

1. After Clicking on create nodered, below ss you will see,click on Deploy your app

Resource list / App details / Node RED ETWJU Add tags Actions...

Details

App URL: You must deploy your app first

Source: Download code

Resource group: Default

Deployment target: You must deploy your app first

Created: 5/9/2020

Services (1) Connect existing services Create service

Name	Resources	Actions
Cloudant	Documentation	

Deployment Automation

Configure Continuous Delivery

Continuous Delivery is not enabled for this app. Enable Continuous Delivery to automate builds, tests, and deployments through Delivery Pipeline, GitLab, and more.

Deploy your app

Credentials Show

```
{
  "cloudant": {
    "name": "node-red-stg-cloudant-3580964",
    "credentials": {
      "apikey": "*****",
      "host": "a513a812-6277-49a9-a68e-80811",
      "password": "*****",
      "port": 443,
      "url": "https://a513a812-6277-49a9-a68e-80811",
      "username": "*****"
    }
  }
}
```

Getting started quickly

Configuring your app

To connect services and DevOps toolchains to your app:

1. Use the **Services** card to connect a service to your app. Select an existing service instance, or create a new one. [Learn more.](#)
2. If you want to view the code before your app is deployed, click **Download code** to obtain the zip file.
3. Click **Deploy your app** in the **Deployment Automation** card to select the deployment target and configure the Continuous Delivery service. The deployment begins automatically.
4. After the deployment begins, you can view the status of the deployment, modify your app, view your repo, or view the app's URL.
5. If you make any changes to your app, be sure to deploy it again.

Building, running, and deploying your app locally

To build and run your app locally:

1. Run the `ibmcloud dev code <APPNAME>` command from the IBM Cloud CLI. [Learn more.](#)
2. Run the following commands in a local development container from the app directory:

```
ibmcloud dev build
ibmcloud dev run
ibmcloud dev deploy
```

2. Click on new

Resource list / App details / Node RED ETWJU

Select the deployment target Configure the DevOps toolchain

Deployment Automation

Select your deployment target and configure your DevOps toolchain. After you click **Create**, the toolchain is created, and the deployment process is started automatically.

Deployment target

Cloud Foundry

Deploy and run your applications without managing servers or clusters. A Lite plan is available for quick and easy deployment.

IBM Cloud API key

IBM Cloud API key New

The value is required.

Number of instances

1

Memory allocation per instance

64 MB 2000 MB 192

Region Organization Space

us-south default

Getting started with apps

Step 1. Select the deployment target

Select your deployment target, and then provide the configuration information.

IBM Cloud Foundry

Cloud Foundry is the premier industry standard Platform-as-a-Service (PaaS) that ensures fast, easy, and reliable deployment of cloud-native apps. Cloud Foundry ensures that the build and deploy aspects of coding remain carefully coordinated with any attached services — resulting in quick, consistent and reliable treating of applications. Cloud Foundry has a Lite plan that allows quick deployments for testing purposes.

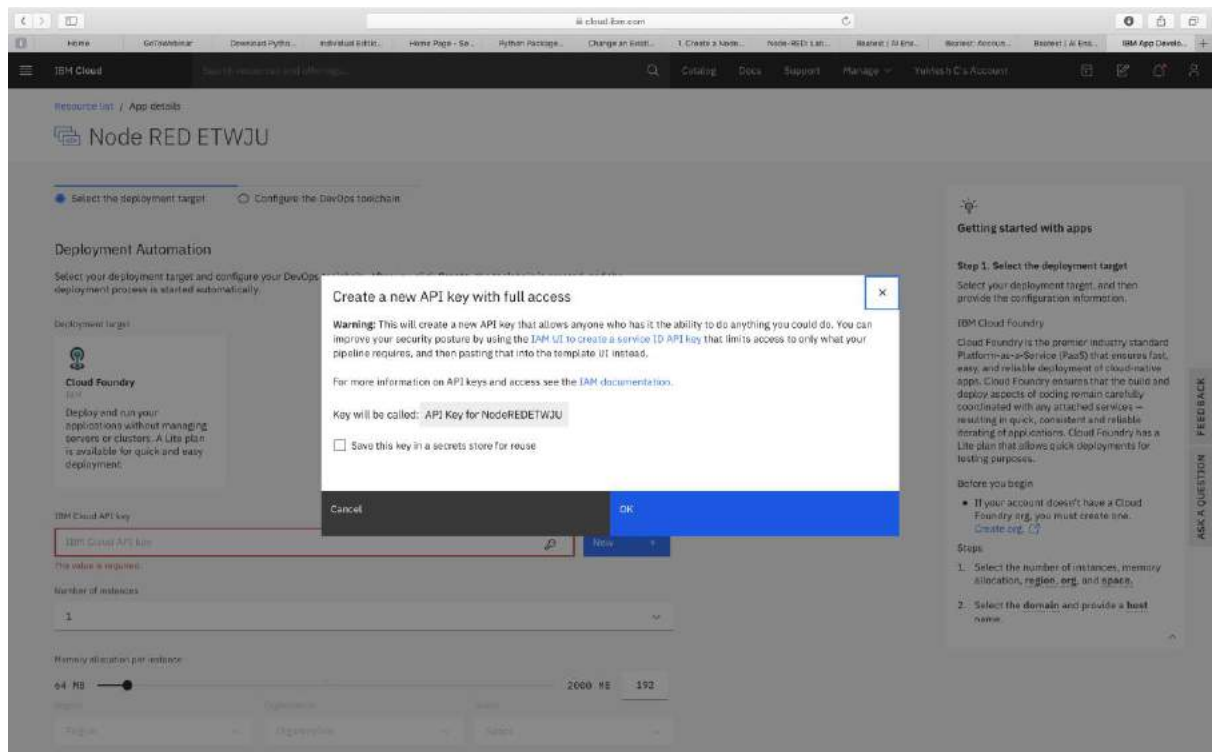
Before you begin

- If your account doesn't have a Cloud Foundry org, you must create one. [Create org.](#)

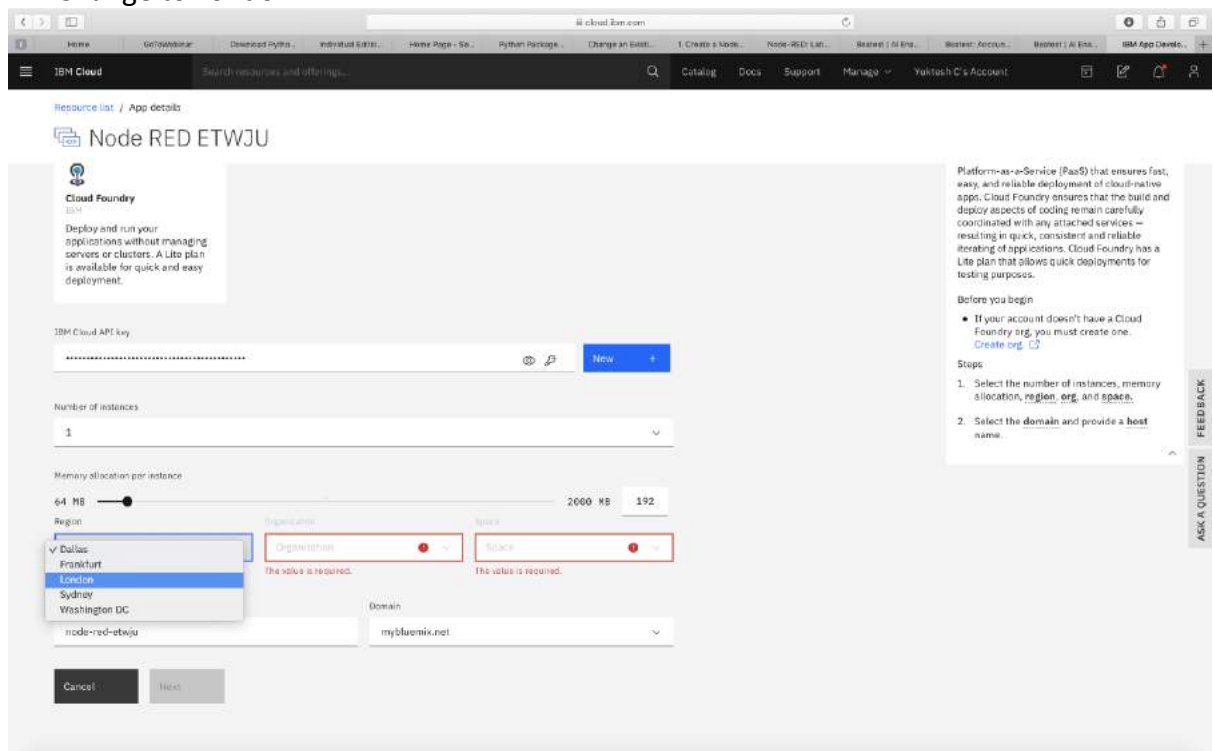
Steps

1. Select the number of instances, memory allocation, **region**, **org**, and **space**.
2. Select the **domain** and provide a **host** name.

3. Click on ok



4. Change to London



5. Click on Next

Node RED ETWJU

Cloud Foundry
Deploy and run your applications without managing servers or clusters. A Lite plan is available for quick and easy deployment.

IBM Cloud API key
[Redacted] [New](#)

Number of instances
1

Memory allocation per instance
64 MB 2000 MB 192

Region
London

Organization
yulpythondia@gmail.com

Space
dev

Host
node-red-etwju

Domain
eu-gb.mybluemix.net

[Cancel](#) [Next](#)

Before you begin

- If your account doesn't have a Cloud Foundry org, you must create one. [Create org](#)

Steps

- Select the number of instances, memory allocation, region, org, and space.
- Select the domain and provide a host name.

Cloud Foundry is the premier industry standard Platform-as-a-Service (PaaS) that ensures fast, easy, and reliable deployment of cloud-native apps. Cloud Foundry ensures that the build and deploy aspects of coding remain carefully coordinated with any attached services – resulting in quick, consistent and reliable iterating of applications. Cloud Foundry has a Lite plan that allows quick deployments for testing purposes.

ASK A QUESTION **FEEDBACK**

6.Click on Create

Node RED ETWJU

Select the deployment target **Configure the DevOps toolchain**

Configure the DevOps toolchain
Give your toolchain a name and select the region to create your toolchain in.

DevOps toolchain name
Accept the default name, or enter a value up to 100 characters.
NodeRED ETWJU

Region
Dallas

[Back](#) [Create](#)

Getting started with apps

Step 2: Configure the DevOps toolchain
The DevOps toolchain includes a Delivery Pipeline tool where you can check the deployment status, start builds, manage deployment, and view logs and history.

- Provide a name for your toolchain.
- Select the region where your toolchain is created.
- Select the resource group that has access to your new toolchain. [Learn more](#)
- After you've finished with your selections, click **Create**.

ASK A QUESTION **FEEDBACK**

6. Now it will process, It will take some time , Observe status it will show “In progress”

Node RED ETWJU Add tags

Details

App URL: You must deploy your app first

Source: [Download code](#)

Resource group: [Default](#)

Deployment target: You must deploy your app first

Created: 5/9/2020

Services (1) [Connect existing services](#) [Create service](#)

Name	Resources	Actions
Cloudant	Documentation	

Deployment Automation

Configure Continuous Delivery

Continuous Delivery is not enabled for this app. Enable Continuous Delivery to automate builds, tests, and deployments through Delivery Pipeline, GitLab, and more.

Credentials [Show](#)

```

{
  "cloudant": {
    "name": "node-red-etwju-cloudant-15889946",
    "credentials": {
      "apikey": "a1d3b812-6127-49a9-ae8c-800114",
      "host": "a1d3b812-6127-49a9-ae8c-800114",
      "password": "a1d3b812-6127-49a9-ae8c-800114",
      "port": 443,
      "url": "https://a1d3b812-6127-49a9-ae8c-800114"
    }
  }
}

```

Getting started quickly

Configuring your app

To connect services and DevOps toolchains to your app:

1. Use the **Services** card to connect a service to your app. Select an existing service instance, or create a new one. [Learn more.](#)
2. If you want to view the code before your app is deployed, click **Download code** to obtain the zip file.
3. Click **Deploy your app** in the **Deployment Automation** card to select the deployment target and configure the Continuous Delivery service. The deployment begins automatically.
4. After the deployment begins, you can view the status of the deployment, modify your app, view your repo, or view the app's URL.
5. If you make any changes to your app, be sure to deploy it again.

