



Going Hands-On with IBM Bluemix Integration Services

Cloud Developer Lab | Miracle Innovation Labs

Manasa Sutapalli

Lead Researcher API M, MIL
Miracle Software Systems, Inc.

February 02nd, 2016

Going Hands-On with IBM Bluemix Integration Services

Goal

In this lab, the user will be creating an Enterprise application using IBM Bluemix where you will be creating a REST API Proxies for existing SOAP Services using IBM API Management; this REST API can be consumed into Bluemix Application.

Pre-Requisites

The following are required for this lab to run successfully,

- Browser for accessing IBM Bluemix
- Active Email ID for registering to IBM Bluemix
- Cloud Foundry
- Public WSDL for testing
- SOAP/REST Formats
- SOAPUI 5.0

Technologies Involved

- IBM Bluemix
- Node.js
- IBM API Management (Cloud Version)
- Cloud Foundry

Lab Steps

So, let us get started with the lab!

#1 | Access Bluemix

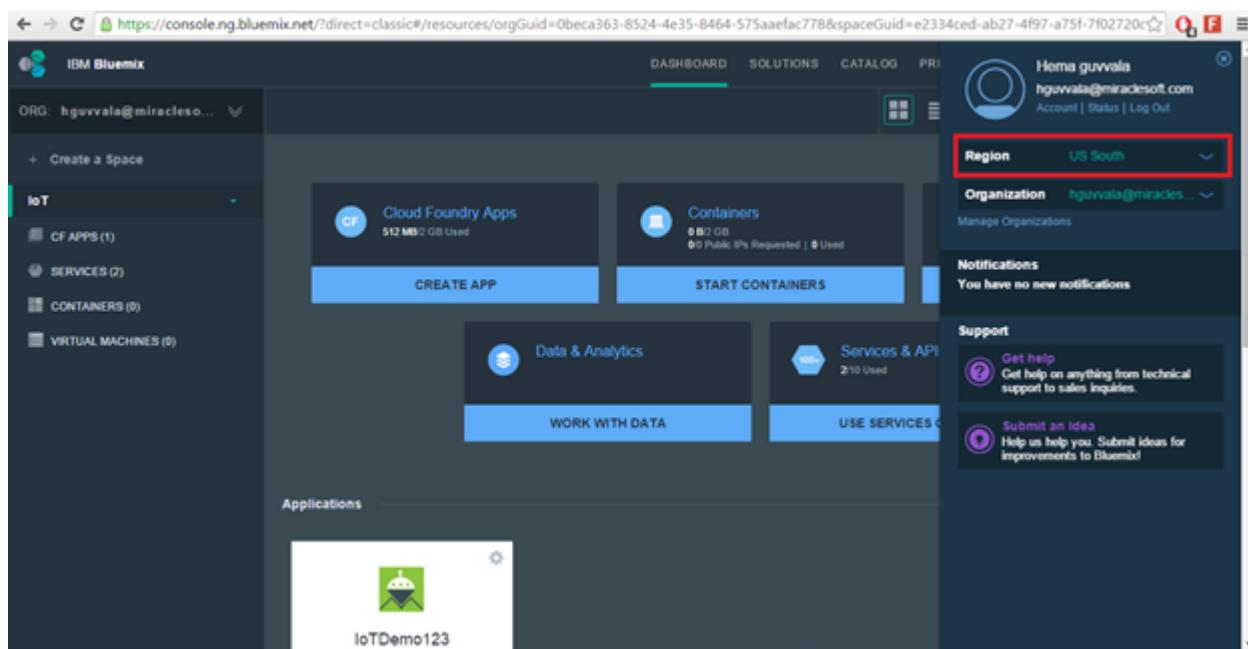
The first step will be to make sure that you have access to the IBM Bluemix Console with either the free trial option (or) the paid subscription option.

Login to Bluemix at <http://bluemix.net>

(or)

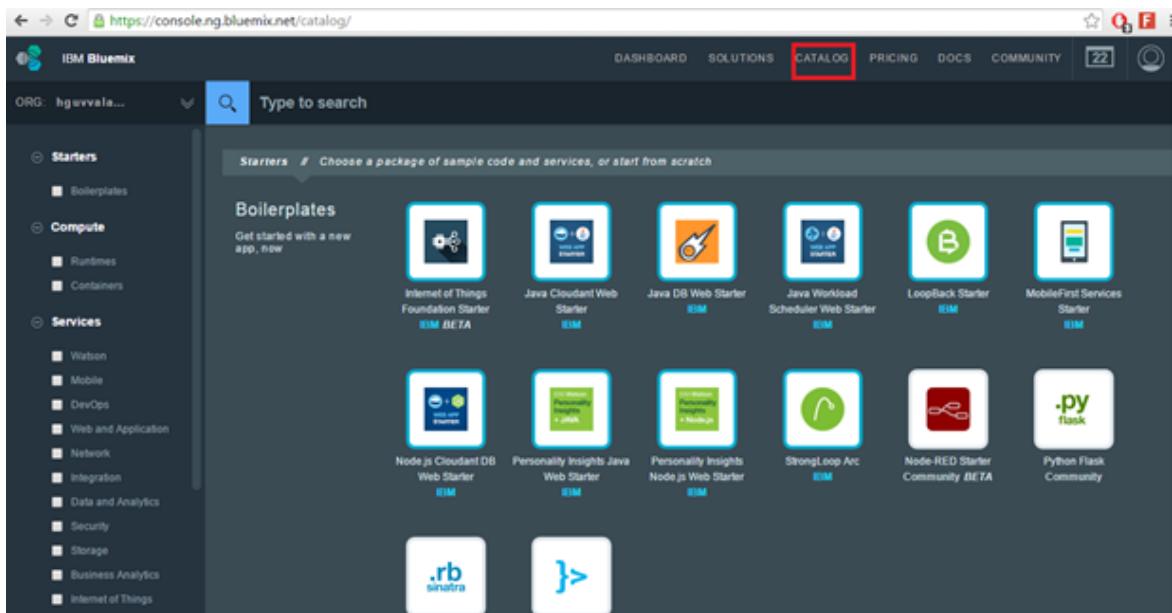
Register today at <https://console.ng.bluemix.net/registration/>

After you login, you can see the dashboard where you can take a look at your applications and services. You can also go to the profile icon at the top and change which Cloud Availability Region you are working in.



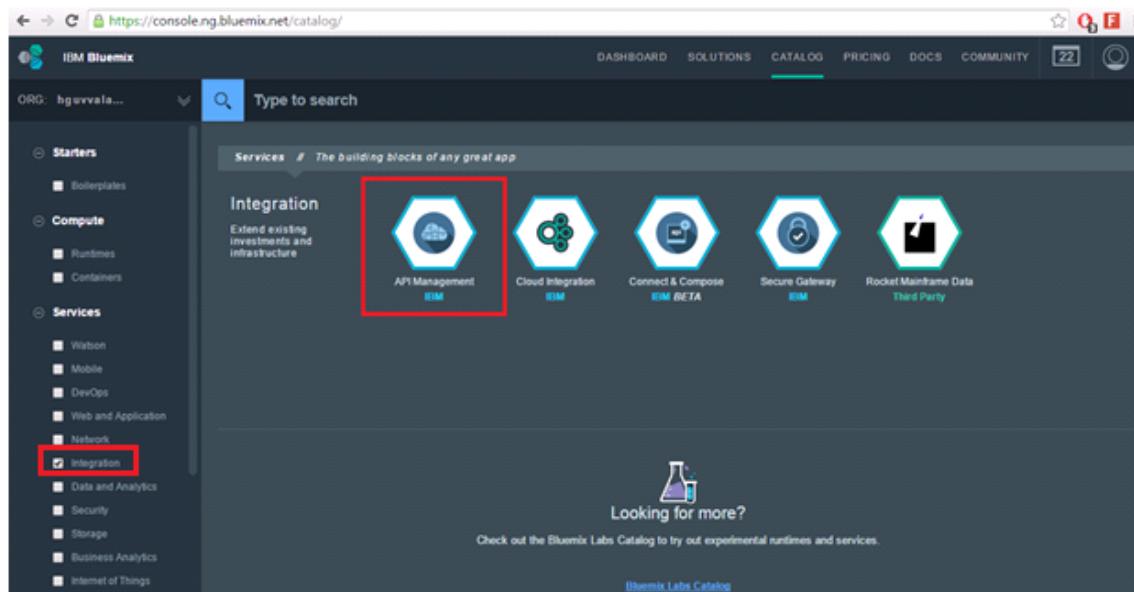
#2 | Bind API Management Service

Go to Catalog, it will show all the services that are provided by IBM Bluemix.



The screenshot shows the IBM Bluemix Catalog page. The top navigation bar includes links for DASHBOARD, SOLUTIONS, CATALOG (which is highlighted with a red border), PRICING, DOCS, and COMMUNITY. A search bar is located above the catalog grid. On the left, a sidebar menu is open under the 'Services' category, specifically the 'Integration' section, which is also highlighted with a red border. The main content area displays a grid of service icons and names. In the first row, the 'API Management' icon is highlighted with a red box. Other visible services include Internet of Things Foundation Starter, Java Cloudant Web Starter, Java DB Web Starter, Java Workload Scheduler Web Starter, LoopBack Starter, and MobileFirst Services Starter. The second row includes Node.js Cloudant DB Web Starter, Personality Insights Java Web Starter, Personality Insights Node.js Web Starter, StrongLoop Arc, Node-RED Starter Community BETA, and Python Flask Community.

In Integration Services, select **API Management Service** for creating API's and publishing them on to Bluemix.



This screenshot shows the same IBM Bluemix Catalog page as the previous one, but with a different selection in the sidebar. The 'Integration' section is now highlighted with a red border. The main catalog grid shows the 'API Management' service as the first item in the 'Integration' category, with its icon and name clearly visible and highlighted with a red box. Other services in this category include Cloud Integration, Connect & Compose, Secure Gateway, and Rocket Mainframe Data. The sidebar's 'Integration' section is also highlighted with a red border. The rest of the interface, including the top navigation and other service categories, remains consistent with the first screenshot.

Click on API Management Service. Choose your space in “Space” field and click on “Create”.

The screenshot shows the IBM Bluemix Catalog interface. On the left, there's a card for the 'API Management' service, which includes details like Publish Date (11/17/2015), Author (IBM), Type (Service), and Location (US South). Below this card is a 'VIEW DOCS' button. The main area displays two sections: 'Discover Existing APIs' and 'Design New APIs'. The 'Discover Existing APIs' section shows a graph of API usage over time and a list of available values. The 'Design New APIs' section shows a flowchart of transformation steps. To the right, there's a form for adding a new service. The 'Space' dropdown is highlighted with a red box and set to 'iot'. The 'Service name' is 'API Management-yg'. The 'Selected Plan' is 'Standard v2'. A large green 'CREATE' button is at the bottom of the form.

Click on “Go To API Manager”.

The screenshot shows the 'API Management-yg' dashboard. On the left sidebar, there are links for 'Manage' and 'Service Access Authorization'. Below that is a section for 'APPS USING SERVICE'. The main content area has a heading 'Get started with API Management' with the sub-instruction 'You can design, publish, and manage APIs through the API Manager console.' Below this is a green bar with the text 'All set! Use the links below to get started..'. It lists four items: 'Import APIs, or compose a new one.', 'Create a new plan, add resources and rate limits, and deploy.', 'Invite other Bluemix organizations to use your APIs.', and 'Publish your plan.'. At the bottom, it says 'Application developers will be able to discover and consume your APIs from the Bluemix catalog.' There are two buttons at the bottom: a large green 'GO TO API MANAGER' button and a smaller 'LEARN MORE' button. The 'GO TO API MANAGER' button is highlighted with a red box.

#3 | Create API

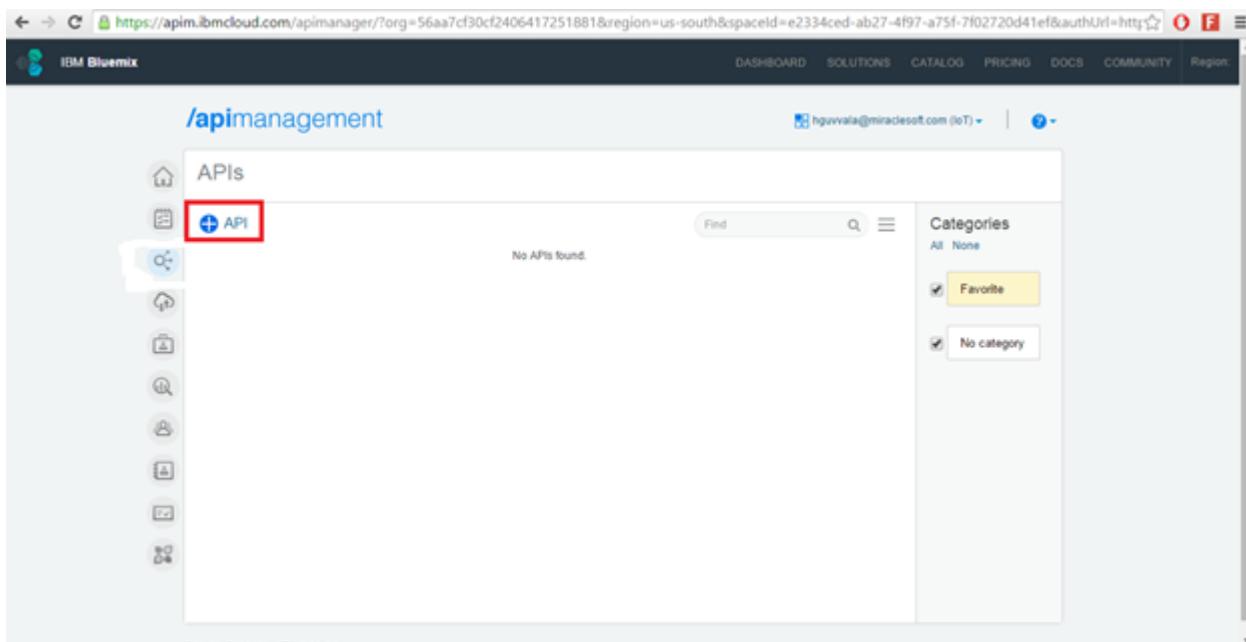
It will be redirecting into API Manager Console.

The screenshot shows the IBM Bluemix API Management console at the URL <https://apim.ibmcloud.com/apimanager/>. The top navigation bar includes links for DASHBOARD, SOLUTIONS, CATALOG, PRICING, DOCS, and COMMUNITY, along with a Region selector. The main content area is titled '/apimanagement' and features a 'Home' section with various icons and statistics. On the right, there are sections for 'Approvals', 'Usage', and 'Payload Logging'. A sidebar on the left contains navigation links for Home, Approvals, Requests, Statistics, APIs, Applications, and Developers.

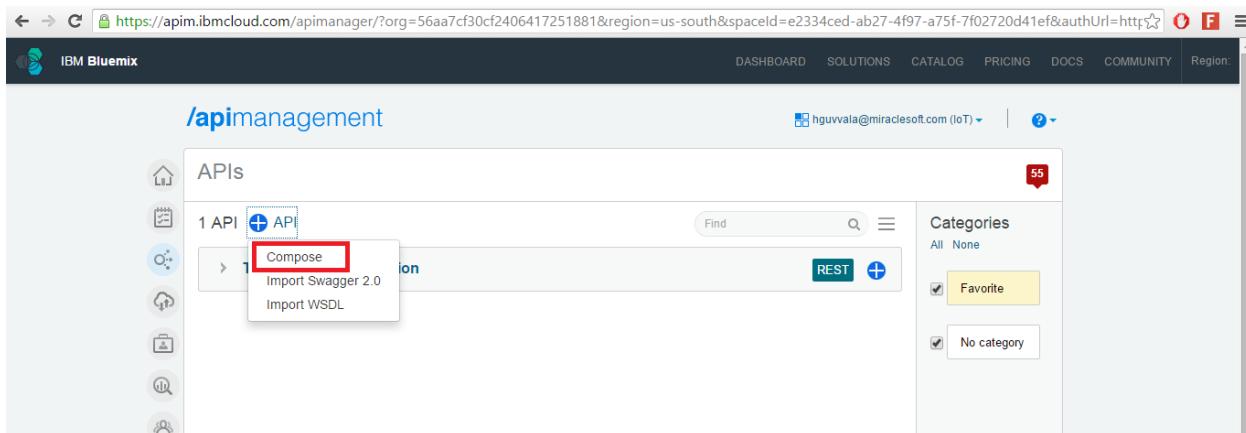
Create API by clicking on API's icon in Navigation Pane, this will take you to Draft API's view.

The screenshot shows the IBM Bluemix API Management console APIs page at the URL <https://apim.ibmcloud.com/apimanager/>. The top navigation bar and sidebar are identical to the previous screenshot. The main content area is titled '/apimanagement' and shows a list of APIs. A red box highlights the search icon in the navigation pane. The results table shows 'No APIs found.' and includes columns for Find, Search, and Actions. To the right, there is a 'Categories' section with options for 'All', 'None', 'Favorite', and 'No category'.

