system shall allow a master data domain to be held as a flat code list where no hierarchy applies, for example, unit of measure, marital status, and gender.
FR-2.5	The system shall allow a record to be moved from one parent to another, and shall record the move in the change history.
FR-2.6	The system shall expose the full hierarchy path of a record, so that a consumer can retrieve a record together with its ancestors.

3.3 Record lifecycle

ID	Requirements
FR-3.1	The system shall allow an authorised tenant user to create a master data record within a domain the tenant is permitted to contribute to.
FR-3.2	The system shall allow an authorised user to amend an existing record, subject to the stewardship
FR-3.3	The system shall support soft delete only. A deleted record is retained in the database and marked inactive, with the reason, the user, and the timestamp recorded.
FR-3.3	The system shall exclude inactive records from default search results and from the default published API response, and shall return them only when a consumer explicitly requests inactive records.
FR-3.3	The system shall allow an inactive record to be reactivated by a data steward.

FR-3.3	Each golden record shall carry a system-generated identifier that is unique across the platform, permanent, and never reused after deactivation or merge.

3.4 Duplicate detection

ID	Requirements
FR-4.1	The system shall run a duplicate check when a new record is submitted and when an existing record is amended in a way that affects the matching attributes.
FR-4.2	Matching rules shall be configurable per entity domain, and shall support exact match, normalised match (case, whitespace, punctuation) and fuzzy match with a configurable similarity threshold.
FR-4.3	On an exact match, the system shall reject the submission and return the identifier of the existing golden record.
FR-4.4	On a suspected match below the exact threshold, the system shall place the submission in a pending state and raise a duplicate review task for a data steward. The submission shall not be published while the task is open.
FR-4.5	The duplicate review screen shall present the submitted record and each candidate match side by side, with the matching score and the differing attributes indicated.
FR-4.6	A data steward shall be able to close a duplicate review task by confirming the duplicate, rejecting it as a distinct record, or returning the submission to the submitter for correction.
FR-4.7	The system shall retain the outcome of every duplicate review, including the steward, the decision, the reason, and the timestamp.

3.5 Merge and unmerge

ID	Requirements
FR-5.1	On a confirmed duplicate, the system shall merge the records into a single golden record. The steward shall select the surviving record and the surviving value for each attribute where the records differ.
FR-5.2	The system shall retain the identifiers of all merged records and map them to the surviving golden record, so that a consumer presenting a superseded identifier is redirected to the surviving record.
FR-5.3	The system shall record the full merge event: the source records, the surviving record, the attribute-level selections, the steward, and the timestamp.

FR-5.4	The system shall support reversal of a merge. On reversal, the source records are restored to their pre-merge state, and the identifier mapping is withdrawn
FR-5.5	Reversal shall be restricted to an authorised role and shall require a recorded reason.
FR-5.6	The system shall notify consuming systems of a merge and of a reversal through the change notification mechanism.

3.6 Conflict resolution

ID	Requirements
FR-6.1	Where two or more tenants submit different values for the same attribute of the same entity, the system shall raise a conflict for steward resolution rather than overwrite silently.
FR-6.2	The system shall hold a configurable source-of-truth precedence per entity domain and per attribute, naming the authoritative register or tenant whose value prevails.
FR-6.3	Where precedence is configured and applicable, the system shall resolve the conflict automatically in favour of the authoritative source and shall log the automatic resolution.
FR-6.4	Where no precedence applies, the system shall raise a conflict task presenting both values, the submitting tenants and the submission timestamps, for manual resolution by the domain steward.
FR-6.5	The resolved value shall become the published golden record value. The non-surviving value shall be retained against the submitting tenant for reference.
FR-6.6	The system shall notify both submitting tenants of the resolution outcome.

3.7 Multi tenancy, users and access

ID	Requirements
FR-7.1	When two or more tenants submit different values for the same attribute of the same entity, the system shall raise a conflict for steward resolution rather than silently overwriting.
FR-7.2	The system shall support multiple tenants, with data partitioned by tenant.

