A blue and black logo

Description automatically generated

Payment Collection Solution

Carl Newton - Solutions Architect

Cisco Systems

January 2024

Table of Contents

[Solution Overview 2](#_Toc155716677)

[Reduce Agent & Supervisor Effort 2](#_Toc155716678)

[Increase Customer Satisfaction Levels 2](#_Toc155716679)

[Create Data To Drive Outbound Campaigns 3](#_Toc155716680)

[Pre-Requisites 3](#_Toc155716681)

[Implementation 3](#_Toc155716682)

[Change Log 3](#_Toc155716683)

# Solution Goals

This document outlines the solution for a payment collection flow within Webex. Some customers currently use an outbound dialler to make outbound calls to clients and make payment demands. The efficiency of these solutions can be improved due to low call answer rates by the consumer who needs to pay, and high effort requirements for the Agents and supervisors to set up and execute the campaigns.

This solution has three aims:

* Reduce Agent & Supervisor effort by completely, or partially automating some of the collections
* Increase customer satisfaction levels by offering the ability to pay at a convenient time
* Create data about which customers have not paid to drive later outbound calling campaigns
* Inject information into the JDS history stream to assist agents in better handling customer interactions

## Reduce Agent & Supervisor Effort

This is achieved by kicking off a Webex connect flow that can be invoked by a webhook. This flow will request payment from the customer by SMS, giving them the opportunity to pay with no interaction from any agent.

In the case that the agent is on a voice call with the customer, they can initiate a payment request in real time, sending an SMS to the customer with a payment link. The payment status is then updated on the screen of the agent once successful.

## Increase Customer Satisfaction Levels

Many people today do not answer calls from numbers they do not recognise. When they do answer, it may not be a good time for them to complete a payment. Being asked to complete a payment while on the phone to an agent at a bad time can greatly increase stress levels in what is already a stressful situation for the consumer. Ignoring those calls and later receiving final demands elevates stress even further.

Using SMS to drive payments allows people to pay when it is convenient for them. They are made aware of the demand in a non time sensitive manner, and can choose to pay at a more convenient time. This lowers consumer stress levels.

## Create Data To Drive Outbound Campaigns

If the consumer chooses not to pay at this time, the flow will add them to a list with the relevant details. This can then be used to drive a subsequent outbound calling campaign, or to kick off a number of other processes depending on the tools in use by the customer.

## JDS Event Injection

These flows also utilise the injection of JDS events into the history stream for a given user. This results in the agent being able to see payment demands being sent to a user, what their responses were, and any payments that have been made directly within the agent desktop.

# Pre-Requisites

The following pre-requisites should be in place before attempting configuration

* An SMS asset in your Webex connect tenant capable of sending and receiving messages. [Details on Customer Engagement Portal](https://cisco.sharepoint.com/sites/GlobalCollaborationSalesEngineering/SitePages/Customer-Engagement-Field-Newsletter.aspx)
* [Postman](https://www.postman.com/downloads/) set up to initiate the flow
* Understanding of OAuth for Webex API executions
* Mockapi.io account set up to receive contacts for the subsequent outbound calling campaign
* JDS Enabled WxCC Tenant (US Available, EU Roadmapped mid 2024)
* JDS capable WxCC Agent Desktop Layout that supports inbound/outbound calling
* Familiarity with [glitch.com](https://glitch.com/)

# Implementation

## Required Info

This solution requires a few variables to be populated across different systems. The table below can be used to store this info. Note the names of the variable may deviate slightly across systems, so multiple names are included to assist identification. **DO NOT SHARE ANY OF THE FOLLOWING INFORMATION WITH ANYONE, ENSURE IT IS STORED SECURELY.**

