

# Phase 7: Integration & External Access

## Goal:

Enable Salesforce to securely communicate with external systems (hotel API) while ensuring data security, scalability, and compliance with Salesforce governor limits

## 1. Named Credentials ( Used)

### Purpose:

Securely store the base URL and authentication settings for the hotel API without hardcoding credentials inside Apex.

### Steps Followed:

- Setup → Named Credentials → New.
- Added **Label**: HotelAPI and **URL**: <https://api.hotel.com> (replace with actual API URL).
- Chose **Identity Type**: Anonymous and **Authentication Protocol**: No Authentication, since using a demo API without login credentials.
- Saved the configuration.
- **Why Important:**
  - Avoids hardcoded API endpoints in Apex.
  - Centralized management of authentication.
  - Secure and easy to update if endpoint changes.

The screenshot shows the Salesforce setup interface for Named Credentials. The left sidebar has 'Named Credentials' selected under 'Security'. The main area title is 'Named Credentials'. Below it, there are tabs for 'Named Credentials' (which is active), 'External Credentials', and 'External Auth Identity Providers'. A search bar at the top says 'Q\_ Named Credentials'. The main content area shows a table with one item: 'HotelAPIConnection' (Label), 'Secured Endpoint' (Type), 'https://api.examplehotel.com' (URL), and 'Password Authentication' (External Credential). There are 'New' and 'Actions' buttons at the top right of the table.

Label	Type	URL	External Credential	Actions
HotelAPIConnection	Secured Endpoint	https://api.examplehotel.com	Password Authentication	New ▾

The screenshot shows the 'SETUP > NAMED CREDENTIALS' interface for a named credential named 'HotelAPIConnection'. The page includes fields for Label (HotelAPIConnection), URL (https://api.examplehotel.com), and Name (HotelAPIConnection). It also features sections for Authentication (External Credential, Password Authentication, Client Certificate), Callout Options (Generate Authorization Header, Allow Formulas in HTTP Header, Allow Formulas in HTTP Body, Outbound Network Connection), and Managed Package Access (Created By Namespace).

## 2. External Services ( Not Used)

### Purpose:

- Allows Salesforce to consume APIs described by OpenAPI/Swagger without writing code.

### Reason for Skipping:

- The hotel API does not provide a Swagger/OpenAPI definition.
- We directly integrated via **Named Credentials + Apex Callout**.

## 3. Web Services (REST/SOAP) ( Used – REST)

### Purpose:

- Salesforce supports integrating with external systems via REST or SOAP.
- REST → lightweight, JSON-based (used in this project).

## **Implementation:**

- Created an Apex class to send REST callouts to the hotel API.
- Example: Fetch hotel data using  
`req.setEndpoint('callout:HotelAPI/listHotels');`

## **Code Skeleton (Apex):**

```
public with sharing class HotelAPIService {  
  
    @AuraEnabled(cacheable=true)  
  
    public static String getHotels() {  
  
        HttpRequest req = new HttpRequest();  
  
        req.setEndpoint('callout:HotelAPI/listHotels');  
  
        req.setMethod('GET');  
  
  
  
        Http http = new Http();  
  
        try {  
  
            HttpResponse res = http.send(req);  
  
            if(res.getStatusCode() == 200){  
  
                return res.getBody(); // JSON response  
  
            } else {  
  
                return 'Error: ' + res.getStatusCode();  
  
            }  
  
        } catch(Exception e){  
  
            return 'Exception: ' + e.getMessage();  
        }  
    }  
}
```

```

    }
}

}

```

## Why Important:

- Enables Salesforce to send HTTP requests (GET/POST) to external systems.
- Fetches real-time hotel data for display in Lightning Web Components.

