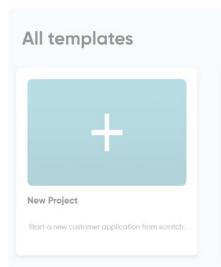


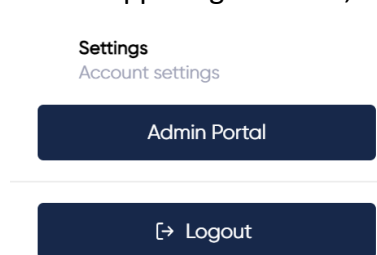
Setting up a Webex visual IVR using the CallVU studio

Follow the guidelines below to setup a visual IVR using the Webex Flow Designer and the CallVU studio. Using CallVU, your Webex visual IVR will have the power to send visual screens and play voice messages in parallel during the flow.

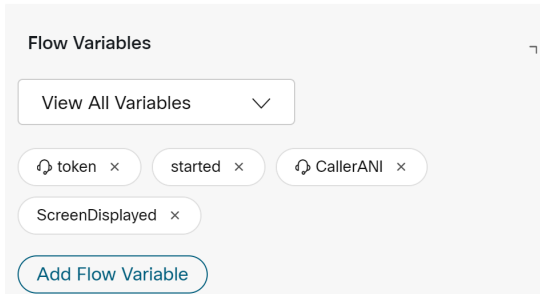
1. Create your CallVU tenant – browse to <https://studio.ficx.app/> and create your tenant
2. Enter your tenant’s studio by browsing to <https://studio.ficx.app/callvu-studio/>
3. In the studio, click the New Project button and create a Micro App



4. Setup your micro app flow (journey) as explained in <https://ficx.gitbook.io/ficx-studio>
5. On the upper right corner, click on the down arrow and enter Settings

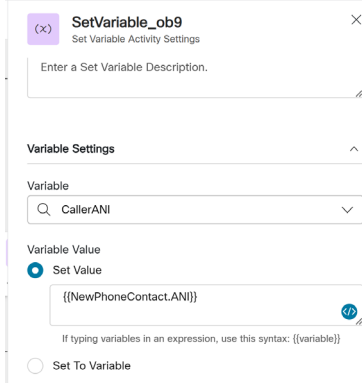


6. Copy the values of the Organization ID and the Access Token
7. Create a connector in the Webex portal
 - a. Log in to your customer organization at <https://admin.webex.com> and navigate to **Services > Contact Center > Integrations**
 - b. Add a Custom Connector
 - i. Assign a name to it, for example, CallVUConnectorv1
 - ii. Select Basic Authentication
 - iii. Set the Resource Domain to: <https://studio.ficx.app/VIVR>
 - iv. In the User Name paste the Organization ID copied in step 5
 - v. In the Password paste the Access Token copied in step 5
 - vi. Set the validation URL to <https://studio.ficx.app>
 - vii. Activate the connector
8. To create a flow using the Webex Flow Designer and initiate your tenant’s visual journey created by the CallVU Studio, in the Webex Flow Designer start a new flow and set the following string variables