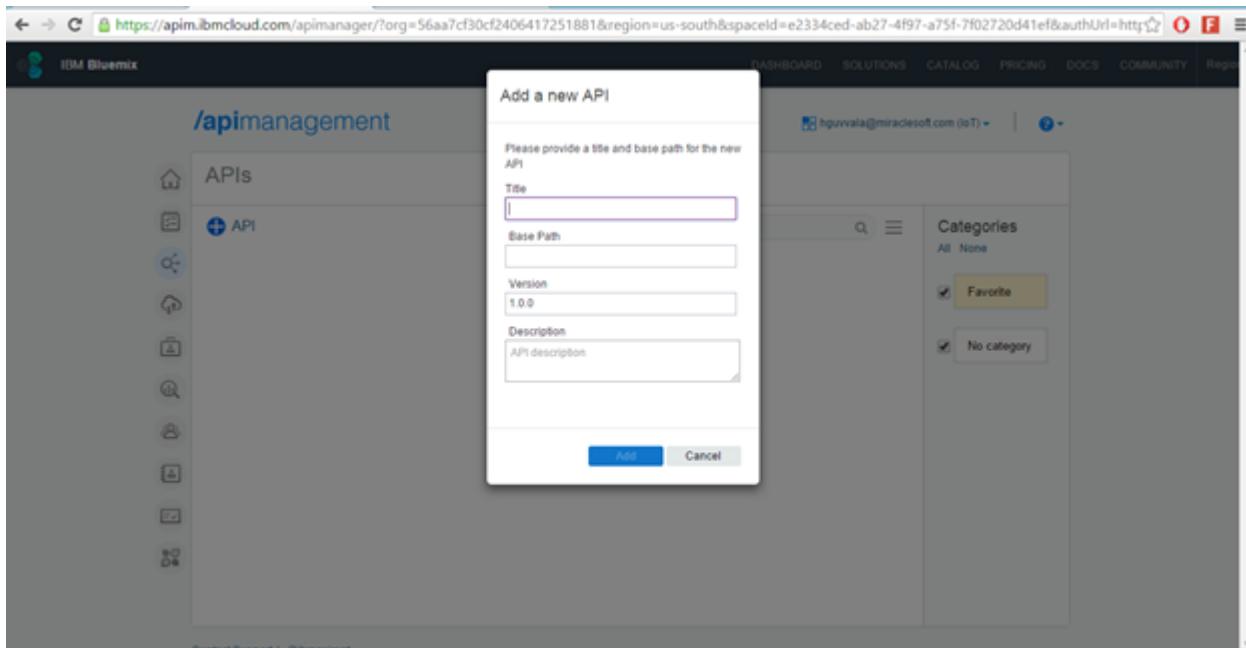
Click on “+API” Button to create New REST API.



Select **Compose** from the list to create a REST API.



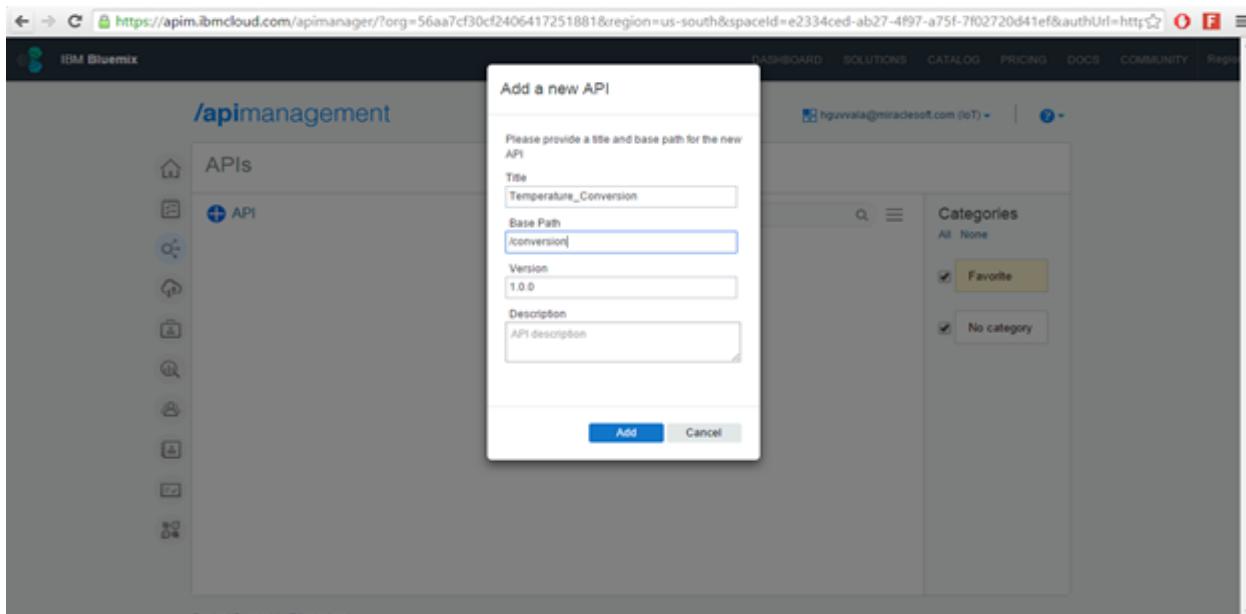
Fill the fields as shown below. Click the **Add button**. This will add the API to the Draft APIs list.



Fill the fields and click add button. This will add the API to the drafts list.

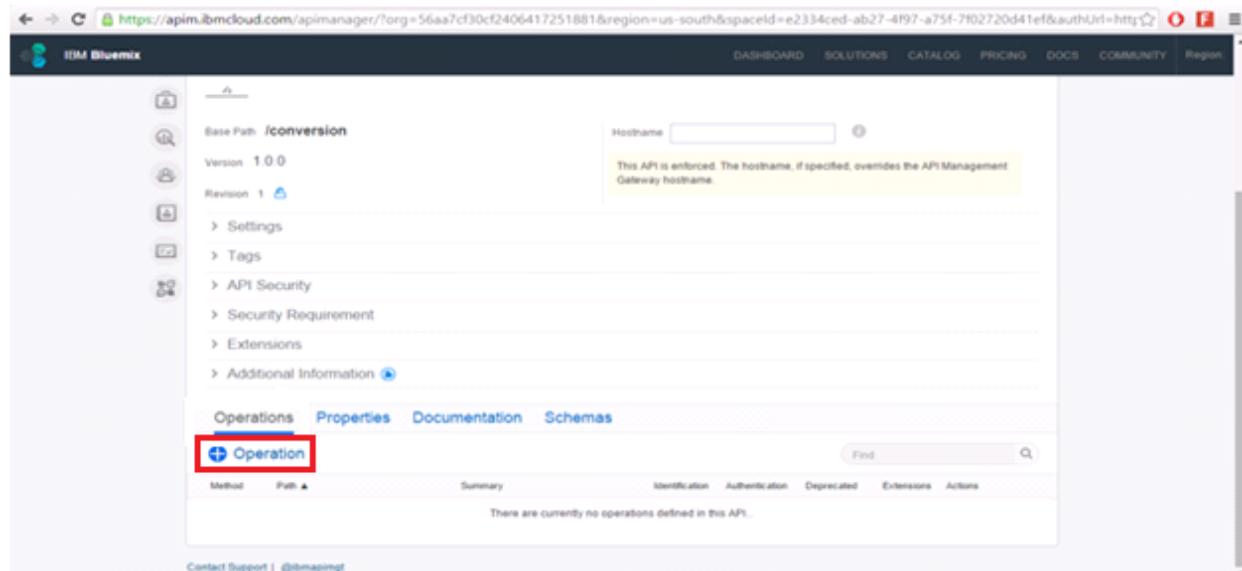
Field Name	Value
API Name(Title)	Temperature_Conversion
Base Path	/conversion
API Description	Converting the temperature from Celsius to Fahrenheit

Note : The Path specified here will become a part of the URL which will eventually be called.



After creating the **Temperature_Conversion** API, it will immediately appear in the list of draft APIs.

Click on the “Operations” pane for defining the API.



By default, the HTTP methods **GET** and **POST** are displayed. Remove the GET method by clicking the 'X' inside of the GET button.

The screenshot shows the 'Operations' tab selected in the top navigation bar. Under the 'Operation' heading, there's a 'Find' search bar. The 'Path' field is set to '/collection/{id}?filter'. In the 'Methods' section, 'GET' is listed with a red box around its 'X' button, and 'POST' is also listed with its own 'X' button. Below the methods are several tabs: 'Method', 'Path ▲', 'Summary', 'Identification', 'Authentication', 'Deprecated', 'Extensions', and 'Actions'.

Populate the fields of the Operation for that API.

Field Name	Value
Path	/FahrenheitToCelsius
Summary	TempConv
Description	Temperature Conversion Resource

The screenshot shows the 'Operations' tab selected in the top navigation bar. Under the 'Operation' heading, there's a 'Find' search bar. The 'Path' field is set to '/FahrenheitToCelsius', the 'Summary' field is set to 'TempConv', and the 'Description' field is set to 'Temperature Conversion Resource'. In the 'Identification' and 'Authentication' sections, the 'Identification' checkbox is checked (with a red box around it) and the 'Authentication' checkbox is unchecked. Below the fields are tabs: 'Method', 'Path ▲', 'Summary', 'Identification', 'Authentication', 'Deprecated', 'Extensions', and 'Actions'.

The screenshot shows the IBM Bluemix API Manager interface. On the left, there's a sidebar with icons for Home, Overview, Catalog, and API Management. The main area displays the details for a resource named '/conversion'. The resource has a base path of '/conversion', version 1.0.0, and revision 1. It includes sections for Settings, Tags, API Security, Security Requirement, Extensions, and Additional Information. Below these, tabs for Operations, Properties, Documentation, and Schemas are visible. Under the Operations tab, a table lists an operation: Method POST, Path /FahrenheitToCelsius, Summary, Identification, Authentication, Deprecated, Extensions, and Actions. The Actions column contains icons for Edit, Delete, and Copy, with the Edit icon highlighted by a red box.

Now that the FahrenheitToCelsius resource has been defined, click on the **Edit** icon to edit the resource details.

This screenshot is identical to the one above, showing the same resource details for '/conversion'. The operations table is the same, with the Edit icon in the Actions column of the first row highlighted by a red box.

In this resource there is no need of adding any request or response headers . So the next step is to add a Request Body to define the structure of the request.

The Temperature Converter request type is SOAP, but we want to use JSON as our data format. So here you need to define your own JSON Request type.

The screenshot shows the IBM Bluemix API Manager interface. A POST method is defined for the path /FahrenheitToCelsius. The Request tab is selected. In the Request Headers section, there is one header named 'FahrenheitTemperature' with the value '25'. The Swagger Schema Object and Request Example sections are currently empty.

Add the Request header as { “Fahrenheit” : “25” }

The screenshot shows the IBM Bluemix API Manager interface. A POST method is defined for the path /FahrenheitToCelsius. The Request tab is selected. In the Request Headers section, there is one header named 'FahrenheitTemperature' with the value '25'. The Swagger Schema Object and Request Example sections are currently empty.

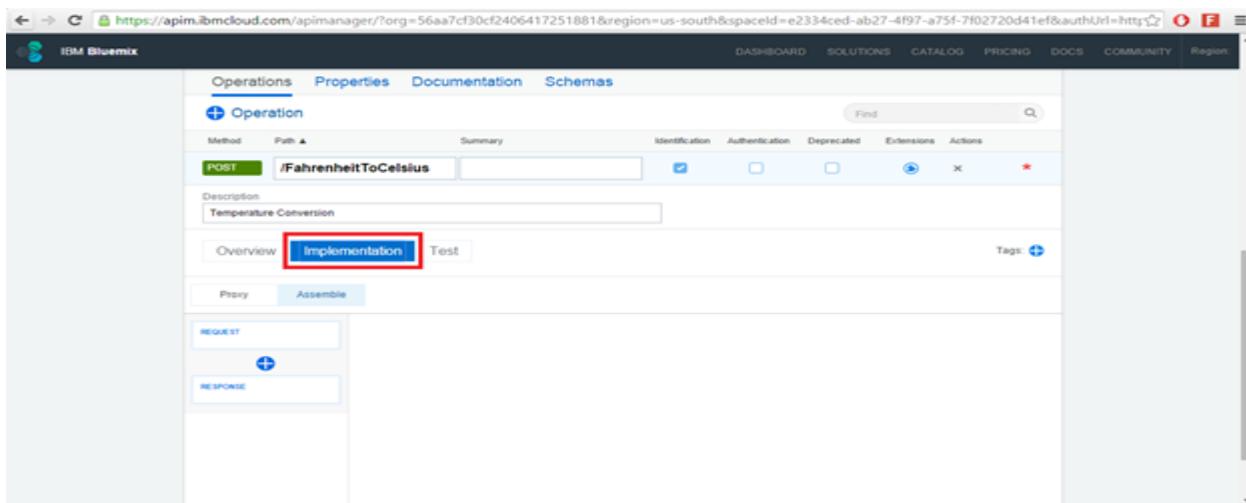
Also Add Response Headers, the request type is SOAP, but we would like to use JSON as the data format. Here 201 will be the default Response code that we need to change to **200** to add the success response.

The screenshot shows the IBM Bluemix API Manager interface. In the top navigation bar, the URL is https://apim.ibmcloud.com/apimanager/?org=56aa7cf30cf2406417251881®ion=us-south&spaceId=e2334ced-ab27-4f97-a75f-7f02720d41ef&authUrl=http://. The page title is "Operations". The main content area shows a POST operation named "/FahrenheitToCelsius". The "Response" tab is selected, and the status code field contains "200".

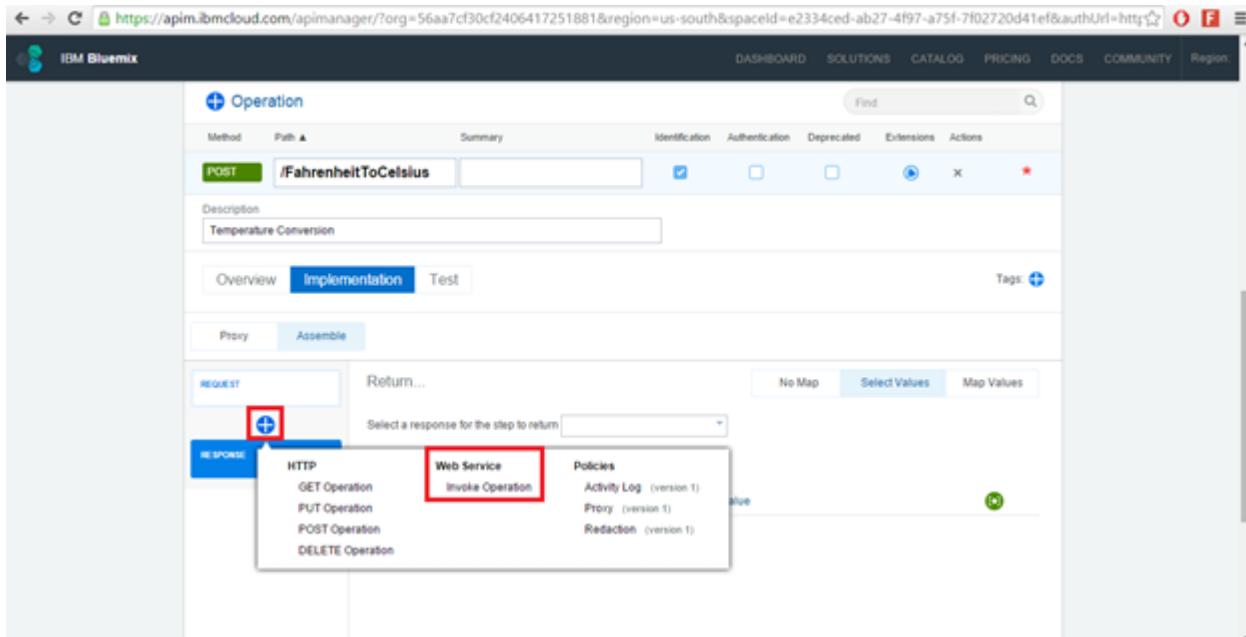
Here you need to define your own JSON Response type. Open the Response "**200 Status**" and add the Response Header format as { "Celsius" : "65" }

The screenshot shows the IBM Bluemix API Manager interface. In the top navigation bar, the URL is https://apim.ibmcloud.com/apimanager/?org=56aa7cf30cf2406417251881®ion=us-south&spaceId=e2334ced-ab27-4f97-a75f-7f02720d41ef&authUrl=http://. The page title is "201". The main content area shows a "Header" section with a Swagger Schema Object field containing "0" and a Response Example field containing a JSON object with "CelsiusTemperature": "65".

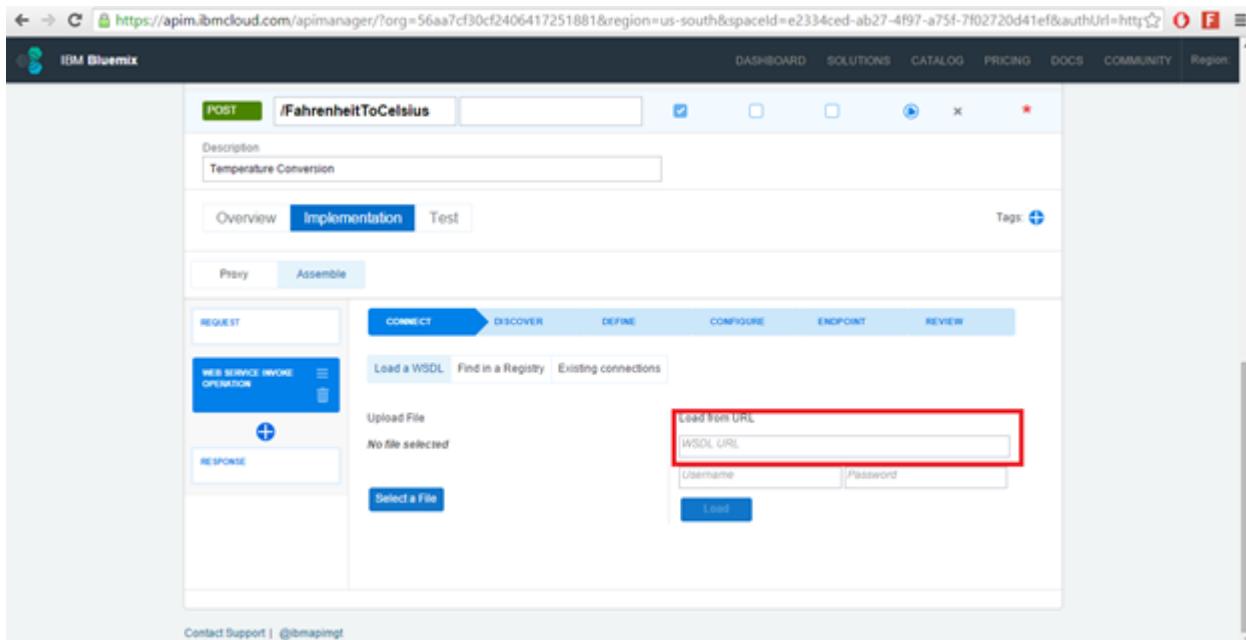
Switch back to API Manager Console, and click on **Implementation** tab for invoking the web Service.



