API-enabled application integration

100-level live demo script

Introduction

Automating customer interactions can remove manual steps, data entry into multiple different applications, and potential errors and delays – all of which are additional costs to your business. This demo automates a series of steps to: obtain and validate input information from a customer with a concern, open a case in Salesforce, attach the incoming information to the case, analyze the tone of the situation, and respond to the customer with the case number and expected date for resolution.

To automate this customer interaction, we will use both APIs and integrations to back-end applications. The demo scenario is related to a car repair, but this is just an example. The same techniques are applicable to your environment in support of your customers.

Let’s get started!

(Demo slides here)

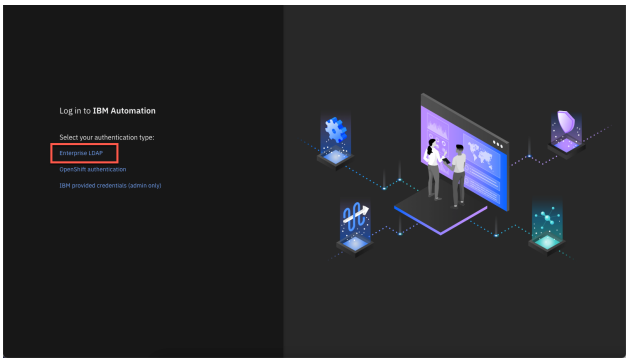
1 - Accessing the environment

Note: There is a known issue with Firefox when trying to access the API. For the best user experience, we recommend that you use the latest version of Safari or Chrome.

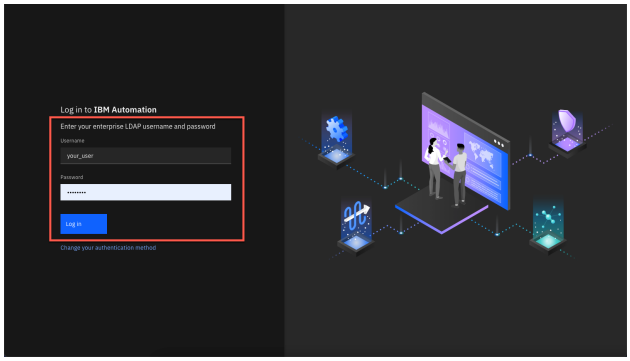
1.1 - Log into Cloud Pak for Integration

Narration
Let's see IBM Cloud Pak for Integration in action. I have a cloud version of the product on IBM Cloud. Let me log in here.

Action 1.1.1
- Open Cloud Pak for Integration. Click **Enterprise LDAP.**



Action 1.1.2
- **Log in** with your user and password.

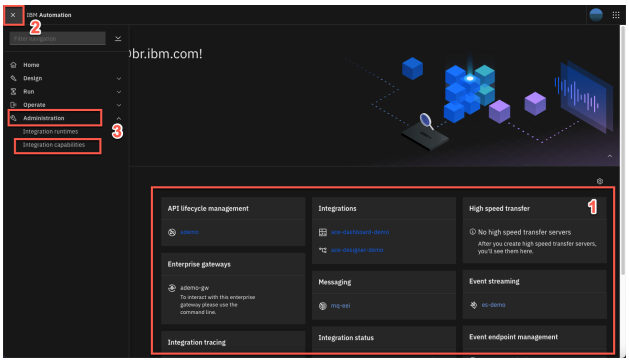


1.2 - View the Cloud Pak for Integration home screen

Narration
Welcome to IBM Cloud Pak for Integration! We're now at the home screen showing all the capabilities of the Pak, brought together in one place. Specialized integration capabilities — for API management, application integration, messaging and more — are built on top of powerful automation services. Let's see the integration capabilities available in Cloud Pak for Integration.

Action 1.2.1

- Show the **Home page** (1). Open the **Menu** (2), and click **Administration > Integration Capabilities** (3).

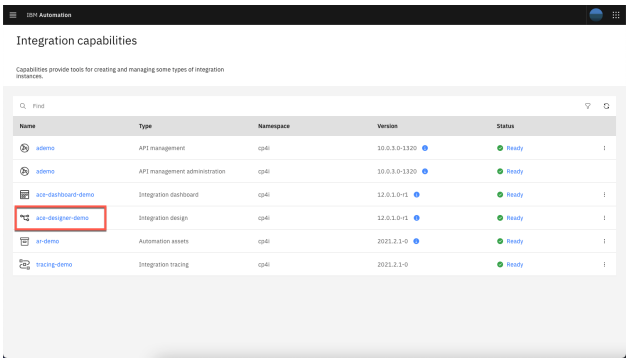


1.3 - Access integration capabilities

Narration
As you can see, you are able to access all the integration capabilities your team needs through a single interface. These include API management, application integration, enterprise messaging, events, and high-speed transfer. In this demo, to automate customer interactions with our company, we will use App Connect for application integration, API Connect for API management, and the Asset Repository as our centralized hub for allowing our teams to work together with integration assets. Let's open our App Connect Designer.

Action 1.3.1

- Show the **Integration Capabilities** page and open the Designer **(ace-designer-demo).**