The screenshot shows the Salesforce Setup interface with the following details:

- Apex Class**: HotelAPIService
- Name**: HotelAPIService
- Namespace Prefix**: (empty)
- Created By**: Shaik Yasmin, 9/26/2025, 1:13 AM
- Status**: Active
- Code Coverage**: 0% (0/13)
- Last Modified By**: Shaik Yasmin, 9/26/2025, 1:13 AM

The **Class Body** tab is selected, displaying the following Apex code:

```

1 public with sharing class HotelAPIService {
2     @AuraEnabled(cacheable=true)
3     public static String getHotels() {
4         HttpRequest req = new HttpRequest();
5         req.setEndpoint('callout:HotelAPI/endpoint'); // Named Credential
6         req.setMethod('GET');
7
8         Http http = new Http();
9         HttpResponse res;
10
11        try {
12            res = http.send(req);
13            If(res.getStatusCode() == 200){
14                return res.getBody(); // JSON data from API
15            } else {
16                return 'Error: ' + res.getStatusCode() + ' - ' + res.getStatus();
17            }
18        } catch(Exception e) {
19            return 'Exception: ' + e.getMessage();
20        }
21    }
22 }

```

- 

## 4. Callouts ( Used)

### Purpose:

- Allow Salesforce to send HTTP requests (GET/POST) to the hotel API for real-time data.

### Implementation:

- Added Named Credential HotelAPI to store API URL and authentication.

- Created Apex class HotelAPIService to make callouts:

```
HttpRequest req = new HttpRequest();
req.setEndpoint('callout:HotelAPI/listHotels');
req.setMethod('GET');

HttpResponse res = new Http().send(req);
```

- Parsed JSON response and displayed hotel data in LWC.

#### Why Important:

- Enables Salesforce to fetch live hotel details without hardcoded credentials.

[Named Credentials](#)   [External Credentials](#)   [External Auth Identity Providers](#)

1 Items · Sorted by Label

Label	Type	URL	External Credential
HotelAPIConnection	Secured Endpoint	<a href="https://api.examplehotel.com">https://api.examplehotel.com</a>	<a href="#">Password Authentication</a>

## 5. Platform Events ( Used)

#### Purpose:

- Support **real-time notifications** inside Salesforce and external systems.

#### Implementation:

- Created BookingCreated\_\_e platform event.
- Published event after a hotel booking is confirmed.
- Subscribed (via Apex trigger / external listener) to react instantly.

#### Why Important:

- Decouples booking logic from notifications.
- Enables **instant alerts** (e.g., send confirmation or fraud check).

The screenshot shows the 'Platform Event Definition Detail' page for 'Booking Created'. At the top, there are buttons for 'Edit' and 'Delete'. Below this, the event details are listed:

- Singular Label:** Booking Created
- Plural Label:** Booking Createds
- Object Name:** Booking\_Created
- API Name:** Booking\_Created\_\_e
- Event Type:** High Volume
- Publish Behavior:** Publish Immediately
- Created By:** Shaik Yasmin, 9/26/2025, 1:26 AM
- Description:** Deployment Status: Deployed
- Modified By:** Shaik Yasmin, 9/26/2025, 1:26 AM

Below the main details, there are sections for 'Standard Fields', 'Custom Fields & Relationships', 'Triggers', and 'Subscriptions'.

Action	Field Label	Field Name	Data Type	Controlling Field	Indexed
	Created By	CreatedBy	Lookup(User)		
	Created Date	CreatedDate	Date/Time		
	Event UUID	EventUuid	Text(36)		
	Replay ID	ReplayId	External Lookup		

## 1. Customer Events

### Purpose:

Notify when a customer creates/updates a booking.

### Implementation:

Created Customer\_Booking\_\_e event.

Published when customer booking is confirmed.

External system (or Salesforce flow) can subscribe to trigger notifications.

### Why Important:

Provides real-time booking updates to other systems.

Improves customer experience with instant confirmation.

The screenshot shows the 'Platform Event Definition Detail' for 'CustomerName'. Key details include:

- Singular Label:** CustomerName
- Plural Label:** CustomerNames
- Object Name:** CustomerName
- API Name:** CustomerName\_\_e
- Event Type:** High Volume
- Publish Behavior:** Publish Immediately
- Created By:** Shaik Yasmin, 9/26/2025, 1:27 AM
- Description:** Deployment Status Deployed
- Standard Fields:** CreatedBy, CreatedDate, EventUuid, ReplayId
- Custom Fields & Relationships:** None defined.

## 2. Hotel Events

### Purpose:

- Notify when hotel availability or booking status changes.

### Implementation:

- Created Hotel\_Update\_\_e event.
- Published when hotel confirms/cancels a room.
- External listeners (fraud check, analytics, or partner systems) get notified.