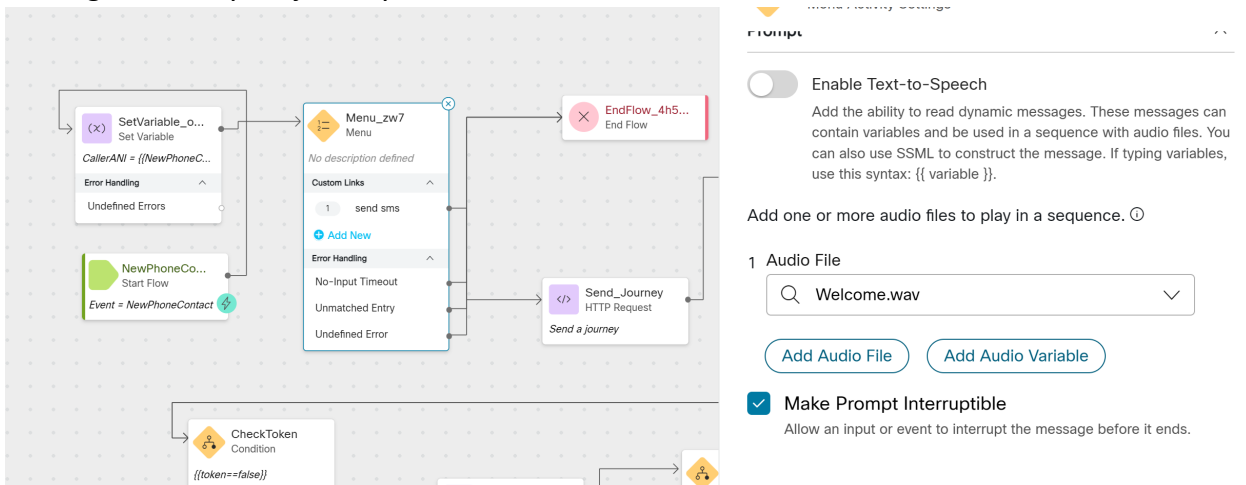


Set the default values of token and started to false

9. Assign the variable CallerANI with the value of {{NewPhoneContact.ANI}}



10. Add a menu block with a voice prompt inviting customers to receive a text message to start the visual journey, for example, Welcome.wav will say “to easily complete your transaction, press 1 and we will send you a text message. Click on the link in the message to start your journey”



11. In this case, if the caller presses 1 the flow will advance to the Send_Journey HTTP Request block. In this block an API request to SetDynamicDisplayJson will be made using the connector that will send an SMS to the calling number and initiate the visual IVR experience.

Send_Journey HTTP Request Activity Settings ✕

HTTP Request Settings ^

Use Authenticated Endpoint

Connector ⊙

CallVUConnector1 ▼

Request Path ⊙

/SetDynamicDisplayJson

Method

GET ▼

Query Parameters

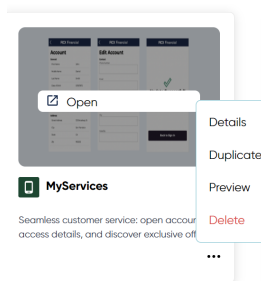
12. Use the following parameters when calling the initial SetDynamicDisplayJson API are:

Key	Value	
Folder	folder	🗑️
CallPhase	Pre-IVR	🗑️
PhoneNumber	{{CallerANI}}	🗑️
File	start	🗑️
Token	starthtml5	🗑️
UrlSlug	24671071-EE38-4F7C	🗑️

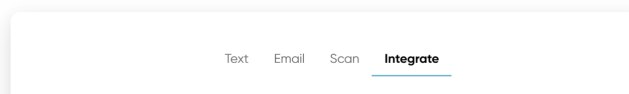
The UrlSlug value is the unique identifier of the Micro App created in the CallVU Studio.

In order to retrieve it, open the CallVU Studio and preview the Micro APP

- Click the 3 dots on the micro app's button and select Preview



- In the Preview screen, select the Integrate tab



Copy the UrlSlug value of the VIVR Integration URL. Clicking the button will copy the entire text in the black window.

```

https://studio.flex.app/VIVR/SetDynamicDisplayJson?
Dns=DSQWYATbr22H&TID=7d776e4b-840f-4b76-83ef-
821faed4d53d&ts=17151719655414&Token=starhtml5&CallPhase=Pre-
IVR&Folder=Folder&File=Start&PhoneNumber=PHONE-
NUMBER&access_token=ACCESS-TOKEN
* Replace PHONE-NUMBER with the ANI
** Replace ACCESS-TOKEN with the access_token value in the
Settings screen

```

VIVR Integration URL

- Use this UriSlug value in the Flow Designer's http request

13. Set the following for the response. The token variable will get a unique session identifier in case the visual session has successfully started, otherwise, it will get false.