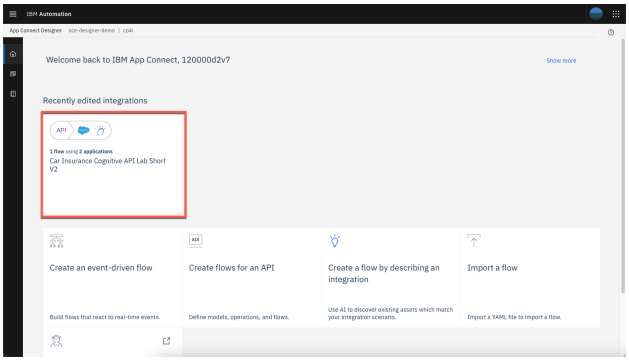


2 - Reviewing the flow

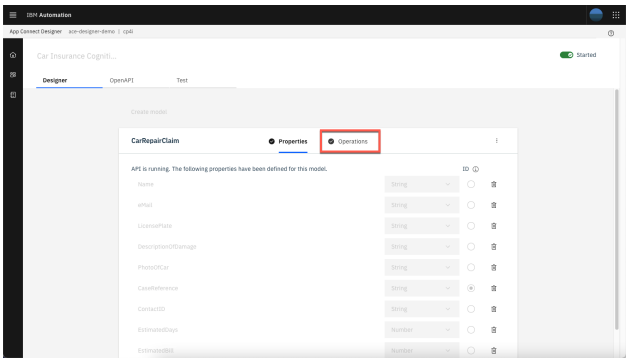
2.1 - Select flow and review properties

Narration
Here we are in the designer tooling. This is where we can create all our API integration flows and manage our connectivity to our services and endpoints. We have a pre-created flow about a customer interaction scenario. Let's use it to simplify our demonstration. These are the fields we are going to use for our API. Note that we tell our API which field is the key – in our case, CaseReference. Let's check the operations.

Action 2.1.1
- Select **Car Insurance Cognitive API Lab Short V2** flow.



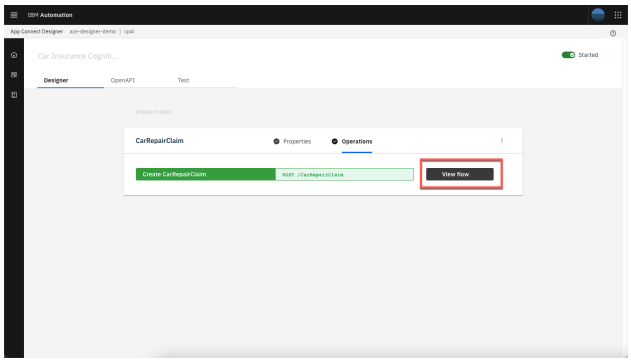
Action 2.1.2
- Show the **Properties view** and click **Operations.**



2.2 - Review operations

Narration
In the operations view are the actions that the API exposes along with the data. In this demo, we're going to build just one operation — "Create Car Repair Claim". We can add more later if we wish. Let's check the flow logic.

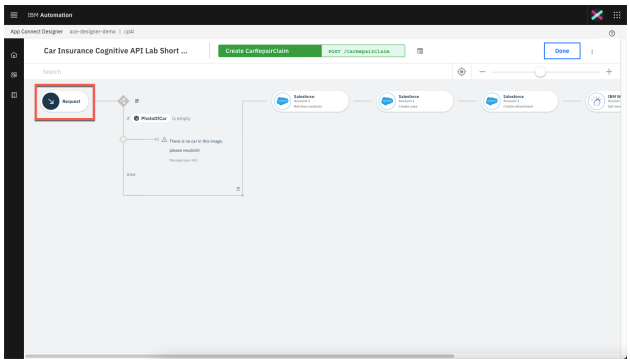
Action 2.2.1
- Show the **Operations** view, and click **View flow.**



2.3 - Review the flow

Narration
Here we have our demo flow. In the designer flow editor, we can edit and change our flow. We are a car repair company that wants to create an API that will enable customers to send us photos of their cars along with descriptions of what needs to be done with them. With this information, we will create a case in Salesforce while using Watson to analyze if the customer is angry or upset. Let's explore our flow in detail.

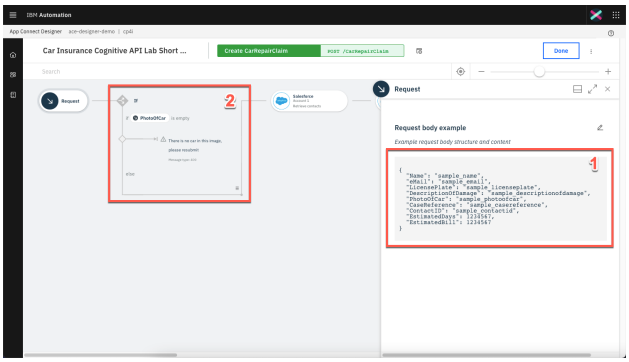
Action 2.3.1
- Explain the flow and scroll through all of the connectors in the flow. Open the **Request** again by clicking the first step of the flow.



2.4 - Review a request

Narration
Our flow starts by receiving the customer's car repair request with photo via an API. Designer automatically creates an API "Request" and "Response" for your API flow.

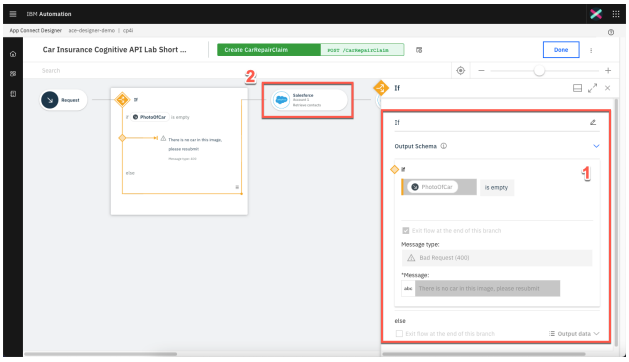
Action 2.4.1
- Show the **Request** dialog (1). Click to **open** the **if** step (2)



2.5 - Check the photo

Narration
Second, we validate the photo. Here, we have a simple IF statement, that checks if the PhotoOfCar is empty. If it is not empty, we move forward to retrieve contacts by connecting to Salesforce.