Click the “+” icon to add a task to the resource implementation, and select **Invoke Operation** from the **web Service** section. This will be used to invoke the Temperature Conversion web Service.



You are now on the **CONNECT** tab for the assembly task. Populate the fields as shown in the table below. The WSDL URL can be copied and pasted from the other browser tab where you viewed the WSDL. When it is completed, click the **Load** button.



Load the WSDL URL for Fahrenheit to Celsius Temperature Conversion.

Field Name	Value
WSDL URL	http://webservice.daehosting.com/services/TemperatureConversions.wso?WSDL
Username	Leave blank
Password	Leave blank

[Overview](#)
[Implementation](#)
[Test](#)

Tags: +

[Proxy](#)
[Assemble](#)

The screenshot shows the Implementation module's interface. On the left, there's a REQUEST panel with a WEB SERVICE INVOKE OPERATION section. The main area has tabs: CONNECT (highlighted in blue), DISCOVER, DEFINE, CONFIGURE, ENDPOINT, and REVIEW. Under the DISCOVER tab, there are three ways to load a WSDL: Load a WSDL, Find in a Registry, and Existing connections. Below these are fields for Upload File (with 'No file selected') and Load from URL (with the URL 'http://s.daehosting.com/services/TemperatureConversions.wso?WSDL'). There are also fields for Username and Password, and a red box surrounds the 'Load' button.

In the **DISCOVER** tab of the implementation module, the WSDL which has been provided has been parsed and presented in a form which allows you to select the fields you wish to expose in the **CONFIGURE** stage. In this case you wish to map only one field i.e. **Fahrenheit** into the request message for the Conversion Temperature operation of the service.

[Overview](#)
[Implementation](#)
[Test](#)

Tags: +

[Proxy](#)
[Assemble](#)

The screenshot shows the Implementation module's interface. On the left, there's a REQUEST panel with a WEB SERVICE INVOKE OPERATION section. The main area has tabs: CONNECT, DISCOVER (highlighted in blue), DEFINE, CONFIGURE, ENDPOINT, and REVIEW. Under the DISCOVER tab, there are sections for Available Objects and Available Fields. Available Objects include TemperatureConversions, TemperatureConversionsSoap, CelsiusToFahrenheit, FahrenheitToCelcius, WindChillInCelcius, WindChillInFahrenheit, and TemperatureConversionsSoap12. Available Fields includes columns for Use, Field, Type, and Sample Data. A red box highlights the 'FahrenheitToCelcius' field in the Available Fields table, and another red box highlights the 'Discover' button in the top navigation bar.

Use	Field	Type	Sample Data
	body	object	...
	FahrenheitToCelcius	object	...
<input checked="" type="checkbox"/>	nFahrenheit	number	...
	header	object	...
	headers	object	...
	Security	object	...
	UsernameToken	object	...
	Username	string	...
	Password	string	...

Select **CONFIGURE** to map the input parameter of your API to the input parameter of the Web Service Invoke Operation.



After selecting the available values select the **Map Values** tab you will be able to map the SOAP values to the JSON values in a drag-and-drop fashion.

The screenshot shows the 'Implementation' tab selected. On the left, there's a diagram with 'REQUEST' and 'RESPONSE' boxes, and a central 'WEB SERVICE INVOKE OPERATION' box. The top navigation bar has tabs: CONNECT, DISCOVER, DEFINE, **CONFIGURE**, ENDPOINT, and REVIEW. The 'CONFIGURE' tab is highlighted with a red box. Below it, there are two tabs: 'Select Values' and 'Map Values', with 'Map Values' highlighted with a red box. The main area shows 'Available values' and 'Input variables' tables. A yellow box highlights a transformation mapping between 'Request' and 'body' fields.

Select **Review**. In the review section you can review the configuration as well as set specific actions if an error occurs when the Web Service Invoke Operation is called. For this document you will not take any actions if an error is returned.

Overview Implementation Test Tags: +

Proxy Assemble

REQUEST

WEB SERVICE INVOKE OPERATION

RESPONSE

CONNECT DISCOVER DEFINE CONFIGURE ENDPOINT **▶ REVIEW**

Error Handling

Inputs

- * Request
- * body
- * FahrenheitToCelcius
- 4.5t nFahrenheit

Request > Body > JSON > EnterTemperatureInFahrenheit

Outputs

- * Response
- * body
- * FahrenheitToCelciusResponse
- 4.5t FahrenheitToCelciusResult
- [] * header
- [] * headers

Select **Response tab** in the Implementation Module for Mapping the values to the Response Header.

https://apim.ibmcloud.com/apimanager/?org=56aa7cf30cf2406417251881®ion=us-south&spaceId=e2334ced-ab27-4f97-a75f-7f02720d41ef&authUrl=https://apimanager.apim.ibmcloud.com/authn

IBM Bluemix

DASHBOARD SOLUTIONS CATALOG PRICING DOCS COMMUNITY Region:

POST /FahrenheitToCelcius

Description
Temperature Conversion

Overview Implementation Test Tags: +

Proxy Assemble

REQUEST

WEB SERVICE INVOKE OPERATION

RESPONSE

Return... No Map Select Values Map Values

Select a response for the step to return 200

Inputs

- 123 Status Select Available Value
- * body
- * JSON
- xy_CelsiusTemperature Web Service Invoke Operation > Response > body > Fa... Change [e]
- * Headers

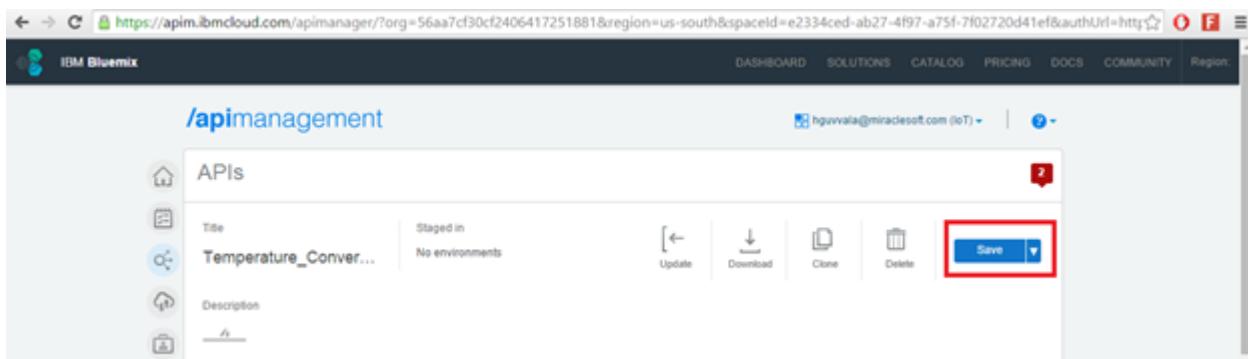
Select **Map Values** to see the alternative way to create mappings using the graphical mapping tool.

Click on the connector for the **FahrenheitToCelsiusResult** output field from the Web Service. Invoke Operation and drag and drop it on the connector for the **FahrenheitToCelsiusResult** field in the API response to map the two fields. Note that the mapping line is green which indicates that the two fields are compatible types and the mapping will succeed.

Map the values for Response as below.

The screenshot shows the MIRACLE software interface for mapping API responses. The top navigation bar has tabs for 'Proxy' and 'Assemble'. The 'Assemble' tab is selected. Below the tabs, there are three main sections: 'REQUEST', 'WEB SERVICE INVOKE OPERATION', and 'RESPONSE'. The 'RESPONSE' section is highlighted with a red box. A dropdown menu labeled 'Return...' is open above the 'RESPONSE' section, with '200' selected. To the right of the dropdown are buttons for 'No Map', 'Select Values', and 'Map Values', with 'Map Values' also highlighted with a red box. The main area shows 'Available values' on the left, 'Transformation' in the center, and 'Input variables' on the right. The 'Available values' section lists fields like 'xy', 'EnterTemperatureInFahrenheit', 'Web Service Invoke Operation', 'Response', 'body', 'FahrenheitToCelsiusResult', 'FahrenheitToCelsiusResult', 'header', 'headers', and 'faults'. The 'Transformation' section contains a mapping diagram where a yellow line connects the 'FahrenheitToCelsiusResult' field in the 'Available values' list to the 'FahrenheitToCelsiusResult' field in the 'Input variables' list. The 'Input variables' section lists 'Status', 'body', 'JSON', 'xy', 'FahrenheitToCelsiusResult' (with a sample value of '51.6666...'), and 'Headers'.

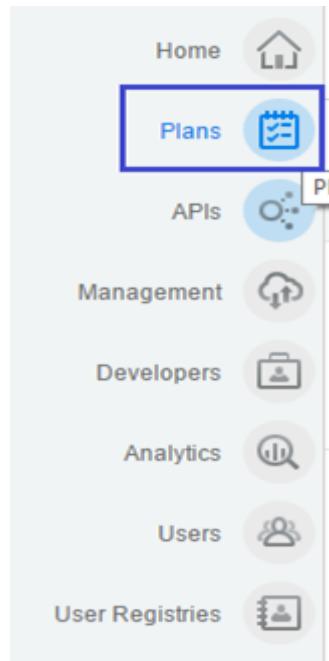
Click **Save** button at the top of the API editor.



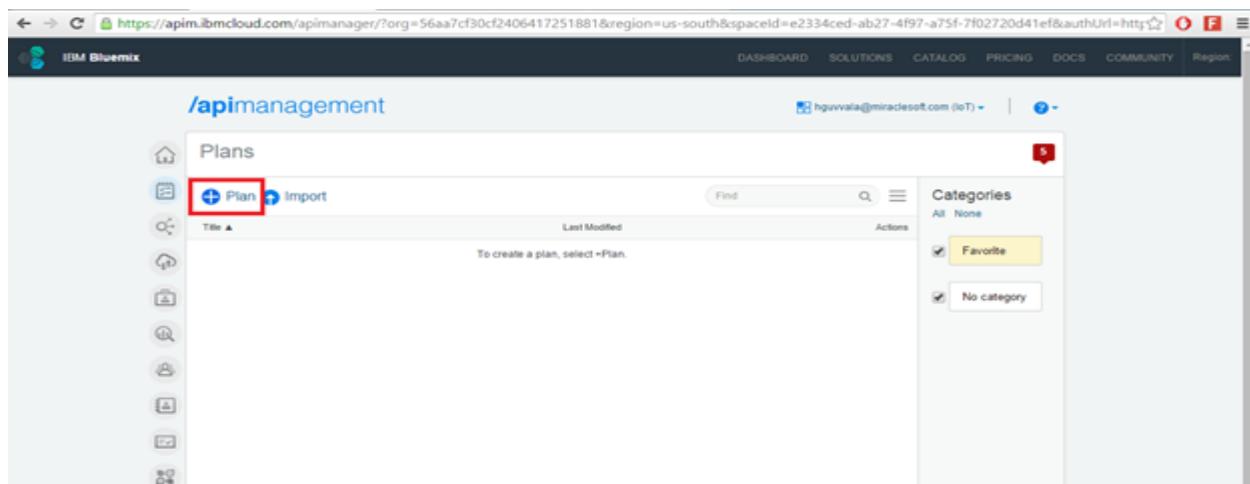
You have now defined your **CurrencyConverter_Mapping** REST API. Before you can test the resource from within the API Manager UI you will need to add it to a Plan.

Now, you need to add this Resource to the existing Sandbox Plan.

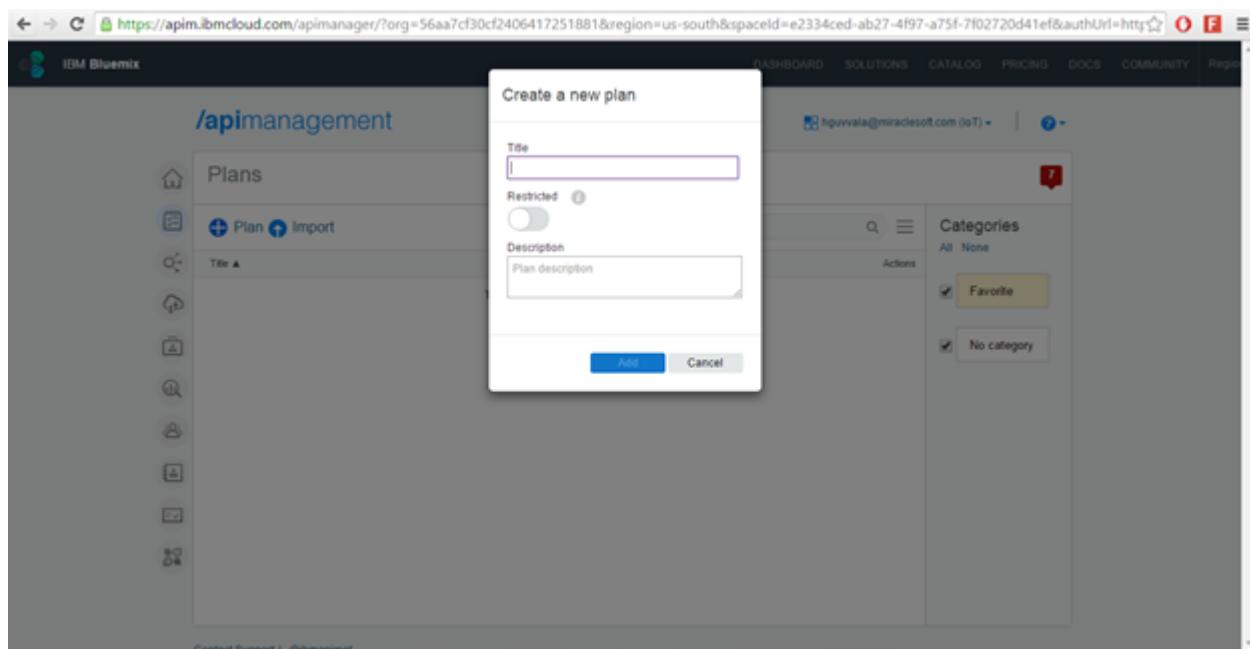
Select **Plans** from the Navigation Pane.

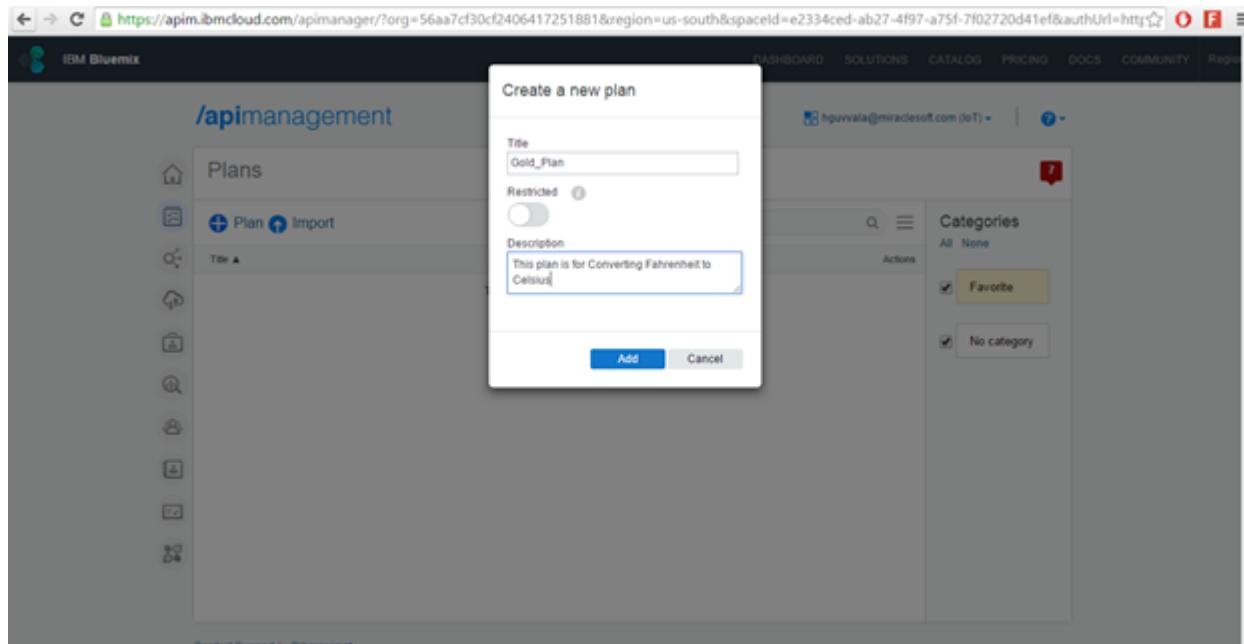


Click on **+Plan** for creating Plan.