Send_Journey
HTTP Request Activity Settings

Response Timeout milliseconds

Number of retries

Parse Settings

Content Type

Output Variable

Path Expression

14. Add a voice message saying the message has been sent

MsgSmsSent
Play Message
No description defined
Error Handling

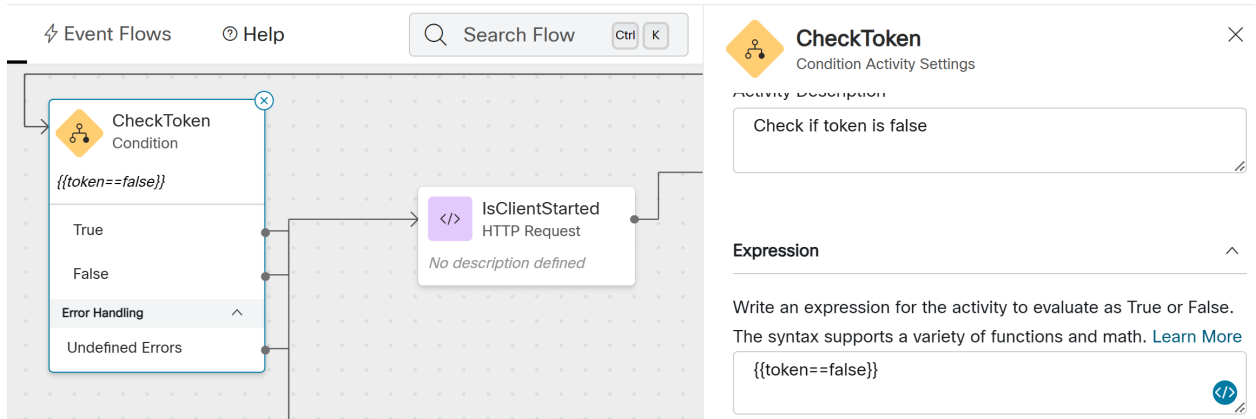
Prompt

Enable Text-to-Speech
Add the ability to read dynamic messages. These messages can contain variables and be used in a sequence with audio files. If typing variables, use this syntax: {{ variable }}. You can also use SSML to construct the message. If using SSML, insert it inside the < speak > < / speak > tags.

Add one or more audio files to play in a sequence.

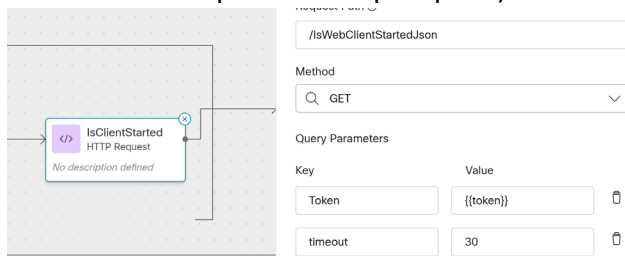
1 Audio File

15. Add a condition to check if the token is false

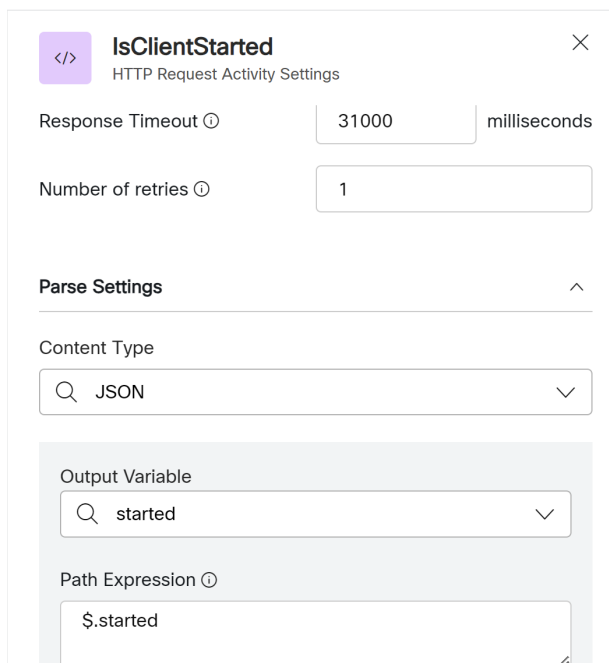


If the token is false, you may continue as a voice only IVR flow, otherwise make an http request to see if the caller has connected to the URL sent in the SMS and started the visual session

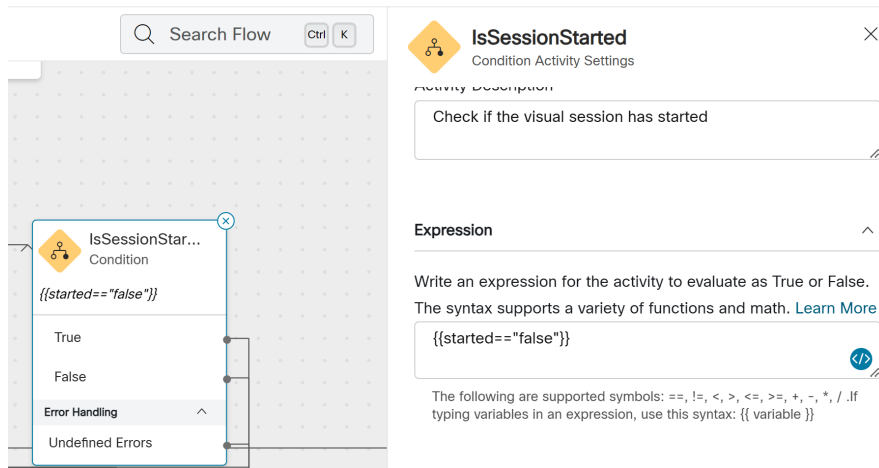
16. Calling the `IsWebClientStartedJson` with the following parameters (the token was retrieved in the previous http request)