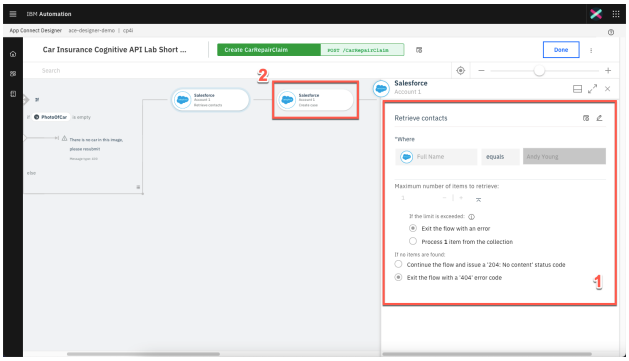
Action 2.5.1
- Explore the **If** step (1) . Click on **Salesforce Connector - Retrieve contacts** (2)



2.6 - Retrieving contacts

Narration
Third, we use a connector to create a case in Salesforce with the data from the API. This case is where we store the details and progress of our repair. We are using a hard-coded contact name: ‘Andy Young’. He’s the contact for the insurance company that sends customers. Salesforce Developer accounts have a prepopulated set of data that you can use to test. ‘Andy Young’ is one of those prepopulated contacts.

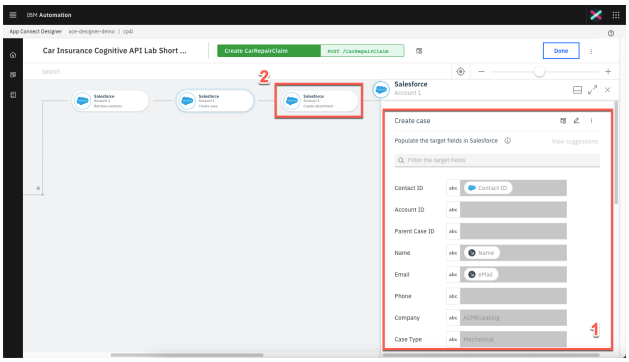
Action 2.6.1
- Explore the **Salesforce Connector - Retrieve contacts** (1). Click on the **Salesforce — Create case** node (2).



2.7 - Create a case

Narration
Now that we have the ID that we need, let’s create our Salesforce case. Note that we just re-use the same Salesforce connector but with a different operation and data. Here we can see that our contact ID comes from the previous ‘retrieve contact’ Salesforce call. The name and email address come from the API request. The connector “knows” that fields like ‘Case Type’ have a limited number of values in Salesforce, so it automatically converts them into pull-down lists of values.

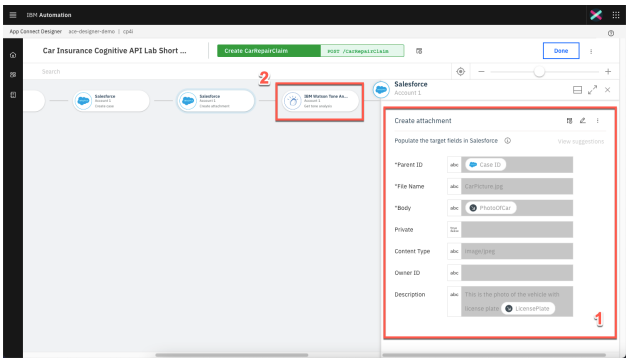
Action 2.7.1
- Explore the **Salesforce – Create case** node (1). Open the **Salesforce - Create Attachment** node (2).



2.8 - Create an attachment

Narration
To add a photograph, we need to create a Salesforce attachment. That will be easy, since we just use the connector again. Note that we use the Case ID that is a returned value from the ‘Create Case’ connector call – it’s been kept in the flow automatically. We send the PhotoOfCar as a base64 string and we tell Salesforce that the content type is image/jpeg. Moving forward to the next step, we want to understand the tone of the request. Is the customer angry? We will use the IBM Watson Tone Analyzer node to understand the situation better.

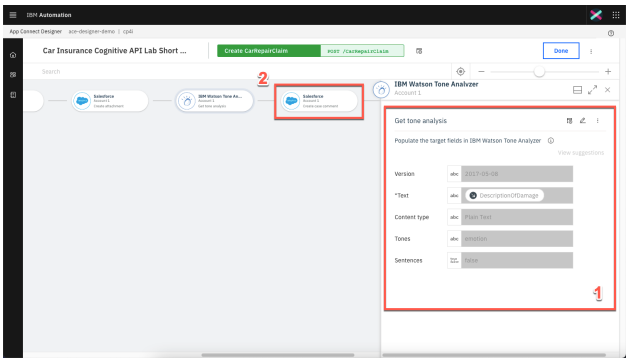
Action 2.8.1
- Explore the **Salesforce – Create attachment** node (1). Open the **IBM Watson Tone Analyzer** node (2).



2.9 - Analyze the description

Narration
Here, the Watson Tone Analyzer service analyzes the tone of the information provided by the customer. This can identify if the customer is angry or upset, allowing us to better tailor our response.

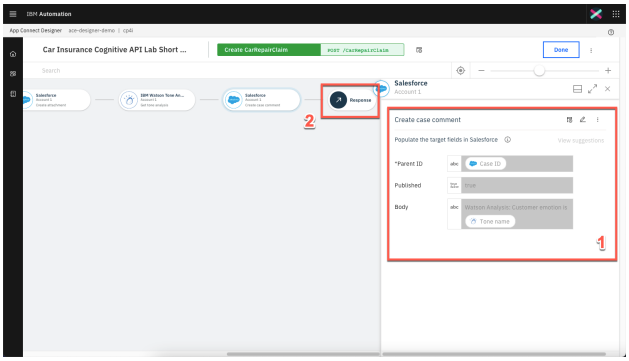
Action 2.9.1
- Explore the **Watson Tone Analyzer** connector (1). Open the **Salesforce - Create case comment** (2).



2.10 - Create a case comment

Narration
Now we'll add a comment to the case with the Salesforce connector and supply the tone name returned from Watson into the body of the comment.

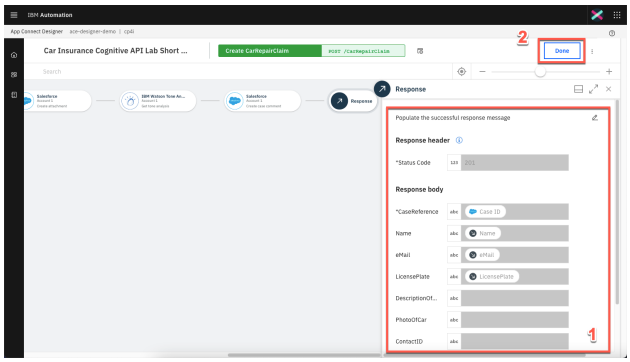
Actions 2.10.1
- Explore the **Salesforce – Create case comment** node (1). Open the **Response** (2).