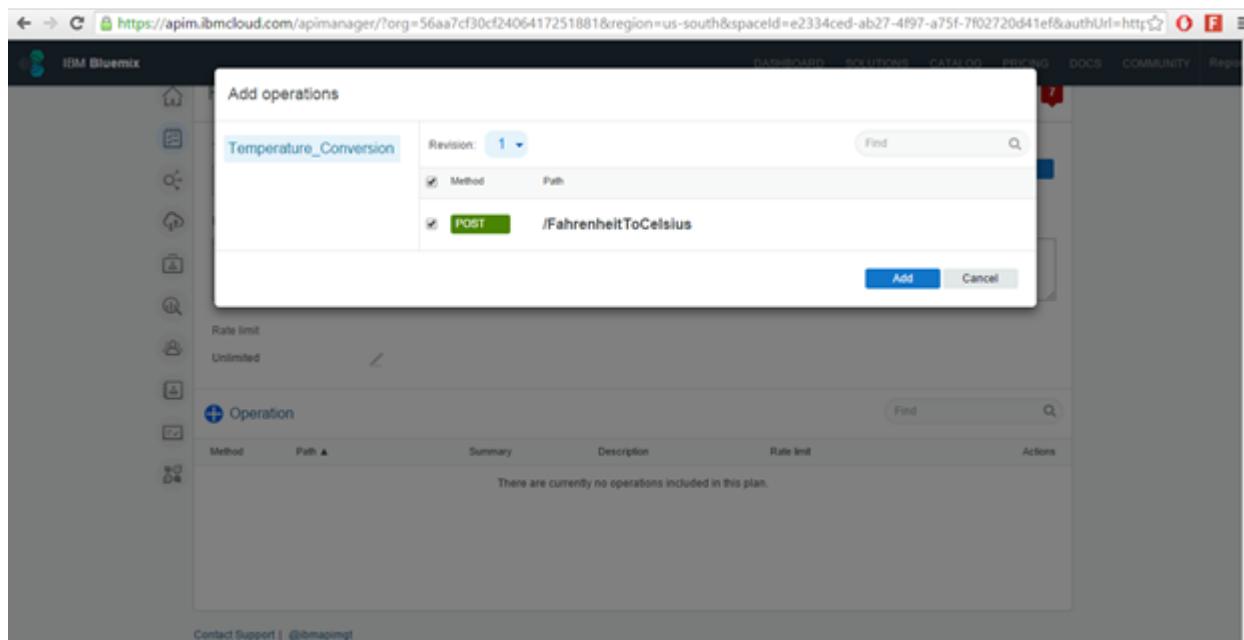


Specify the Plan name and Description for creating the plan.

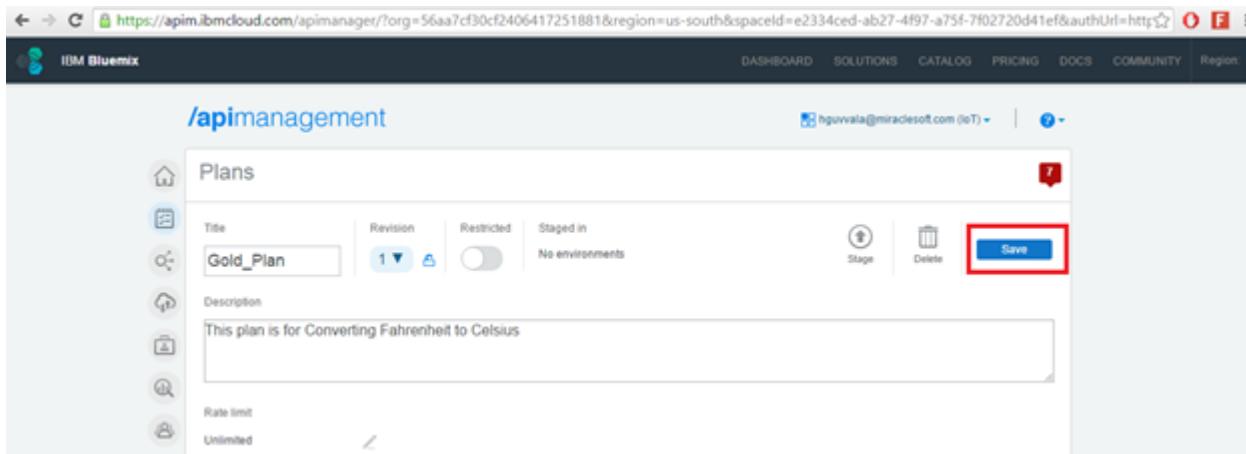




In order to expose the resource you have created you need to add it to the Public plan. Click **+ Operation**.

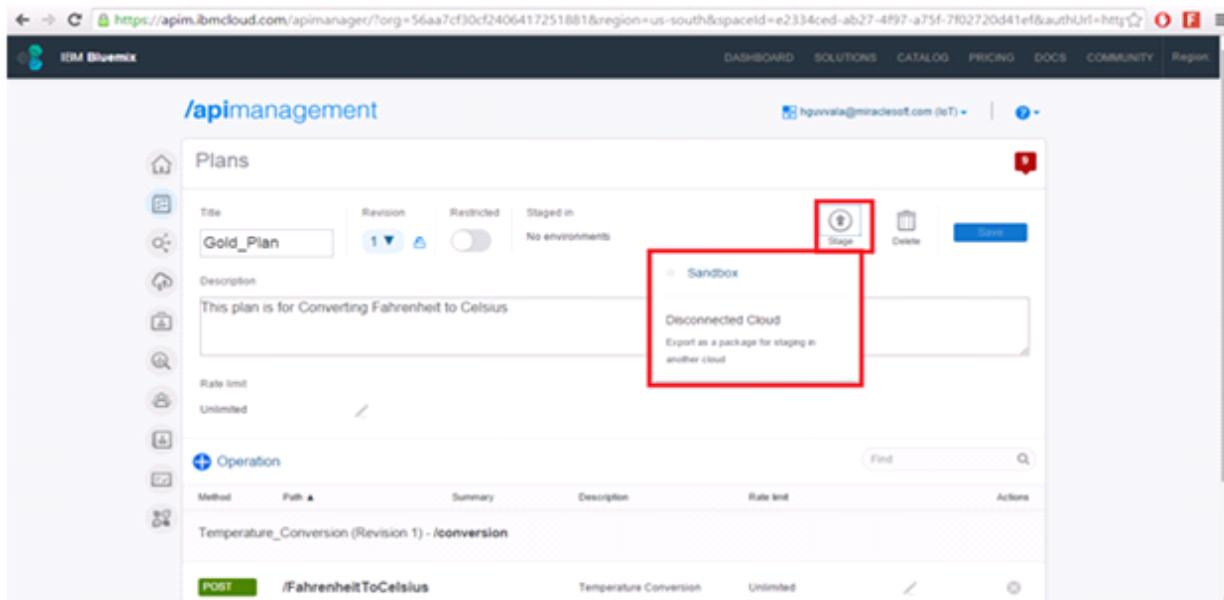


Click on Save to save the Plan.



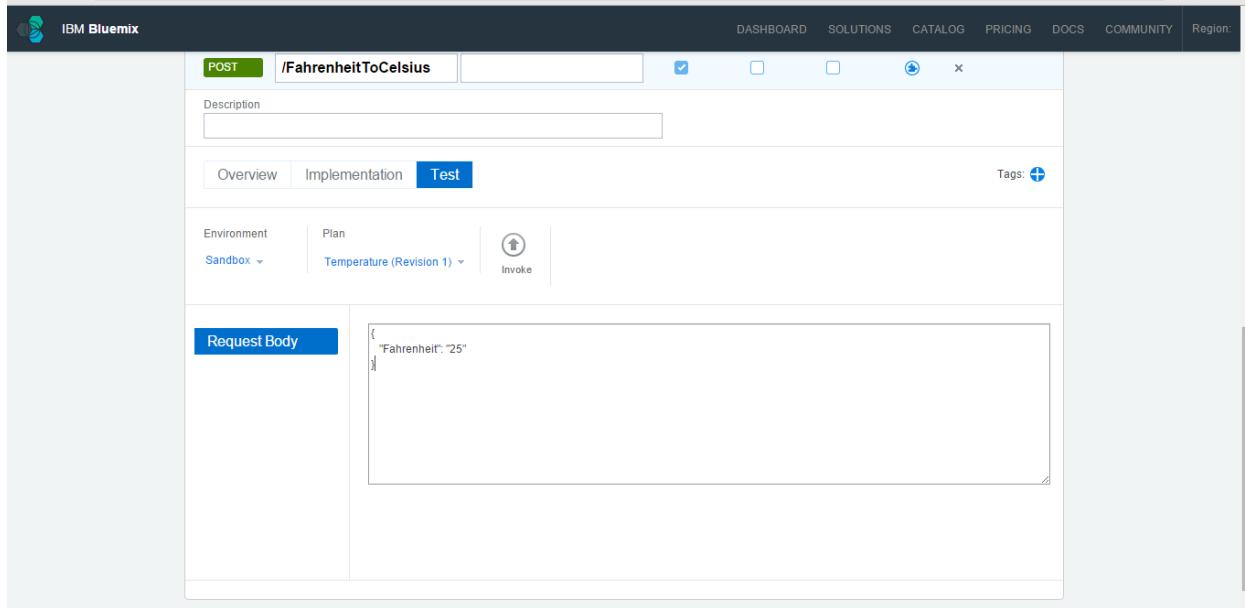
The screenshot shows the IBM Bluemix API Management interface. In the top navigation bar, there are links for DASHBOARD, SOLUTIONS, CATALOG, PRICING, DOCS, COMMUNITY, and Region. The URL in the address bar is <https://apim.ibmcloud.com/apimanager/?org=56aa7cf30cf2406417251881®ion=us-south&spaceId=e2334ced-ab27-4f97-a75f-7f02720d41ef&authUrl=https://>. The main content area is titled '/apimanagement'. On the left, there's a sidebar with icons for Home, API, Application, Environment, Operation, and Help. The 'Plans' section is active, showing a table with one row. The row contains the title 'Gold_Plan', revision '1', a 'Restricted' toggle switch (unchecked), and a note 'Staged in No environments'. Below the table is a description box containing the text 'This plan is for Converting Fahrenheit to Celsius'. At the bottom of the table are 'Stage', 'Delete', and a large blue 'Save' button, which is highlighted with a red box. To the right of the table is a large red 'X' icon.

Now, you need to deploy this application to Sandbox Environment.



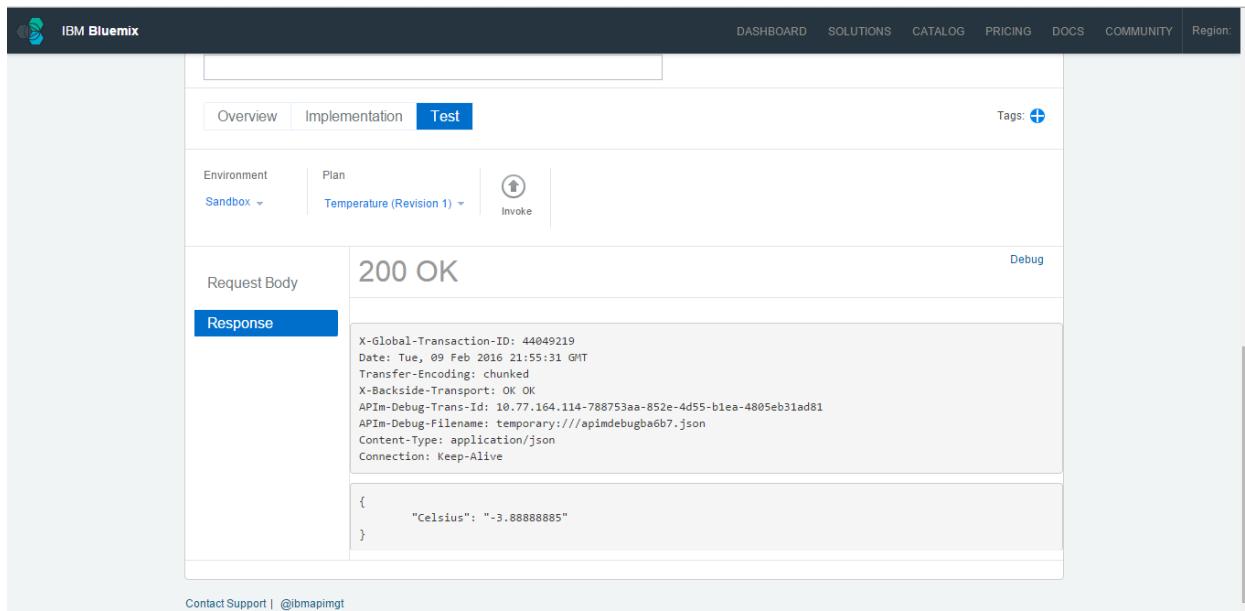
The screenshot shows the same IBM Bluemix API Management interface as the previous one, but with a different focus. The 'Stage' button in the 'Plans' section is highlighted with a red box. A tooltip 'Sandbox' is displayed above the button. A callout box with a red border highlights the 'Disconnected Cloud' section, which contains the text 'Export as a package for staging in another cloud'. The rest of the interface is similar to the first screenshot, showing the 'Operation' section with a single entry for 'Temperature_Conversion'.

To test your REST resource to ensure that it is defined and implemented correctly, **Select APIs from the navigation pane**. Click the **Edit icon** for the POST quote resource. Now, Test your API.



The screenshot shows the IBM Bluemix API Test interface. At the top, there's a header with the IBM Bluemix logo and navigation links: DASHBOARD, SOLUTIONS, CATALOG, PRICING, DOCS, COMMUNITY, and Region. Below the header, a modal window is open for a POST request to the endpoint /FahrenheitToCelsius. The modal has tabs for Overview, Implementation, and Test, with the Test tab selected. Under the Test tab, there are sections for Environment (set to Sandbox) and Plan (set to Temperature (Revision 1)). On the right, there's a button labeled 'Invoke'. In the Request Body section, the JSON payload is shown as: { "Fahrenheit": "25" }.

Click on Invoke to invoke your API.



The screenshot shows the IBM Bluemix API Test interface after invoking the API. The Response tab is selected, displaying the status 200 OK and the response body. The response body contains the converted Celsius value: { "Celsius": "-3.88888885" }. Above the response, the X-Global-Transaction-ID, Date, Transfer-Encoding, X-Backside-Transport, APIM-Debug-Trans-Id, APIM-Debug-Filename, Content-Type, and Connection headers are listed.

Once you get the response, then redirect to API Manager Console. In order to publish your API you need to redirect to Developer Console.

The screenshot shows the IBM Bluemix API Management console at the URL <https://apim.ibmcloud.com/apimanager/>. The page title is "/apiManagement". On the left, there's a sidebar with icons for Home, Environment, API, Data, Functions, Pipelines, and Watson. The main area is titled "Environments" and shows a single entry: "Sandbox APIMOMT_GATEWAY". Below it, there are tabs for "Configuration", "Portal", and "Permissions". The "Configuration" tab is active. The configuration details include:

- Name:** Sandbox
- Sandbox:** (checkbox checked) If enabled, any staging and publishing actions will be forced. If conflicts are found, they will be automatically resolved by the system. Unpublish actions will happen automatically.
- Gateway Cluster:** APIMOMT_GATEWAY
- API Endpoint:** Base URL: <https://api.apim.ibmcloud.com/hguvalamiraclesoftcom-iot/sb>
- Default:** The default environment can be accessed using a shorter URL, omitting the environment URL path segment.
- Path Segment:** sb

Click on **Portal** Tab.

The screenshot shows the same IBM Bluemix API Management console, but the "Portal" tab is now active. The configuration details remain the same as in the previous screenshot. The "Portal" tab section includes:

- Basic Developer Portal:** Portal URL: <https://developer.apim.ibmcloud.com/hguvalamiraclesoftcom-iot/sb>, Customize URL: <https://example.com>, Customize Portal
- Advanced Developer Portal:** URL: <https://example.com>
- Other:** URL: <https://example.com>

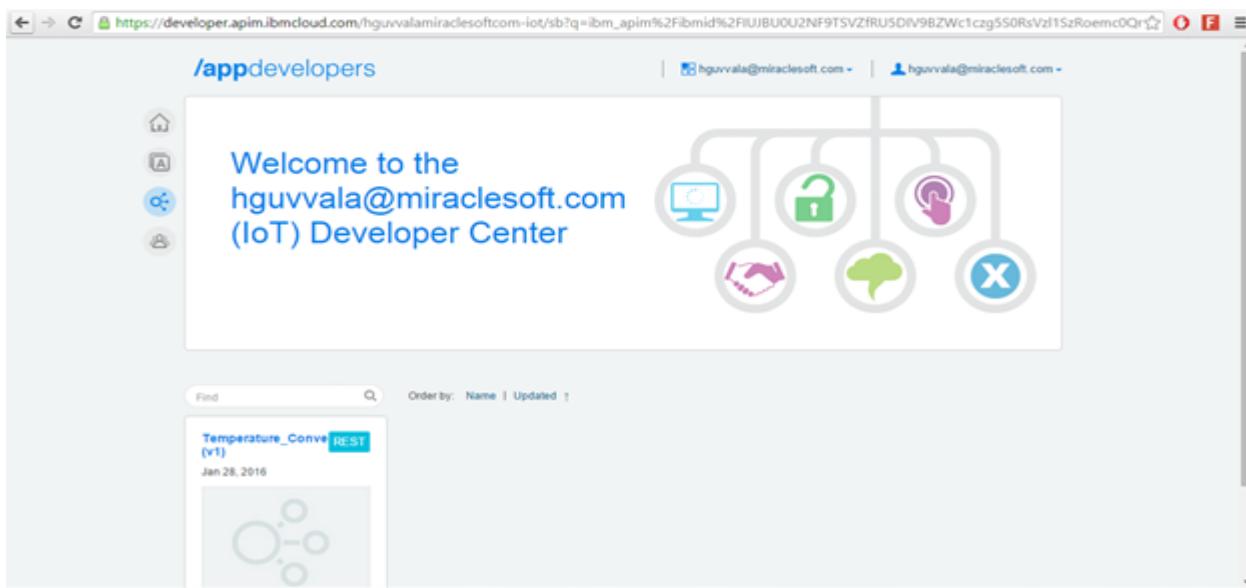
Click on the Basic Developer Portal checkbox and open the URL which was specified.

