

CASE STUDY

Integrating Salesforce

with Slack for Improved Incident Management

Author:

Marko Balant

Salesforce Engineer, Comtrade 360

At a glance:

The integration of Salesforce with Slack is a pivotal strategy for enhancing operational efficiency. We delve into our innovative approach to solving the challenge of real-time incident notification and management within Slack, bypassing the need for expensive, feature-heavy solutions that exceed the requirements of large organizations. By developing a bespoke integration that not only notifies but also periodically compiles and shares incident reports through Slack, we've showcased our ability to provide scalable, cost-effective solutions tailored to our client's specific needs.

Industry

Customer Relationship Management (CRM)

Technology

- Salesforce
- Slack
- Apex
- Webhooks for Real-Time Data Transmission
- OAuth & Permissions Configuration for Slack

Introduction

As a provider of Salesforce Administration services, we understand the critical need for seamless integrations. We often see organizations seeking solutions to alerts, reports, or notifications from Salesforce within Slack or Teams, aiming to avoid unnecessary expenditure on features they don't need. However, existing integration options are often part of larger product suites or solutions that require the purchase of costly licenses dependent on the number of Salesforce users. Especially for large and growing entities with numerous platform users, this quickly becomes an expensive, unnecessary investment in unused features, all to access one straightforward feature.

This leaves many organizations grappling with the challenge of finding an affordable, scalable solution that provides streamlined Salesforce integration with Slack. Additionally, they hope to create an automation that compiles active incidents in a spreadsheet and sends them via Slack at periodic intervals.

Identifying the Problem

We recently collaborated with a client seeking to streamline their incident management process by receiving real-time notifications in Slack when new incidents were generated in Salesforce. Additionally, they aimed to automate the periodic dissemination of active incident reports to maintain oversight and facilitate delegation.

The Challenge

The initial step was to explore potential providers who could offer a solution tailored to our clients' needs. While our analysis outlined that some providers did offer solutions that met and even exceeded the customer's requirements, we determined that the cost, totaling close to \$56,000 annually, was too expensive given they needed just one feature. We felt that we could explore a better solution to meet the customer's request in a more economical way. The customer also agreed that these market solutions weren't the right option either.

The Solution

1. Send Salesforce Incident Details Notification to Slack

To fulfill the client's need for incident alerts, we utilized webhooks to integrate Salesforce with Slack. This allowed us to correctly configure Slack channels and implement Apex logic in Salesforce. Next, we began configuration and implementation on the Salesforce side. We configured a Remote Site to enable access to Slack from Salesforce.

CASE STUDY

Following that, we initiated the implementation of Apex logic to send Incident Details to the Slack channel. With these configurations in place, an incident generated in Salesforce would trigger the creation of a new Slack channel, including all involved engineers. A notification including the incident information, like incident status, description, and recommended action, is sent to the channel, allowing involved agents the ability to discuss the incident management in a centralized location, with real-time, updated data.

Here's an example of the delivered message:

ApexTest - December 22nd at 8:17 AM

New Incident Registered in Salesforce

- Incident Status: Open
- Incident #: *****
- Correspondent: *****
- Description: Issue with connection, Code: A453, message: transfer has failed
- Number of affected transfers: 6
- API Enabled?: N/A
- Status of Transaction: API Failed
- Processing Time: Within 10 minutes
- Service: *****
- Receive Country: *****
- CC Actions: Can change and/or cancel (Follow Info Icon)
- What to advise: Please get in contact with Tech Team
- Latest Update: 22 Oct 7:16 AM BST - notified agents, awaiting response

While this feature proved invaluable for on-call engineers, it proved difficult for management who didn't have Salesforce access, and consequently, didn't have a complete overview of incidents and the number of items being handled at any time. To address this gap in information for management, we developed an additional feature that sent a copy of the incident reports to email, providing management with a quick overview of incidents without being added to relevant engineer Slack channels or managing individual changes in Salesforce.

The configuration for this feature can be seen below:

```
global class BatchJobSendSlackMessage implements Database.Batchable<SObject>, Database.AllowsCallouts
{
    ...
    public Set<Id> IncidentIds;
    public String IncidentStatus;
    Operations_Update__c oldIncident;

    global BatchJobSendSlackMessage(Set<Id> Incident_Ids, String status_Incident, Operations_Update__c Incident_old)
    {
        IncidentIds=Incident_Ids;
        IncidentStatus= status_Incident;
        oldIncident=Incident_old;
        System.debug('Incident Ids: '+ IncidentIds);
    }

    global Database.QueryLocator start(Database.BatchableContext bc )
    {
        // get the list on new incidents
        String newLstIncident = 'Select id,Name, Incident_Status__c,Error_Code__c, Incident_Number__c, Correspondent__c,
        return Database.getQueryLocator(newLstIncident);
    }

    public class IncidentBody
    {
        public String IncidentNumber;
        public String IncidentStatus;
    }
}
```

2. Scheduled Incident Report Distribution

To address the client's request for comprehensive incident updates, we developed a feature to generate and distribute periodic incident reports via Slack. Using Apex logic, **we automated the creation of a spreadsheet containing detailed incident summaries**, including all opened and active incidents.

```

.....public static String generateCsvFile(String fileName, List<List<String>> data)
.....{
.....    String orgUrl = URL.getSalesforceBaseUrl().toExternalForm();
.....    String csvFileBody = '';
.....    //Build the CSV content
.....    for (List<String> row : data) {
.....        csvFileBody += String.join(row, ',') + '\n';
.....    }

.....    //Create a Document record
.....    Document doc = new Document();
.....    doc.Name = fileName;
.....    doc.IsPublic = true;