### Why Important:

- Keeps hotel system and Salesforce synced.
- Enables instant alerts for room allocation, cancellations, etc.

The screenshot shows the 'Platform Event Definition Detail' for 'HotelName'. Key details include:

- Singular Label:** HotelName
- Plural Label:** HotelNames
- Object Name:** HotelName
- API Name:** HotelName\_\_e
- Event Type:** High Volume
- Publish Behavior:** Publish Immediately
- Created By:** Shaik Yasmin, 9/26/2025, 1:27 AM
- Description:** Deployment Status Deployed
- Standard Fields:** CreatedBy, CreatedDate, EventUuid, ReplayId
- Custom Fields & Relationships:** None defined.

### 3. Room Events

#### Purpose:

- Notify when **room status changes** (booked, available, cleaned).

#### Implementation:

- Created Room\_Status\_\_e event.
- Published when a room is **booked, released, or updated**.

#### Why Important:

- Helps maintain **accurate availability in real-time**.
- Supports **hotel operations** like cleaning, check-in, and check-out.

The screenshot shows the Salesforce Platform Events setup interface. At the top, there's a navigation bar with a gear icon labeled "SETUP" and the title "Platform Events". Below the header, the page title is "Platform Event RoomType". There are two links at the top right: "Standard Fields (4)" and "Custom Fields & Relationships (0)".

The main section is titled "Platform Event Definition Detail". It contains the following information:

Singular Label	RoomType	Description
Plural Label	RoomTypes	Deployment Status Deployed
Object Name	RoomType	
API Name	RoomType_e	
Event Type	High Volume <span style="color: blue;">i</span>	
Publish Behavior	Publish Immediately <span style="color: blue;">i</span>	
Created By	Shaik Yasmin, 9/26/2025, 1:28 AM	Modified By Shaik Yasmin, 9/26/2025, 1:28 AM

Below this, there's a section titled "Standard Fields" with a table:

Action	Field Label	Field Name	Data Type	Controlling Field
<u>Created By</u>	CreatedBy	Lookup(User)		
<u>Created Date</u>	CreatedDate	Date/Time		
<u>Event UUID</u>	EventUuid	Text(36)		
<u>Replay ID</u>	ReplayId	External Lookup		

At the bottom, there's a section titled "Custom Fields & Relationships" with a "New" button and the message "No custom fields defined".

```
1 public with sharing class HotelBookingEventPublisher {
2
3     public static void publishBookingEvent(String bookingId, String customerName, String hotelName) {
4         // Create Platform Event instance
5         BookingCreated__e bookingEvent = new BookingCreated__e(
6             BookingId__c = bookingId,
7             CustomerName__c = customerName,
8             HotelName__c = hotelName
9         );
10
11        // Publish the event
12        Database.SaveResult sr = EventBus.publish(bookingEvent);
13    }
14 }
15
```

## 6. Salesforce Connect (Not Used)

### Purpose:

- Connects external databases (Oracle, SAP, AWS) as virtual objects in Salesforce without storing data locally.

### Reason for Skipping:

- The hotel API provides **API access only**, not a database connection.

---

## 7. API Limits

### Purpose:

- Salesforce imposes limits to ensure system performance.

### Important Limits:

- REST API calls per 24 hours → based on license (e.g., 15,000 for Enterprise Edition).
- Callout timeout → 120 seconds max.

- Concurrent callouts → up to 10 per Apex transaction.

**Why Important:**

- Helps design integrations without exceeding limits.
  - Ensures API quota is not exhausted while fetching hotel data.
- 

## 8. OAuth & Authentication (Not Used – No Auth)

**Purpose:**

- Standard method for connecting Salesforce to third-party services.

**Reason for Skipping:**

- Hotel API is a demo and does not require credentials.
- Chose **No Authentication** in Named Credentials.

Note: In real projects, APIs usually require **OAuth2 or Username-Password** for security.

---

## 9. Remote Site Settings (Used)

**Purpose:**

- Allow Salesforce to call external services securely.

**Steps Followed:**

- Using **Named Credentials**, Remote Site Settings are automatically handled.
- If calling the API endpoint directly, you must manually add the **base URL** in **Setup → Remote Site Settings**.

**Why Important:**

- Prevents unauthorized external callouts.
- Adds a layer of security by whitelisting allowed domains