2.11 - Respond to the customer

Narration
Here we have the response that we submit to the customer after the API call. This response includes their Salesforce case reference for future enquiries, an estimate of how long it will take to repair, and also how much it will cost. Great! Our API is running in our Integration Runtime. Later, we will test this API using the developer portal. But now, let's explore another great capability: API Manager. Let's see the benefits of having the API Manager control our API lifecycle.

Action 2.11.1
- Explore the **Response** dialog (1). Click **Done** (2).

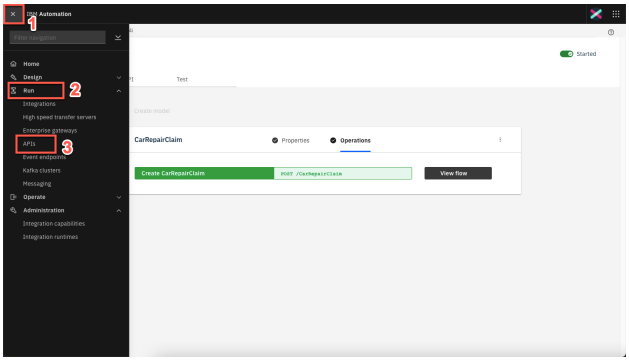


3 - Managing APIs

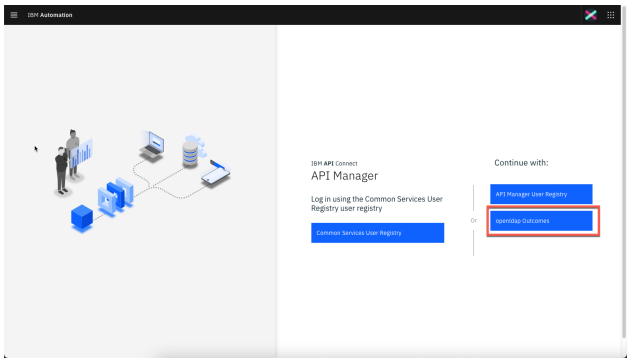
3.1 - Access API Connect

Narration
Now, let’s open the API management component inside Cloud Pak for Integration - API Connect. IBM API Connect is an integrated API management offering, with capabilities and tooling for all phases of the API lifecycle. Key steps of the API lifecycle include create, secure, manage, socialize, and analyze. API Connect has four major components: API Manager, Analytics, Developer Portal, and Gateway. Now, let’s explore the API Manager.

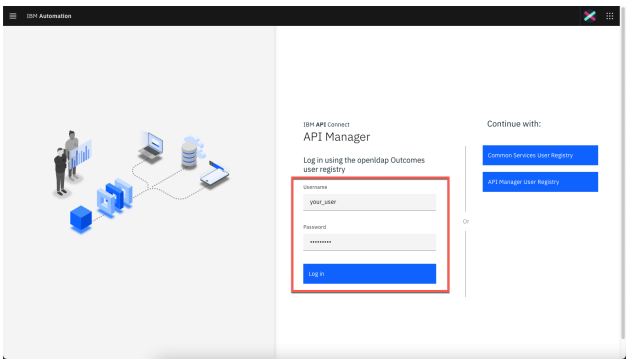
Action 3.1.1
- Open the **Menu** (1) and on **Run** (2) section, select **APIs** (3).



Action 3.1.2
- On the **API Connect** page, click **openldap Outcomes**.



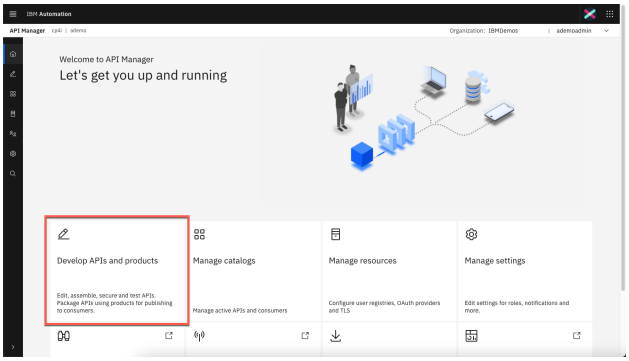
Action 3.1.3
- **Log in** with your Outcomes username and password.



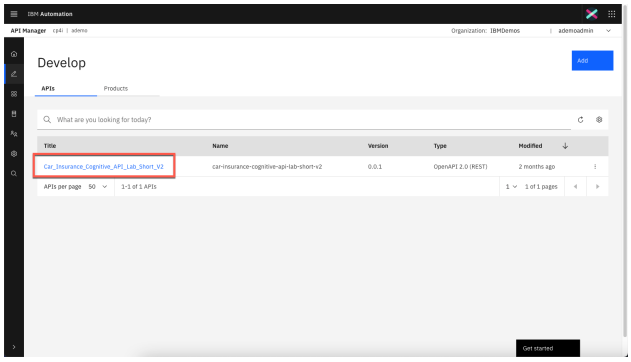
3.2 - Develop an API

Narration
Let’s check our API here in the API Manager. To simplify the demo, our API is pre-assembled here in the API Manager. In the Develop page, we are able to edit, assemble, secure and test APIs. In the security definitions, we can control client access to API endpoints (for example, using API key validation).

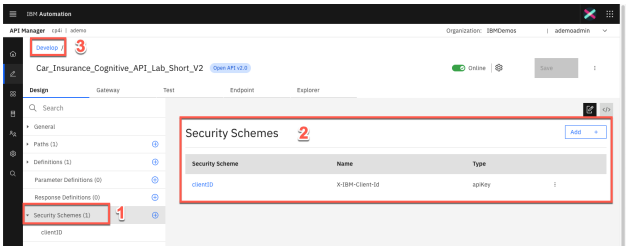
Action 3.2.1
- Click **Develop APIs and Products**.



