Test Strategy Document: QR Code-Based Payment Method

Table of Contents

1. Introduction

This document outlines the comprehensive test strategy for validating a new digital payment method-**QR Code-Based Payments** being introduced alongside traditional payment systems such as SEPA bank transfers. The primary goal is to ensure functional correctness, robust integration, regulatory compliance, and user-friendly experiences across both web and mobile platforms.

This structured approach is designed to be extensible and applicable to other new digital payment methods planned in future:
- **Cryptocurrency Payments**
- NFC-based Tap to Pay
- Digital Wallets
- Buy Now Pay Later (BNPL)
- Instant Payments (SEPA Instant)
- Account-to-Account (A2A) via Open Banking

2. Payment Method in Focus

QR Code-Based Payments:
- Users scan a merchant-generated QR code containing payment metadata (amount, merchant ID, currency).
- Payment is confirmed and authorized by the user using SCA (biometric/OTP).
- Transactions are processed via SEPA or A2A rails, with real-time acknowledgment using pacs.002 and camt.054.

3. Test Objectives

- Ensure reliable execution of all user and system-level payment flows.
- Validate backend processing, including SEPA integration (e.g., pain.001, pacs.002).
- Confirm compliance with PSD2, GDPR, and ISO 20022 standards.
- Ensure seamless experience across mobile and web platforms.
- Establish an automation-ready framework for CI/CD integration.

4. Test Scope

In Scope:
- End-to-end functional validation of QR code-based payment flows
- Backend processing and SEPA message validation
- UI and UX testing across devices
- Security and fraud simulation
- Automation of critical paths, regression, and smoke tests

Out of Scope:
- Manual SEPA clearing handled by banks
- Direct user interaction with ISO 20022 message formats
- Validation of pacs.008 (unless explicitly owned)

5. Functional Testing

- **QR scan flow:** successful and failed scans
- Payment confirmation and success/failure handling
- **Edge cases:** expired QR, incorrect metadata, unsupported currency
- Wallet funding, top-up, and refunds (for digital wallet reuse)
- BNPL flow (eligibility checks, payment schedules)
- Retry mechanisms and fallbacks

6. Integration Testing

- **Open Banking APIs:** consent flow, account linking, SCA enforcement
- **SEPA backend validations:**
  - pain.001: Credit transfer initiation
  - pain.008: Direct debit initiation
  - pacs.002: Payment status updates
  - pacs.004: Return processing
  - camt.053: End-of-day statements
  - camt.054: Real-time transaction reports
  - pacs.008: Optional interbank transfers (where applicable)
- Notification systems (email, SMS, in-app, push notifications)

7. Mobile & Web Testing

- **Mobile (iOS/Android):**
  - Biometric login
  - QR code scanner integration
  - Offline and low-network resilience
- **Web:**
  - Responsive QR consent journeys
  - Accessibility testing (WCAG)
  - Localization (multi-language)

8. Security Testing

- Strong Customer Authentication (BankId, OTP, biometrics)
- Data-in-transit and at-rest encryption (TLS, AES)
- Session management, replay attack prevention
- Rate limiting and brute-force detection
- **Fraud scenarios:** QR tampering, impersonation, duplicate scan prevention

9. Performance Testing

- API throughput under varying loads
- Concurrent QR payments (1000+ TPS)
- Spike and soak testing for backend services
- Resilience and recovery benchmarks
- **Tools:** k6, Gatling, JMeter for scenario design, execution, and metrics

10. Automation Strategy

-Design and maintain acceptance tests for APIs, business logic, and integration points
-Automate integration tests across web and mobile platforms
-Validate backend processes and message flows in pre-production environments
-Run integration tests as part of pull requests to catch issues early
-Schedule daily health check runs to verify system availability and critical functionality
-Execute full regression suite at the end of each sprint for release readiness
-Integrate automated tests into CI/CD pipelines for continuous feedback
-**Tools**: SpecFlow, RestSharp, Postman, PactumJS, Playwright, Cypress, WebdriverIO, Appium, Flutter Integration Driver

11. Compliance Testing

- **PSD2:** SCA, consent workflows, access delegation
- **GDPR:** data masking, user rights (deletion, consent)
- **ISO 20022:** SEPA format validation for all applicable message types
- AML/KYC validation in onboarding flows

12. Test Data Management

- Dynamic test users and wallet configurations
- Mock SEPA IBANs, credentials, and identity docs
- Secure handling of PII in pre-prod and CI environments
- Shared anonymized datasets for automation and load tests

13. Summary

This strategy ensures comprehensive functional validation of the QR Code-Based Payment system, addressing core areas such as integration, security, performance, and regulatory compliance. Each test layer is structured to align with financial industry standards and user experience expectations, ensuring robustness and reliability. The approach is designed to be modular and reusable, supporting future digital payment methods including Cryptocurrency Payments, NFC Tap-to-Pay, BNPL, Instant Payments, Open Banking, and Wallet-based transactions.

The automation-first strategy enhances development agility by embedding test execution into CI/CD pipelines, enabling rapid feedback and faster releases. Automated tests are maintained for API, web, and mobile platforms, supporting acceptance, regression, and health check scenarios. This continuous testing approach ensures sustained product quality, seamless rollout of new features, and efficient validation in regulated environments.