

# Integrating Twilio with Mule 4

This article covers the basic integration and the various operations performed using Twilio by integrating it with MuleSoft.

Let's start with...

## **What is Twilio, and why do we use it?**

Communications have become a paramount aspect in today's world, whether notifying the customers about shipment status, providing support over the phone, or confirming the reservations by text. It is very important to be connected with the customers on the communication channels they use every day like calls, text messages, WhatsApp conversations, and many more. But, building it requires a mesh of thousands of carriers, global regulations, expensive hardware, and a plethora of other problems. However, with Twilio, all these problems get resolved under one roof.

Twilio provides the platform for communication using the communication APIs, where we can send or receive a text, place or receive a call or kick-off a conversation over chat, and perform many more communication functions!

## **How can we connect MuleSoft with Twilio?**

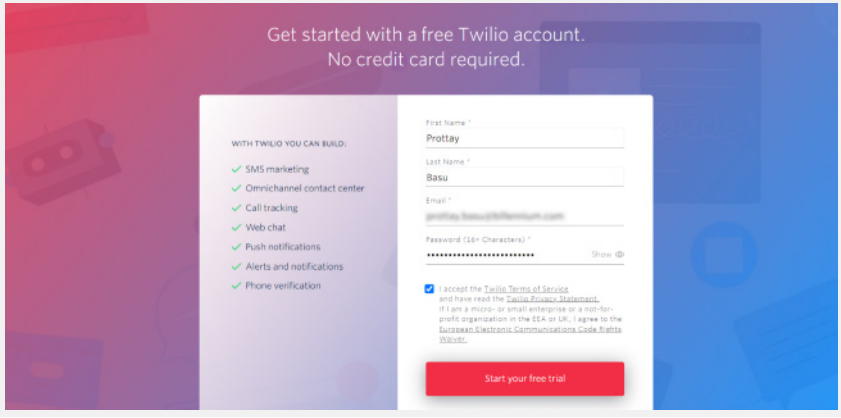
Anypoint Connector for Twilio (4.2) provides an interface to the Twilio REST APIs, enabling the users to create connectivity between Twilio and other platforms and serve as an intermediary for connecting to various communication channels. Some of the example operations that can be performed with the Twilio connector are:

- Create Message,
- Create Call,
- Fetch Account Balance,
- List Call,
- Fetch Message,
- and many more!

# Implementation

## → Step 1: Registering to Twilio

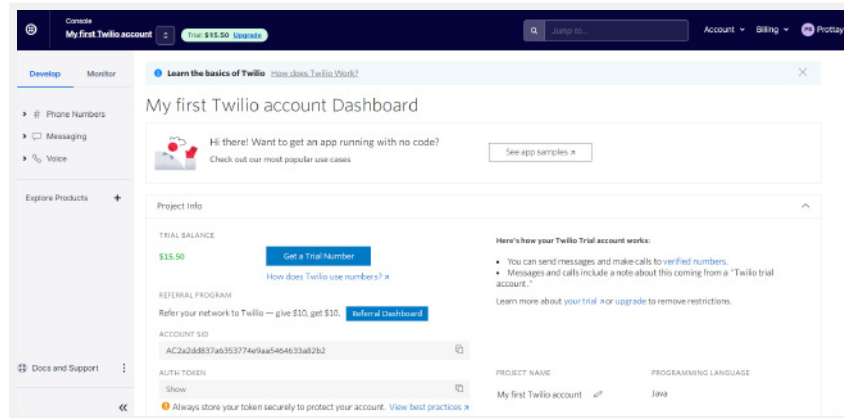
To create a Twilio account, please click on the URL  
<https://www.twilio.com/try-twilio>.

The image shows the Twilio registration page. At the top, it says "Get started with a free Twilio account. No credit card required." Below this, there's a section titled "WITH TWILIO YOU CAN BUILD:" followed by a list of services: SMS marketing, Omnichannel contact center, Call tracking, Web chat, Push notifications, Alerts and notifications, and Phone verification. To the right of this list is a registration form with fields for First Name (filled with "Protlay"), Last Name (filled with "Basu"), Email (filled with "protlay.basu@epam.com"), and Password (16+ Characters). There's a "Show" icon next to the password field. Below the password field is a checkbox that is checked, with text indicating acceptance of the Terms of Service and Privacy Policy. At the bottom of the form is a red button labeled "Start your free trial".

A verification mail has been sent to the registered email, and please verify your account.

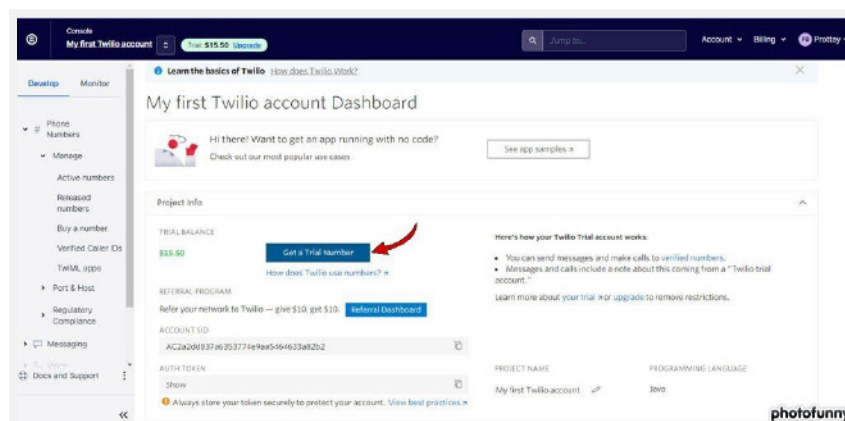
## → Step 2: Login

Login using the credentials you used while registering yourself on the Twilio platform. After login, you will be redirected to your console dashboard. This will be the home for finding Twilio credentials, checking usage, procuring a number, and more. A small preloaded balance is also given to test out Twilio's functionality.