The screenshot shows the IBM Bluemix API Management interface. On the left, there's a sidebar with icons for environments, applications, databases, and more. The main area is titled '/apimanagement' and shows a list of environments, with 'Sandbox APIMGMT_GATEWAY' selected. Below the environment list are tabs for Configuration, Portal (which is highlighted with a red box), and Permissions. Under the Portal tab, there's a section for 'Basic Developer Portal' which is checked. It displays a 'Portal URL' field containing 'https://developer.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb'. There are also fields for 'Customize URL' (set to 'https://example.com') and 'Customize Portal'. Below these are sections for 'Advanced Developer Portal' and 'Other', both with a URL field set to 'https://example.com'. At the bottom, there's a link 'User Registration and Invitation'.

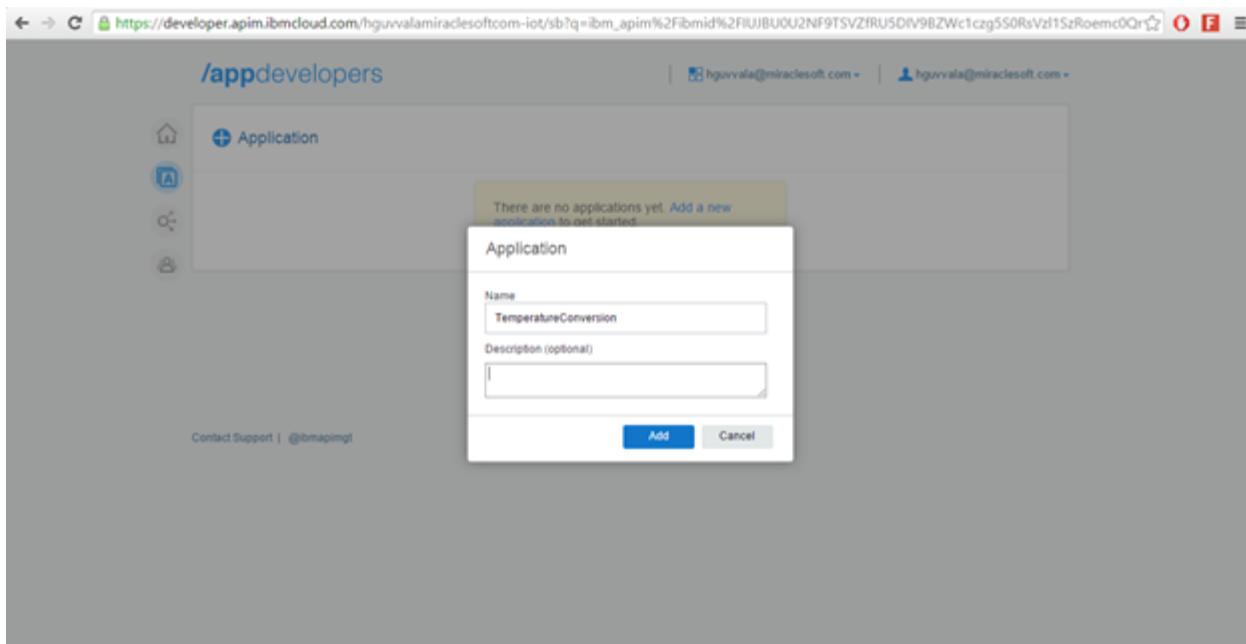
After clicking the URL you will be redirecting to API Developer Portal. Click on **SignIn**, it will automatically synchronize your credentials into Developer Portal and lets you login into that.

The screenshot shows the API Developer Portal at 'https://developer.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb#/.apis/appdevelopers'. The page has a header with 'Sign in' and 'Join' buttons. The main content area says 'Welcome to the hguvvala@miraclesoft.com (IoT) Developer Center'. To the right is a graphic of interconnected circles with icons for a computer monitor, a lock, a microphone, a handshake, a brain, and an 'X'. Below this are sections for 'Find' and 'Order by: Name | Updated'. A small preview window shows a card for 'Temperature_Conver (v1)' with a 'RESET' button and a date 'Jan 28, 2016'.

Click on **SignIn**.



Click on **+Application** and specify Application Name and Description for the API and click Add Button.



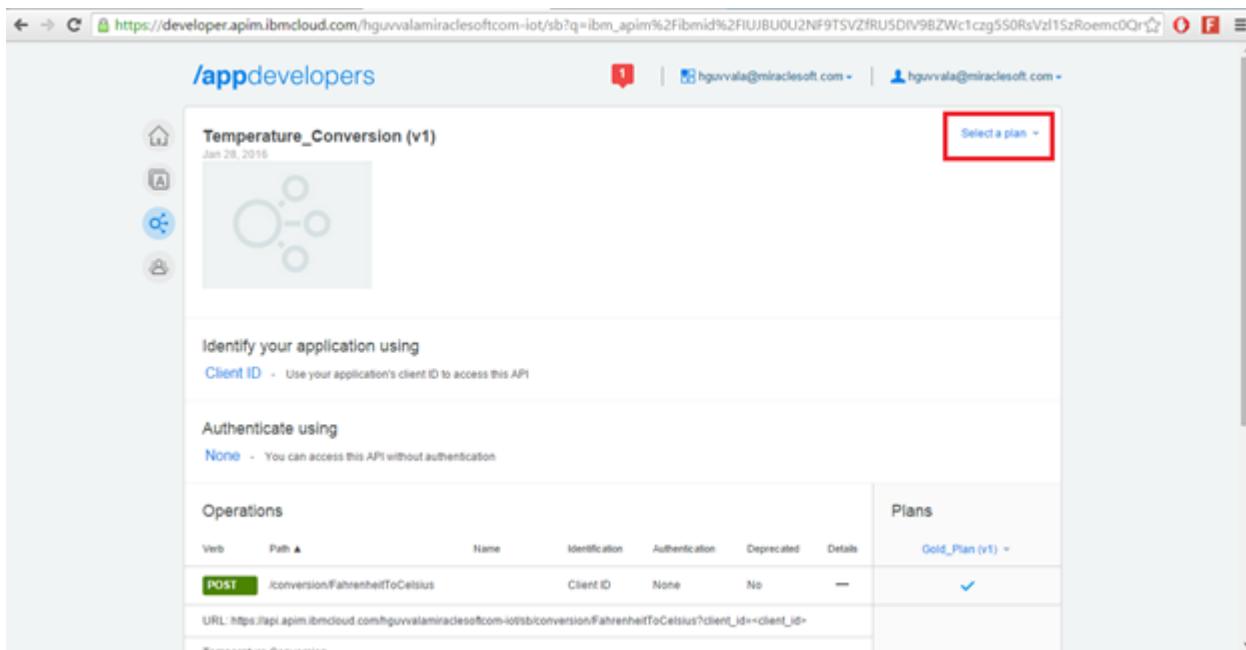
Click on the required API and copy the client ID as shown below.

The screenshot shows the IBM API Management developer portal at https://developer.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb?q=ibm_apim%2Fibmid%2FIUJB002NF9TSVZFRU5D1V9BZWc1czg550RsVzl1SzRoemc0Qr. The left sidebar has icons for Home, Applications, Catalog, and User. The main area is titled '/appdevelopers' and shows an application named 'TemperatureConversion'. A red box highlights the 'Client ID' input field, which contains a long string of characters. Below it is a 'Client Secret' field with a placeholder 'Click show to retrieve the client secret' and 'Show' and 'Reset' buttons.

Go back to Developer Portal Console, Click on API's in Navigation Pane and open Temperature_Conversion API.

The screenshot shows the IBM API Management developer portal at https://developer.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb?q=ibm_apim%2Fibmid%2FIUJB002NF9TSVZFRU5D1V9BZWc1czg550RsVzl1SzRoemc0Qr. The left sidebar has icons for Home, Applications, Catalog, and User. The main area is titled '/appdevelopers' and shows an API named 'Temperature_Conversion (v1)'. Below it is a 'Select a plan' dropdown. The API details section includes fields for 'Client ID' (with a placeholder 'Use your application's client ID to access this API!') and 'Authentication' (set to 'None'). The 'Operations' table lists a single endpoint: a POST method for '/conversion/FahrenheitToCelsius' with 'Client ID' as the required parameter, 'None' authentication, and 'No' deprecation status. The 'Plans' column shows a 'Gold_Plan (v1)' assigned to this endpoint.

For invoking the API in Developer Portal, you need to add plan by clicking on **Select a plan** option.



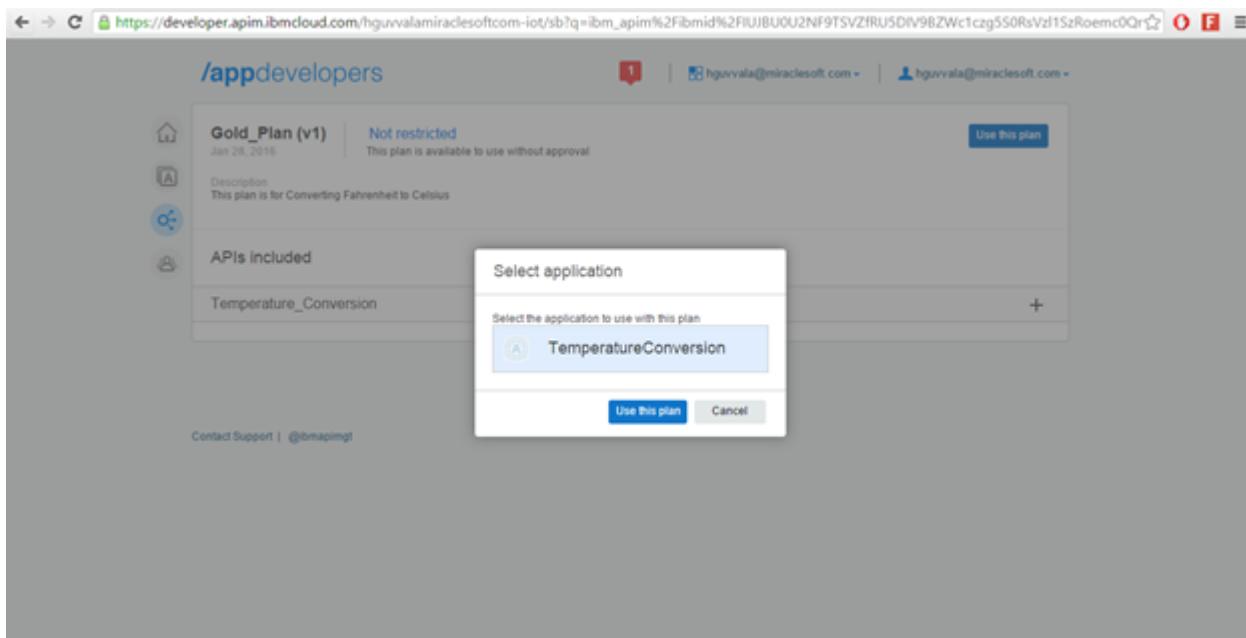
The screenshot shows the IBM API Management Developer Portal interface. At the top, there's a navigation bar with icons for back, forward, search, and user authentication. The URL in the address bar is https://developer.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb?q=ibm_apim%2Fibmid%2FIUJB0U2NF9TSVZfRU5DfV9BZWc1czg550RsVzI5zRoemc0Qr. Below the address bar, the page title is **/appdevelopers**.

The main content area displays the **Temperature_Conversion (v1)** API, last updated on Jan 28, 2016. It includes a small icon representing the API. To the right of the icon, a red box highlights the **Select a plan** button.

Below the API title, there are sections for identifying the application using Client ID and authenticating using None. The Operations table lists a single endpoint:

Operations	Plans
Verb Path Name Identification Authentication Deprecated Details	Gold_Plan (v1) ✓
POST /conversion/FahrenheitToCelsius Client ID None No —	
URL: <a href="https://api.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb/conversion/FahrenheitToCelsius?client_id=<client_id>">https://api.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb/conversion/FahrenheitToCelsius?client_id=<client_id>	

After selecting the plan, click on the application to which you are going to add, to the plan and then click “Use this plan”.



The screenshot shows the IBM API Management Developer Portal interface. The URL in the address bar is https://developer.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb?q=ibm_apim%2Fibmid%2FIUJB0U2NF9TSVZfRU5DfV9BZWc1czg550RsVzI5zRoemc0Qr. The page title is **/appdevelopers**.

The main content area displays the **Gold_Plan (v1)** plan, last updated on Jan 28, 2016. The plan is described as "Not restricted" and "This plan is available to use without approval". A blue button labeled "Use this plan" is located in the top right corner of this section.

Below the plan details, there's a section for "APIs included" showing the **Temperature_Conversion** API. A modal dialog box titled "Select application" is open, listing "TemperatureConversion" as the selected option. The dialog also has "Use this plan" and "Cancel" buttons at the bottom.

After adding the Application to the API, Click on API's and open Temperature_Conversion API. Test the API to get the response.

Click on plus(+) symbol to show the entire URL as shown in the figure.

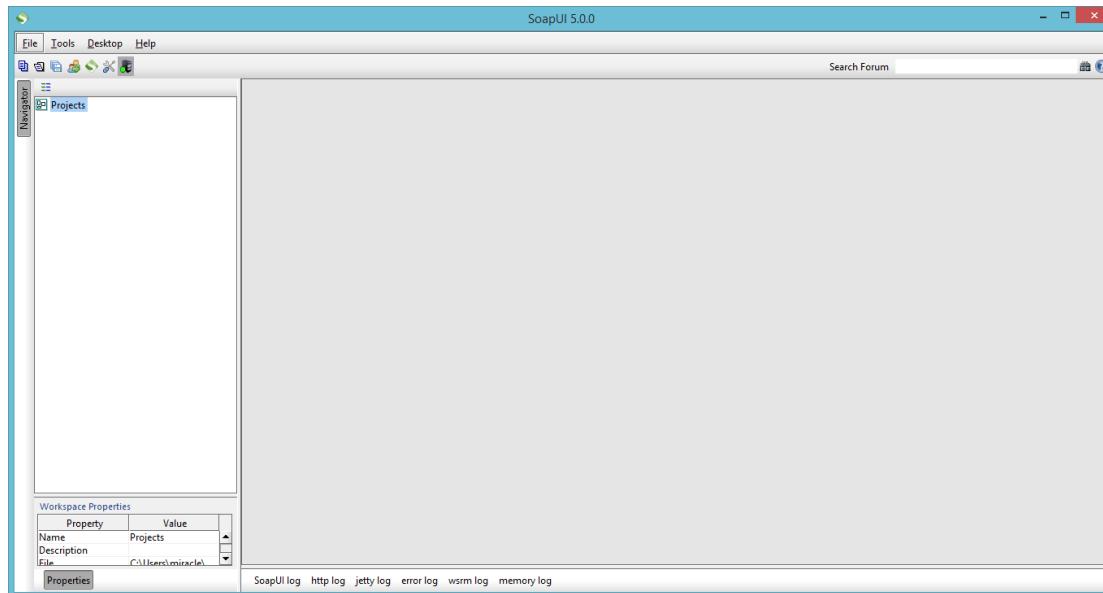
The screenshot shows the IBM API Management interface. The URL in the address bar is https://developer.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb?q=ibm_apim%2Fibmid%2FIUJBU0U2NF9TSVZfRU5DIV9BVytBWIE1ZzIKRXIYWDQzMnpHK. The page title is /appdevelopers. The main content area displays the 'Temperature_Conversion (v1)' API, which was created on Feb 1, 2016. It includes sections for identifying the application using Client ID and authenticating using None. The Operations table lists a single endpoint: POST /conversion/FahrenheitToCelsius. The 'Details' column for this endpoint has a red box around the '+' button, indicating it can be expanded. The Plans section shows 'Gold_Plan (v1)'. At the bottom left, there is a link to Contact Support (@ibmapimg).

Here is the URL, which has to be consumed by Node.js Application. Copy the link.