Building, running, and deploying your app locally

To build and run your app locally:

1. Run the `ibmcloud dev code <APPNAME>` command from the IBM Cloud CLI. [Learn more.](#)
2. Run the following commands in a local development container from the app directory:

```

ibmcloud dev build
ibmcloud dev run
ibmcloud dev deploy

```

7. Click on status "In progress" it should show the below

NodeRED ETWJU | Delivery Pipeline [Add Stage](#) [Actions...](#)

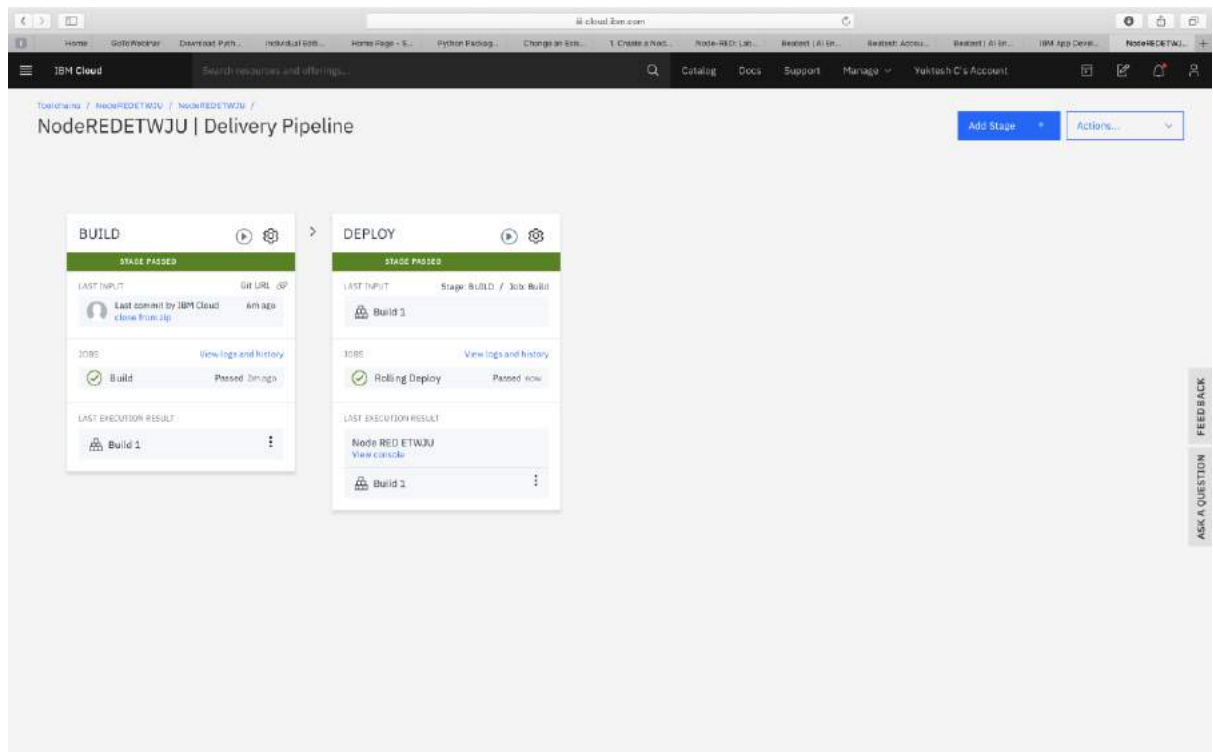
Build [View logs and history](#)

DEPLOY [View logs and history](#)

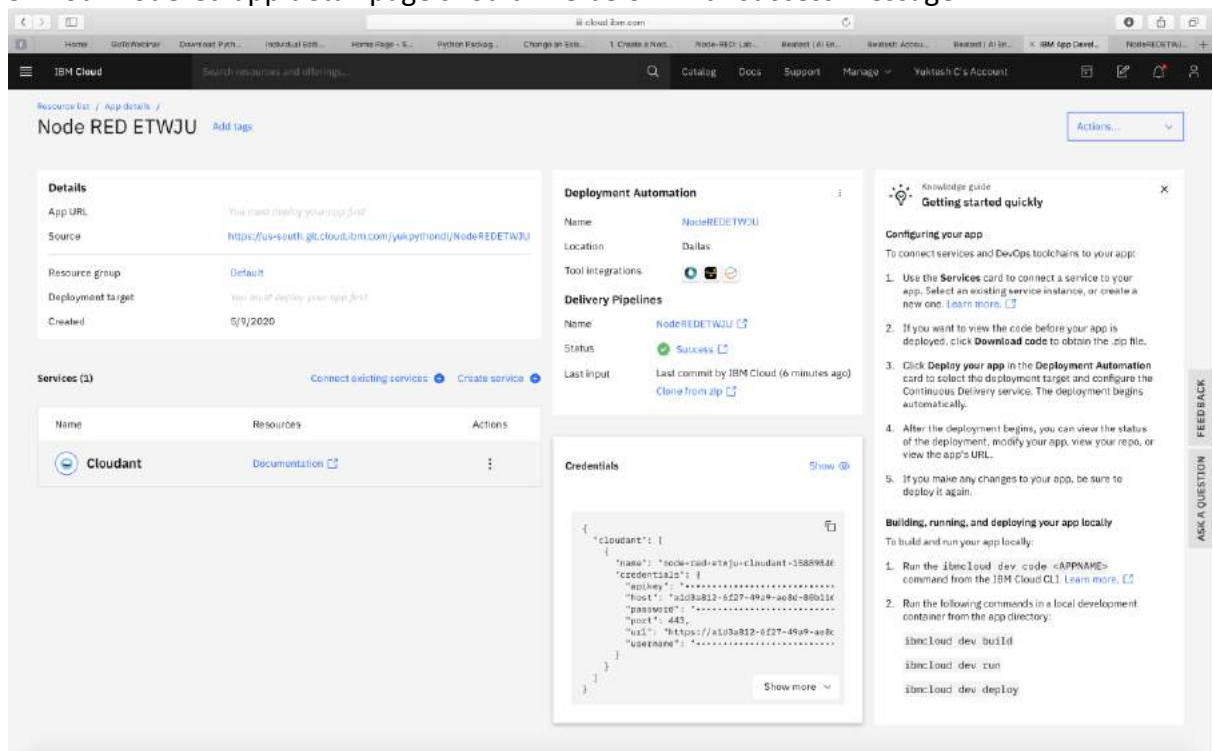
Build [View logs and history](#)

Rolling Deploy [View logs and history](#)

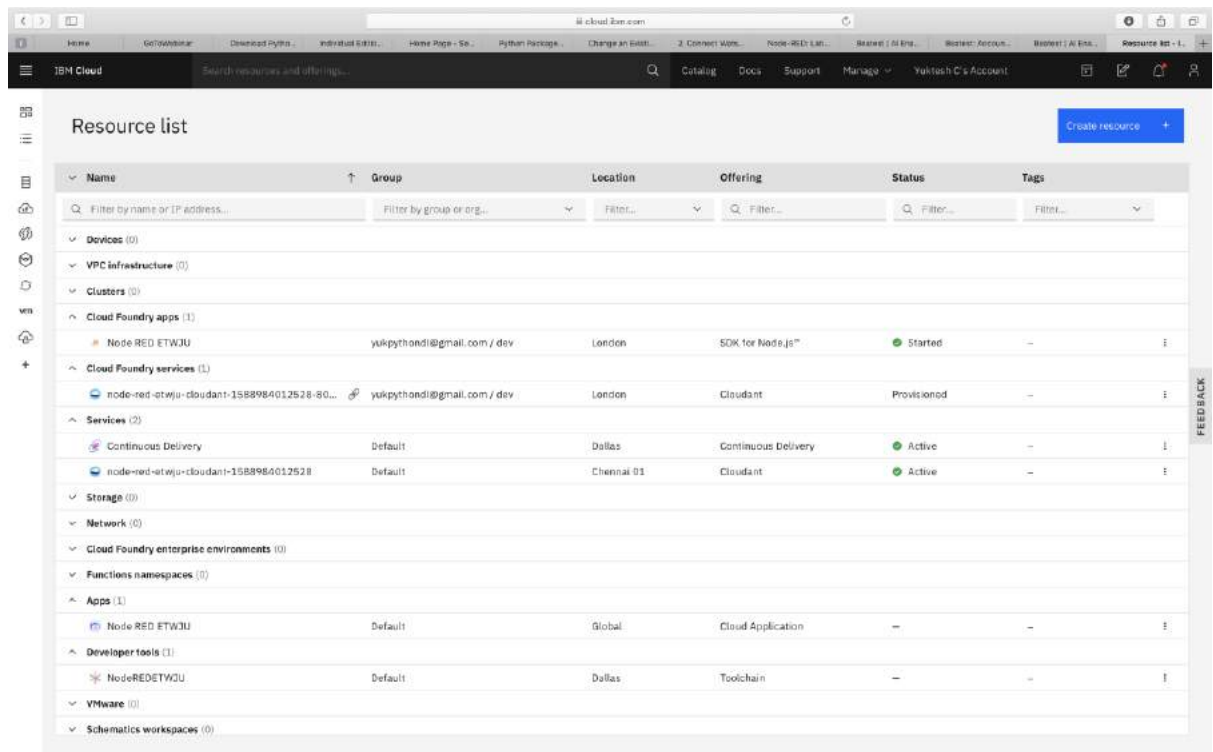
8. After some time it should show like below



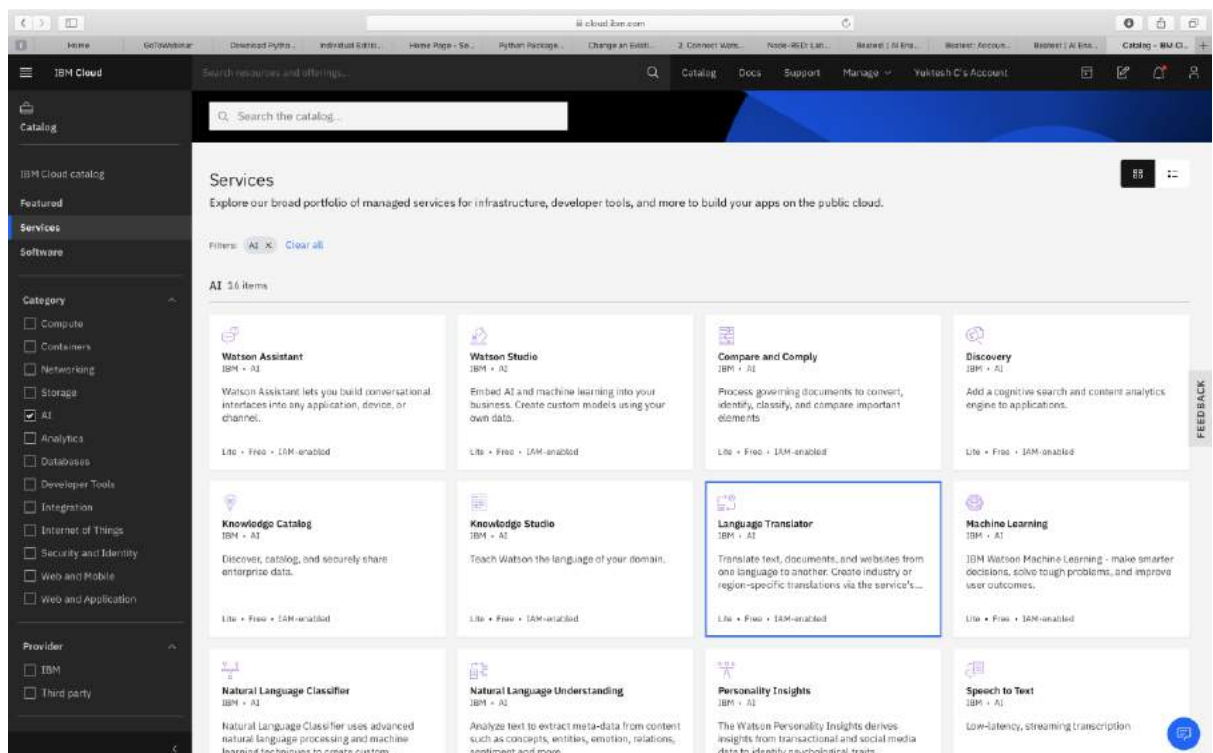
9. Your nodered app detail page should like below with success message



10. Goto Dashboard and click on Resource list you should be able to see below



11. Goto catalog and goto services and choose AI and click on “Language Translator”



12. Click on create

Language Translator

Author: IBM Watson • Date of last update: 04/24/2020 • Docs • API docs

Create **About**

Select a region

Select a region

London

Select a pricing plan

Displayed prices do not include tax. Monthly prices shown are for country or region: [United States](#)

Plan	Features	Pricing
Lite	Translate up to 1,000,000 Characters per Month Identify up to 60 languages with language identification Document Translation supporting 12 file types The Lite plan gets you started with 1,000,000 characters per month at no cost and includes the default translation models. When you upgrade to a paid plan, you can create custom models. Lite plan services are deleted after 30 days of inactivity.	Free
Standard	Everything in Lite plus... Removal of monthly 1M character translation limit First 250,000 characters are free	\$0.02 USQ/THOUSAND CHAR
Advanced	Standard Translations Custom Translations Build Domain Specific Custom Models (Pro-Rated Daily)	\$0.02 USQ/THOUSAND CHAR \$0.08 USQ/THOUSAND CHAR \$15.00 USD/INSTANCE MONTH
Premium	Everything in Advanced plus... Usage and Training Data is Private + Stored in an Isolated Single-Tenant Environment High Availability and Service Level Uptime Guarantee	\$11,500.00 USQ/INSTANCE \$0.02 USQ/THOUSAND CHAR \$0.08 USQ/THOUSAND CHAR

Summary

Language Translator **Free**

Region: London
Plan: Lite
Service name: Language Translator-99
Resource group: Default

Create

Add to estimate

[View terms](#)

13. You should see below screen

Language Translator-99 **Active** **Add tags**

Manage

Getting started

Service credentials

Plan

Connections

Getting started with Language Translator

Last Updated: 2020-04-22

This tutorial walks you through the commands to translate text from English to Spanish, and to identify the language of a text sample.

