Payment gateway integration service for a U.S. startup

Overview

Custom backend engineering for a U.S.-based startup launching a universal payment gateway

Touchlane collaborated with a fintech startup from the United States that planned to introduce a cross-platform payment gateway for online merchants. The company needed a secure and scalable backend capable of connecting different payment providers under one system and processing large transaction volumes with real-time verification.

The client's priority was a **fast MVP delivery,** but with a foundational architecture designed for a multi-tenant environment from the outset. Touchlane handled the architecture design, API creation, and deployment and monitoring in the AWS cloud.

Initial task

Description

The startup needed a backend that would process payments through multiple third-party services. These included:

- PayPal
- payment scheme integration (Visa / Mastercard / AmEx)
- local banking APIs.

In addition, the backend should offer real-time transaction tracking for both customers and merchants. A high availability setup was required to guarantee uninterrupted payment flows – even during system maintenance or unexpected traffic surges. PCI-DSS compliance was also a must.

Technology stack

- **Programming language /** Java
- **Framework /** Spring Boot
- **Cloud platform /** AWS
- **Database /** PostgreSQL, Amazon DynamoDB
- **Messaging /** Apache Kafka
- **CI/CD /** AWS CodePipeline, Docker
- **Security /** AWS Secrets Manager, JWT, TLS 1.3

Challenges

Two main challenges included:

- **Building a scalable authorization gateway.** Direct integration with card schemes using the ISO8583 protocol presented complex challenges. Legacy systems and the need for constant updates to meet scheme standards created risks of failed authorizations, financial discrepancies, and compliance issues.
- **Automating clearing and settlement.** The complex, often manual processes for clearing and settlement with schemes like Visa and Mastercard carried risks of reconciliation errors, delayed funds, scheme fines, and financial exposure. Automating this was critical.

Additionally, Touchlane's team had to keep in mind the following:

- Integrating and normalizing data across several payment providers with different API structures
- Maintaining transaction integrity across distributed services with guaranteed message delivery
- Implementing a foundational rules engine to flag suspicious transactions in near real-time
- Meeting strict SLAs (response time under 150 ms) and maintaining full data encryption and audit trails required for **PCI-DSS compliance.**

Development

Phase 1. Architecture planning and core setup

An architectural blueprint for a microservices system served as the project's starting point. Each service was responsible for a specific part of the transaction lifecycle. That included authorization, settlement, refunds, and audit logging.

To meet PCI-DSS requirements, the system was designed with multi-layered security controls:

- network-level isolation
- TLS 1.3 for data in transit
- AES-256 encryption at rest
- centralized key management via AWS KMS.

Touchlane engineers used Kafka to implement asynchronous message processing, which helped manage concurrent transactions and minimize latency. To make version management and scaling easier, containerized deployment was done using AWS ECS.

Phase 2. API development and third-party integrations

During this stage, the engineering team implemented a dedicated service for card scheme communication. This gateway featured async processing for high throughput and real-time monitoring. To maintain compliance with evolving scheme standards, the team designed a

system for automated rule updates. They integrated HSM-backed key management for secure transaction signing.

For the post-authorization lifecycle, engineers built API-based clearing connectors. These systems automated the entire process and provided full transaction reconciliation, tracking scheme fee updates, and maintaining complete audit logs. All sensitive data related to clearing used HSM-secured storage.

Additionally, the team created RESTful APIs to connect the platform with external payment gateways. A routing engine was built to dynamically choose a provider based on transaction type, location, and currency.

Touchlane developers implemented custom retry logic and circuit breakers to maintain stability in case of third-party downtime. Security modules included authentication based on JSON Web Token and AWS Secrets Manager for key rotation.

Phase 3. Monitoring, load testing, and go-live

Before launch, the backend underwent extensive stress testing. Touchlane's team used simulated traffic that mirrored real transaction behavior. CloudWatch and Prometheus dashboards were added for operational visibility and performance tracking.

To guarantee consistency across all endpoints, a QA engineer conducted integration and regression testing. Following validation, the system was set up for disaster recovery and high availability across several AWS regions.

Results

The final solution processes **over 10,000 transactions per minute** with an uptime rate above **P99**. Average API latency remained below **120 ms** during high-load periods.

After launch, the startup onboarded **more than 40 online merchants** within the first quarter and expanded payment coverage to North America.