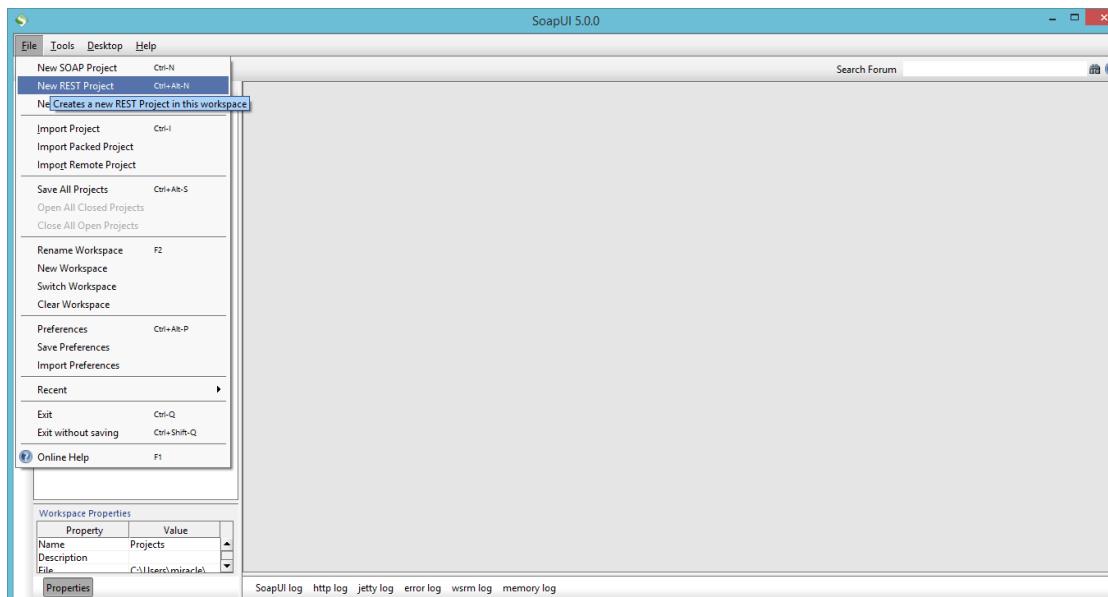
This screenshot shows the same IBM API Management interface as the previous one, but with the URL expanded. The URL https://api.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb/conversion/FahrenheitToCelsius?client_id=<client_id> is highlighted with a red box. The rest of the interface remains the same, showing the API details and operations table.

Replace the <client_Id> of the URL with the client id generated in the previous step. You can test that URL in SOAPUI. Double click on SOAPUI icon to open it.

The SOAPUI Dashboard will appear like this,



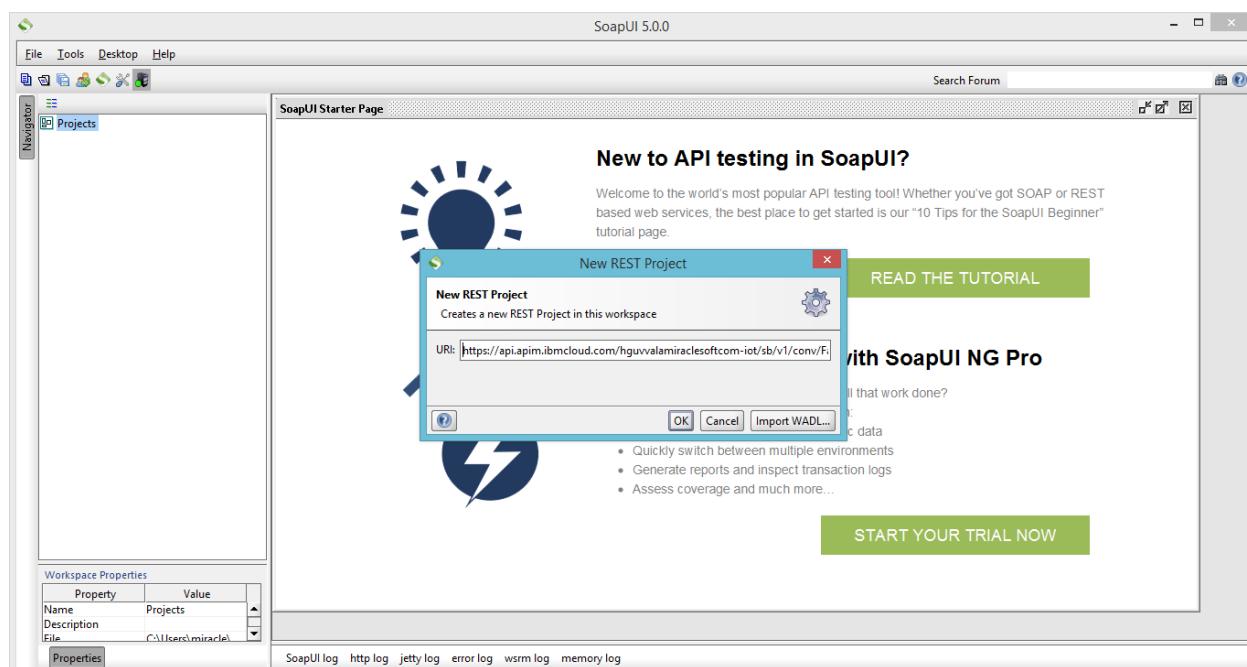
Create a new REST project from the File menu by choosing the "New REST Project" option in the File menu.



Specify the following API Management URL in the Service Endpoint Field

https://api.apim.ibmcloud.com/hguvvalamiraclesoftcom-iot/sb/v1/conv/FahrenheitToCelsius?client_id=<client-id>

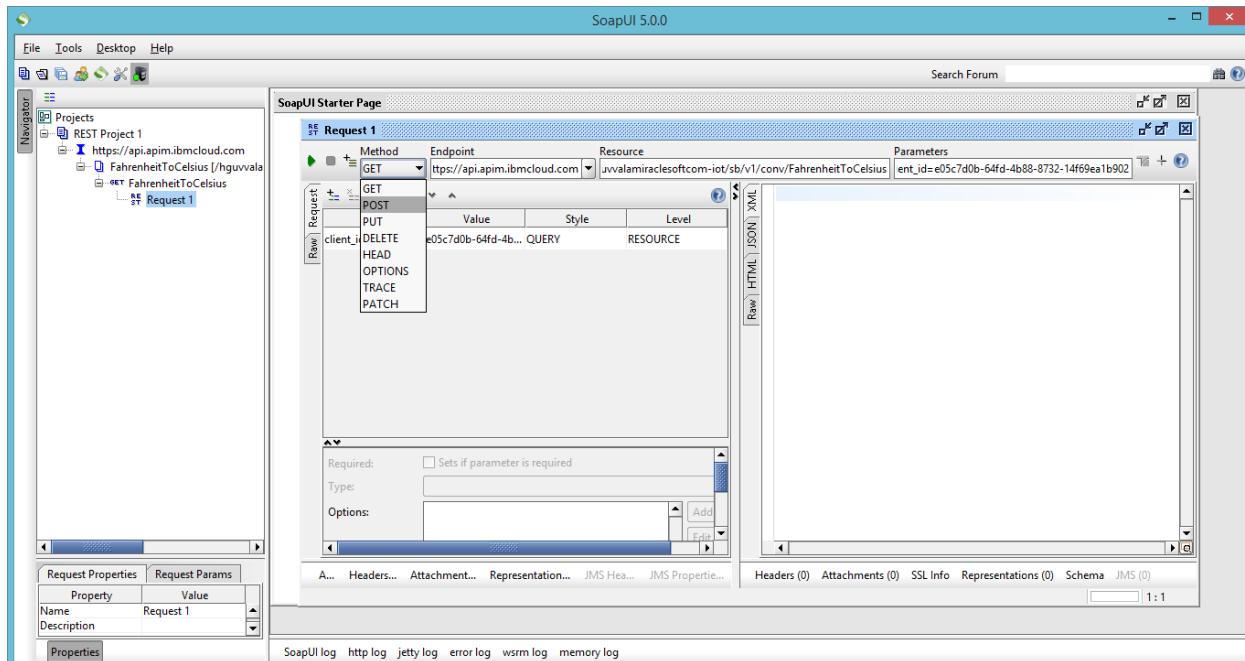
Specify the Client ID which was generated in IBM Developer Portal and the process okay.



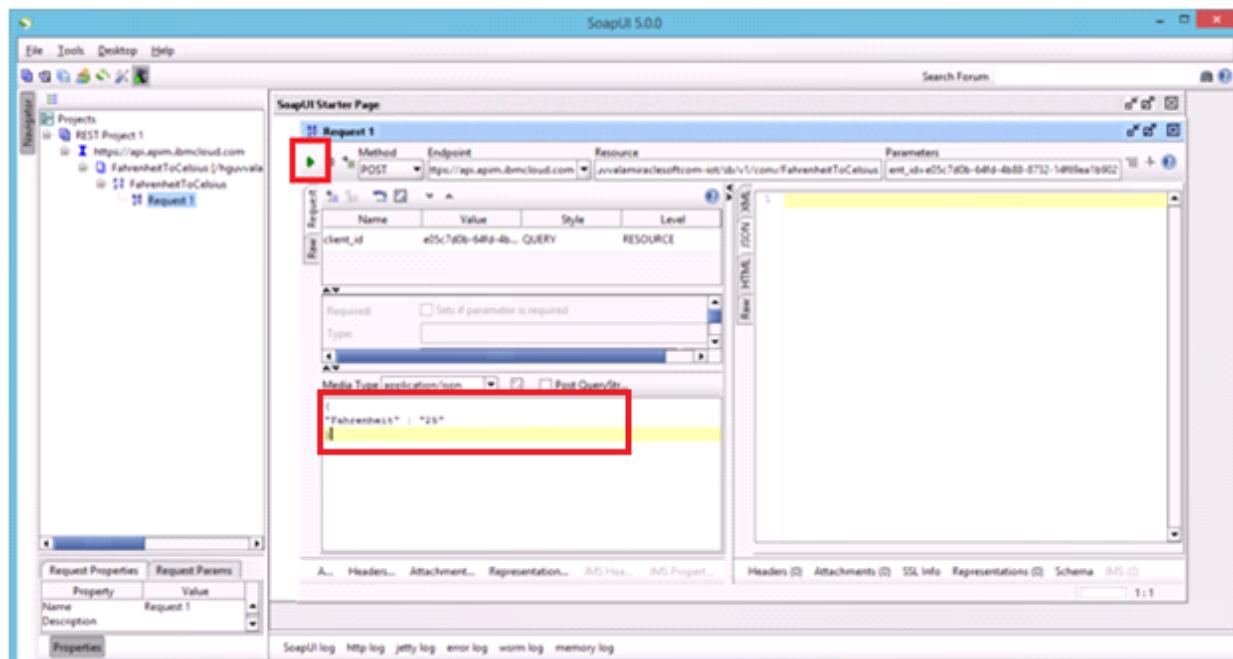
SOAPUI creates the project completely with a Service, Resource, Method and the actual Request and opens the Request editor.

By default, it will give GET method. Click on drop down list to change it to POST method.

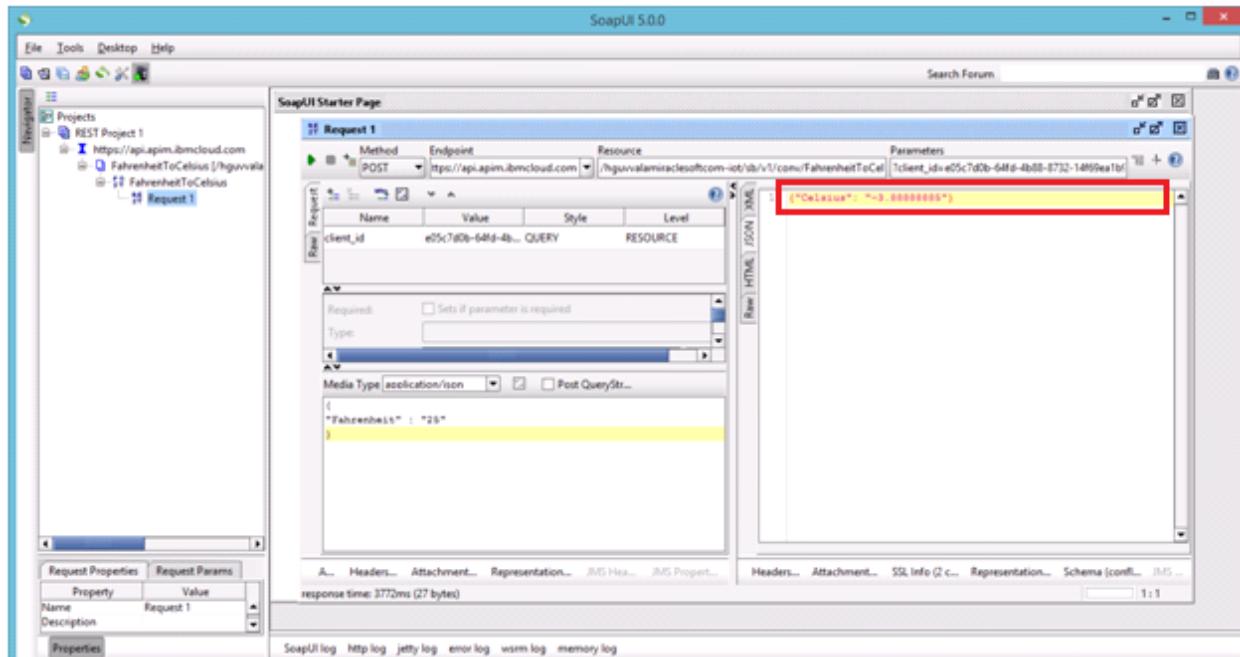
Click on JSON tab in the Right Navigation Pane for getting the result in JSON format.



Enter the Request Data in JSON format as { “Fahrenheit” : “25” }

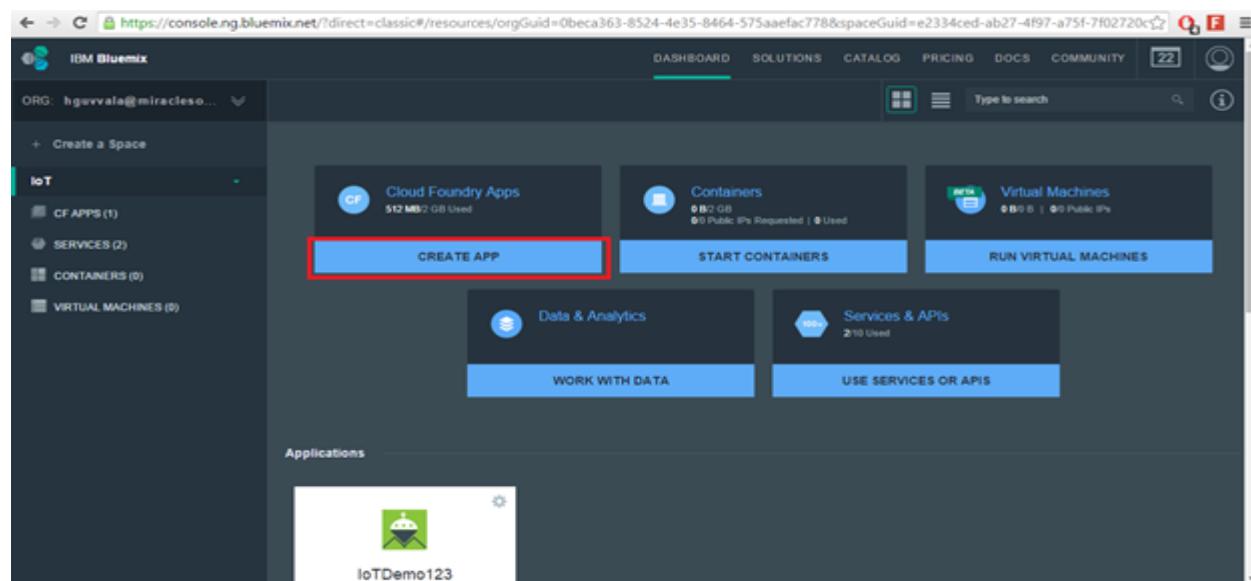


Press the green arrow at the top left in the Request editor and you can see the JSON output returned by the service.

