Boosting digital adoption of MSME ecosystem

TSI DPDP Consent Management System

API Specification and Security Plan

28th May 2025
V1.0

TSI Tech Solutions Cooperative Foundation
A section 8 company

https://tsicoop.org

Table of Contents

Version History.....3
Abbreviations, Terms and Definitions.....3
Intended Audience.....3
About Us.....3
Scope.....4
API Design Principles.....5
Authentication & Authorization:.....5
Common API Structure.....6
API Endpoints for User Facing Applications.....7
API Endpoints for CMS Client SDK Integration.....9
API Endpoints for CMS Client Service Adapters.....9
Security Plan.....10
    Security Guiding Principles:.....10
    Security Domains/Layers:.....10
Compliance Alignment with DPDP Act.....12

Version History

Author(s)	Date	Version	Description
Satish Ayyaswami TSI Tech Solutions Cooperative Foundation	28/05/2025	0.1	Draft

Abbreviations, Terms and Definitions

DPDP Act	Digital Personal Data Protection Act 2023
CMS	Consent Management System
DF	Data Fiduciary
DP	Data Processor
DPB	Data Protection Board

Intended Audience

Architects, Developers, DevOps, QA, Project Managers, Stakeholders

About Us

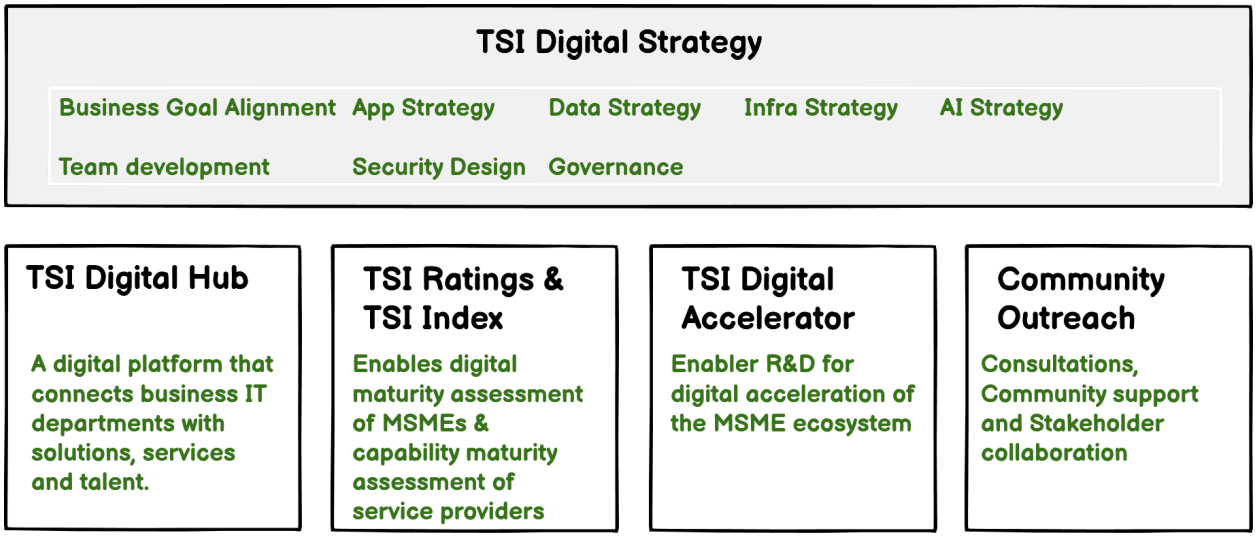
TSI Tech Solutions Cooperative Foundation (TSI Coop) initiative aims to address critical gaps in the MSME digital ecosystem by enabling businesses to implement successful technology strategies with the help of niche providers. Our mission also focuses on creating sustainable pathways for graduates from tier-3 and tier-4 institutions while fostering a more equitable and efficient domestic IT supply chain.

TSI stands for Technology & Social Impact. 'Coop' embodies the spirit of a cooperative economic model, where smaller providers and MSMEs collectively drive digital adoption within the ecosystem, fostering shared growth and opportunity.

While our IT machinery prioritizes foreign markets, GCCs, high profile startups and large enterprises, MSMEs are underserved. We aim to unlock their potential by facilitating the following:

- Discovery of niche providers, products, services and talent. Direct Interactions.
- Continuous digital maturity assessment of MSMEs ecosystem participants, helping stakeholders' identify areas for improvement
- Enabler R&D for digital acceleration of MSME ecosystem
- Community support and stakeholder collaboration

Our offerings for the MSME ecosystem below:



TSI DPDP Consent Management System is a key component of our TSI Digital Accelerator program.

Scope

This section outlines the design principles, authentication mechanisms, and common structures for the DPDP Solution's RESTful APIs. These APIs enable seamless integration with Data Fiduciaries, Data Processors, and various internal/external systems.