Action 3.2.2
- Click **Car_Insurance_Cognitive_API_Lab_Short_V2**.



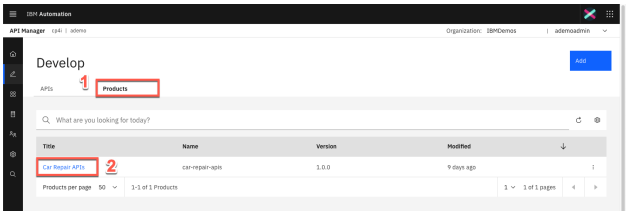
Action 3.2.3
- Open **Security Schemes** (1). Explore the **Security Schemes** page (2). Then, click **Develop** on the breadcrumbs (3).



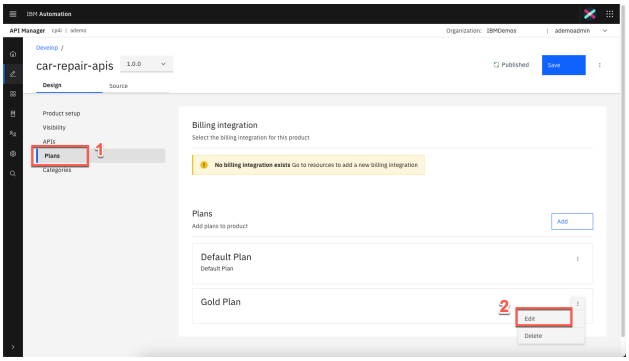
3.3 - Develop a product

Narration
Now, let’s see how we package our APIs using products for publishing to consumers. To simplify the demo, our product is already created here in the API Manager. Products are packages that contain both the APIs and the accompanying plans. The providers use plans to control access to APIs and to manage API usage. Plans can use differing rate limits to provide different levels of service to API consumers. In our product, we already have two plans: the Gold plan and the Default plan. We can have multiple plans for different consumers. For example, we can add approval steps for consumers when they sign up, or we can allocate them plans as a provider. For example, our Gold plan permits up to 100 calls per minute. Great, our API and projects are ready to be published for our API consumers.

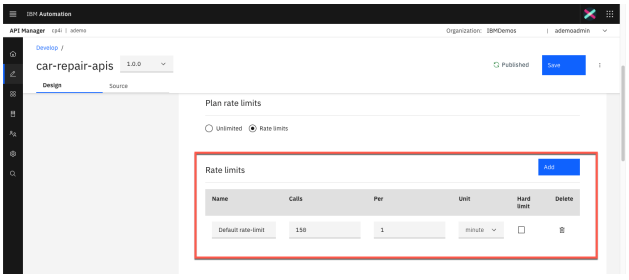
Action 3.3.1
- Click **Products** (1). Click **Car Repair APIs** product (2).



Action 3.3.2
- Open the **Plans** (1) tab. Open the context menu of **Gold Plan** and select **Edit** (2).

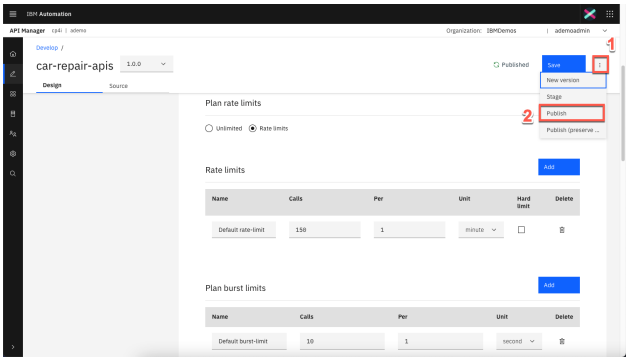


Action 3.3.3
- Explore the **Rate Limits** section.



Action 3.3.4

- Show how simple is to publish the Product and API by clicking **Menu** (1) > **Publish** (2).
- **Note**: you don't need to publish, because the product is already published.

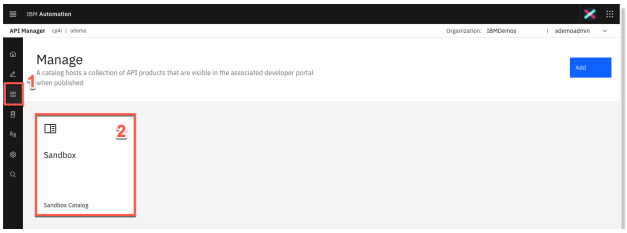


4 - Working with the Portal

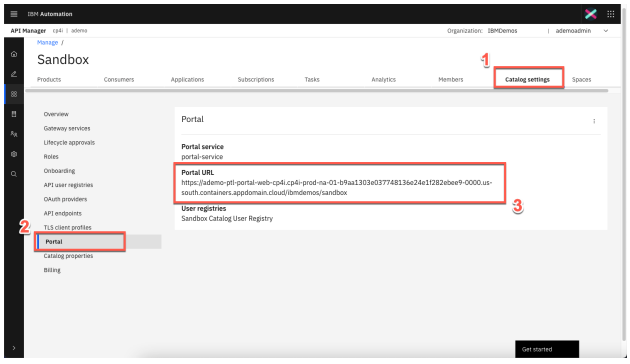
4.1 - Access the portal

Narration
For this demo, our API is already published. Now we need to make sure that our API consumers can discover it and use it. Our portal allows potential API consumers to view the APIs, sign up and subscribe to plans in a self-service manner, test the APIs, download the OpenAPI - Swagger documents, and more. Let's get our portal URL and sign up as a consumer of our API using portal self-service.

Action 4.1.1
- On the left navigator menu, open **Manage** page (1). Open the **Sandbox** catalog.