## → Step 3: Getting Trial Number

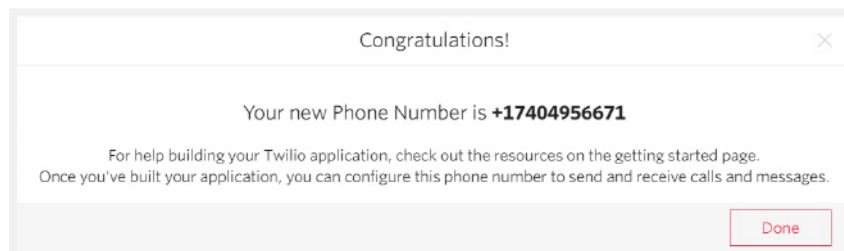
- a) Click on the **Get a Trial Number Button**. This number will be used to send messages, make calls or do other communication activities.



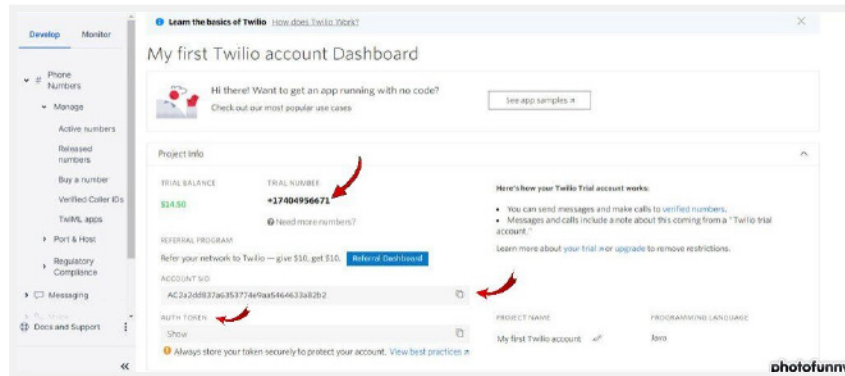
- b) A dialog box opens with the details of the number.  
Click on **Choose this Number**.



- c) A dialog box appears with the new phone number. Click on **Done**.

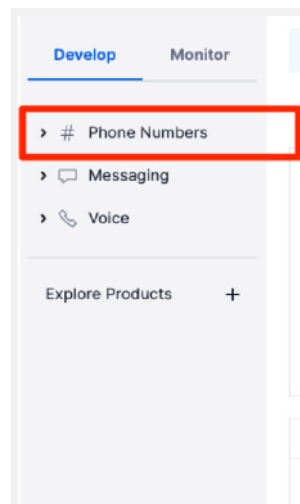


- d) Your account will be assigned with the number. The **Account SID** and the **AUTH Token** will be later used while configuring the Twilio connector.



#### → Step 4: Verify Caller Ids

- a) While in trial mode, we have to verify any non-Twilio phone numbers we wish to send SMS messages or place phone calls to. However, after upgrading the account, this is no longer needed. Click on **the Phone Numbers** dropdown on the side navigation menu.



- b) Click **Verified caller IDs** under the **Manage** dropdown. Once on the Verified caller IDs page, click the **Add a new Caller ID** button to add a new number:

**Verified Caller IDs**

Verify a number that you own to use it as a caller ID or as the 'To' number for outbound calls/messages from the Sandbox Number. The phone numbers you buy from Twilio, or port to Twilio, can always be used as caller IDs. [Learn more about Verified Caller IDs](#)

**Number** **Friendly Name**

**Filter** [Clear filter](#)

Enter the exact number Enter the exact friendly name

Number	Friendly Name	Actions
		<a href="#">Remove</a>

- c) Enter the desired phone number with extension and country to verify. Select the preferred verification method (SMS or Call), and then click **Verify Number**.

**Add a Caller ID**

The phone numbers you buy from Twilio, or port to Twilio, can always be used as Caller IDs.

**Country**

(+1) United States - US

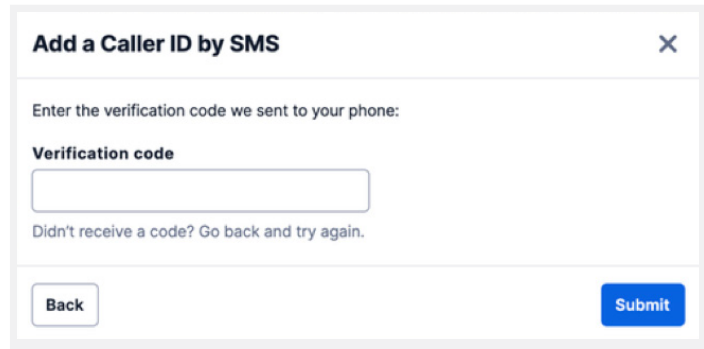
**Number** **Extension**

**Send verification code via**

☒ SMS ☐ Call

[Cancel](#) [Verify number](#)

- d) Enter the verification code. You're now ready to text or call this number with your trial Twilio account.