Before you begin

- Make sure that you have the `curl` command.
 - Test whether `curl` is installed. Run the following command on the command line. If the output lists the `curl` version with SSL support, you are set for the tutorial.

```
$ curl -V
```
- If necessary, install a version with SSL enabled from [curl.haxx.se](#). Add the location of the file to your PATH environment variables if you want to run `curl` from any command-line location.

Tip: This tutorial uses an API key to authenticate. In production, use an IAM token. For more information see [Authenticating to Watson services](#).

Step 1: Translate text

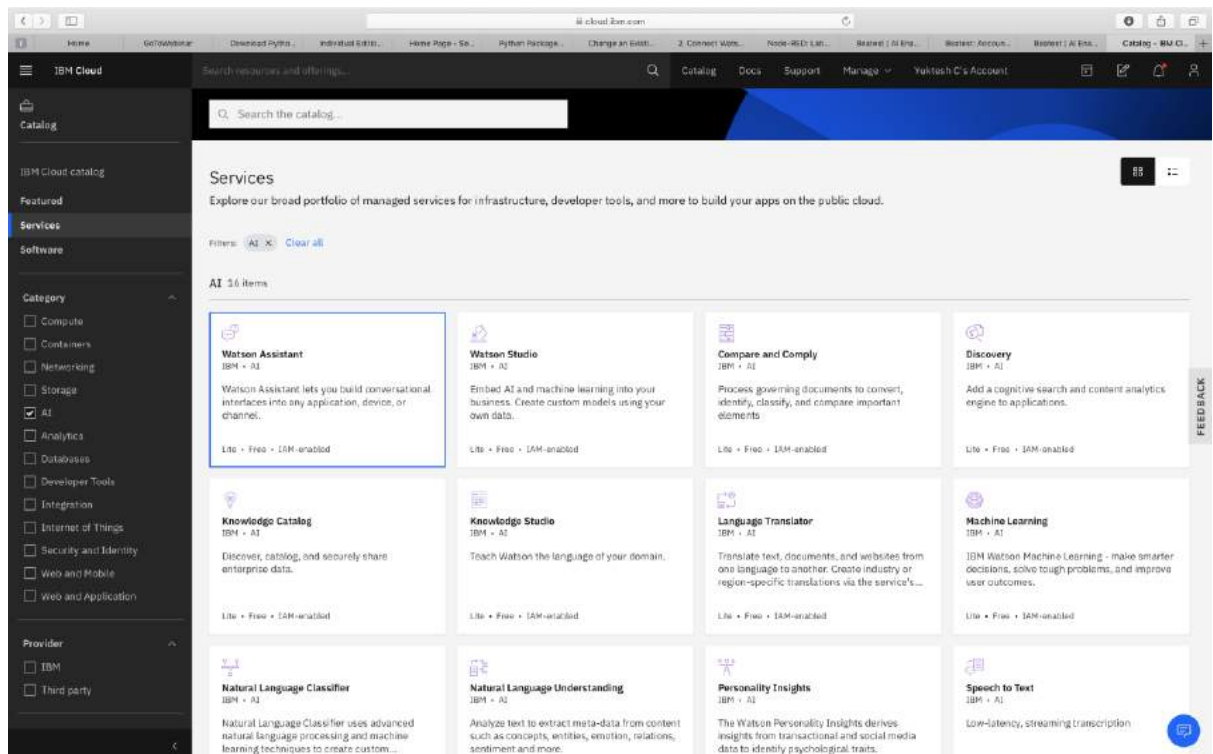
Use the following example to translate two phrases, "Hello, world!" and "How are you?" from English to Spanish.

```
$ curl -X POST -d '{"apikey":"apikey"}' --header 'Content-Type: application/json' --data '{"text": "Hello, world! How are you?"}'
```

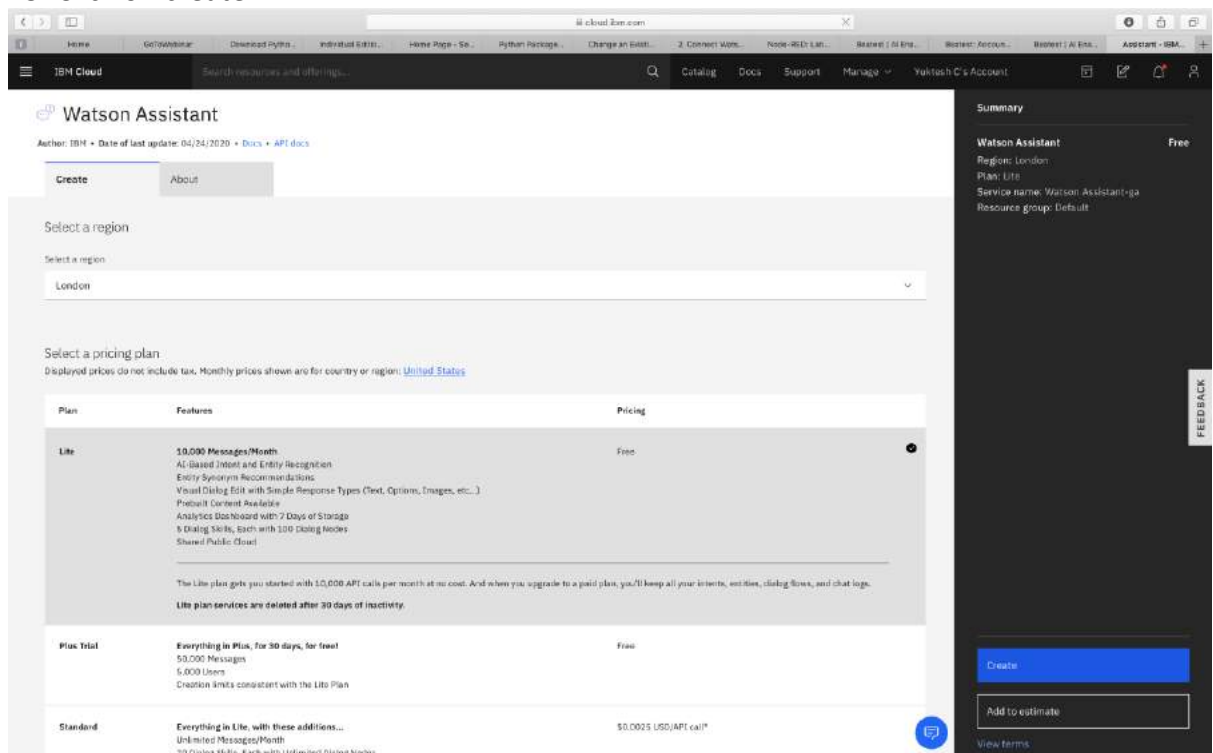
Step 2: Identify language

Use the following example to identify the language of text.

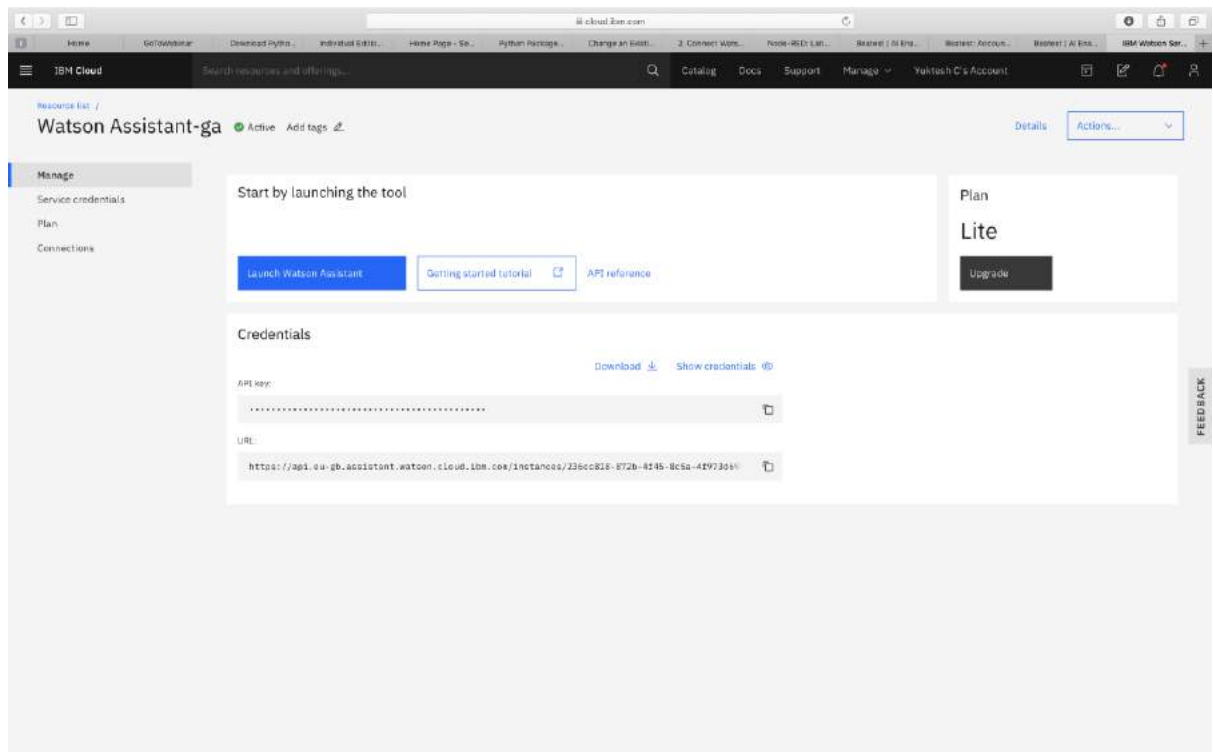
14. Now goto Catalog again and select services->AI->Watson Assistant service and click on it