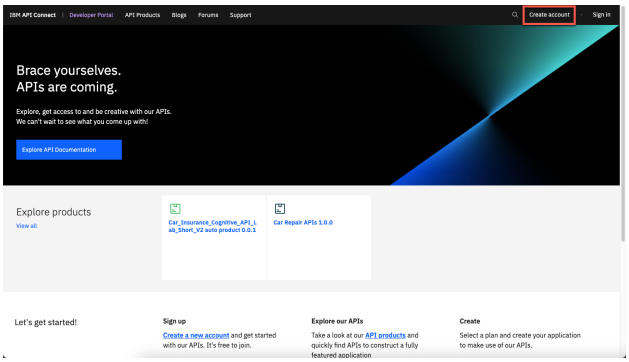


Action 4.1.2
- Open the **Catalog settings** (1). Click **Portal** (2). Copy the **Portal URL** (3).



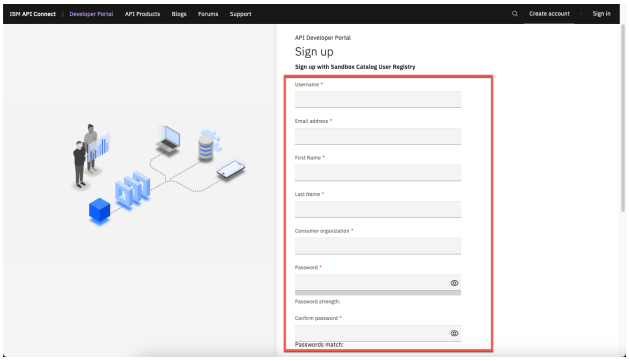
Action 4.1.3

- Open a new browser tab and access the portal URL. Create a developer account by clicking **Create account.**



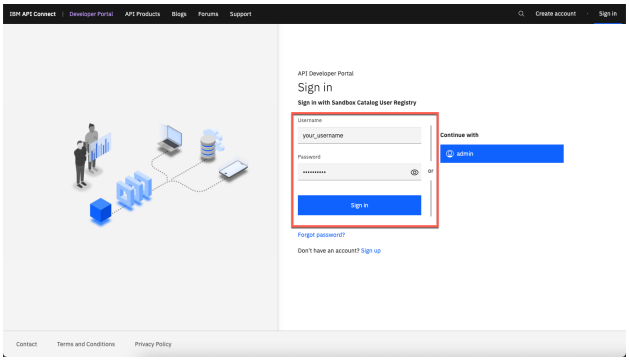
Action 4.1.4

- Use your personal email to complete the registration.



Action 4.1.5

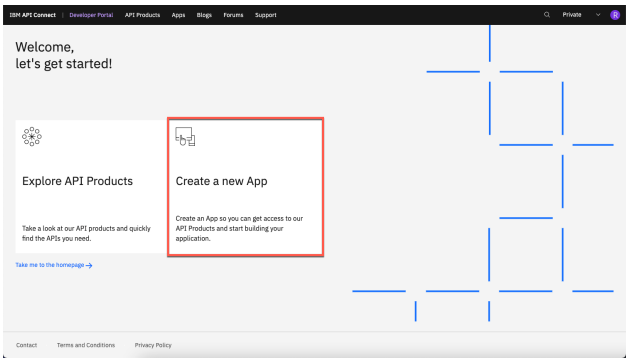
- You need to access your email to accept the invitation, and then **sign in.**



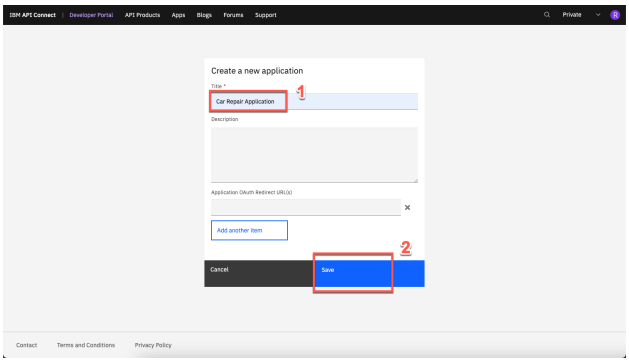
4.2 - Create a new app

Narration
As a consumer/developer, we're going to create a new application in the portal. This will give us an API key so we can call our APIs. We just need to give an application title and copy the API Key and Secret.

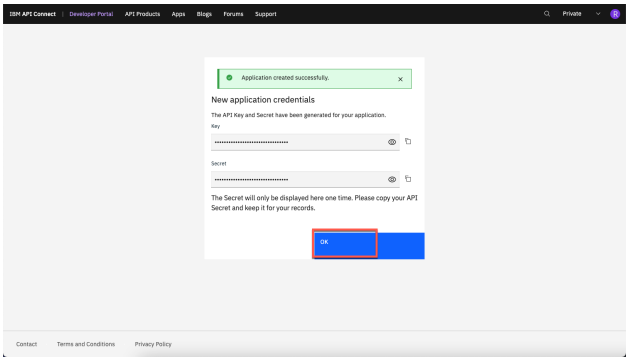
Action 4.2.1
- Click **Create a new App.**



Action 4.2.2
- Enter **Car Repair Application** as the Title (1). Click **Save** (2).

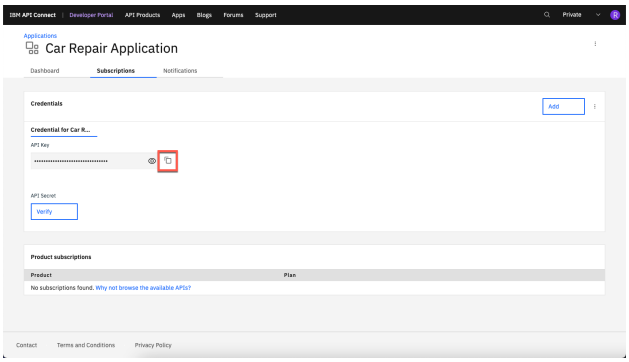


Action 4.2.3
- On the credentials dialog, click **OK.**



Action 4.2.4

- On the **Subscription** tab, copy the **Client ID**.