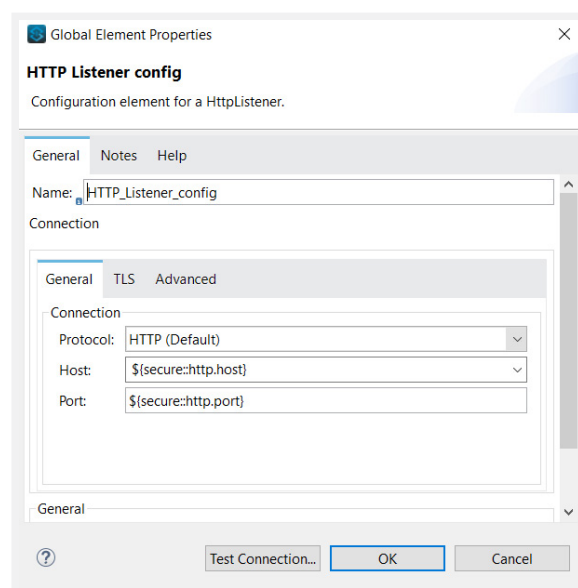


A dialog box titled "Add a Caller ID by SMS" with a close button (X) in the top right corner. The main text says "Enter the verification code we sent to your phone:". Below this is a label "Verification code" followed by a text input field. Under the input field, there is a link that says "Didn't receive a code? Go back and try again.". At the bottom, there are two buttons: "Back" on the left and "Submit" on the right.

## → Step 5: Integrating Twilio with MuleSoft

### 1. Basic Configuration

- a) Create a **New Mule Project** in Anypoint Studio.
- b) Create a flow with **HTTP Listener** which will trigger the flow. In the listener configuration, provide the desired **host** and **port**. Click on **OK**.

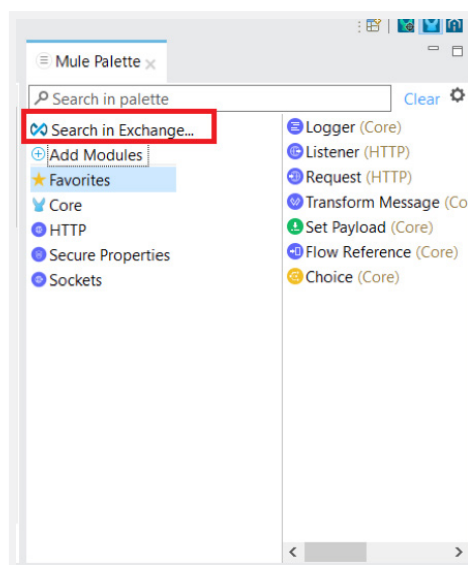


A "Global Element Properties" dialog box for "HTTP Listener config". The title bar says "Global Element Properties" and "HTTP Listener config". Below the title, it says "Configuration element for a HttpListener.". There are three tabs: "General", "Notes", and "Help". The "General" tab is selected. It shows a "Name:" field with the value "HTTP\_Listener\_config". Below that is a "Connection" section with three sub-tabs: "General", "TLS", and "Advanced". The "General" sub-tab is selected. It contains a "Protocol:" dropdown set to "HTTP (Default)", a "Host:" dropdown set to "\${secure:http.host}", and a "Port:" dropdown set to "\${secure:http.port}". At the bottom of the dialog, there is a "General" tab label, a help icon (?), and three buttons: "Test Connection...", "OK", and "Cancel".

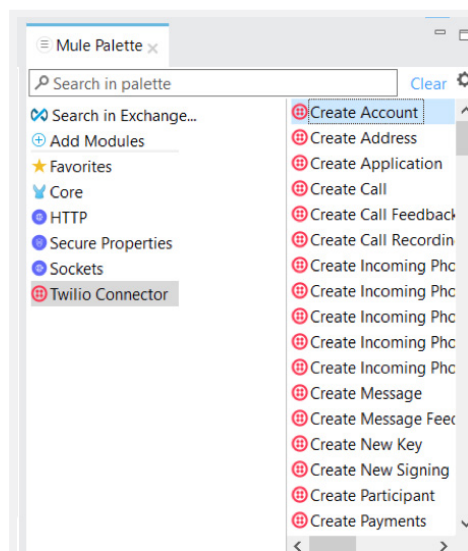
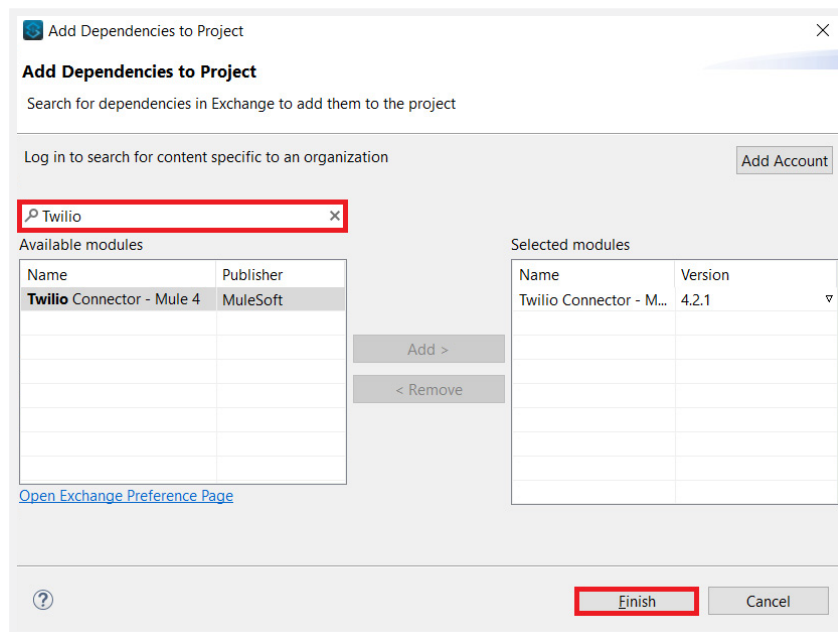
c) Under the path field, provide the desired listener **path**.