Set the following parameters for the response



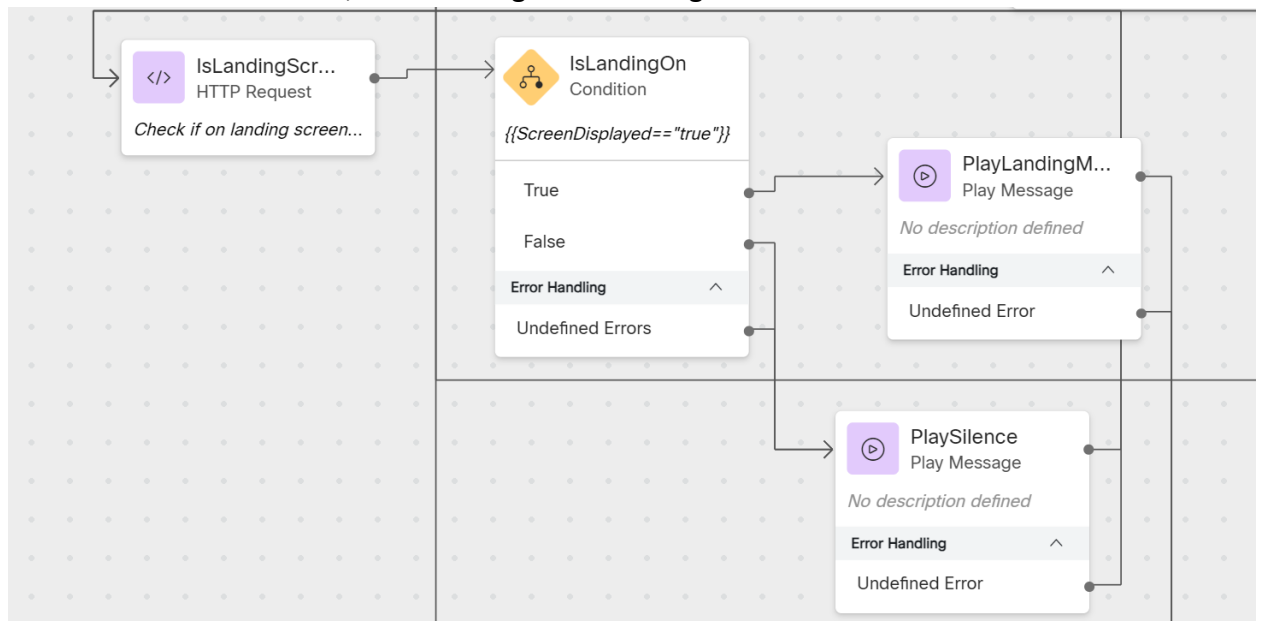
Then set a condition for the started variable



If the value is False, you may continue as a voice only IVR or disconnect the call. In case the value is True, it means the visual interaction is on and the caller already sees the first visual screen.

Playing voice prompts when displaying screens (optional)

Using Webex and the CallIVU platform, you can play voice prompts while displaying screens within the VIVR interaction. For example, while displaying a screen that collects your address details, you can play a message like “please fill in your updated address details”. In order to do so, we are using the following 4 blocks for each such screen:



In this example we check if the landing screen is displayed and play a message we have recorded in advance for that screen.