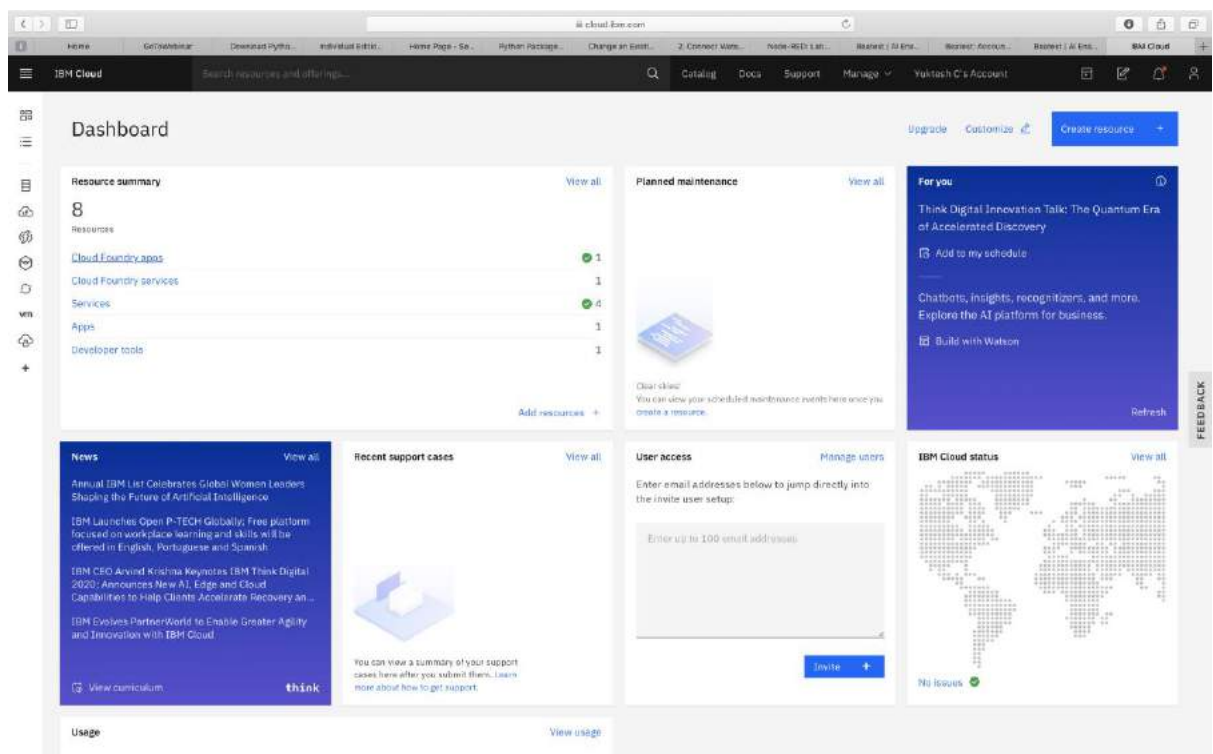
15. Click on create



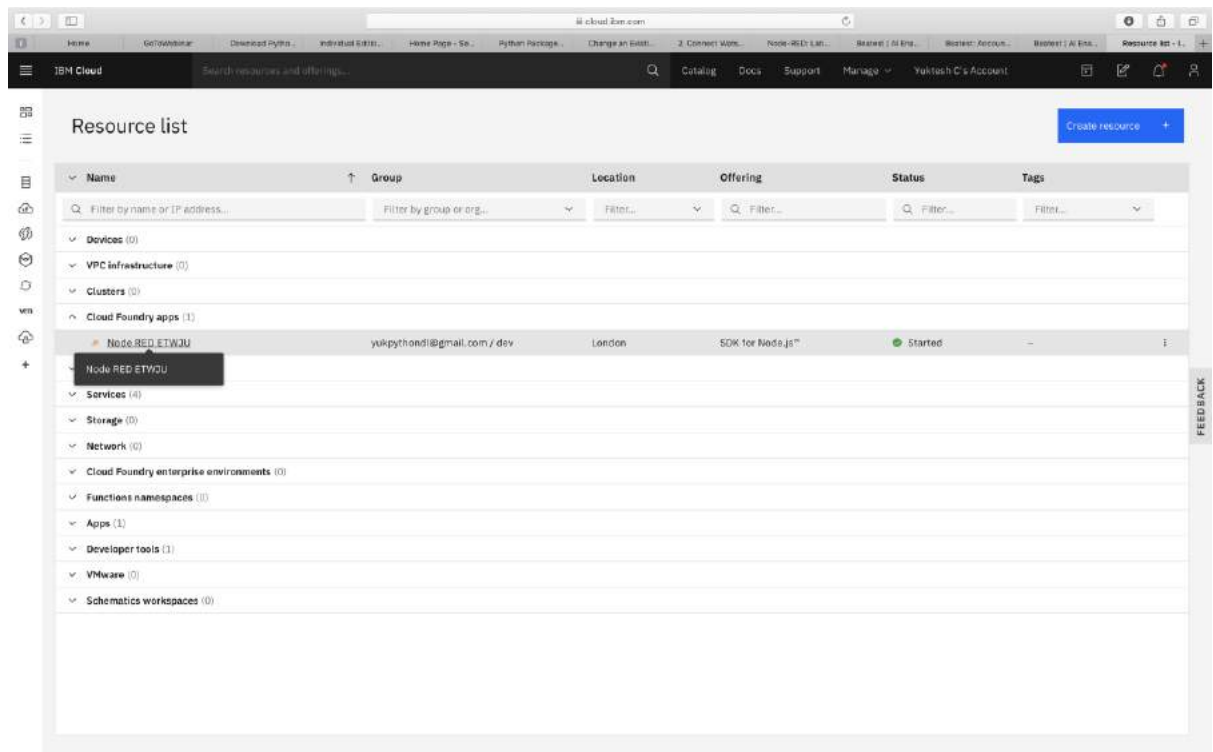
16. You should see below page



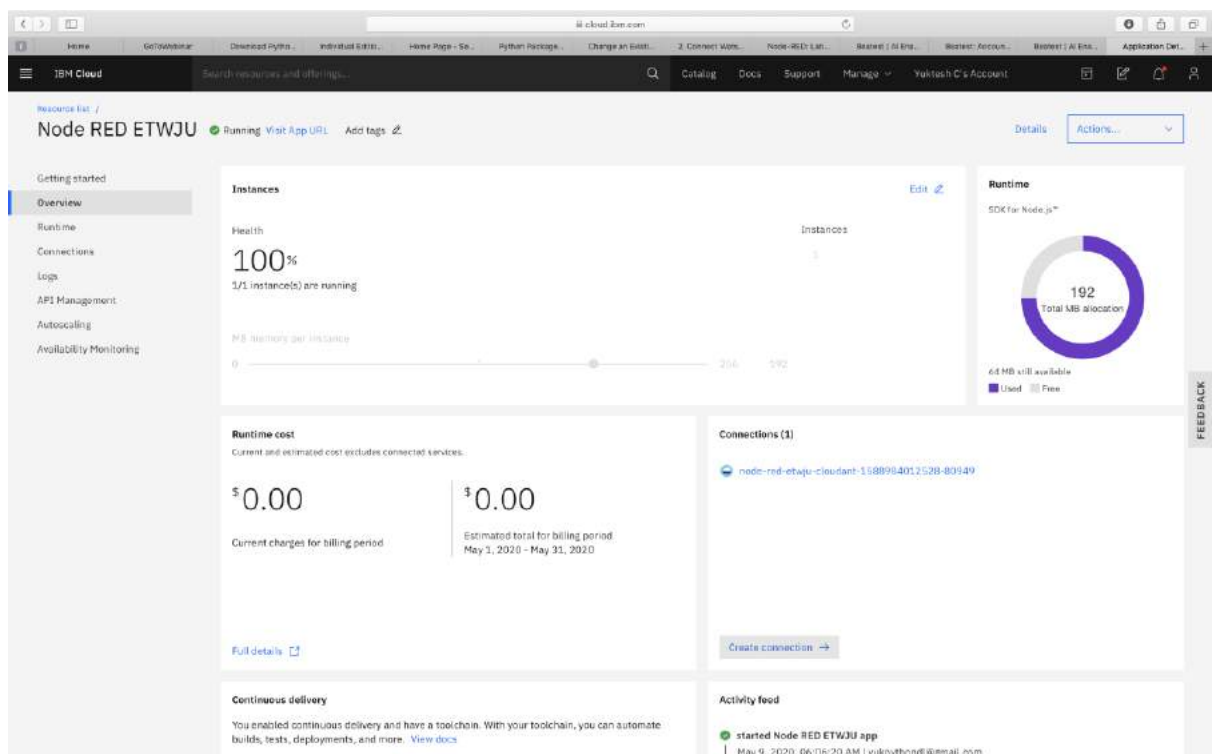
Now add these two services to Node Red APP
 17. Goto Dashboard and click on Cloud Foudry App



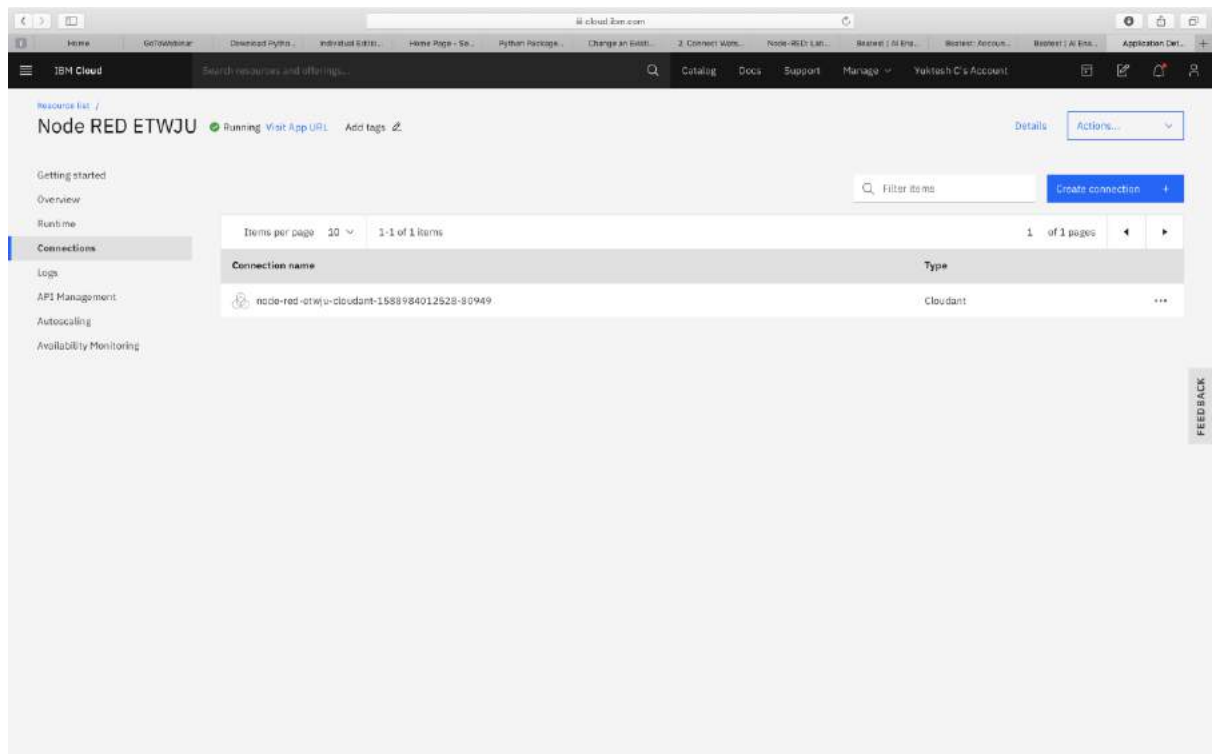
18. Click on your Node Red App



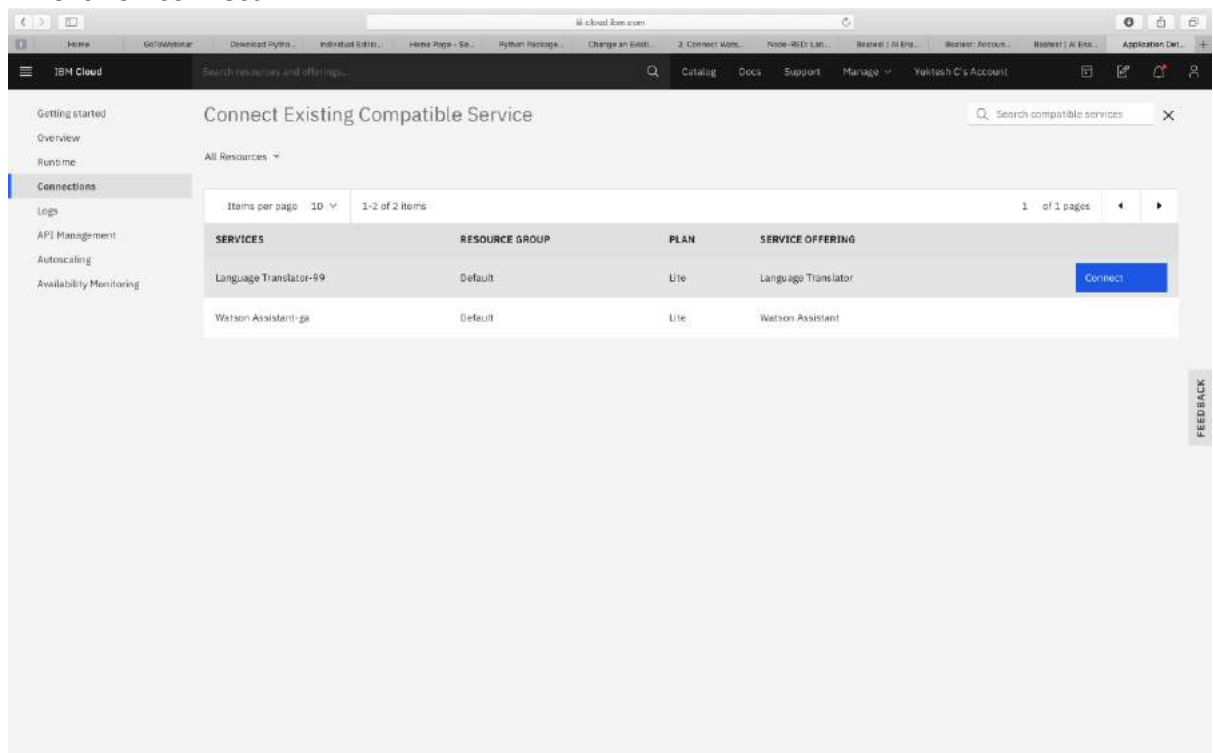
19. You will be in overview page, click on create connection on rightside in the middle of page



