Shuttlemate Payment Data Protection & Transaction Security Policy

215068P

H.KodikaraProtection

Document Control	**Details**
Policy Number	SM-POL-S2-002
Topic	Payment Data Protection & Transaction Security
Classification	Internal Use Only
Owner	Student 2 (Payment Security Lead)
Approver	CISO / Project Board
Effective Date	January 30, 2026
Version	1.0 (Final)

1. Purpose

This Security Policy establishes mandatory controls for the handling of financial transactions within the ShuttleMate application to ensure compliance with the Payment Card Industry Data Security Standard (PCI-DSS) and to protect sensitive cardholder data from unauthorized access, theft, or compromise.

2. Scope

This policy applies to

- All ShuttleMate systems that process, store, or transmit payment-related data
- All developers, engineers, and technical staff involved in payment integration
- All API endpoints and backend services that receive payment webhooks

- Third-party payment processors integrated with ShuttleMate (currently: Stripe)
- Development, testing, staging, and production environments

3. Policy Statements

3.1 Payment Card Data Handling (PCI-DSS Enforcement)

- ShuttleMate shall not store, process, or transmit raw Primary Account Numbers (PAN), CVV/CVC codes, or card expiration details on its servers.
- All payment processing must be delegated exclusively to a PCI-DSS–compliant third-party payment processor (e.g., Stripe).
- Tokenized payment identifiers provided by the payment processor may be stored only for transaction reference and reconciliation purposes.
- Any attempt to log, cache, or persist sensitive cardholder data is strictly prohibited.

3.2 Webhook Authentication & Integrity

- All payment-related webhook requests received from the payment processor must be cryptographically verified using the official signing secret provided by the processor.
- Webhook payloads must not be processed unless signature validation is successfully completed.
- Requests with missing, invalid, or mismatched signatures shall be rejected immediately with an HTTP 400 (Bad Request) response.
- Webhook endpoints must use HTTPS to prevent man-in-the-middle attacks.

3.3 Secure Configuration & Access Control

- Payment-related API keys and webhook secrets **must be stored securely** using environment variables or a secrets management service.
- Access to payment configuration settings is restricted to authorized personnel (Payment Lead and backend administrators).
- Secrets must never be hard-coded or committed to version control systems.

5. Roles and responsibilities

5.1 Payment Lead (S2)

- Own and maintain this security policy

- Conduct annual PCI-DSS compliance reviews
- Manage relationships with payment processors
- Review and approve all payment integration code
- Investigate and respond to security incidents involving payment systems

5.2 Development Team

- Implement payment features in accordance with this policy
- Never store prohibited data elements
- Implement proper webhook signature verification on all payment endpoints
- Use secure coding practices when handling payment-related functionality
- Report any suspected policy violations or security concerns immediately

5.3 Security Team

- Monitor webhook verification logs for suspicious patterns
- Conduct periodic code reviews of payment integration code
- Perform penetration testing on webhook endpoints
- Maintain and update secrets management infrastructure
- Respond to security incidents and coordinate remediation

5.4 DevOps/Infrastructure Team

- Maintain secure secrets management systems
- Ensure webhook signing secrets are properly configured in all environments
- Implement network security controls for payment endpoints
- Monitor and alert on webhook verification failures

6. Compliance and Enforcement

6.1 Mandatory Code Review

All code changes affecting payment processing MUST undergo security-focused code review by the Payment Lead or designated security reviewer before merging to production branches.

6.2 Automated Scanning

Static code analysis tools MUST be configured to detect

- Potential storage of PAN, CVV, or other prohibited data
- Hardcoded secrets or credentials
- Missing webhook signature verification
- Insecure cryptographic implementations

6.3 Testing Requirements

All webhook endpoints MUST have automated tests that verify

- Valid signatures are accepted
- Invalid signatures are rejected with HTTP 400
- Missing signatures are rejected
- Expired signatures are rejected
- Tampered webhook data is detected and rejected

7. INCIDENT RESPONSE

7.1 Security Incidents

The following events constitute security incidents requiring immediate escalation

- Detection of PAN, CVV, or prohibited data stored in ShuttleMate systems
- Suspected compromise of webhook signing secrets
- Repeated webhook verification failures from unexpected sources
- Unauthorized access to payment processor accounts
- Unusual patterns in payment transaction logs

7.2 Response Procedures

Upon discovering a security incident

- **Immediate containment -** Disable affected endpoints or services if necessary
- **Notification -** Alert Payment Lead and Security Team within 15 minutes
- **Assessment -** Determine scope, impact, and root cause
- **Evidence preservation -** Capture relevant logs, system states, and audit trails
- **Remediation -** Implement fixes and validate effectiveness

- **External notification -** Notify payment processor and regulatory authorities as required by law
- **Post-incident review -** Document lessons learned and update security controls

7.3 Data Breach Notification

In the event of a confirmed or suspected breach of payment card data

- Notify Stripe (or applicable payment processor) immediately
- Engage forensic investigators if compromise is suspected
- Follow PCI-DSS breach notification requirements
- Comply with applicable data breach notification laws (GDPR, CCPA, etc.)
- Document all actions taken and maintain detailed incident records