FR-7.3	A system administrator shall be able to onboard a new tenant, including tenant profile, permitted entity domains, and the initial tenant administrator account.
FR-7.4	Users shall be created and managed within a tenant. A user shall belong to one tenant.
FR-7.5	Access shall be role-based. The system shall provide, at minimum, the (eg., roles system administrator, tenant administrator, data steward, data contributor, and read-only consumer) and shall allow further roles to be defined.
FR-7.6	A tenant user shall be able to view all master data belonging to that tenant.
FR-7.7	A tenant user shall be able to view published golden records shared across government, regardless of the tenant that originated them.
FR-7.8	A tenant user shall not be able to view another tenant's unpublished or draft master data.
FR-7.9	A tenant user shall be able to submit a new requirement for golden record data, including a request for a new field or a new domain value.
FR-7.10	The system shall support authentication integrated with the government identity provider, and shall enforce session timeout and account lockout policy.

3.8 Search and retrieval

ID	Requirements
FR-8.1	The system shall provide search across all entity domains the user is permitted to see.
FR-8.2	Search shall support keyword search, search restricted to a named field, and filters on domain, tenant, status, active or inactive, and date of last change.
FR-8.3	Search shall support partial and normalised matching, and shall return results ranked by relevance.
FR-8.4	A user shall be able to open a record from the search results and see its attributes, its hierarchy path, its status and its change history.
FR-8.5	The system shall allow search results to be exported by an authorised user in a machine-readable format.

3.9 Publication and consumption

ID	Requirements

FR-9.1	The system shall publish golden records to consuming agency systems, services and GA systems through an API.
FR-9.2	The API shall support retrieval of a single record by identifier, retrieval of a code list or hierarchy in full, and retrieval of records changed since a given timestamp.
FR-9.3	The API shall be versioned, and a version shall remain available for a defined deprecation period after its successor is released.
FR-9.4	The system shall support a change notification or subscription mechanism, so that a consuming system is informed of a creation, amendment, deactivation, merge or merge reversal affecting records it consumes.
FR-9.5	API access shall be authenticated and authorised per consuming system, and every call shall be logged with the consumer, the endpoint, the parameters and the timestamp.
FR-9.6	The API shall enforce rate limiting per consumer.

3.10 Change history and audit

ID	Requirements
FR10.1	The change history for every golden record shall record the user, the tenant, the attribute changed, the previous value, the new value, and the date and time. It is all audited in the Journaling Block (JLB).

4. Conceptual Data Model (Indicative)

The following conceptual entities are sufficient to support the capabilities in Section 4. Golden Record domains (Organization, Agency, Department, Location, Address, and similar) hold configurable, tenant-contributed master data; they do not hold an individual citizen's personal data.

4.1 Golden Record

Field	Description
record_id	System-generated identifier for this record. Permanent, unique across the platform, and never reused after deactivation or merge.
domain	The entity domain this record belongs to (e.g. Organization, Agency, Department, Location, Address).

Field	Description
tenant_id	The tenant that created this record and is accountable for its accuracy.
status	active | inactive | pending_review | merged.
created_by / created_at	The user and timestamp of the record's original creation.
last_updated_by / last_updated_at	The user and timestamp of the most recent amendment.

4.2 Field Definition

Field	Description
field_id	Unique identifier for this field definition.
domain	The entity domain this field belongs to.
field_name / data_type / length	The field's name, data type (text, number, date, boolean, or reference to another record or code list), and maximum length where applicable.
mandatory_flag / default_value	Whether the field is required, and its default value where applicable.
validation_rule	The rule applied to a submitted value for this field at data entry and API submission.
version	Version number of this field definition, so a change does not invalidate records captured under an earlier definition.
active_flag	Whether the field is active or has been deactivated (soft-deleted).

4.3 Hierarchy Relationship

Field	Description
relationship_id	Unique identifier for this relationship.
child_record_id / parent_record_id	The two golden records this relationship connects.
hierarchy_type	Which classification or hierarchy this relationship belongs to.
effective_from / effective_to	The period this parent-child link has been, or was, in effect — used when a record moves to a different parent.