API Design Principles

1. **RESTful Principles:** Adherence to Representational State Transfer (REST) architectural style. Resources are identified by URLs, and standard HTTP methods (GET, POST, PUT, DELETE, PATCH) are used for operations.
2. **Statelessness:** Each API request from a client to server contains all necessary information; the server does not store client context between requests.
3. **Standard Data Formats:** JSON (JavaScript Object Notation) will be the primary format for all request and response bodies due to its lightweight nature and widespread adoption.
4. **Versioning:** API versions will be managed via the URL path (e.g., `/api/v1/consent`). This ensures backward compatibility for integrated clients during upgrades.
5. **Readability & Discoverability:** Clear, intuitive resource naming and comprehensive documentation (e.g., OpenAPI/Swagger).
6. **Idempotency:** Where applicable, operations (e.g., POST for initial consent recording) will be designed to be idempotent if replayed.

Authentication & Authorization:

The API Gateway will enforce authentication and authorization for all incoming requests.

- **API Keys (for Data Fiduciaries/Processors):**
  - Used for most automated, server-to-server integrations (e.g., frontend sending consent, backend validation calls).
  - Keys are unique per integrating client/application.
  - Generated securely by the CMS backend.
  - Passed via `X-API-KEY` HTTP header or as a query parameter (less secure).
  - Associated with specific permissions.
- **JWTs (for Admin UI / DPO Dashboard / User Dashboard):**
  - Used for user-facing applications requiring user login.
  - Clients obtain JWTs after user authentication (username/password, MFA).
  - JWTs are passed via `Authorization: Bearer <token>` header.
  - JWTs contain user roles and permissions.
- **Mutual TLS (Two-Way SSL) (for DPB Integration):**
  - Required for highly sensitive, server-to-server communication with the Data Protection Board (DPB).
  - Both client (CMS) and server (DPB) present and verify X.509 certificates.

- **Role-Based Access Control (RBAC):**
    - Authorization logic on the backend will check the authenticated user's/API key's associated roles and permissions against the requested resource and action.
    - Example permissions: `consent:read`, `consent:write`, `policy:publish`, `user:manage`, `audit:read`.

Common API Structure

Request Headers:

- `Content-Type: application/json`
- `Accept: application/json`
- `Authorization: Bearer <token>` (for user-based access)
- `X-API-KEY: <key>` (for service-to-service access)
- `Accept-Language: en, ta, hi` (for localized content)

Response Headers:

- `Content-Type: application/json`

Success Response Body:
JSON

```
Unset

{
  "data": { /* resource object or array of objects */ },
  "metadata": { /* pagination, etc. */ }
}
```

Error Response Body:
JSON

```
Unset

{
```

```
  "error": {
    "code": "string_error_code_e.g._INVALID_CONSENT_PAYLOAD",
    "message": "Human-readable error message explaining the
issue.",
    "details": "Optional: More specific details, validation
errors array."
  }
}
```

HTTP Status Codes:

- `200 OK`: Successful GET, PUT, PATCH.
- `201 Created`: Successful POST.
- `204 No Content`: Successful DELETE.
- `400 Bad Request`: Invalid request payload, missing parameters.
- `401 Unauthorized`: Authentication failed (invalid/missing token/key).
- `403 Forbidden`: Authenticated but not authorized to perform action.
- `404 Not Found`: Resource not found.
- `409 Conflict`: Request conflicts with current state (e.g., duplicate unique entry).
- `500 Internal Server Error`: Generic server error.
- `503 Service Unavailable`: Temporary server overload or maintenance.

API Endpoints for User Facing Applications

[For Admin UI / DPO Dashboard / User Dashboard]

HTTP Method	Endpoint	Description	Authentication / Authorization Scope
POST	/app/v1/consentmanager	Register CM, Validate Domain, Retrieve CM details	JWT / Admin (cm:write, cm:read), DPO (cm:read)

POST	/app/v1/datafiduciary	Register DF, Validate Domain, Retrieve DFdetails	JWT / Admin (df:write, df:read), DPO (df:read)
POST	/app/v1/dataprocessor	Register DP, Validate Domain, Retrieve DP details	JWT / Admin (dp:write,dp:read), DF DPO (dp:read), DP DPO (dp:read)
POST	/app/v1/policy	Publish Policy, Retrieve Policy	JWT / Admin (policy:write, policy:read), DF DPO (policy:write,policy:read), DP DPO (policy:read)
POST	/app/v1/consent	Retrieve Consents	JWT / Admin (consent:read), DF DPO (consent:read), DP DPO (consent:read)
POST	/api/v1/webhook	Enable DF & DP to register their webhook with the CMS for the purge event handling	JWT / Admin, DPO (for DF & DP) / webhook:write
POST	/app/v1/grievances	Submit a new privacy grievance from a Data Principal.	User JWT / grievance:submit
POST	/app/v1/grievances/{id}/status	Update the status of a specific grievance (e.g., from 'New' to 'In Progress').	Admin/DPO JWT / grievance:manage
GET	/app/v1/audit-logs	Retrieve filtered system audit logs.	Admin/Auditor JWT / audit:read
POST	/api/v1/dpb/reports/{type}	Submit a specific type of report (e.g., breach notification) to the DPB.	DPB Mutual TLS Auth + Admin/DPO JWT / dpb_report:submit