|  |  |  |
| --- | --- | --- |
| Variable Name(s) | Variable Value | Relevant Section |
| Client ID  Client\_id.JDS | C12eb15a284c4ebc336049c41f263b105235942f32e4ec236ceaf5b5300b18c33 | From [Create Oauth Section](#_Create_OAuth_Integration) |
| Client Secret  Client\_sectet.JDS | 756174ef119c93e50adfca9c1f23c9f6919cb1a84f97944b866562d5ec12ebfd | From [Create Oauth Section](#_Create_OAuth_Integration) |
| OAuth Auth URL | https://webexapis.com/v1/authorize?client\_id=C12eb15a284c4ebc336049c41f263b105235942f32e4ec236ceaf5b5300b18c33&response\_type=code&redirect\_uri=https%3A%2F%2Fadmin.webex.com%2F&scope=cjp%3Auser%20spark%3Apeople\_read%20cjp%3Aconfig\_write%20cjp%3Aconfig%20cjp%3Aconfig\_read%20cjds%3Aadmin\_org\_read%20cjds%3Aadmin\_org\_write&state=set\_state\_here | From [Create Oauth Section](#_Create_OAuth_Integration) |
| Redirect URI | https://admin.webex.com/ | From [Create Oauth Section](#_Create_OAuth_Integration) |
| Project ID  workspaceid | 64da3f9d21a4034e04bc54c8 | From [Install Postman Section](#_Install_Postman_Collection) |
| Org ID  Orgid.JDS | 77dbece8-9e08-4e32-acd2-2f1a89c50d9a | From [Install Postman Section](#_Install_Postman_Collection) |

## Create OAuth Integration

An OAuth token is required to perform API operations for JDS event injection and other functionalities.

Important Note: OAuth tokens created may only last for a single day. Be sure to re-populate your token in your flows right before your customer demo to avoid failures. This will be described in a later portion of this document. Alternatively, set up a [token store](https://github.com/WebexSamples/webex-contact-center-api-samples/tree/main/token-management-samples/token-service-sample) using the [tutorial video](https://app.vidcast.io/share/ed971770-49bb-47e5-96d0-7c920074fd53) to automatically manage the tokens from flows.

1. Go to <https://developer.webex-cx.com/> and log in using an administrator account.
2. Click “My Webex Apps” from the sub menu in the top right, then select “Create a new app”  
   A screenshot of a computer

   Description automatically generated
3. Configure your integration as follows. **Any fields marked in red must not be changed from the below values.**

|  |  |
| --- | --- |
| Field | Value |
| Integration Name | PaymentDemand |
| Description | Integration to support SMS payment demand from Connect |
| Redirect URI | https://admin.webex.com/ |
| Scopes | Select All |

1. You will be presented with a screen of information. Go to the [required information](#_Required_Info) table and fill out the relevant fields.

## Install & Configure Postman Collection

The postman collection has several functionalities. Download it from the [Git Repo](https://github.com/canewton-ccep/PaymentCollections). This collection uses JDS API samples by Arunabh located [here](https://github.com/WebexSamples/webex-contact-center-api-samples/blob/main/customer-journey-samples/cjds-postman-example/JDS%20CiscoLive.postman_collection.json)

1. Open Postman, choose File -> Import
2. Load the file “CCEP Payment Collection.postman\_collection.json”
3. Log into control hub. Select the “Customer Journey Data” Tab, and copy your project ID. Populate it, and your Org ID in the [Required Info Table](#_Required_Info).  
   A screenshot of a computer

   Description automatically generated
4. Go back to Postman. Select the collection you just imported, and go to the variables screen. Populate all of the fields using info from the [Required Info Table](#_Required_Info). except for MockAPIID, baseURL & identity, these will be used later. **SAVE THE COLLECTION**A screenshot of a computer

   Description automatically generated
5. Click on the Authorization tab. Scroll down to the “Configure new token” area. The fields should be pre-populated to reference the variables you just set up.   
   Scroll to the bottom and select “Get New Access Token”. You will be asked to authenticate, and then accept the integration.  
   A screenshot of a computer

   Description automatically generatedA screenshot of a computer

   Description automatically generated
6. On success, You will be presented with the token details. Select “Use this token”. Feel free to close this screen for now. If the token fails, check the postman log to see why and re-try. Clearing cookies before choosing “Get new access token” can help.
7. Open the “JDS” Section of the collection, and open the “Get History Stream” request. Now just click “Send” and you should receive a 200OK and get some kind of data. Do not proceed until you have a 200OK.

A screenshot of a computer

Description automatically generated

## Create MockAPI DB

## Create Glitch Site

## Modify Agent Desktop Layout

## Set Up Flows

# Change Log