4.4 Duplicate Review Task

Field	Description
task_id	Unique identifier for this review task.
submitted_record_id / candidate_record_id	The newly submitted record and the existing golden record it was matched against.
match_score / matching_attributes	The similarity score and which attributes triggered the match.
status	pending | confirmed | rejected | returned_for_correction.
assigned_steward / decided_at	The data steward who closed the task, and when.

4.5 Merge Event

Field	Description
merge_id	Unique identifier for this merge event.
source_record_ids / surviving_record_id	The records merged, and the one that survives.
attribute_selections	The steward's chosen surviving value for each attribute where the merged records differed.
steward / merged_at	Who performed the merge, and when.
reversed_flag / reversed_by / reversed_at / reversal_reason	Whether this merge was later reversed, and the details of that reversal, if so.

4.6 Conflict Resolution Task

Field	Description
conflict_id	Unique identifier for this conflict.
record_id / attribute	The golden record and the specific attribute in conflict.
submitting_tenant_a / value_a / submitting_tenant_b / value_b	The two tenants and the differing values they each submitted.

Field	Description
resolution_type	automatic_precedence | manual_steward.
resolved_value / resolved_by / resolved_at	The value that was published, who resolved the conflict, and when.

4.7 Tenant

Field	Description
tenant_id	Unique identifier for this tenant.
tenant_name	The tenant's display name.
permitted_domains	The entity domains this tenant is authorised to contribute to.
status / onboarded_at	Whether the tenant is active, and when it was onboarded.

4.8 User Account

Field	Description
user_id	Unique identifier for this user.
tenant_id	The single tenant this user belongs to.
role	system_administrator | tenant_administrator | data_steward | data_contributor | read_only_consumer.
status	Whether the user's account is active.

4.9 Change History Entry

Field	Description
record_id / tenant_id / user_id	The golden record changed, and who changed it, on behalf of which tenant.
field_changed / previous_value / new_value	What changed, and its value before and after.
change_type	create | amend | deactivate | reactivate | merge | merge_reversal.
timestamp	When the change occurred. Every entry is written to the Journal / Audit Log Block (JLB).

5. Integration with Other Building Blocks

Integration	Direction	Data Exchanged
NDE (National Data Exchange)	MDMB ⇄ NDE	All lookup requests/responses and Agency tenant-update requests
JLB (Journal / Audit Log Block)	MDMB → JLB	Audit records of lookups and Agency updates

6. Non-Functional Requirements

ID	Category	Requirement
NFR-1	Availability	The system shall be available at least 99.9% of the time monthly, measured excluding scheduled maintenance windows.
NFR-2	Performance	At least 95% of API lookups, searches, and record submissions shall complete within 2 seconds under normal load.
NFR-3	Scalability	The system shall scale to support all onboarded tenants and their peak concurrent submission and lookup volumes without breaching NFR-2.
NFR-4	Security	All traffic shall be encrypted in transit and at rest; every request shall be authenticated and authorised per the roles defined in FR-7.5.
NFR-5	Data Integrity	Referential integrity across hierarchies, merge mappings, and identifier assignments shall be enforced at all times (FR-2.3, FR-5.2, FR-3.3).
NFR-6	Auditability	Every creation, amendment, deactivation, merge, reversal, conflict resolution, and access shall be logged immutably and shall remain retrievable for audit (FR-10.1).
NFR-7	Interoperability	All external integration shall occur through documented, versioned APIs, with a superseded version remaining available for an agreed deprecation period (FR-9.3).
NFR-8	Usability	The duplicate review and conflict resolution screens shall present all information a steward needs for a decision on a single screen, without requiring separate navigation (FR-4.5).
NFR-9	Maintainability	Entity domains, fields, and hierarchy structures shall be configurable by an authorised administrator without requiring a code release (FR-1.2, FR-1.3).

7. Roles and Governance

MDMB does not run a formal review workflow for everyday value updates — those are self-certified by the owning Agency, individually or by batch. It does, however, require a Data Steward review for the much rarer events of onboarding or retiring a whole taxonomy (Capability 3).