API Endpoints for CMS Client SDK Integration

[For retrieving policy, linking principal, recording consent]

HTTP Method	Endpoint	Description	Authentication / Authorization Scope
POST	/api/v1/consent	Record a Data Principal's new consent decision or update existing preferences.	API Key (for Fiduciary App) / consent:write
GET	/api/v1/consent/{userId}/active	Retrieve a Data Principal's current active consent preferences.	API Key (for Fiduciary App) / consent:read
GET	/api/v1/policies/active?fId={id}&j={j}	Retrieve the currently active consent policy (for frontend rendering).	Public / API Key (for Fiduciary Frontend App)

API Endpoints for CMS Client Service Adapters

[For consent validation, process purge]

HTTP Method	Endpoint	Description	Authentication / Authorization Scope
POST	/api/v1/purge-status	DF / DP posts the purge status to CMS	API Key (for DF & DP) / purge:write

Security Plan

This section details the comprehensive security measures implemented across the DPDP Solution's architecture, adhering to "Security by Design" and "Privacy by Design" principles, as well as the DPDP Act's "reasonable security safeguards" requirement.

Note: Some of the security measures need to be implemented outside the CMS solution highlighted using *

Security Guiding Principles:

Least Privilege: Users and services are granted only the minimum necessary permissions to perform their functions.
Data Minimization: Only collect, process, and retain data that is necessary for the defined purposes.
Auditability & Transparency: All security-relevant actions are logged and auditable.

Security Domains/Layers:

Application Layer:

- **Secure Coding Practices:** Adherence to OWASP Top 10 guidelines (e.g., preventing injection, XSS, insecure deserialization). Regular static (SAST) and dynamic (DAST) application security testing.
- **Input Validation & Sanitization:** Strict validation and sanitization of all user-supplied inputs to prevent malicious data injection.
- **API Security:** Robust authentication, authorization, rate limiting*, and API schema validation.
- **Secure Error Handling:** Generic error messages; detailed error logs are internal only.

Authentication & Authorization Layer:

- **Multi-Factor Authentication (MFA):** Mandatory for all CMS administrators, DPOs, and other privileged users (e.g., using Email OTP/ TOTP*).
- **Strong Password Policy:** Enforced for all CMS users (minimum length, complexity, history, expiry).
- **Role-Based Access Control (RBAC):** Fine-grained permissions managed through the User Role Management module, applied at API and UI levels.
- **Session Management:** Secure session IDs, inactivity timeouts, forced re-authentication for sensitive actions.

Data Layer:

- **Encryption at Rest*:** Leverage Managed Database service from cloud vendors with built-in at-rest encryption for production deployments.
- **Encryption in Transit:** All communication (client-to-server, inter-service, API calls, webhooks, DPB integration) is encrypted using TLS 1.2 or higher. Two-Way SSL for DPB.
- **Database Access Control:** Strict database-level access policies (least privilege) for backend services.

Infrastructure Layer:

- **Network Segmentation*:** Use VPCs, subnets, Security Groups, and Network ACLs (cloud) to isolate components and limit network access.
- **Host Security*:** Hardened OS images, regular patching, minimal services, host-based firewalls, anti-malware.
- **Container Security*:** Use minimal base images, scan container images for vulnerabilities, run containers with least privilege (non-root users, restricted capabilities).
- **Secret Management*:** Securely manage API keys, database credentials, and certificates using dedicated secret management solutions (e.g., AWS Secrets Manager).
- **Load Balancing*:** Use secure load balancers with SSL termination.

Operational Security:

- **Centralized Logging & Monitoring*:** Comprehensive, immutable audit logs captured for all security-relevant events (access, configuration changes, data operations, policy changes). Monitored in real-time for anomalies.
- **Security Information and Event Management (SIEM)*:** Integration with a SIEM system for advanced threat detection and analysis.
- **Vulnerability Management*:** Regular penetration testing, vulnerability scanning (network, application, container images), and security audits.
- **Incident Response Plan*:** A well-defined and regularly tested plan for detecting, containing, eradicating, recovering from, and reporting security incidents (especially data breaches, aligned with DPDP Act notification requirements).
- **Disaster Recovery & Business Continuity*:** Robust backup strategies (with cross-regional replication for critical data) and DR plans to ensure data availability and service continuity.

- **Security Awareness Training*:** Regular training for all personnel (including DPOs, Admins) on data protection best practices and security policies.

Compliance Alignment with DPDP Act

Reasonable Security Safeguards: The layered and comprehensive security measures described fulfill the Act's requirement for implementing "reasonable security safeguards."
Breach Notification: The Incident Response Plan specifically addresses the Act's obligations for notifying the Data Protection Board and affected Data Principals of a personal data breach.
Accountability: The robust audit logging and monitoring capabilities provide the necessary evidence for demonstrating compliance and accountability to the DPB.
Consent & Purpose Enforcement: Security controls work hand-in-hand with functional controls to ensure data is only processed for consented purposes by authorized entities.