# Hospital Appointment Management System - Phase 7: Integration & External Access

## Overview

In Phase 7, we implement integration with external systems to automate sending SMS notifications to patients when their appointments are confirmed. Key concepts implemented in this phase:

* Remote Site Settings (Required)
* Apex Callouts (POST request to SMS API)
* Trigger automation on Appointment confirmation
* Debugging and testing callouts

## 1️⃣ Remote Site Settings

**Use Case:** Allow Salesforce to make HTTP callouts to external API.

**Steps:** 1. Go to Setup → Security → Remote Site Settings. 2. Click New Remote Site. 3. Fill in: - Remote Site Name: Hospital\_SMS\_API - Remote Site URL: https://example-sms-api.com - Description: Used to send SMS notifications to patients 4. Click Save.

## 2️⃣ Apex Callout Class – AppointmentSMSHandler

**Use Case:** Send SMS notifications to patients automatically.

**Steps:** 1. Developer Console → File → New → Apex Class → Name: AppointmentSMSHandler 2. Paste the following code:

public with sharing class AppointmentSMSHandler {  
  
 @future(callout=true)  
 public static void sendSMS(String patientPhone, String message) {  
 HttpRequest req = new HttpRequest();  
 req.setEndpoint('https://example-sms-api.com/send');  
 req.setMethod('POST');  
 req.setHeader('Content-Type', 'application/json');  
  
 Map<String, String> body = new Map<String, String>();  
 body.put('phone', patientPhone);  
 body.put('message', message);  
 req.setBody(JSON.serialize(body));  
  
 Http http = new Http();  
 try {  
 HttpResponse res = http.send(req);  
 System.debug('SMS Response: ' + res.getBody());  
 } catch(Exception e) {  
 System.debug('Error sending SMS: ' + e.getMessage());  
 }  
 }  
}

## 3️⃣ Apex Trigger – AppointmentTrigger

**Use Case:** Automatically call the SMS handler when an appointment is confirmed.

**Steps:** 1. Setup → Object Manager → Appointment → Triggers → New → Name: AppointmentTrigger 2. Paste code:

trigger AppointmentTrigger on Appointment\_\_c (after update) {  
  
 for(Appointment\_\_c app : Trigger.new) {  
 Appointment\_\_c oldApp = Trigger.oldMap.get(app.Id);  
 if(app.Status\_\_c == 'Scheduled' && oldApp.Status\_\_c != 'Scheduled') {  
 String phone = app.Patient\_\_r.Contact\_Number\_\_c;  
 String message = 'Your appointment on ' + String.valueOf(app.Appointment\_Date\_\_c) +  
 ' at ' + String.valueOf(app.Appointment\_Time\_\_c) + ' is confirmed.';  
 AppointmentSMSHandler.sendSMS(phone, message);  
 }  
 }  
}

## 4️⃣ Testing the Apex Trigger and Callout

**Steps:** 1. Create a test contact with phone number. 2. Create a test appointment linked to this contact with Status = Requested. 3. Update appointment Status → Scheduled. 4. Setup → Debug Logs → View → Look for ‘SMS Response:’ 5. Check for errors ‘Error sending SMS:’ if any. 6. Optional: Developer Console → Execute Anonymous to simulate status change.

## 5️⃣ Apex Test Class

**Use Case:** Provide Salesforce test coverage.

@isTest  
private class TestAppointmentSMS {  
  
 @isTest static void testSendSMS() {  
 Contact c = new Contact(FirstName='Test', LastName='Patient', Contact\_Number\_\_c='9999999999');  
 insert c;  
  
 Appointment\_\_c app = new Appointment\_\_c(  
 Patient\_\_c = c.Id,  
 Appointment\_Date\_\_c = Date.today(),  
 Appointment\_Time\_\_c = Time.now(),  
 Status\_\_c = 'Requested'  
 );  
 insert app;  
  
 Test.startTest();  
 app.Status\_\_c = 'Scheduled';  
 update app;  
 Test.stopTest();  
 }  
}