1. API Security Policy

4.1 Purpose

To ensure the secure design, implementation, and operation of APIs to prevent unauthorized access and data breaches.

4.2 Scope

This policy applies to all APIs developed, hosted, or managed by the entity.

4.3 Principles

4.3.1 **Least Privilege:** APIs must enforce the principle of least privilege, granting only the minimum access necessary for functionality.
4.3.2 **Data Protection:** Sensitive data must be encrypted in transit and at rest.
4.3.3 **Authentication and Authorization:** All API requests must be authenticated and authorized.
4.3.4 **Input Validation:** All API inputs must be validated to prevent injection attacks.
4.3.5 **Monitoring and Logging:** API activity must be monitored and logged for security and auditing purposes.

4.4 Authentication and Authorization

4.4.1 **Authentication:**

a. Use **OAuth 2.0**, **API keys**, or **mutual TLS (mTLS)** for authentication.

4.4.2 **Authorization:**

a. Implement **RBAC** or **ABAC** to ensure users have appropriate permissions.

4.4.3 **Token Management:**

a. Access tokens must have short expiration times and be securely stored.

4.5 Data Protection

1. Encrypt all sensitive data transmitted via APIs using **TLS 1.2 or higher**.
2. Mask sensitive data (e.g., PII, financial information) in API responses where possible.
3. APIs must not retain sensitive data longer than necessary, in compliance with data retention policies.

4. Citizen information or any type of critical information will not be provided unless three parameters is submitted
5. All APIs should be protected by a well configured WAF.
6. API Security solution is highly recommended.
7. Citizen information APIs or any type of critical APIs information cannot be accessed without passing the following requirements (user name,passowrd and OTP).
8. The system that accesses the API should be Free from any vulnerabilities
9. A penetration test and vulnerabilities scanning should be conducted on any systems that access the APIs
10. All API Access should be logged in a safe place with all needed details (API url, Time and date, real source address, user identification)
11. Prevent access to the system except from previously authorized addresses.
12. The DATA producer should approve the access of the data consumer.
13. Admissible Data owner's consent should be taken and stored in a safe place
14. All APIs usage shall be conducted through government service bus(GSB).
15. It is completely not allowed to share encryption keys with the encrypted data.
16. Any bulk data should not be delivered to any entity before approval of Data committee.
17. The bulk data should contain the ID info only without images.
18. The entity should declare the place which will host the data.
19. The entity should destroy any data rather than the ID info under the supervision of MODEE Team.(Databases, backups and flat files)
20. The entity should provide MODEE with a list of the counter measures which are being used to protect the data from abuse and attacks.
21. MODEE cyber security should be applied on the system that hosts the data.

4.6 Input Validation and Output Encoding

4.6.1 Input Validation:

a. Validate all API inputs (e.g., query parameters, headers, payloads) to prevent injection attacks.

4.6.2 Output Encoding:

a. Encode API responses to prevent cross-site scripting (XSS) and other injection attacks.

4.7 Rate Limiting and Throttling

4.7.1 Rate Limiting:

a. Implement rate limiting to prevent abuse .

4.7.2 **Throttling:**

a. In ordedr to smooth spikes ,throttle requests to ensure fair usage and maintain system performance.

4.8 Error Handling

4.8.1 **Generic Error Messages:**

a. Return generic error messages to avoid exposing sensitive information.

4.8.2 **Logging:**

a. Capture detailed error logs for internal debugging but do not expose them to end-users.

4.9 Monitoring and Logging

4.9.1 **Activity Logs:**

a. Log all API requests and responses for auditing and monitoring purposes.

4.9.2 **Anomaly Detection:**

a.Implement tools to detect and alert on unusual API activity (e.g., spikes in traffic, unauthorized access attempts).

4.9.3 **Log Retention:**

a. Retain logs for a minimum of **180 days**, in compliance with regulatory requirements.

4.10 Security Testing

4.10.1 **Penetration Testing:**

a. Conduct regular penetration testing of APIs.

4.10.2 **Static and Dynamic Analysis:**

a. Test APIs using static and dynamic analysis tools during development and deployment.

4.10.3 **Vulnerability Scanning:**

a. Scan APIs for vulnerabilities (e.g., OWASP API Security Top 10) before production release.

4.11 Third-Party APIs

4.11.1 **Vendor Assessment:**

a. Assess third-party APIs for security compliance before integration.

4.11.2 **Data Sharing Agreements:**

a. Include data protection and security requirements in contracts with third-party API providers.

4.11.3 **Monitoring:**

a. Monitor third-party API usage for compliance with security policies.

4.12 Incident Response

4.12.1 **Incident Reporting:**

a. Report any suspected or confirmed API security incidents immediately to the cyber security team.

4.12.2 **Response Plan:**

a. Maintain a documented incident response plan for API-related security breaches.

4.12.3 **Post-Incident Review:**

a. Conduct post-incident reviews to identify root causes and implement corrective actions.

4.13 Compliance

4.13.1 **Regulatory Requirements:**

a. Ensure APIs comply with applicable regulations (e.g cyber security law,personal data protection law).

4.13.2 **Audits:**

a. Conduct regular audits to ensure compliance with this policy and API Minimum Base Line(Reference).

4.14 API Documentation Security

1. Swagger/OpenAPI documentation **must not** be publicly accessible in production environments.
2. If Swagger/OpenAPI documentation is required for internal use, it must be:
    - Hosted behind authentication and authorization controls.
    - Accessible only from approved IP addresses or networks.
    - Reviewed regularly to ensure it does not contain sensitive data or endpoints.
3. Any Swagger/OpenAPI files generated during development must be removed or disabled before deployment to production.