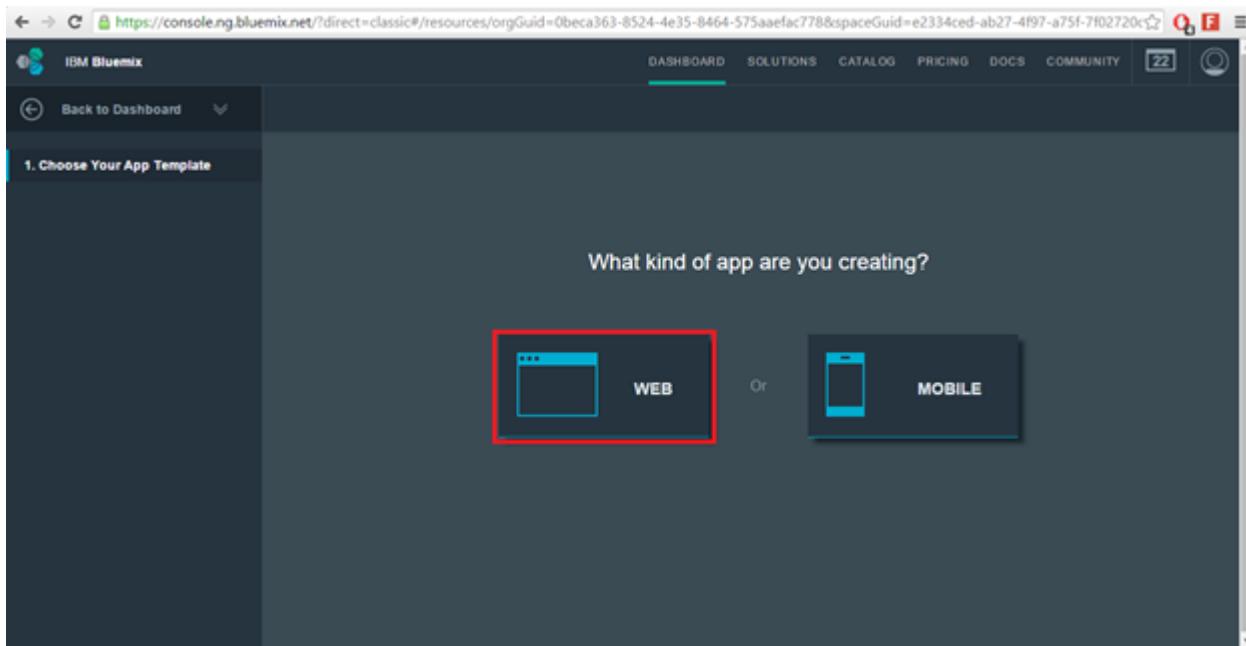


#5 | Create Application in Bluemix

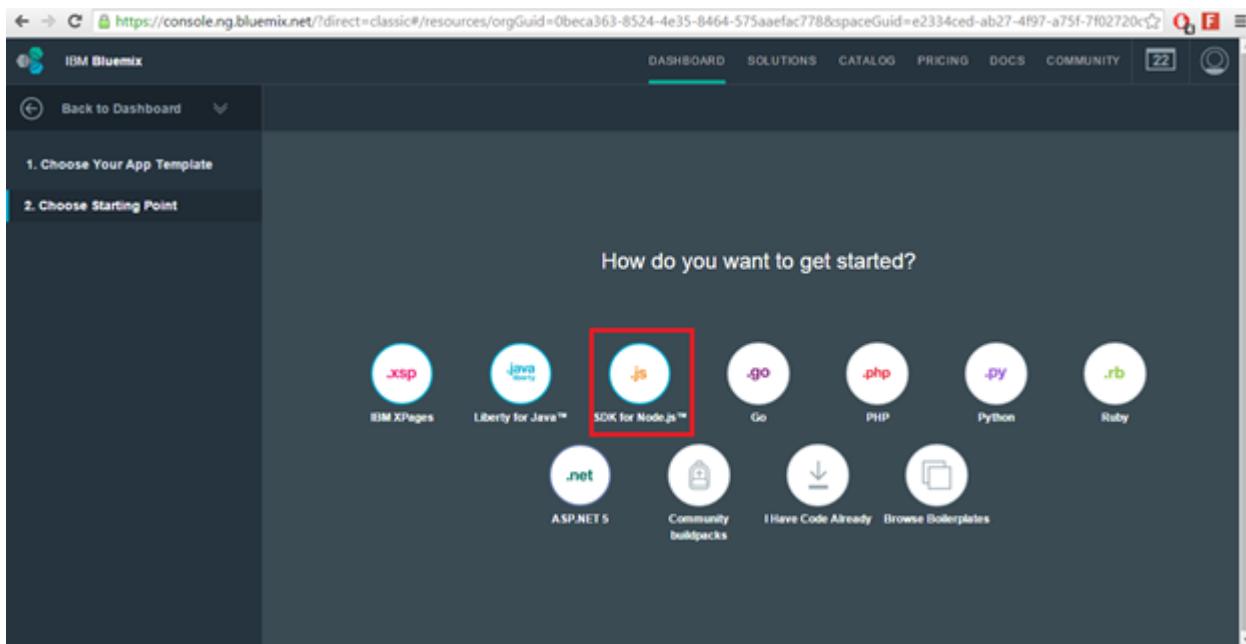
Click on the Cloud Foundry Apps module on the Dashboard to start creating the application.



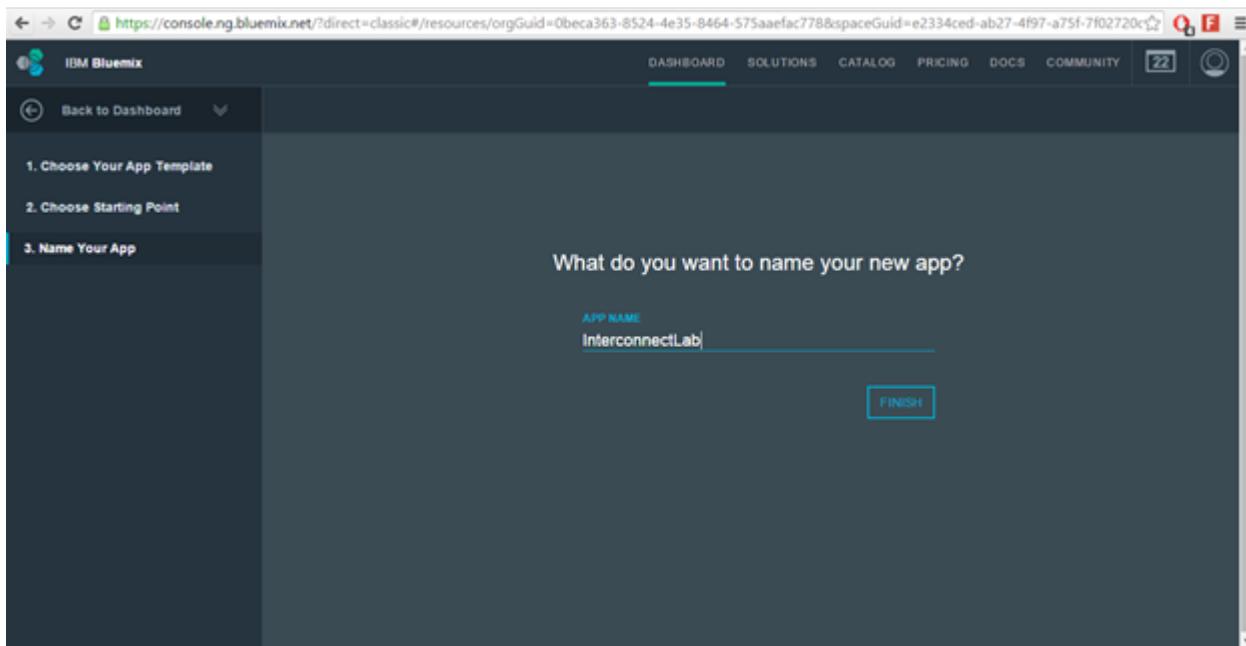
When prompted for the kind of application you want to create, please select Web Application.



Select “SDK for Node.js” in the page.

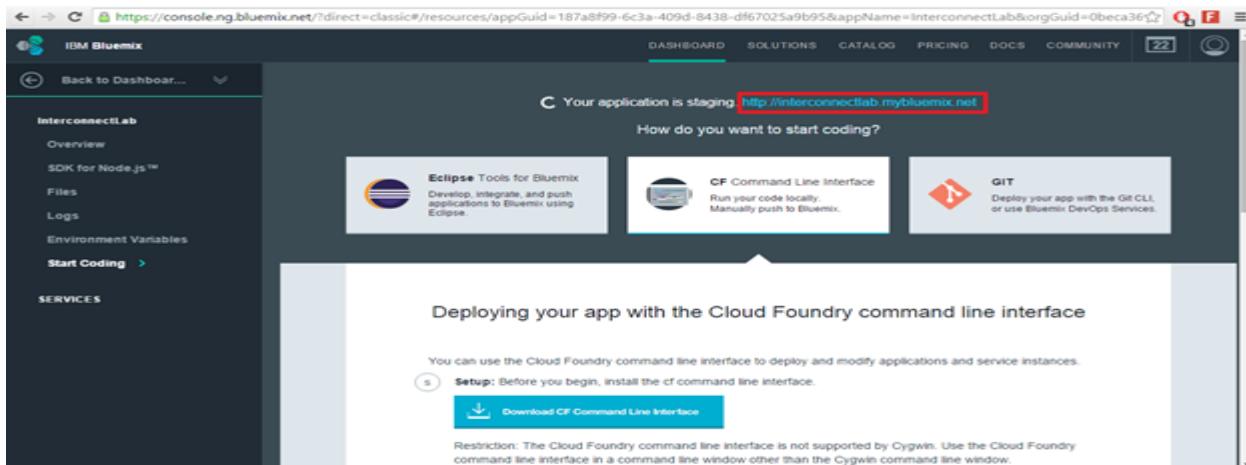


Click on continue and enter an application name and click finish. Application names must be unique as they will be on a public domain.

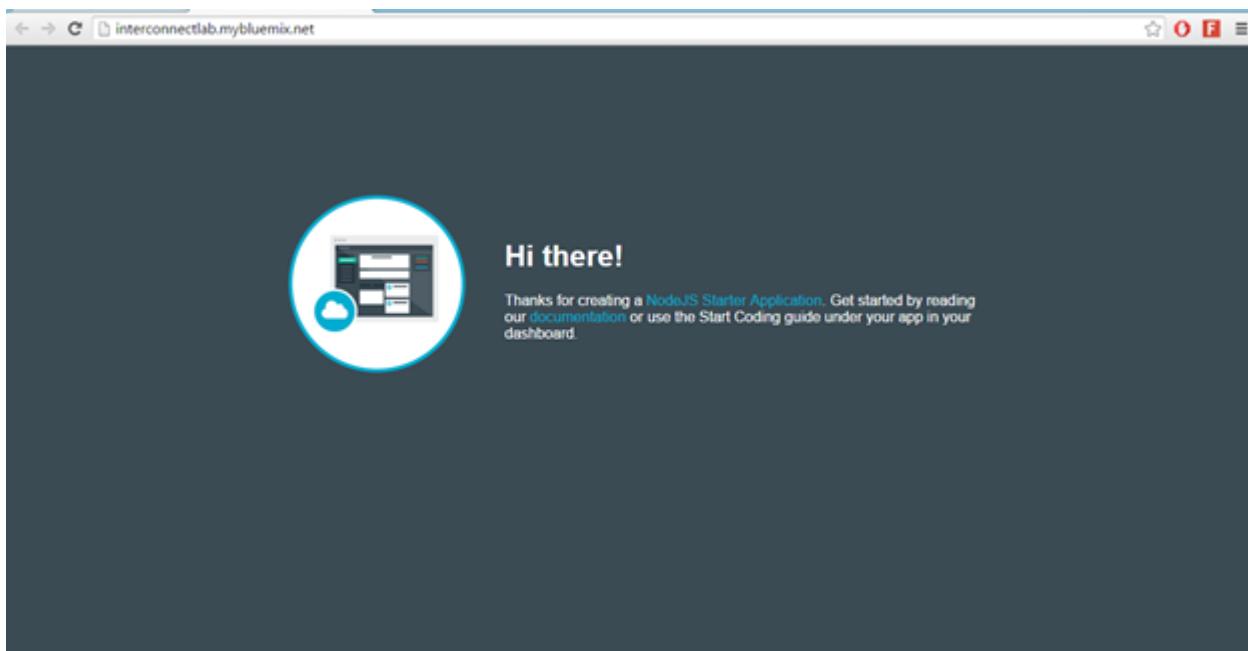


Note : Once created application will take about 2 minutes for staging and starting the application.

After creating the application you can use the application URL and view the sample application in the Browser. Click on the URL given on the page.



After clicking the URL you can view the page as below.



#6 | Download and Edit Starter Code

At this point, you will be asked how you want to start coding your application. For this lab, you will be using the Cloud Foundry (CLI) option. Download and Install the **CF CLI Tool** (any version) as the next step.

The screenshot shows the IBM Bluemix dashboard with the 'DASHBOARD' tab selected. On the left, there is a sidebar with 'InterconnectLab' and 'Start Coding >'. The main area has three cards: 'Eclipse Tools for Bluemix' (Develop, integrate, and push applications to Bluemix using Eclipse), 'CF Command Line Interface' (Run your code locally. Manually push to Bluemix), and 'GIT' (Deploy your app with the Git CLI, or use Bluemix DevOps Services). The 'CF Command Line Interface' card is highlighted. Below it, the text 'Deploying your app with the Cloud Foundry command line interface' is displayed, along with instructions and a 'Download CF Command Line Interface' button. A note states: 'Restriction: The Cloud Foundry command line interface is not supported by Cygwin. Use the Cloud Foundry command line interface in a command line window other than the Cygwin command line window.' At the bottom, there is a numbered list: '1. Download your starter code.' followed by a 'Download Starter Code' button.

For this, Click on “Download CF Command Line Interface” then it will open a new webpage, where you can download your CF installer file. Select the Installer file which suits your OS.

The screenshot shows the GitHub releases page for the 'cloudfoundry / cli' repository. The 'Releases' tab is active. A green button labeled 'Latest release' points to version v6.15.0. Below the release notes, there's a section titled 'Installers' with a list of supported operating systems. The 'Windows 64 bit' option is highlighted with a red rectangle.

Then you will get a zip file. After extracting the zip file, you can find a .exe file inside. **Install it.** To check whether **CF** is installed properly or not, open command prompt and execute CF command. Then it will show you a set of **CF** commands, which indicates that CF is successfully installed on your machine.

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\miracle>cf
NAME:
  cf - A command line tool to interact with Cloud Foundry

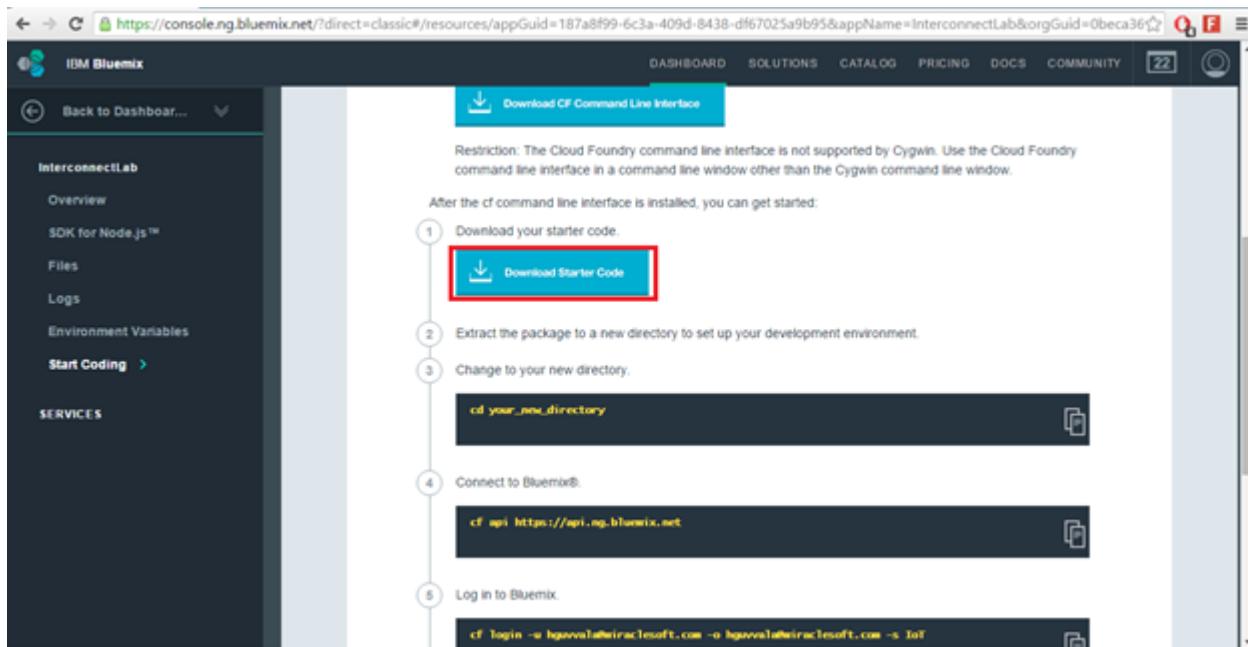
USAGE:
  [environment variables] cf [global options] command [arguments...] [command options]

VERSION:
  6.5.1-4aaf45f-2014-08-27T20:43:36+00:00

BUILD TIME:
  2016-02-09 04:33:58.9440598 -0500 EST

GETTING STARTED:
  login, l                               Log user in
  logout, lo                             Log user out
  passwd, pw                           Change user password
  target, t                            Set or view the targeted org or space
  api                                 Set or view target api url
  auth                                Authenticate user non-interactively
```

Once you have CF CLI running on your machine go ahead and download the Starter Application code for your application. After the application is downloaded, save to your working directory and extract the zip download (unzip).



#7 | Let's Start Coding

Once you have the started files, go to your application folder in your local file system. Delete the index.html file which is present in **Public** folder. Our goal will be to modify some pages in the downloaded starter code folder,

Open “**package.json**” file using an editor (Notepad++). Replace the code in “**package.json**” file with the following content,

```
{  
  "name": "NodejsStarterApp",  
  "version": "0.0.1",  
  "description": "A sample nodejs app for Bluemix",  
  "scripts": {
```

```
"start": "node app.js"
},
"dependencies": {
    "cfenv": "^1.0.3",
    "ejs": "^2.4.1",
    "express": "4.12.x",
    "https": "^1.0.0",
    "path": "^0.12.7",
    "request-json": "^0.5.5"
},
"repository": {},
"engines": {
    "node": "0.12.x"
}
}
```

Note : This above code is used to add dependencies which are used in this application.

Open **app.js** file using an editor (Notepad++). Replace the code in **app.js** file with the following content,

```
/*eslint-env node*/
// This application uses express as its web server
// for more info, see: http://expressjs.com
var express = require('express');
var path=require('path');
// cfenv provides access to your Cloud Foundry environment
// for more info, see: https://www.npmjs.com/package/cfenv
var cfenv = require('cfenv');

// create a new express server
```

```
var app = express();

// serve the files out of ./public as our main files
app.use(express.static(__dirname + '/public'));
var routes = require('./routes/index');
var users = require('./routes/users');
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');
app.use('/', routes);
app.use('/users', users);
// get the app environment from Cloud Foundry
var appEnv = cfenv.getAppEnv();