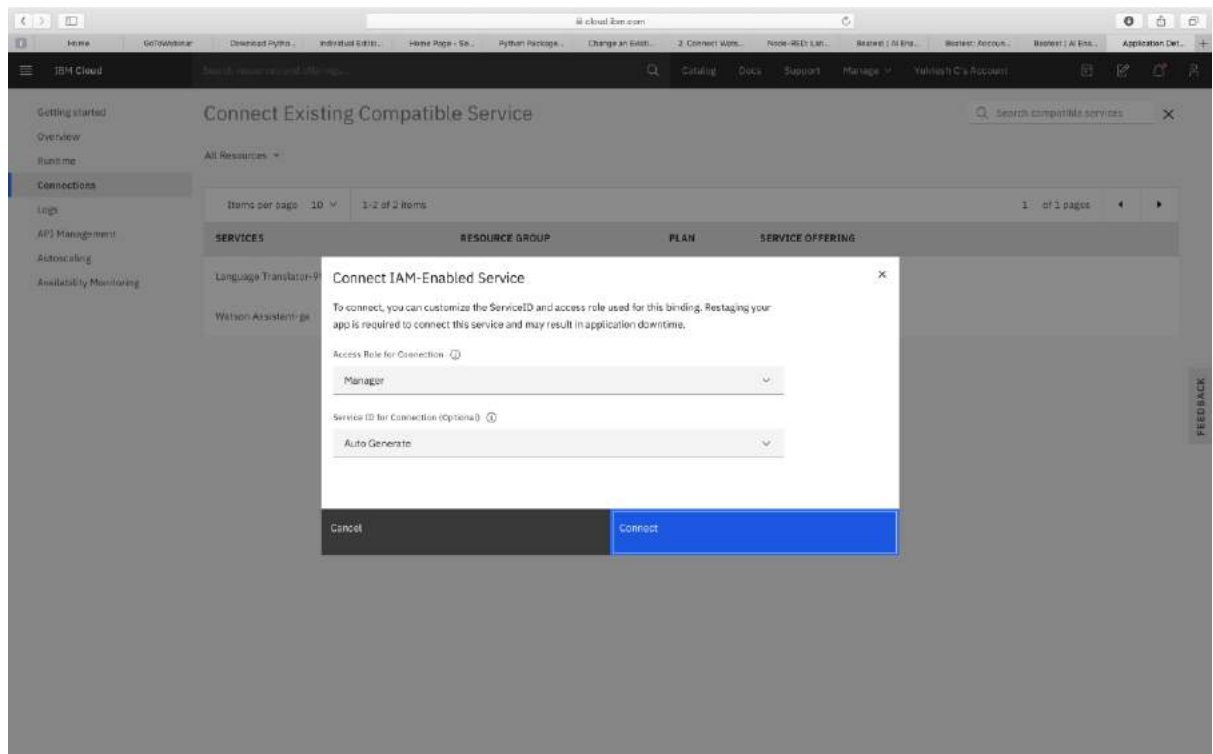
20. Click on create connection



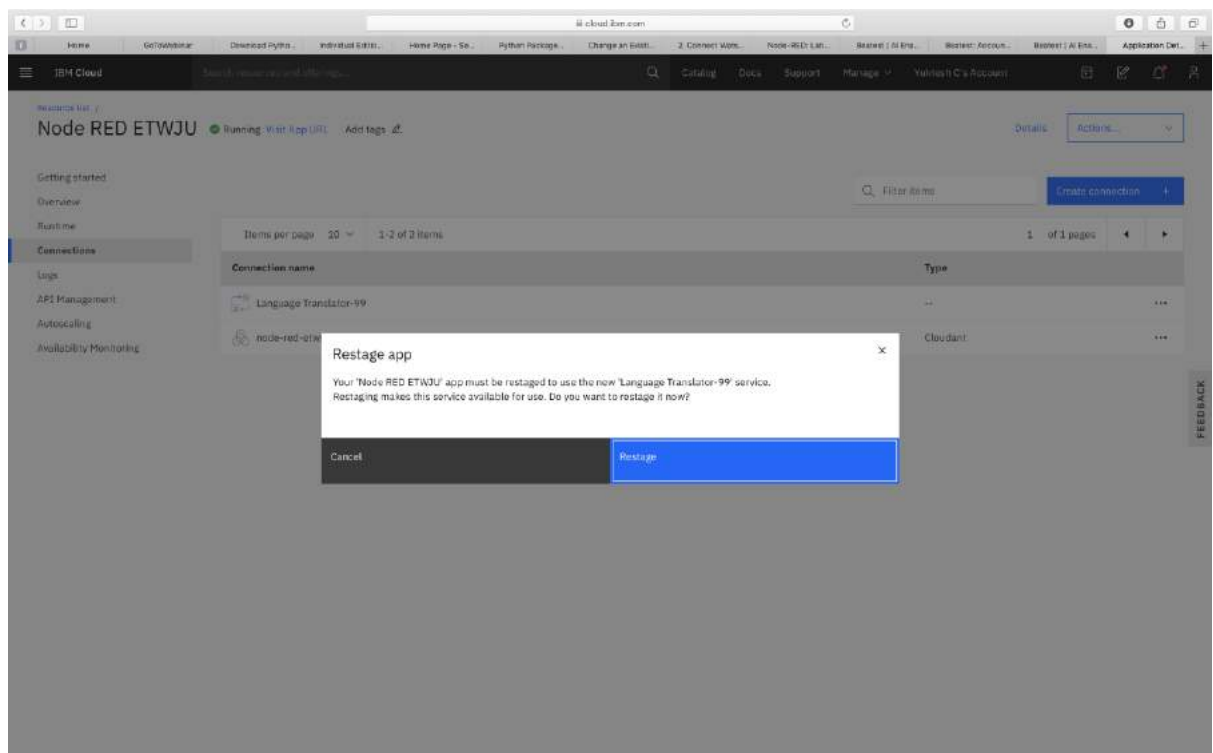
21.Click on connect



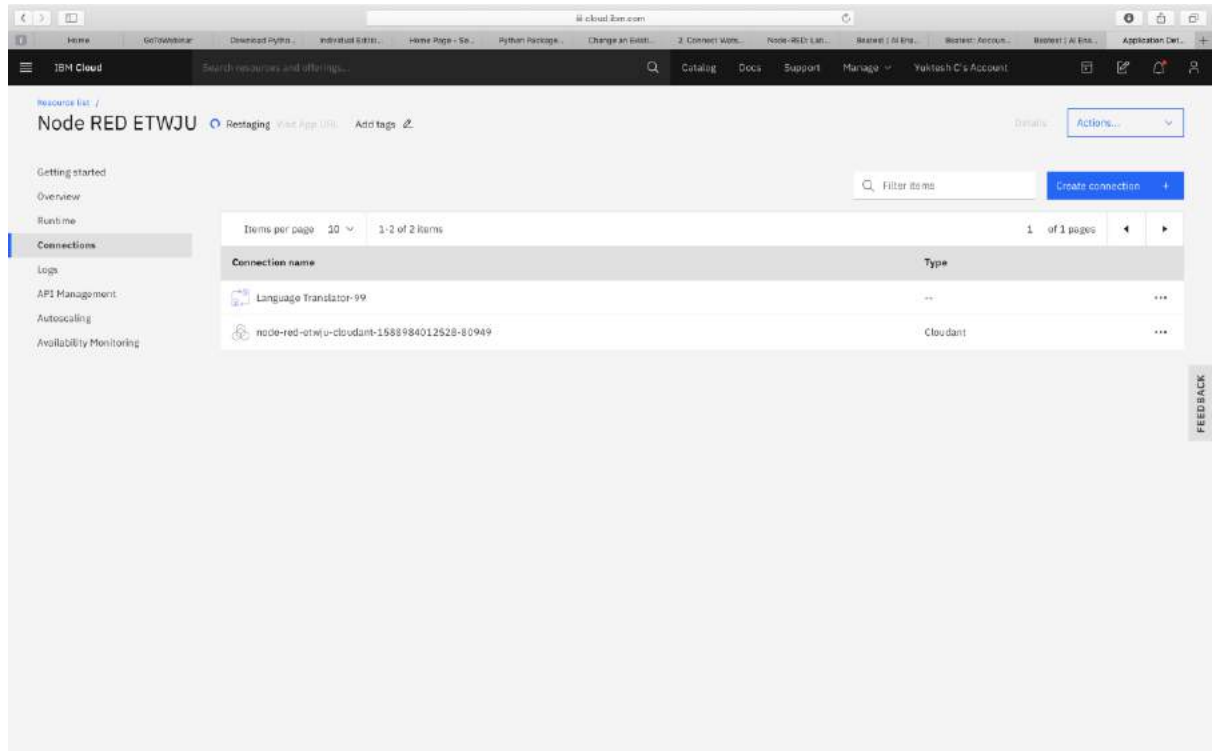
22.Again click on connect



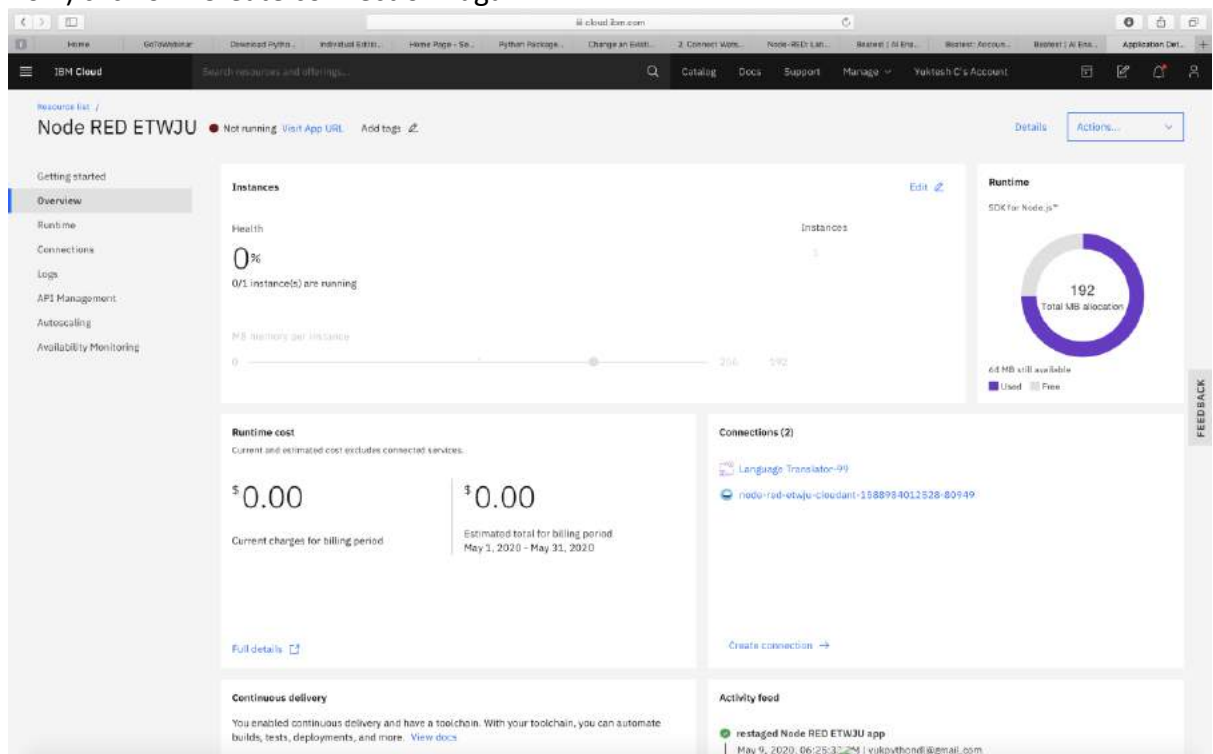
23. Click on Restage



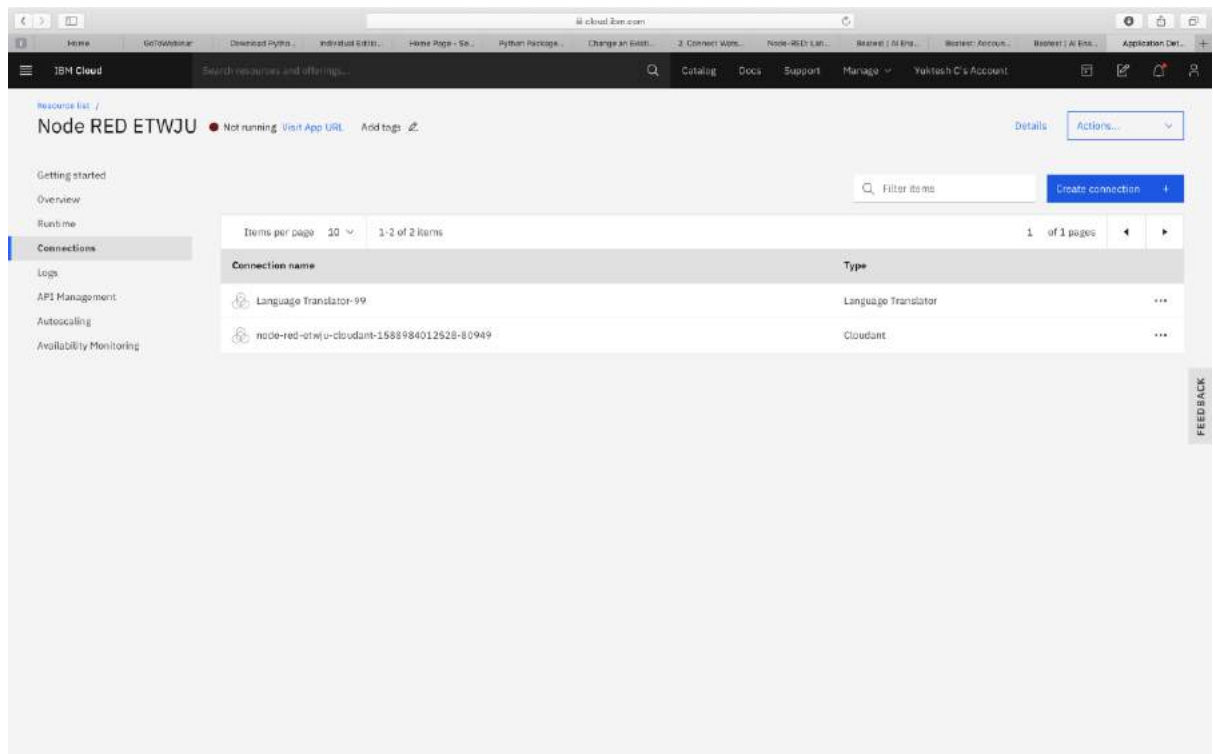