1. Make a request to IsEndFlowJson with the following request parameters

IsLandingScreenDispl...
HTTP Request Activity Settings

Connector

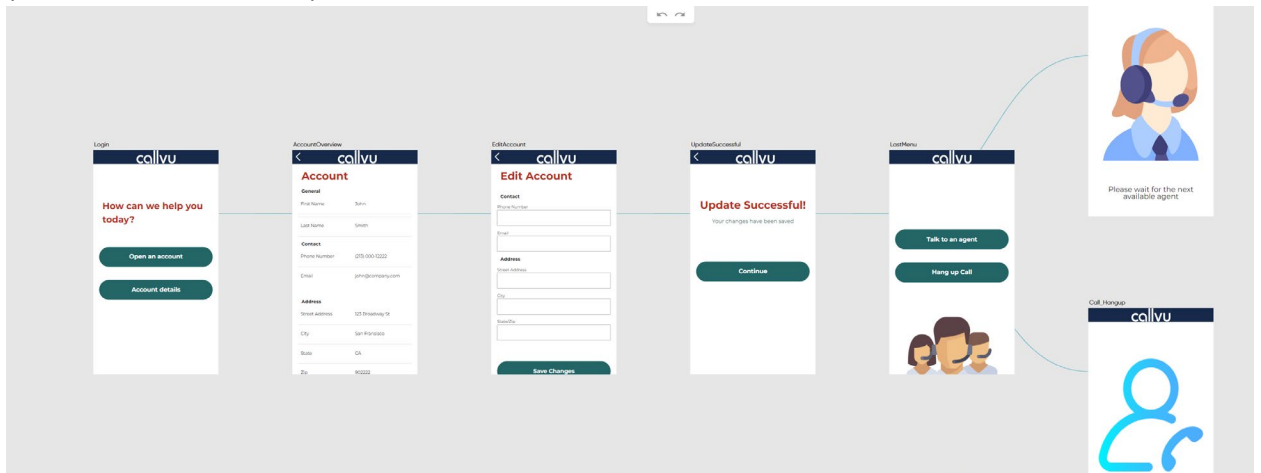
Request Path

Method

Query Parameters

Key	Value
Token	{{token}}
Screen	Login

The Login, is the screen's name in the visual Micro App created in the CallIVU Studio (first screen of the flow)



Set the response parameters as follows

Parse Settings

Content Type

Output Variable

Path Expression

2. Set a condition to check if the screen is displayed to the caller

IsLandingOn
Condition Activity Settings

General Settings

Activity Label
IsLandingOn

Activity Description
Check if the landing screen is displayed

Expression

Write an expression for the activity to evaluate as True or False.
The syntax supports a variety of functions and math. [Learn More](#)

```
{{ScreenDisplayed=="true"}}
```

3. If the screen is not displayed play a one second silence message and go back to step 1

PlaySilence
Play Message Activity Settings

Enable Text-to-Speech
Add the ability to read dynamic messages. These messages can contain variables and be used in a sequence with audio files. If typing variables, use this syntax: {{ variable }}. You can also use SSML to construct the message. If using SSML, insert it inside the <speak></speak> tags.

Add one or more audio files to play in a sequence.

1 Audio File
SilenceOneSecond.wav

[Add Audio File](#) [Add Audio Variable](#)

Activity Output Variables

4. If the screen is displayed, play the screen's voice message you have recorded and continue to the next screen

PlayLandingMsg
Play Message Activity Settings

Enter a Description

Prompt

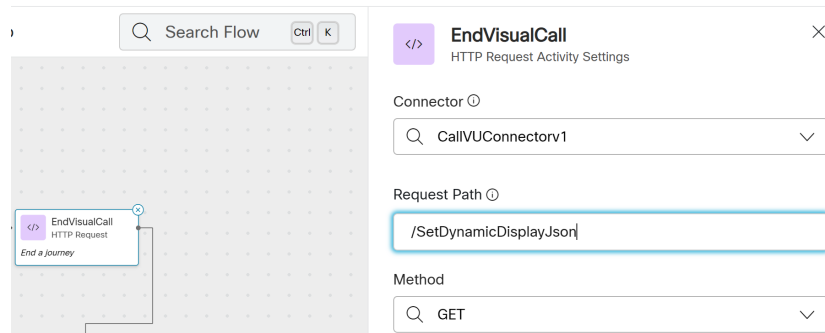
Enable Text-to-Speech
Add the ability to read dynamic messages. These messages can contain variables and be used in a sequence with audio files. If typing variables, use this syntax: {{ variable }}. You can also use SSML to construct the message. If using SSML, insert it inside the <speak></speak> tags.

Add one or more audio files to play in a sequence.

1 Audio File
Greetings.wav

Termination of the visual journey

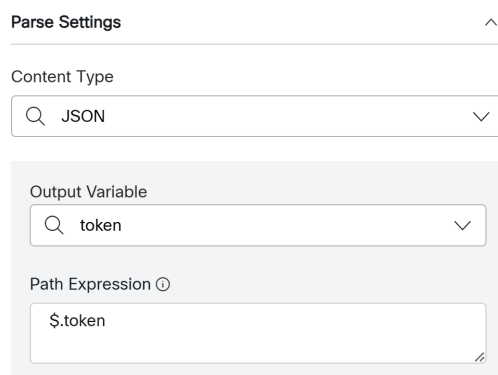
The visual journey can be terminated within an IVR flow or at the end of it, upon hang-up based on the business decision of the client that builds the IVR flow. In order to terminate the visual journey, an http request to SetDynamicDisplayJson with the following parameters should be called



Key	Value	
PhoneNumber	{{CallerANI}}	🗑️
File	!EndCall	🗑️
Token	{{token}}	🗑️
UriSlug	24671071-EE38-4F7C	🗑️

The UriSlug value is the one retrieved previously

The response is defined as follows



Download Sample VIVR Flow.json to see the implementation above.