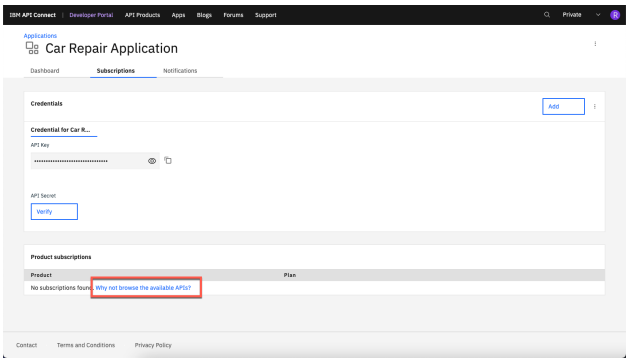
4.3 - Subscribe to the API

Narration

We've not subscribed to any APIs. Let's do it now. There's only one API product to subscribe to in our demo (normally, there would be many). Now that we selected our API product, we can see the plans available. You'll need to hover over to get the limits – we want the Gold plan. We want to subscribe to the plan, but which application do we want to use to subscribe? We can have many applications, but in this demo, we've only created one. So, we just need to select the app that we created earlier and confirm our subscription. And done - we are subscribed to our API!

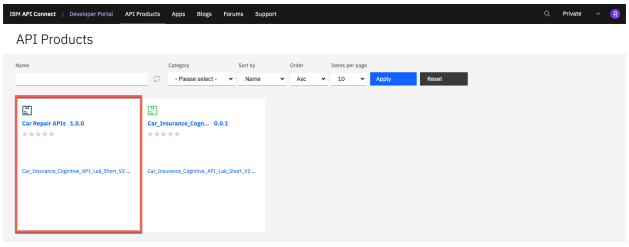
Action 4.3.1

- Click **Why not browse the available APIs?**



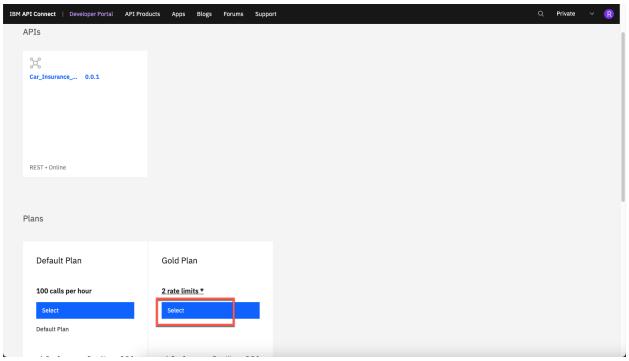
Action 4.3.2

- Click **Car Repair APIs 1.0.0.**



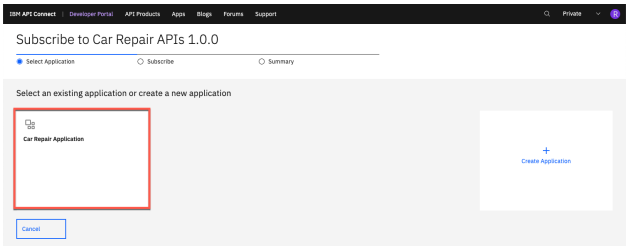
Action 4.3.3

- In the **Gold Plan** section, click **Select.**



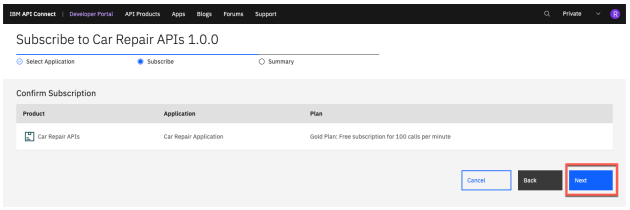
Action 4.3.4

- Select **Car Repair Application.**



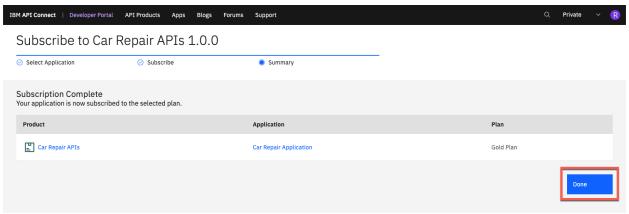
Action 4.3.5

- Confirm the subscription by clicking **Next.**



Action 4.3.6

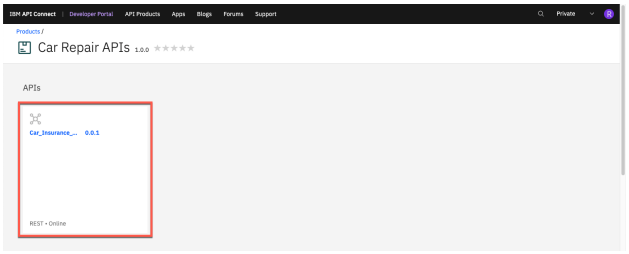
- Click **Done.**



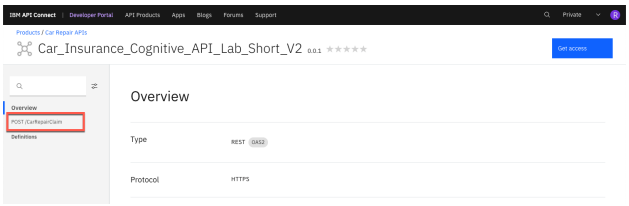
4.4 - Test the API

Narration
We're now back at the product screen. Let's explore our API here. From the Overview page, we can download the OpenAPI Document and get the API endpoint. Note the portal has everything you need to call your API, it's even generated clients in various languages for you to copy/paste into your calling application. You can try the API on the "Try it" area. Using the Generate button, the portal generates a request with random sample data for you. Now, let's test it. Great, we got a response, our API is running, and we've gone through the gateway to access it.