24. You should see app restaging on top left



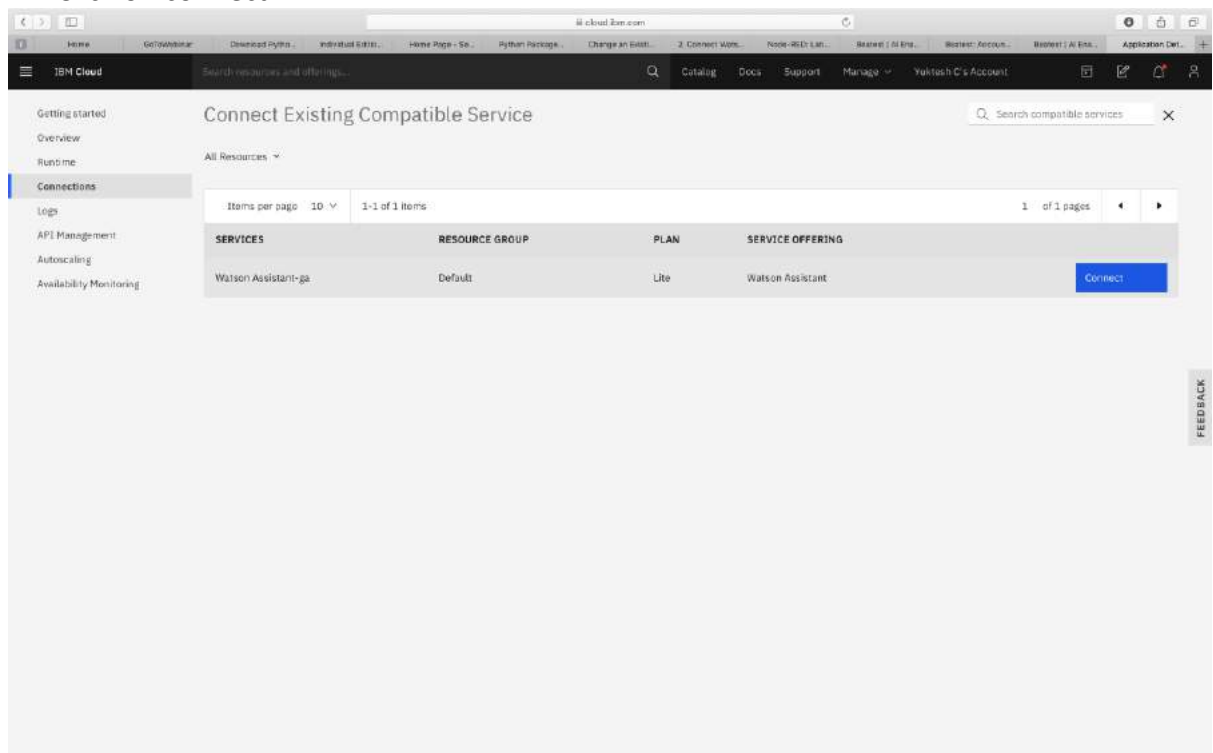
25. After restaging goto overview page and check whether “Language Translator” services has added in connection or not. (Don’t bother if your app is not running for now) click on “Create connection” again