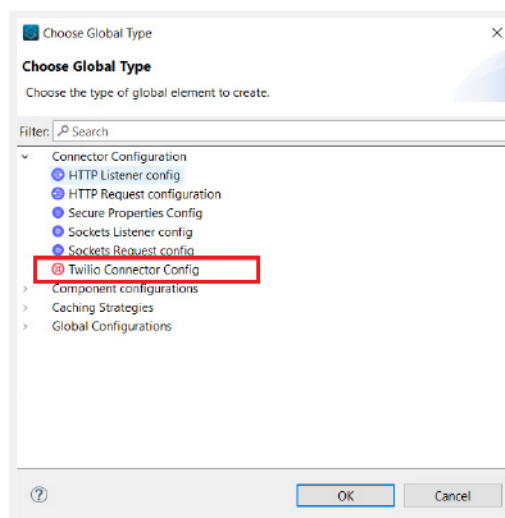
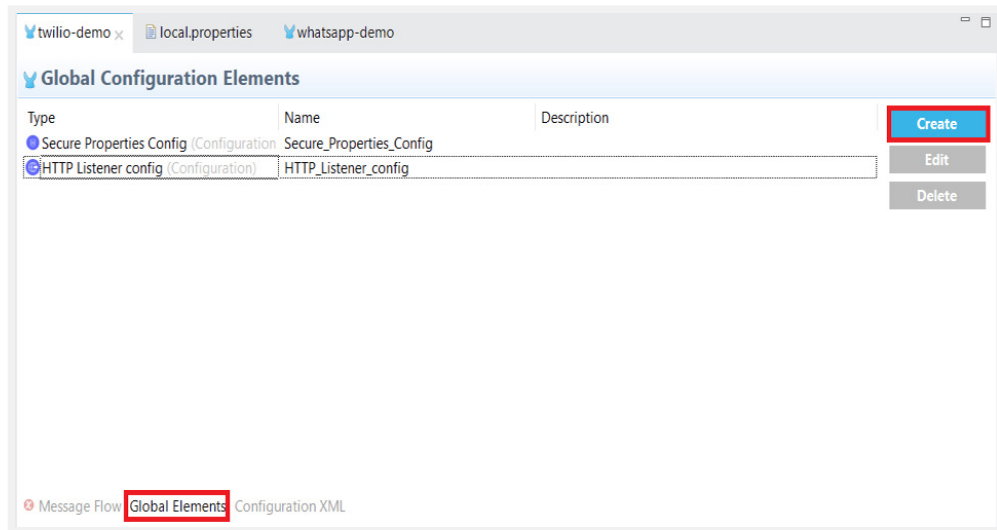
d) Twilio connector is not present by default in the Mule palette. We need to import it from Anypoint Exchange. Click on **Search in Exchange** in the Mule Palette.



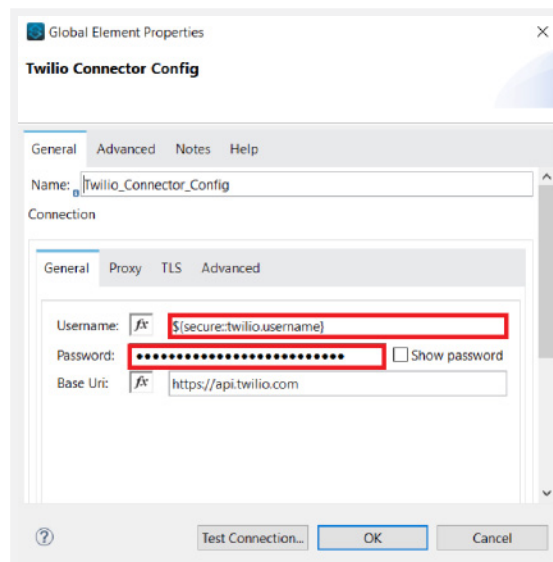
- e) Type Twilio in the search box. **Twilio Connector – Mule 4** will populate under the available modules. Click on **Add >** button, and then click on **Finish**. The Twilio module will get imported. After the Twilio module gets imported, you can see all the operations performed using the connector.



- f) Click on **Global Elements**. Then, click **Create**. A dialog box appears for choosing the **Global Type**. Click on **Connector Configuration**, and then click on **Twilio Connector Config**.



- g) The Twilio Connector Config dialog box, provides the **Twilio Account SID** under the **Username** field and **Twilio Auth Token** under the **Password** field. Please note that the Account SID and the Auth Token are fetched from the Twilio console dashboard.