```

We also configured Slack's 'OAuth & Permissions' settings to enable file sharing. Once the correct permissions were granted on the Slack side, **the next step was to create and send the message** containing the incident details to the Slack channel.

```

.....//This method will pick excel spreadsheet file on SF and send it to Slack Chanel
.....global void execute(Database.BatchableContext bc, List<Document> documentList)
.....{
.....    String orgUrl = URL.getSalesforceBaseUrl().toExternalForm();
.....    for (Document myDoc : documentList) {
.....        if (myDoc != null && myDoc.IsPublic && myDoc.ContentType == 'text/csv') {
.....            //Construct the file URL
.....            String fileUrl = URL.getSalesforceBaseUrl().toExternalForm() + '/' + myDoc.Id;

.....            //Prepare the message for Slack
.....            String initialCommentSf = 'Salesforce Excel/CSV file: ';
.....            String title = 'IncidentStatus.csv';

```

We were able to create a system to create a new message, retrieve and attach the generated incident file to the message, and then send it to the Slack channel. We scheduled this message to be sent four times per day to the Slack channel, providing a regular, comprehensive update available for both management and agents.

Friday, April 5th

ApexTest APP 12:05 PM
Download Excel/CSV file directly from Slack or download it from Salesforce link
IncidentStatus.csv

Incident Number	Incident Status	Correspondent	Created Date	Error Code	Transactions Affected	API Enabled	Status of Transaction	Standard Processing Time	Service	Receive Country	Affected Se
AP_24_000000	Open	Marko	2023-11-24 13:28:41	N/A			N/A	N/A	N/A	N/A	
AP_24_000001	Open	Marko	2023-11-17 12:07:17		1		API Cancelled	Within 10 minutes	Airtime Topup	N/A	Do Not Acti
AP_24_000002	Open	Marko	2023-09-15 16:27:26	test marko sf admin	34234234		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000003	Open	Marko	2023-08-07 11:06:55	MARKO BALANT TEST SF ADMIN	234234234		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000004	Open	Marko	2023-08-01 13:21:43	MARKO BALANT TEST SF ADMIN	23423423423		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000005	Open	Marko	2023-08-01 13:18:16	MARKO BALANT TEST SF ADMIN	423423423		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000006	Open	Marko	2023-08-01 13:15:59	MARKO BALANT TEST SF ADMIN	32423423		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000007	Open	Marko	2023-08-01 13:13:09	MARKO BALANT TEST SF ADMIN	432423432		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000008	Open	Marko	2023-08-01 12:54:44	TEST MARKO BALANT SF ADMIN	3423423423		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000009	Open	Marko	2023-08-01 12:44:17	TEST MARKO BALANT SF ADMIN	24234234234		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000010	Open	Marko	2023-07-25 14:41:18	TEST SF ADMIN MARKO	TEST		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000011	Open	Marko	2023-07-25 13:50:14	TEST MARKO BALANT SF ADMIN	423423423		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000012	Open	Marko	2023-07-25 12:25:56	TEST SF ADMIN MARKO	213213213213		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000013	Open	Marko	2023-05-02 15:29:42	Code: CreateRemitterFalle..	1		API Failed				
AP_24_000014	Open	Marko	2023-04-27 08:15:02	TEST SALESFORCE ADMIN MARKO	TEST SALESFORCE ADMIN MARKO		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000015	Open	Marko	2023-04-26 12:41:48	TEST MARKO BALANT MULTIPL..			N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000016	Ongoing Issues	Marko	2023-04-26 11:49:50	TEST MARKO BALANT SF ADMIN	TEST MARKO BALANT SF ADMIN		N/A	Within 3 working days	N/A	N/A	N/A
AP_24_000017	Open	Marko	2023-04-26 10:29:07		423432432		API Failed	Within 3 working days	Airtime Topup	N/A	N/A

This snippet was truncated for display; see it in full

Results and Outcomes

The implementation of custom automation yielded significant improvements in operational efficiency for our client’s organization. By integrating their Salesforce incident notifications into Slack, agents experienced enhanced time management, resulting in heightened productivity. Previously, the absence of such automation often led to oversight of old open incidents, contributing to inefficiencies within the system.

With the new automation framework in place, management gained invaluable insight into daily incident activities, empowering them to exercise effective task delegation, even for long-standing or neglected incidents. This comprehensive oversight facilitated proactive measures in addressing issues, ensuring a streamlined workflow and timely resolution of incidents.

Conclusion

This case study demonstrates the necessity of customized solutions for organizations dealing with unique operational hurdles. While off-the-shelf options are available, cost concerns often deter their adoption. By introducing custom automation, our client not only addressed these challenges but also saved over \$56,000 annually compared to standardized solutions. This success highlights the practical benefits of tailored approaches in improving operational efficiency and cost-effectiveness. In essence, our case study illustrates how personalized solutions effectively meet specific business needs in a straightforward and economical way.



info@comtrade360.com
+1 617-546-7400
comtrade360.com