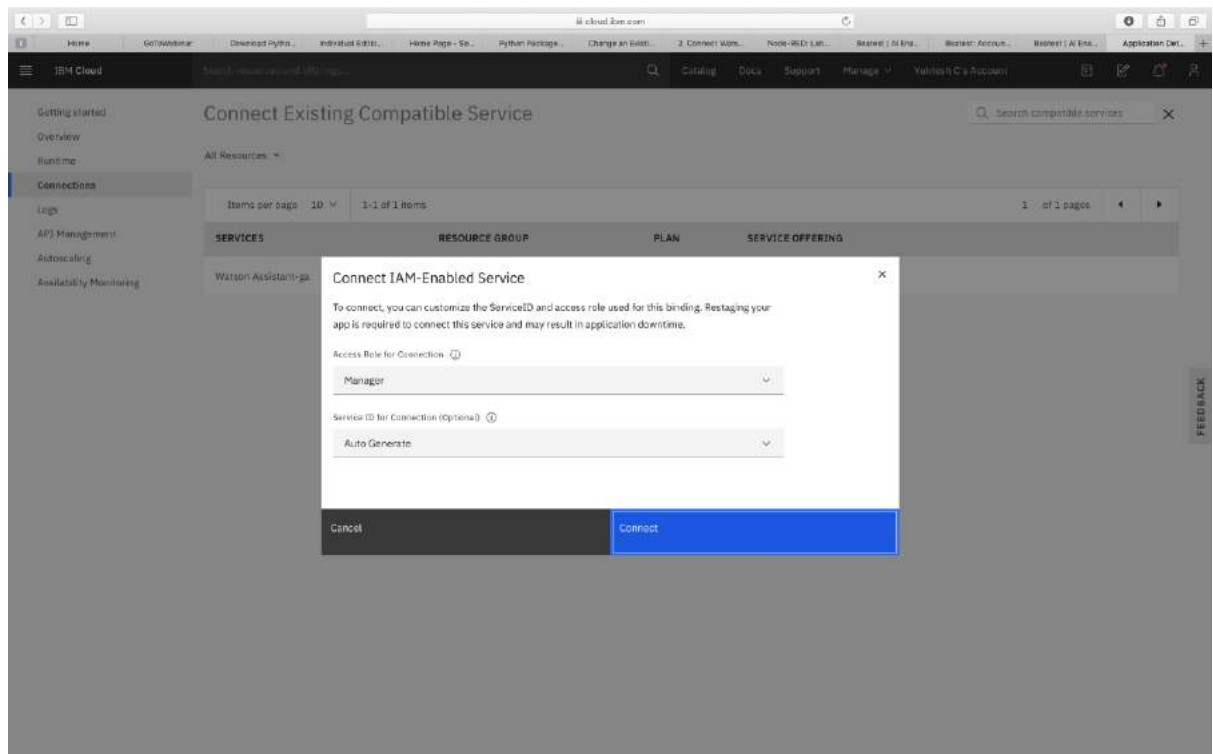
26. Click on create connection



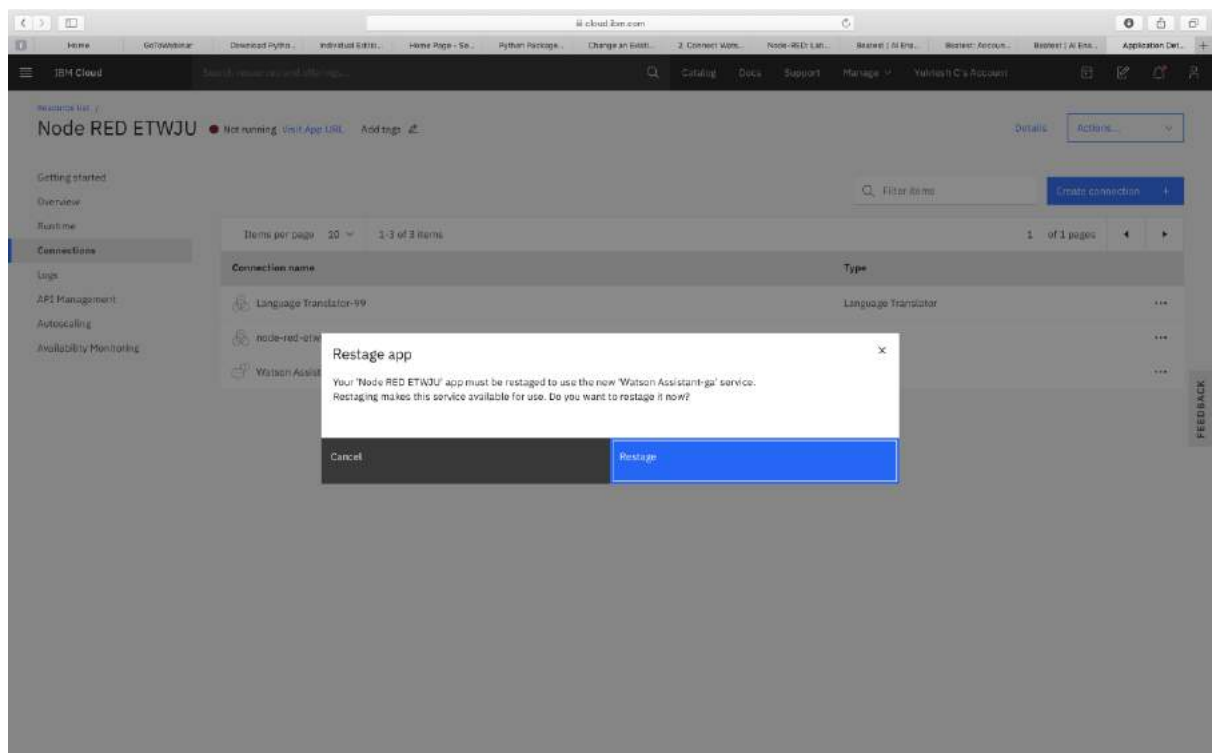
27. Click on connect



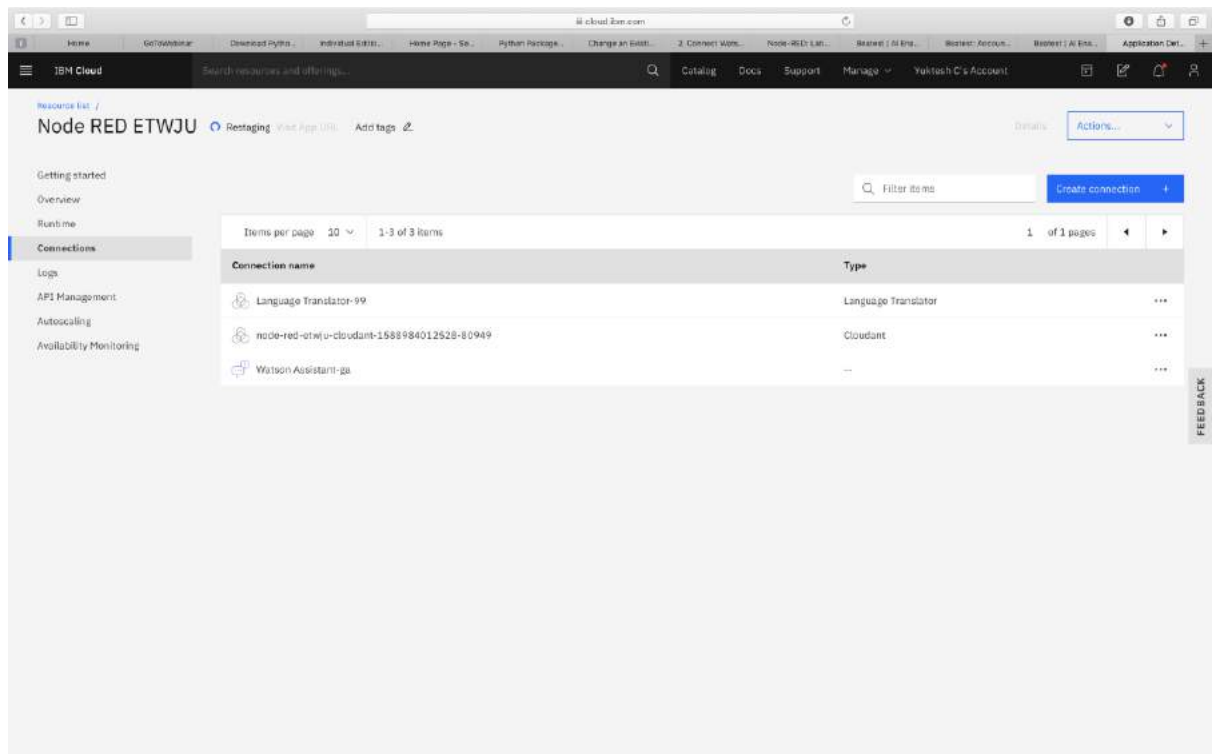
28. Click on connect



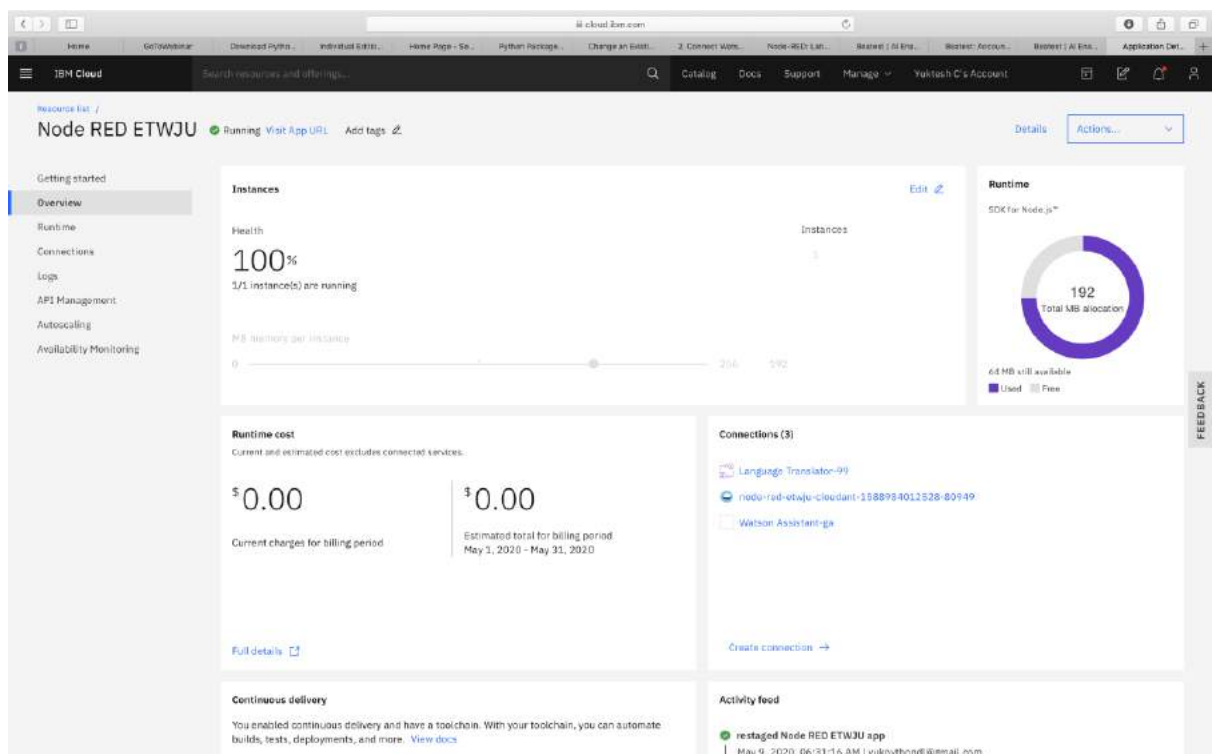
29. Click on Restage



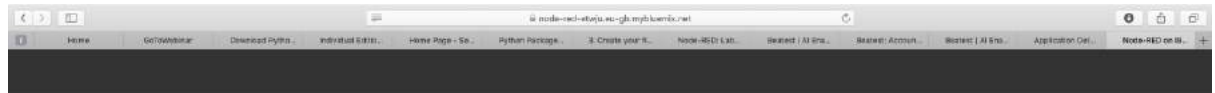
30. It should Restaging on top left



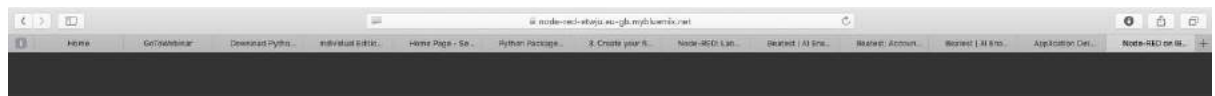
31. After restaging goto overview page and ensure 3 connections and click on visit app url



32. Click on Next



33. Give user name and password (choose anything)



Secure your Node-RED editor

☒ Secure your editor so only authorised users can access it

Username:

Password:

☒ Allow anyone to view the editor, but not make any changes

☐ Not recommended: Allow anyone to access the editor and make changes

☐ ☐ ☐ ☐

Previous Next

34. Click on Next



Learn how to install additional nodes

Node-RED provides a **huge catalog of extra nodes** you can install into the editor.

Many of these nodes can be installed directly from the editor's palette manager feature. However that can cause issues due to the limited memory of the default Node-RED starter application.

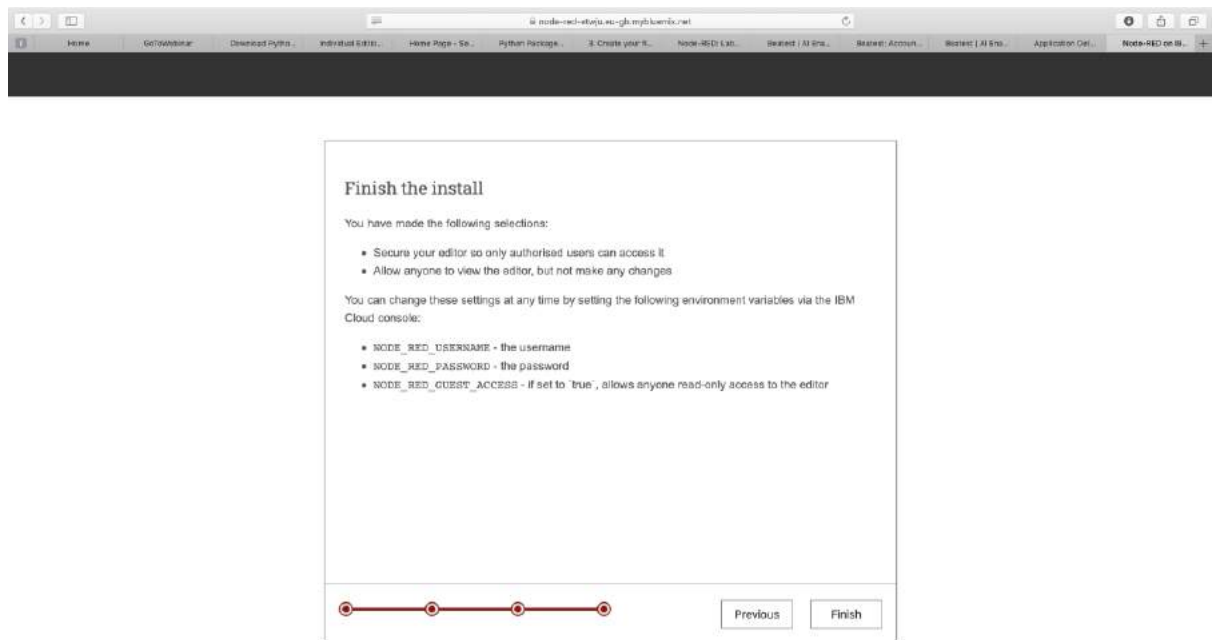
The recommended approach is to edit your application's `package.json` file to include the additional node modules and then redeploy the application. This can be done using the Continuous Delivery feature on the application's IBM Cloud dashboard.

For more information, follow [this tutorial on IBM Developer](#).