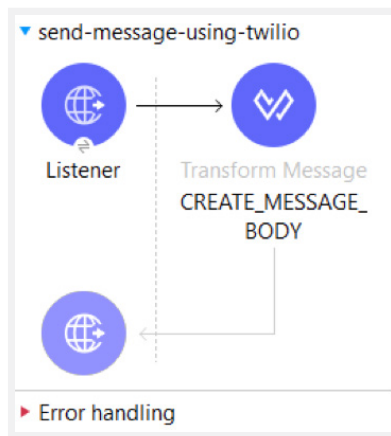
So, we are done with the Basic Configuration Settings of the Twilio Connector. Now let's touch upon a different operation that can be performed using different use cases:

## 2. Use Cases

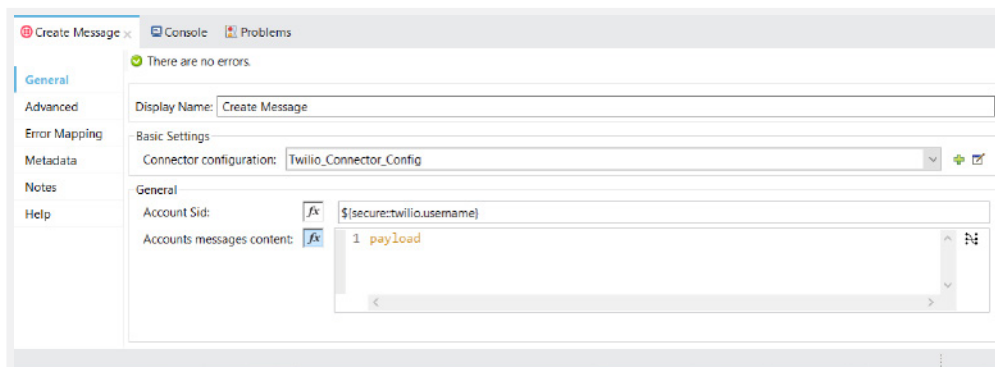
### → Case 1: Sending Text Message in Twilio

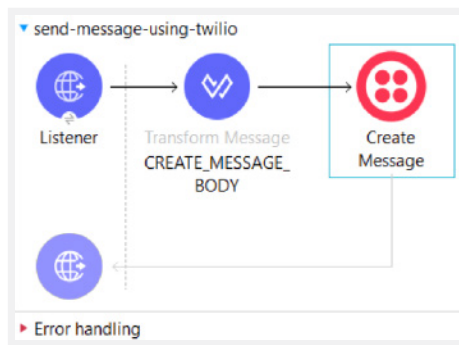
- Follow the steps as mentioned in the **Basic Configuration** to set up the Twilio connector.
- Configure the **Transform Message** with the following code:

```
%dw 2.0
output application/x-www-form-urlencoded
---
{
    Body: payload.Body,
    From: p('secure::twilio.fromNumber'),
    To: payload.To
}
```



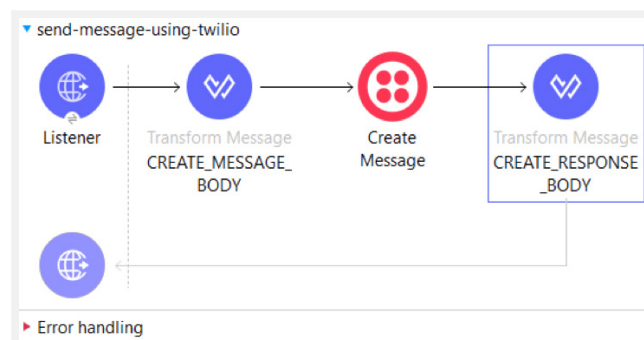
- **Body** – The text message that we want to send to the recipient.
  - **From** – The Twilio Number from which the message is being sent.
  - **To** – The message recipient's phone number.
- c) Configure the **Create Message** of Twilio Connector. Use the same Twilio connector configuration that we have already configured. Under the **Account Sid**, use the same Twilio username that we have already used.





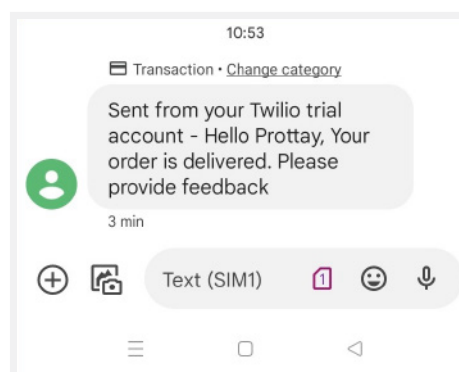
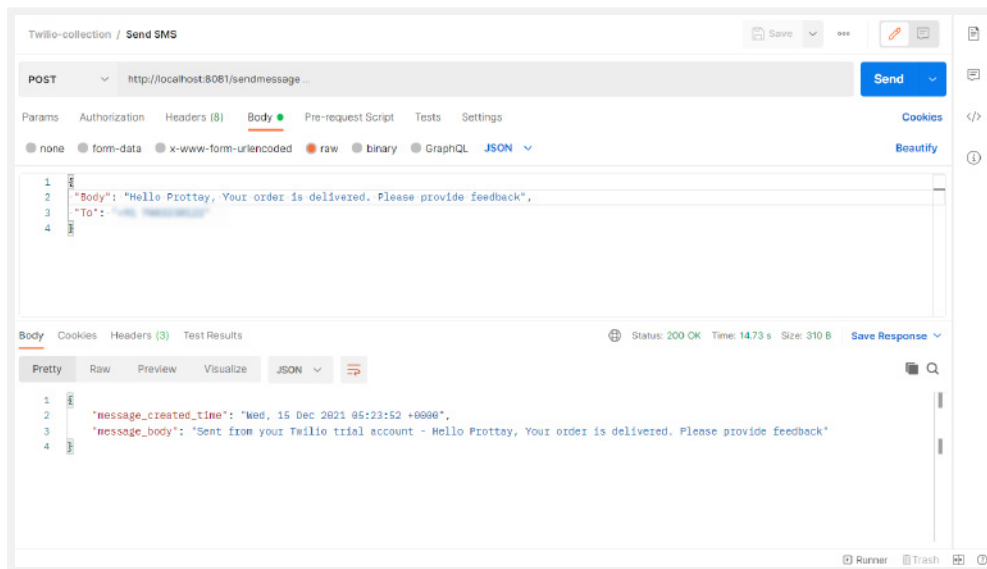
d) Configure the response body using **Transform Message** as mentioned.

```
%dw 2.0
output application/json
---
{
  "message_created_time": payload.date_created,
  "message_body": payload.body
}
```



e) Deploy the Mule application.