Role	Accountability
System Administrator	Accountable for platform-wide configuration: onboarding tenants, defining entity domains and fields, and managing system-level settings.
Tenant Administrator	Accountable for their own tenant's users, the entity domain the tenant is permitted to contribute to, and day-to-day data contribution activity.
Data Steward	Accountable for reviewing duplicate matches, confirming or reversing merges, resolving cross-tenant conflicts, and approving the onboarding or retirement of a taxonomy or domain.
Data Contributor	Accountable for creating and amending golden records within the domains their tenant is permitted to contribute to.
Read-Only Consumer	Accountable only for their own access; may view published golden records and any data they are separately authorised to see, and nothing else.
MDMB Operations Team	Accountable for platform availability, monitoring, and responding to operational alerts.

8. Business Rules

ID	Rule
BR-1	A golden record is only created, amended, or deactivated through the record lifecycle in Capability 3.3 — never directly in storage.
BR-2	Every new or amended record is checked for duplicates before publication; an exact match is rejected, and a suspected match is held for steward review.
BR-3	A merge may only be confirmed by an authorised data steward, and every merge remains reversible by an authorised role with a recorded reason.
BR-4	Where two tenants submit conflicting values for the same attribute, the value is published only once resolved — automatically by a configured precedence rule, or manually by a domain steward.
BR-5	A record's system-generated identifier is permanent and is never reused, even after deactivation or merge.

ID	Rule
BR-6	A tenant may view all data belonging to its own tenant and all published golden records, but never another tenant's unpublished or draft data.
BR-7	A field or entity domain may only be deactivated if no active validation rule, hierarchy, or published API contract still depends on it.
BR-8	A deleted record is always a soft delete — the record is retained and marked inactive, never physically removed.
BR-9	Every change to a golden record — creation, amendment, deactivation, merge, or merge reversal — is recorded in the change history with the user, tenant, and timestamp.
BR-10	All external consumption of golden records occurs through a versioned, authenticated API; a superseded version remains available for an agreed deprecation period.

9. Glossary

Term	Definition
MDMB	Master Data Management Block: the shared store of taxonomy and hierarchy data, queried by My Bhutan Portal via NDE, only after SSOT has been checked and does not have a value. Holds no personal or business data.
SSOT	Single Source of Truth: the Agency that holds a citizen's actual record for a given kind of information. Always checked first, outside MDMB.
Taxonomy	A named list MDMB holds, either hierarchical (with parent-child levels) or flat.
Hierarchical Taxonomy	A taxonomy with parent-child levels: a child value always belongs to exactly one parent value.
Mapping	The recorded parent-child relationship within a hierarchical taxonomy. A taxonomy that needs to be mapped is mapped in MDMB, and that mapping remains changeable by the owning Agency.
Agency	The individual government agency responsible for maintaining one particular taxonomy, or for holding a citizen's actual record as an SSOT.
Multi-Tenancy	The architectural model under which MDMB hosts each Agency's taxonomy in its own isolated tenant partition, with no Agency able to read or write another Agency's partition.
Tenant Partition	The isolated portion of MDMB's storage that belongs to a single Agency.

Term	Definition
NDE	National Data Exchange: the data-exchange fabric that carries all lookup and Agency tenant-update traffic between My Bhutan Portal and MDMB.
Data Steward	The role accountable for reviewing and approving (or rejecting) the onboarding of a new taxonomy and the retirement of an existing one.
Grace Period	A configurable window, after an approved retirement is soft-deleted, during which an objection can still be raised before the taxonomy is archived or permanently deleted.
Soft-Delete	Marking a taxonomy inactive and revoking its API access without yet removing its data, the first, reversible step of an approved retirement.
JLB	Journal / Audit Log Block: the immutable audit sink MDMB writes every lookup and Agency update to.
My Bhutan Portal	The single front-end through which a citizen logs in and avails a government service, and through which service providers deliver and render services. The only channel through which services are availed.