## Phase 7: Integration & External Access – Event Management & Ticketing System

### 🔹 Objective

To enable the *Event Management & Ticketing System* to securely integrate with **external services** for payments, QR code generation, email/SMS notifications, and survey tools. This ensures real-time updates, automation, and scalability.

### 🔹 Integration Components

#### 1. Named Credentials

* Securely store authentication details for external systems.
* **Use Case:** Payment Gateway API (e.g., Stripe/PayPal) for ticket payments.

HttpRequest req = new HttpRequest();  
req.setEndpoint('callout:PaymentGatewayAPI/charge');  
req.setMethod('POST');  
req.setHeader('Content-Type','application/json');  
req.setBody(JSON.serialize(paymentDetails));  
Http http = new Http();  
HttpResponse res = http.send(req);

#### 2. External Services

* Import OpenAPI schema of payment/feedback APIs.
* Use declarative Flows to call services.
* **Example:** Call external survey API after event completion to collect attendee feedback.

#### 3. REST/SOAP Web Services

* **REST Example:** Send ticket confirmation to external notification service.

HttpRequest req = new HttpRequest();  
req.setEndpoint('https://sms-service.com/api/send');  
req.setMethod('POST');  
req.setHeader('Content-Type','application/json');  
req.setBody(JSON.serialize(smsObj));  
Http http = new Http();  
HttpResponse res = http.send(req);

* **SOAP Example:** Legacy system integration for event sponsorship records.

#### 4. Callouts

* Callouts used for:
  + Payment verification
  + QR code generation
  + Email/SMS notifications
  + Feedback collection
* Asynchronous callouts via @future(callout=true) for high-volume processes.

#### 5. Platform Events

* Publish events when:
  + Ticket is purchased
  + Attendee checks in
  + Event is completed
* External systems subscribe for real-time updates.

#### 6. Change Data Capture (CDC)

* Enable CDC on \*\*Event\_\_c, Ticket\_\_c, Attendee\_\_c\*\*.
* Sync changes (ticket status, feedback updates) with external reporting systems.

#### 7. Salesforce Connect

* Connect external datasets (e.g., external sponsorship/partner data) without storing in Salesforce.
* Display external financial data for admins alongside Salesforce data.

#### 8. API Limits

* Monitor API usage to avoid exceeding Salesforce limits.
* Use batching and Platform Events to optimize callouts.

#### 9. OAuth & Authentication

* Use OAuth 2.0 for secure integrations.
* **Use Case:** Authenticate with third-party payment and email services without hardcoding credentials.

#### 10. Remote Site Settings

* Whitelist external URLs used for callouts (e.g., Stripe API, Mailgun API, Survey API).

### 🔹 Example Workflow

1. Attendee registers for an event → Payment Gateway API callout for ticket purchase.
2. On success → Ticket\_\_c record created + QR Code generated.
3. Confirmation email sent via Email Service API.
4. At check-in → QR code validated using external QR API.
5. After event completion → Feedback API invoked to collect survey responses.
6. Reports generated with Salesforce + external analytics.

### 🔹 Outcome of Phase 7

✅ Secure integration with payment gateways, email/SMS services, and feedback APIs.  
✅ Real-time updates with Platform Events & CDC.  
✅ External data displayed inside Salesforce using Salesforce Connect.  
✅ Scalable and secure integrations using OAuth + Named Credentials.

This completes **Phase 7** documentation for the *Event Management & Ticketing System*. The system is now **UI-ready** and **integration-enabled**.