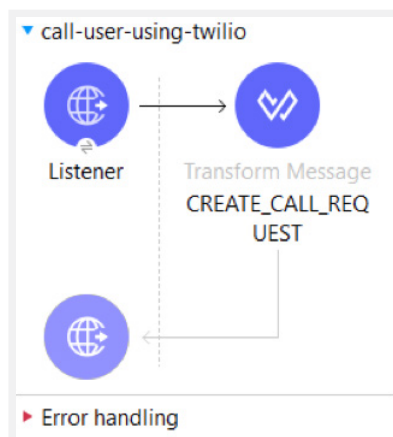
f) Test the application using Postman.



→ **Case 2: Calling user with Call Recording and Transcribe Feature**

- a) Follow the steps as mentioned in the **Basic Configuration** to set up the Twilio connector.
- b) Configure the **Transform Message** with the following body:

```
%dw 2.0
output application/x-www-form-urlencoded
---
{
  "To": payload.tonum,
  From: p('secure::twilio.fromNumber'),
  Twiml: "<Response><Say>Hi, Hope you are doing good!.
        Your order has been delivered </Say>
        <Record transcribe='true' />
        </Response>"
}
```



- **From** – The Twilio Number from which the call is being placed.
- **To** – The phone number which you want to call.
- **Twiml** – TwiML is the Twilio Markup Language, an XML document with special tags defined by Twilio. Every TwiML document will have the root **<Response>** element, which can contain one or more than one verbs.
  - TwiML Voice: **<Say>** - The **<Say>** verb converts text to speech that the caller reads back. **<Say>** is useful for development or saying dynamic text that is difficult to pre-record.

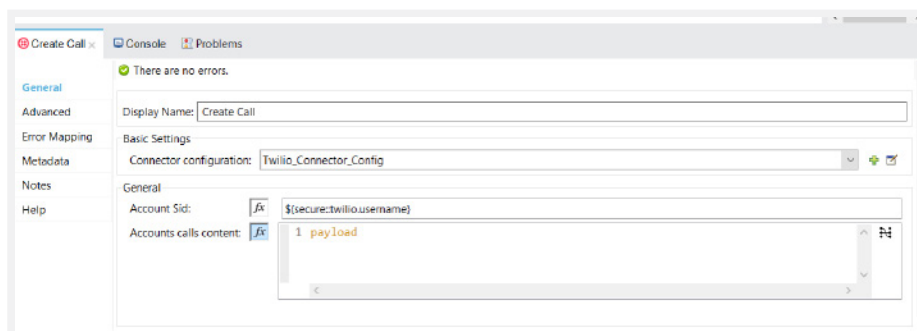
- **Twiml Voice: <Record>** - The <Record> verb records the caller's voice and returns to you the URL of a file containing the audio recording. The **transcribe** attribute tells Twilio that you would like a text representation of the audio recording.
  - Twilio will pass this recording to a speech-to-text engine and convert the audio to human-readable text. The transcribe option is not enabled by default. You have set it to **true**.
- a) **Note:** You can also use the **Record** field in the body to use the recording feature if you don't want to use TwiML.
- b) Keep the **Record** field as **true** if you're going to record the call, otherwise keep the **Record** field as **false**. The Url can be optionally used, instead of passing the **Twiml** parameter, you can provide a Url that returns Twiml voice instructions. The request transform message body will look like:

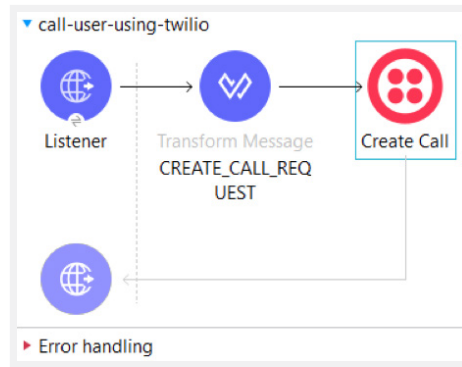
```
%dw 2.0
output application/x-www-form-urlencoded
---
{
    "To": payload.tonum,
    From: p('secure::twilio.fromNumber'),
    Record: true,
    Url: "http://demo.twilio.com/docs/voice.xml"
}
```

You can explore more on the TwiML feature:

<https://www.twilio.com/docs/voice/twiml>.

- c) Configure the **Create Call** of Twilio Connector. Use the same Twilio connector configuration that we have already configured. Under the **Account Sid**, use the same Twilio username that we have already used.

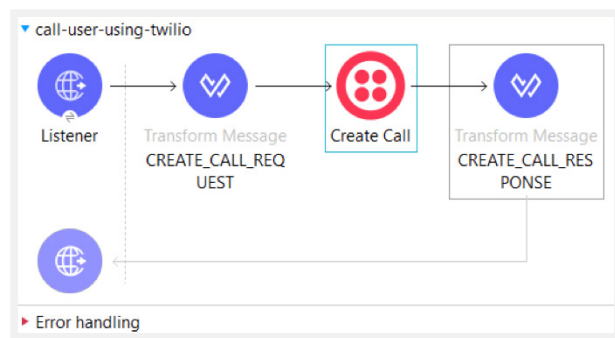




d) Set the response message using the **Transform Message** component.