// start server on the specified port and binding host
app.listen(appEnv.port, '0.0.0.0', function() {

    // print a message when the server starts listening
    console.log("server starting on " + appEnv.url);
});
```

Go to Public folder in your application and open Stylesheets folder. Open “**style**” file (css) using an editor (Notepad++). Replace the code in “**style**” file (css) with the following content,

```
body {
    padding: 50px;
    font: 14px "Lucida Grande", Helvetica, Arial, sans-serif;
}

a {
    color: #00B7FF;
```

}

Create a “**styles.css**” file in public→stylesheets. Place the following content in it.

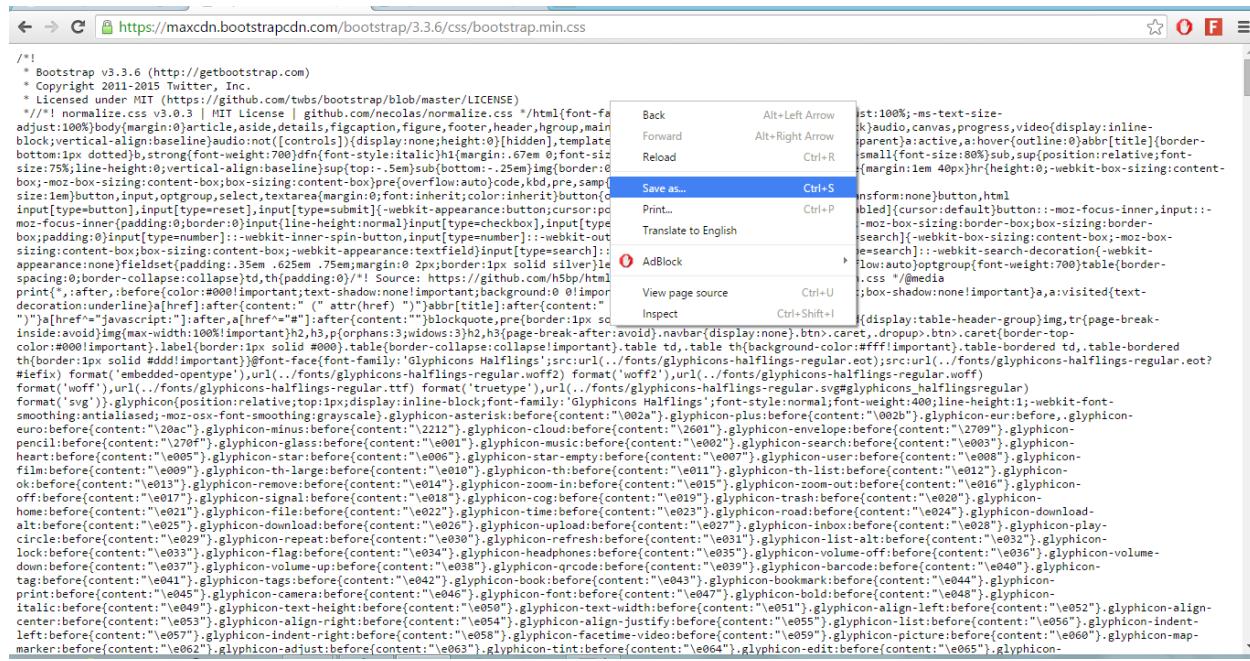
```
.modal-footer { border-top: 0px; }
```

Go to this URL,

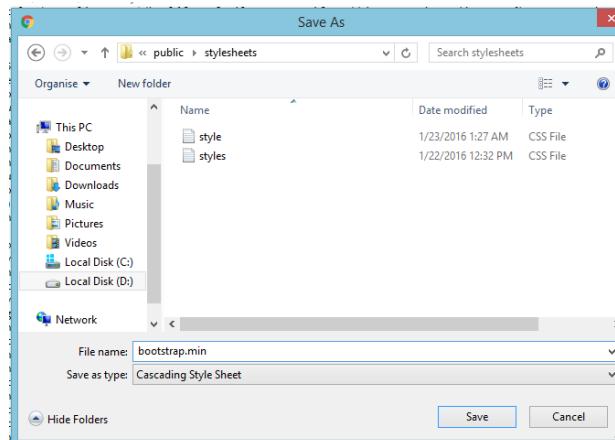
<https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css>

A page with bootstrap code appears.

Right click anywhere and select “Save As”.



This will prompt you to select storage location. Go to your application → Public → stylesheets. When stylesheets folder is opened, click on “Save”.



Go to your application → Public folder and create a “javascripts” folder.



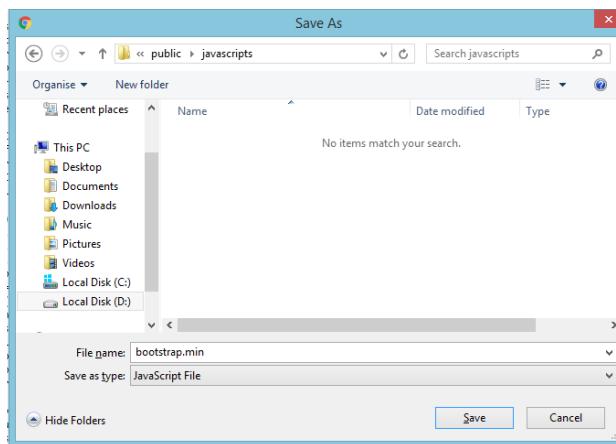
Go to this URL,

<https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js>

A page with bootstrap code appears.

Right click anywhere and select “Save As”.

This will prompt you to select storage location. Go to your application → Public → javascripts. When javascripts folder is opened, click on “Save”.



Now, create two folders with names as “**routes**” and “**views**” in the application which you downloaded from Bluemix.

Note: **Views** is used for designing Frontend, **Routes** is used for Controlling.

In **routes** folder, create two files - “**index.js**” and “**users.js**”

Open “**index.js**” file using an editor (Notepad++). Copy the following code, and paste it in “**index.js**” file

Also replace <**your.ibm-api-management-URL**> from this code with the URL that was generated in IBM API Management Developer Portal.

```
var express = require('express');
var https=require('https');
var request = require('request-json');
var client = request.createClient('http://localhost:8888/');
var router = express.Router();
```

```
/* GET users listing. */
router.get('/', function(req, response, next) {
    var data = {
        'Fahrenheit':'25'
    };
    client.post('<your.ibm-api-management-URL>', data, function(err, res,
body) {
        response.render('index',{title:body.Celsius});

    });
});
module.exports = router;
```

Note : Copy the Client ID and paste in the place of <your_client_id> in the above code.

Open “*users.js*” file using an editor (Notepad++). Copy the following code, and paste it in “*users.js*” file

Also repacle <your.ibm-api-management-URL> from this code with ther URL that was generated in IBM API Management Developer Portal.

```
var express = require('express');
var https=require('https');
var request = require('request-json');
var client = request.createClient('http://localhost:8888/');
var router = express.Router();

/* GET users listing. */
router.get('/', function(req, response, next) {
    var data = {
```

```
'Fahrenheit':'25'
};

client.post('<your.ibm-api-management-URL>', data, function(err, res,
body) {
    response.render('index',{title:body.Celsius});

});

module.exports = router;
```

Note : In the path, you should give the URL which was generated in IBM API Developer Portal. Also Copy the Client ID and paste in the place of <your_client_id> in the above code.

Open **Views** folder in application. Create a file – “index.ejs”

Open “**index.ejs**” file using an editor (Notepad++). Copy the following code, and paste it in “**index.ejs**” file

```
<html lang="en">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-
8">
    <meta charset="utf-8">
    <title>Fahrenheit to Celsius Converter</title>
    <meta name="generator" content="Bootply" />
    <meta name="viewport" content="width=device-width, initial-
scale=1, maximum-scale=1">
    <link href="stylesheets/bootstrap.min.css" rel="stylesheet">
    <!--[if lt IE 9]>
      <script
src="//html5shim.googlecode.com/svn/trunk/html5.js"></script>
```

```
<![endif]-->
<link href="stylesheets/styles.css" rel="stylesheet">
</head>
<body>
<!--login modal-->
<form action="/users" method="get">
<div id="loginModal" class="modal show" tabindex="-1" role="dialog" aria-hidden="true">
<div class="modal-dialog">
<div class="modal-content">
<div class="modal-header" style="background: black">

    <h2 class="text-center" style="color: white"><b>Miracle Software Systems</b></h2>
    <h4 class="text-center" style="color: #3498DB"><b>Fahrenheit to Celsius Converter</b></h4>
</div>
<div class="modal-body" style="background: #3498DB">
<form class="form col-md-12 center-block">
<div class="form-group"><br><br>
    <center style="color: black"><h3><b>Celsius Result</b> : <%= title %></h3>
</div>

</form>
</div>
<div class="modal-footer" style="background: #3498DB">
<div class="col-md-12">

    </div>
</div>
</div>
</div>
</form>
```

```
<!-- script references -->
<script
src="//ajax.googleapis.com/ajax/libs/jquery/2.0.2/jquery.min.js"></script>
<script src="js/bootstrap.min.js"></script>
</body>
</html>
```

Note : This file will be used for displaying the output.

Now, you need to check whether the application is running fine or not.

#8 | Deploying to the Cloud with CF CLI

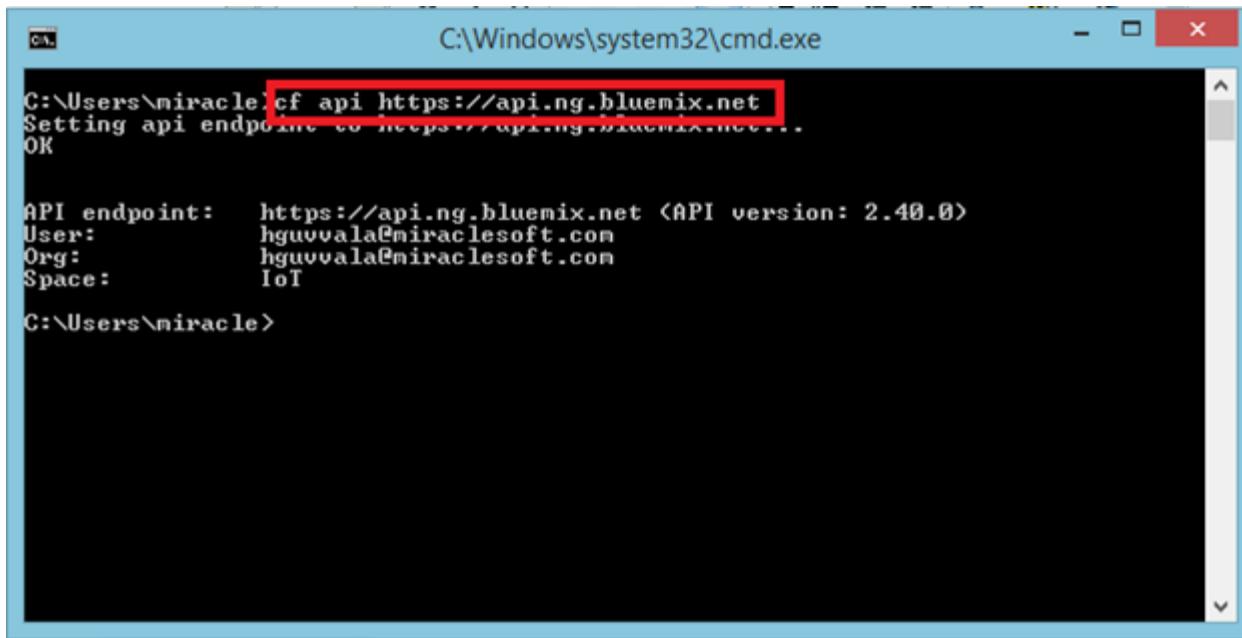
The next step will be to take your application and deploy it back to Bluemix so that you can share it with your friends.

Open to Command Prompt and go to the location where you have extracted your application. Then connect to Bluemix using one of the following commands (Depends on which region they have selected in your profile).

For Sydney : cf api https://api.au-syd.bluemix.net

For US South : cf api https://api.ng.bluemix.net

For US North : cf api https://api.eu.gb.bluemix.net



A screenshot of a Windows Command Prompt window titled 'cmd' with the path 'C:\Windows\system32\cmd.exe'. The window contains the following text:

```
C:\Users\miracle>cf api https://api.ng.bluemix.net
Setting api endpoint to https://api.ng.bluemix.net...
OK

API endpoint: https://api.ng.bluemix.net (API version: 2.40.0)
User: hguvvala@miraclesoft.com
Org: hguvvala@miraclesoft.com
Space: IoT

C:\Users\miracle>
```

Login to Bluemix using the command,

cf login

When prompted enter username and password.

```
C:\Windows\system32\cmd.exe - cf login
OK

API endpoint: https://api.ng.bluemix.net (API version: 2.40.0)
User: hguvvala@miraclesoft.com
Org: hguvvala@miraclesoft.com
Space: IoT

C:\Users\miracle cf login
API endpoint: https://api.ng.bluemix.net

Email> hguvvala@miraclesoft.com

Password>
Authenticating...
OK

Targeted org hguvvala@miraclesoft.com
Select a space (or press enter to skip):
1. IoT
2. Raspberry
3. Raspberry-Pi

Space>
```

Specify the space where you deployed your application.

```
C:\Windows\system32\cmd.exe

Email> hguvvala@miraclesoft.com

Password>
Authenticating...
OK

Targeted org hguvvala@miraclesoft.com
Select a space (or press enter to skip):
1. IoT
2. Raspberry
3. Raspberry-Pi

Space> IoT
Targeted space IoT

API endpoint: https://api.ng.bluemix.net (API version: 2.40.0)
User: hguvvala@miraclesoft.com
Org: hguvvala@miraclesoft.com
Space: IoT

C:\Users\miracle>
```

Make sure that you are in application's directory.

Use **cf push** to push your application to your Bluemix Organization.

```
C:\Windows\system32\cmd.exe - cf push

API endpoint: https://api.ng.bluemix.net (API version: 2.40.0)
User: hguvvala@miraclesoft.com
Org: hguvvala@miraclesoft.com
Space: IoT

C:\Users\miracle>d:
D:\>cd Interconnect_2016
D:\Interconnect_2016>cd InterconnectLab
D:\Interconnect_2016\InterconnectLab>cf push
Using manifest file D:\Interconnect_2016\InterconnectLab\manifest.yml

Updating app InterconnectLab in org hguvvala@miraclesoft.com / space IoT as hguvvala@miraclesoft.com...
OK

Using route interconnectlab.mybluemix.net
Uploading InterconnectLab...
Uploading app files from: D:\Interconnect_2016\InterconnectLab
Uploading 18.8K, 11 files
```

```
C:\Windows\system32\cmd.exe - cf push

----> Installing binaries
  engines.node <package.json>: 0.12.x
  engines.npm <package.json>: unspecified <use default>
  Resolving node version 0.12.x via 'node-version-resolver'
  Installing IBM SDK for Node.js (0.12.9) from cache
  Using default npm version: 2.14.9
----> Restoring cache
  Loading 1 from cacheDirectories <default>:
    - node_modules
----> Building dependencies
  Pruning any extraneous modules
  Installing node modules <package.json>
----> Installing App Management
----> Caching build
  Clearing previous node cache
  Saving 1 cacheDirectories <default>:
    - node_modules
----> Build succeeded!
  └─ cfenv@1.0.3
    └─ express@4.12.4
----> Uploading droplet <14M>

0 of 1 instances running, 1 starting
```

Note : This will take around 3 minutes for completion

Once your application is pushed, your Command prompt should look as below,

```
C:\Windows\system32\cmd.exe
----> Build succeeded!
    └─ cfenv@1.0.3
        express@4.12.4
----> Uploading droplet <14M>
0 of 1 instances running, 1 starting
1 of 1 instances running
App started
Showing health and status for app InterconnectLab in org hguvvala@miraclesoft.com / space IoT as hguvvala@miraclesoft.com...
OK
requested state: started
instances: 1/1
usage: 256M x 1 instances
urls: interconnectlab.mybluemix.net
#0  state      since
#0  running    2016-02-01 07:14:10 AM  cpu      0.0%   memory  71.3M of 256M  disk    55M of 1G
D:\Interconnect_2016\InterconnectLab>
```

Now, you can go back to your Bluemix Dashboard in Browser and access your applications URL through **Dashboard--> Application Overview--> Application URL**.

The screenshot shows the IBM Bluemix Application Overview page for the 'InterconnectLab' application. The URL in the browser is <https://console.ng.bluemix.net/direct=classic#/resources/appGuid=187a8f99-6c3a-409d-8438-df67025a9b95&appName=InterconnectLab&orgGuid=0beca361>. The page displays the following information:

- Routes:** interconnectlab.mybluemix.net (highlighted with a red box)
- SDK FOR NODE.js™:** Shows 1 instance, 256 MB memory quota, and 256.0 MB available memory.
- APP HEALTH:** Your app is running. Buttons for RESTART and STOP are present.
- ACTIVITY LOG:** A log of recent events:
 - 2/1/16 7:24 AM hguvvala@miraclesoft.com stopped InterconnectLab app
 - 2/1/16 7:24 AM hguvvala@miraclesoft.com started InterconnectLab app
 - 2/1/16 7:13 AM hguvvala@miraclesoft.com started InterconnectLab app
 - 2/1/16 7:13 AM hguvvala@miraclesoft.com stopped InterconnectLab app
- Navigation:** Back to Dashboard, Overview, Services, and a button to Estimate the cost of this app.

The application, which you created and deployed in Bluemix should be as below.