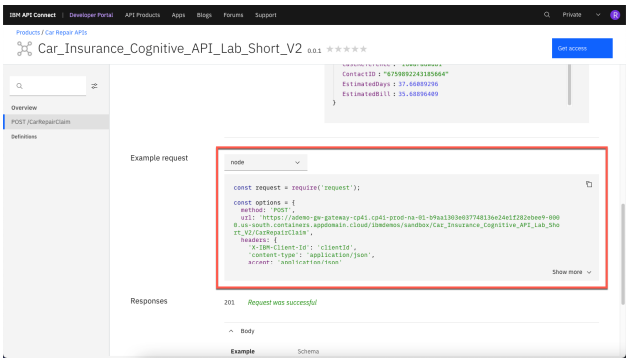
Action 4.4.1
- Click on the **Car_Insurance API.**



Action 4.4.2
- Open the **POST/CarRepairClaim.**

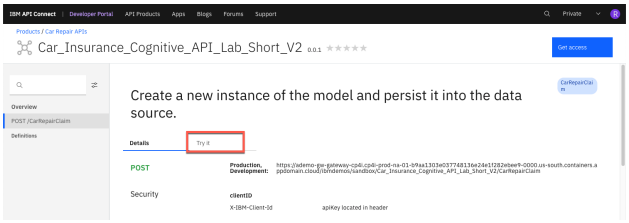


Action 4.4.3
- Explore the **Example request** area.



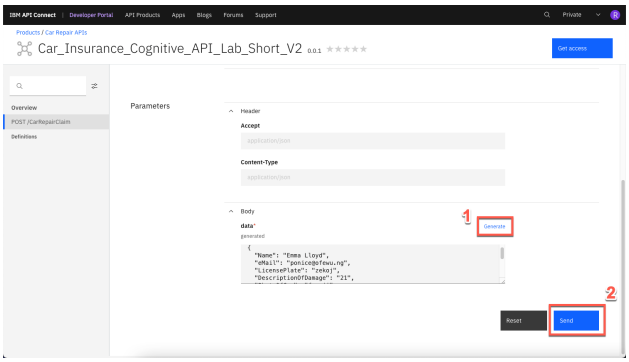
Action 4.4.4

- Open the **Try it** tab.



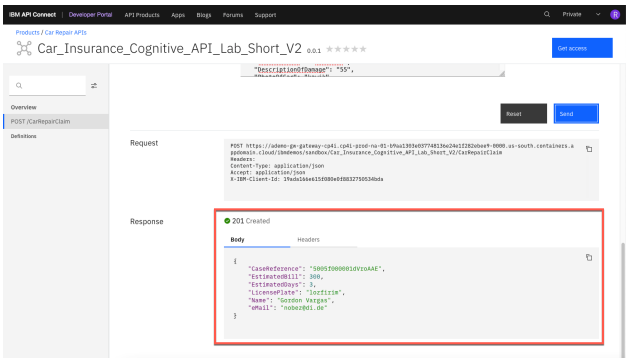
Action 4.4.5

- Click **Generate** (1) and click **Send** (2).



Action 4.4.5

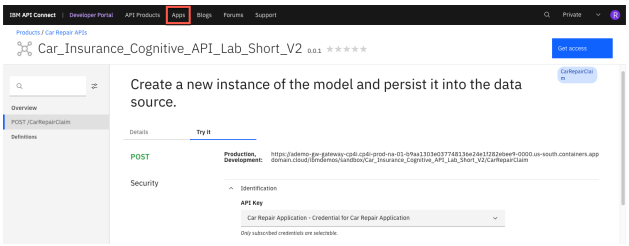
- Explore the **Response.**



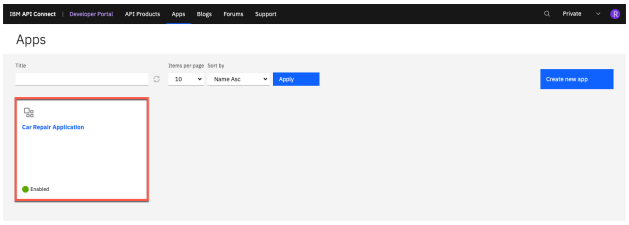
4.5 - View the API statistics

Narration
We can see our API statistics in the portal. We just need to select our app, and here we can see all the API calls, including any possible errors. If you make more calls, you'll see larger statistic results.

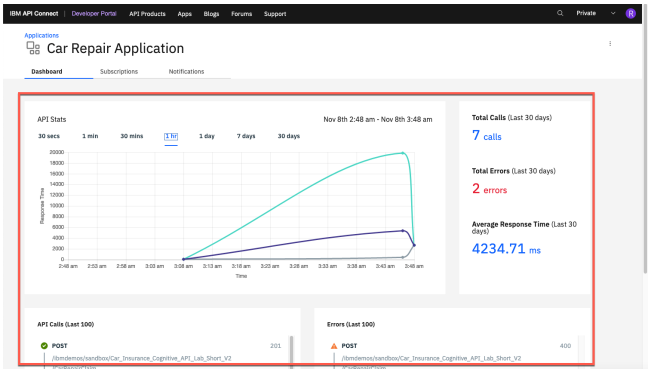
Action 4.5.1
- Click on **Apps** on the top menu.



Action 4.5.2
- Click on the **Car Repair Application** .



Action 4.5.3
- Explore the API stats.



Summary

Let’s summarize what we’ve done today. In the demo we:

- Accessed the Cloud Pak for Integration environment and explored the capabilities
- Reviewed the automated customer interactions integration flow
- Managed access to the flow as an API and set up the security and rate limits
- Demonstrated how a developer can use the API portal to perform self-service consumption of the API

From a business perspective, we used APIs and application integrations to automate a series of steps to: obtain and validate input information from a customer with a concern, open a case in Salesforce, attach the incoming information to the case, analyze the tone of the situation, and respond to the customer with the case number and expected date for resolution.

The customer expressing a concern or needing assistance obtains rapid response to their interaction and the confidence that your business is handling their request.