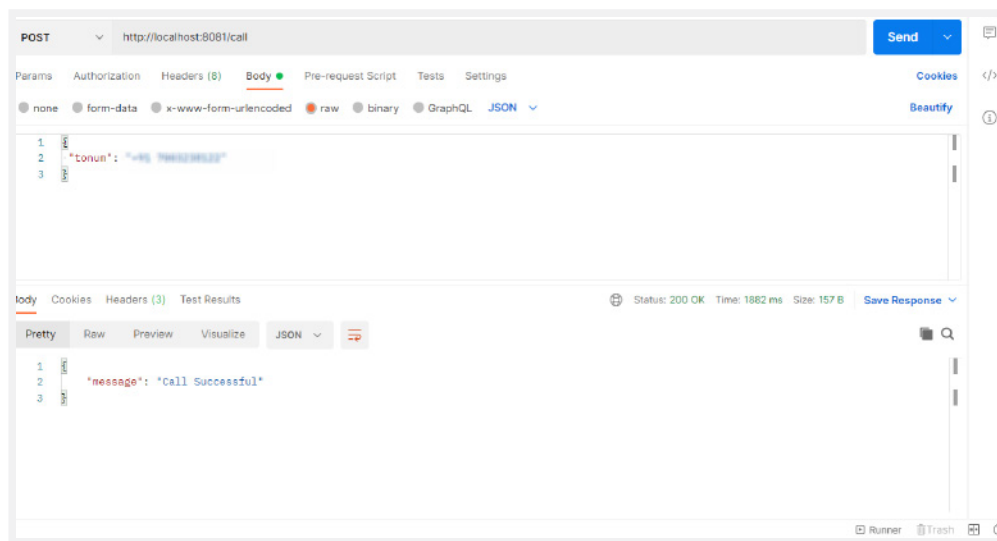
```

Output Payload  ▾  ≡  ✎  ≡  Preview
1  %dw 2.0
2  output application/json
3  ---
4  {
5    "message": "Call Successful"
6  }
  
```



e) Deploy the Mule Application.

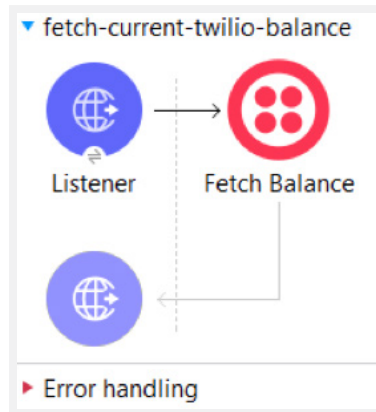
f) Test the application using Postman.



As we can observe, the call came from the same number assigned while setting up the account.

→ **Case 3: Fetch Twilio Account Balance**

- a) Follow the steps as mentioned in the **Basic Configuration** to set up the Twilio connector.
- b) Configure the **Fetch Balance** of Twilio Connector. Use the same Twilio connector configuration that we have already configured. Under the **Account Sid**, use the same Twilio username that we have already used.



There are no errors.

Display Name:

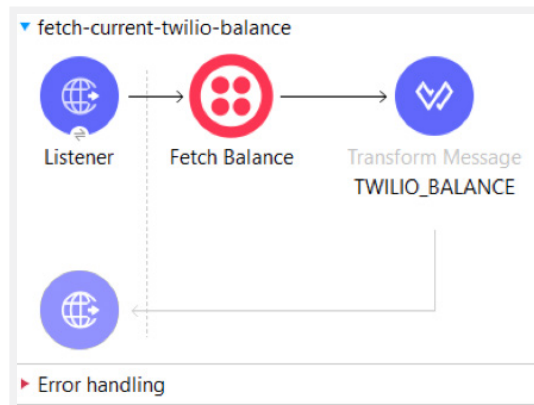
Basic Settings

Connector configuration:

General

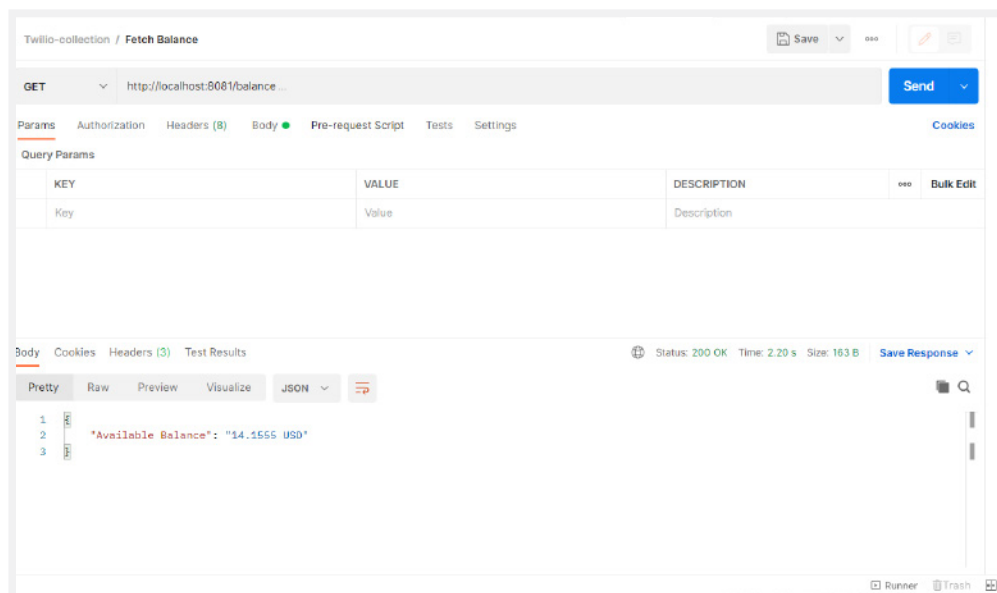
Account Sid:

- c) Set the response message in **Transform Message**.



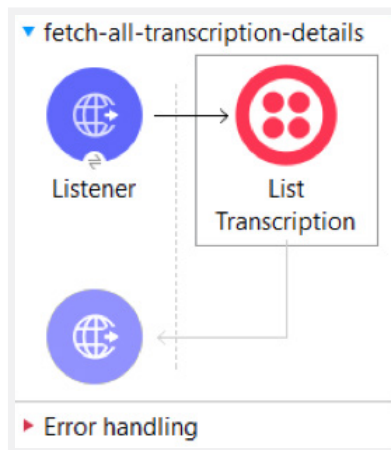
```
Output Payload 1 issue found
1 %dw 2.0
2 output application/json
3 ---
4 {
5   "Available Balance": payload.balance ++ " " ++ payload.currency
6 }
```

- d) Deploy the Mule Application.
- e) Test the application using Postman.



→ **Case 4: Fetch all the successfully recorded transcriptions**

- a) Follow the steps as mentioned in the **Basic Configuration** to set up the Twilio connector.
- b) Configure the **List Transcription** of Twilio Connector. Use the same Twilio connector configuration that we have already configured. Under the **Account Sid**, use the same Twilio username that we have already used. You can also define the **page size** if required.

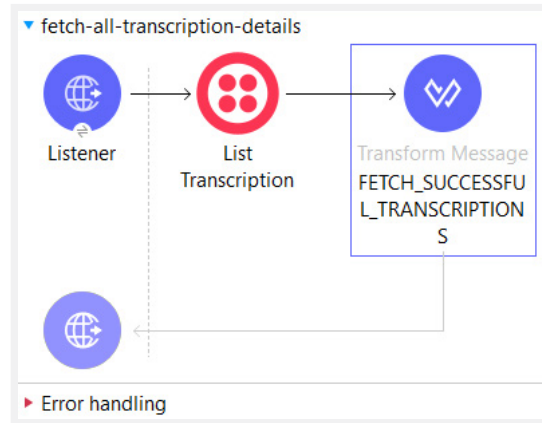


The screenshot shows the 'List Transcription' configuration window in Mule 4. The window has a sidebar on the left with tabs for 'General', 'MIME Type', 'Advanced', 'Error Mapping', 'Metadata', 'Notes', and 'Help'. The 'General' tab is selected. The main area displays the following configuration:

- Display Name:** List Transcription
- Basic Settings:**
  - Connector configuration:** Twilio\_Connector\_Config
- General:**
  - Account Sid:**
  - Page Size:**

At the top of the main area, there is a green checkmark icon and the text 'There are no errors.'.

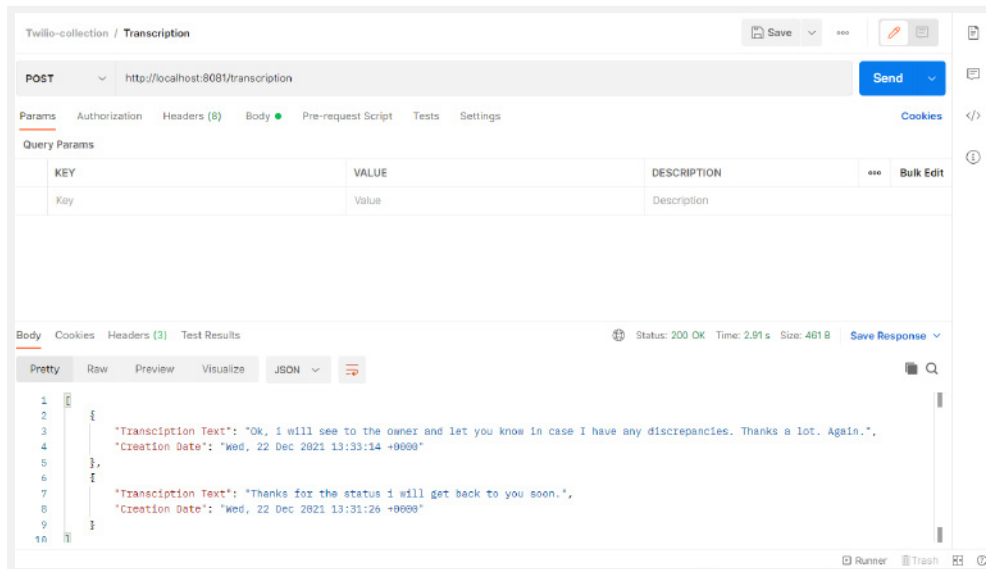
- c) Set the response body using **Transform Message**. Only those transcription details will be fetched, which is marked as **completed**. The transcriptions which have failed or are in progress will not be displayed.



Output Payload ▾ ⚙️ 🗑️ Preview

```
1 @%dw 2.0
2   output application/json
3   ---
4   payload filter($.status == "completed") map(
5     {
6       "Transcription Text": $.transcription_text,
7       "Creation Date": $.date_created
8     }
9   )
10 )
```

- d) Deploy the Mule application
- e) Test the application using Postman.



**Conclusion:** In this part, we covered Twilio Account Setup and the different use cases of integrating Twilio with MuleSoft.

The next part will cover integrating **WhatsApp** using Twilio with MuleSoft.