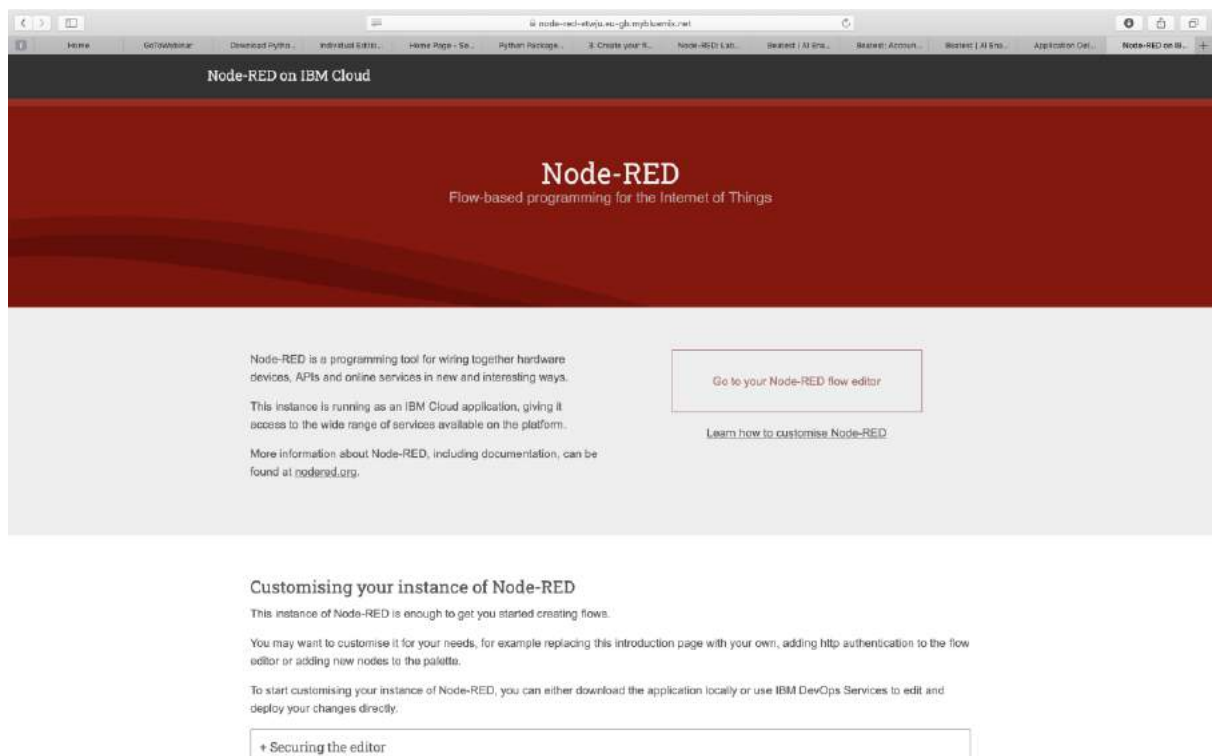
☐ ☐ ☐ ☐

Previous Next

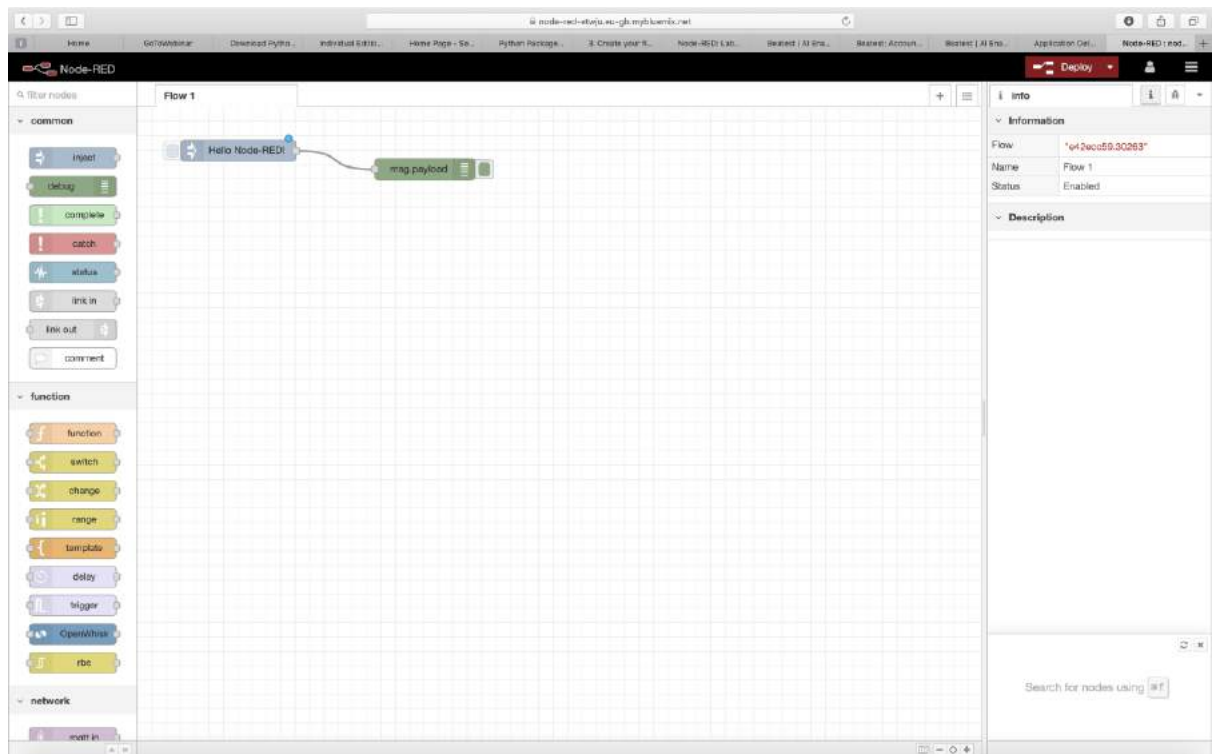
35. Click on Finish



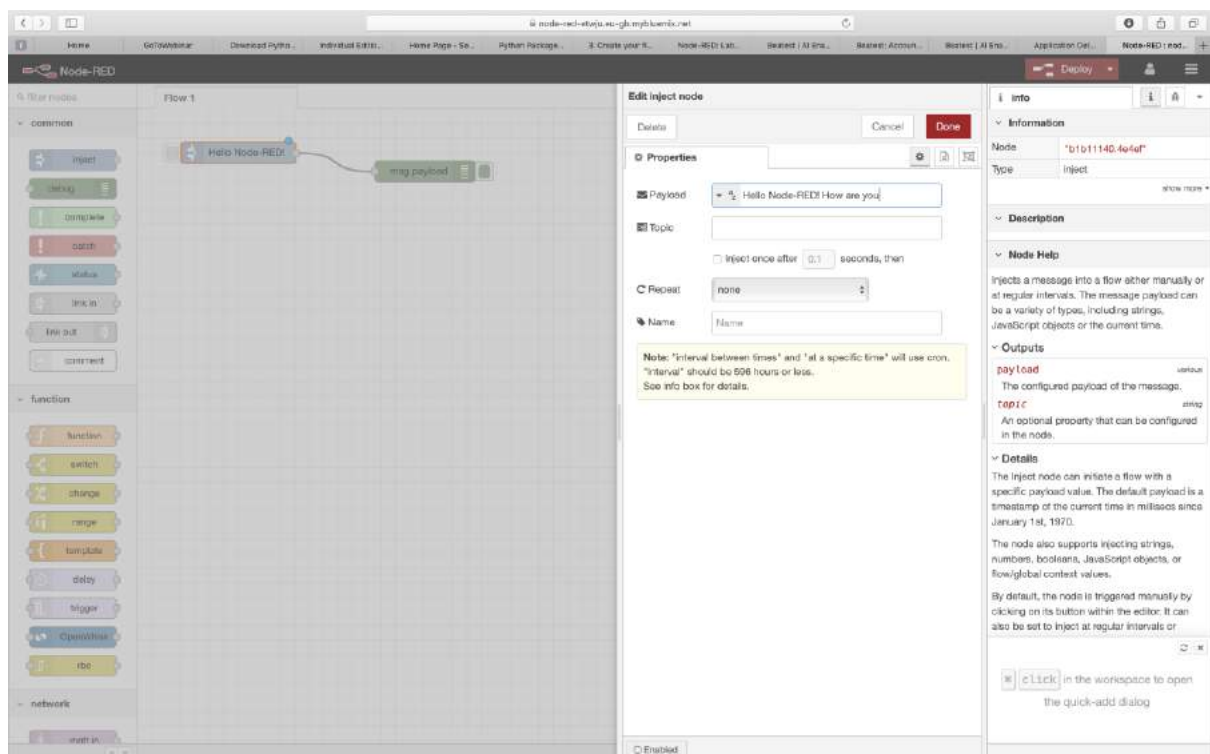
36. Click on “Goto your NodeRed flow editor”



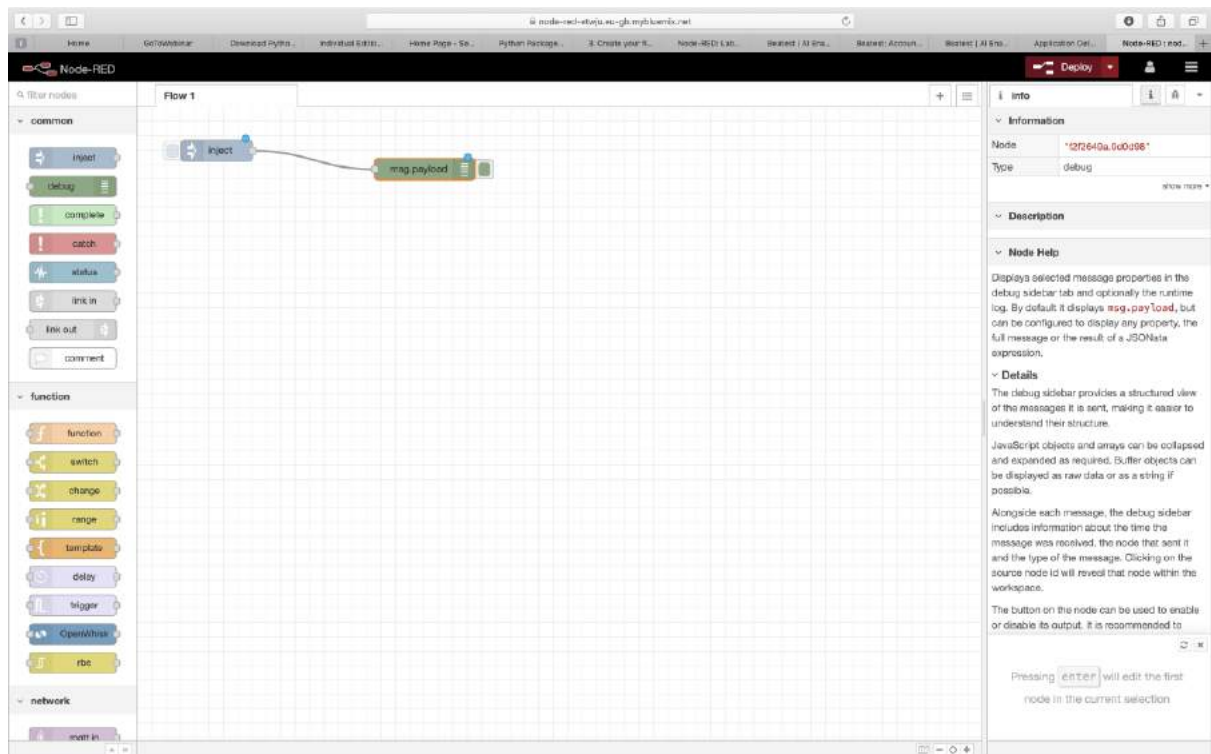
37. This by default show 2 nodes one is inject node and other is debug node



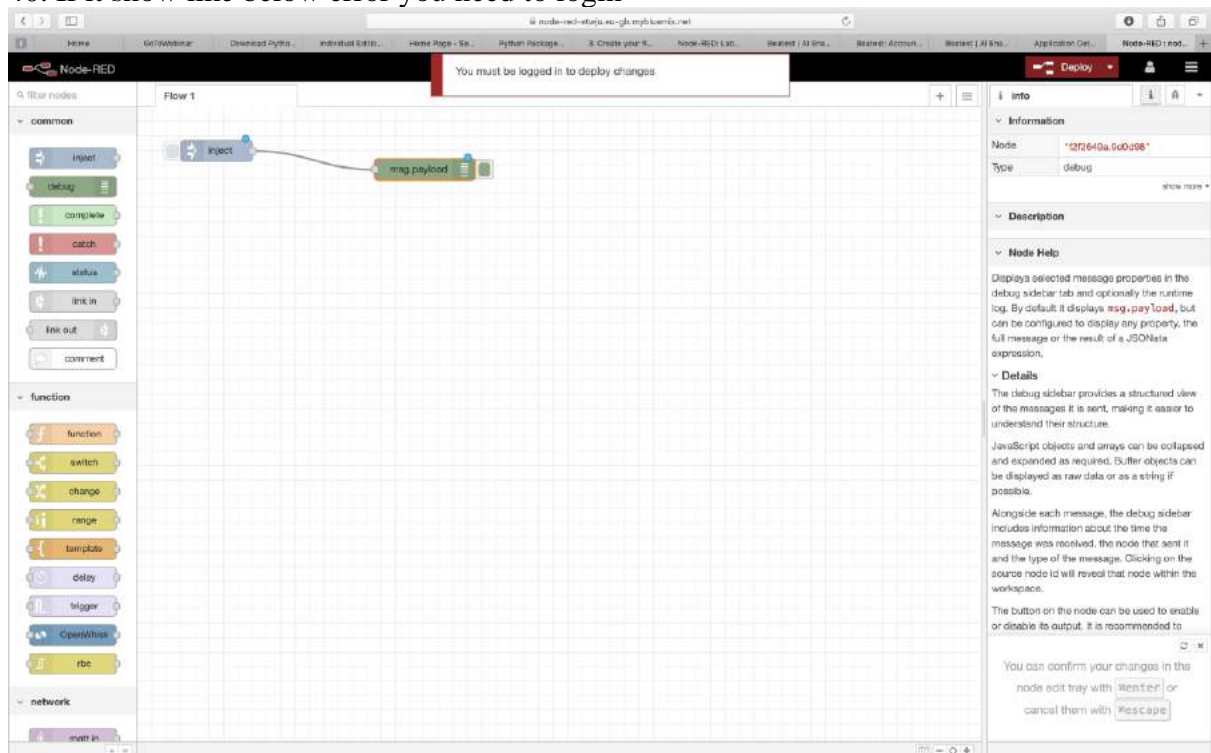
38. Double-click the **inject** node. In the payload field add How are you after Hello Node-RED! And click on Done



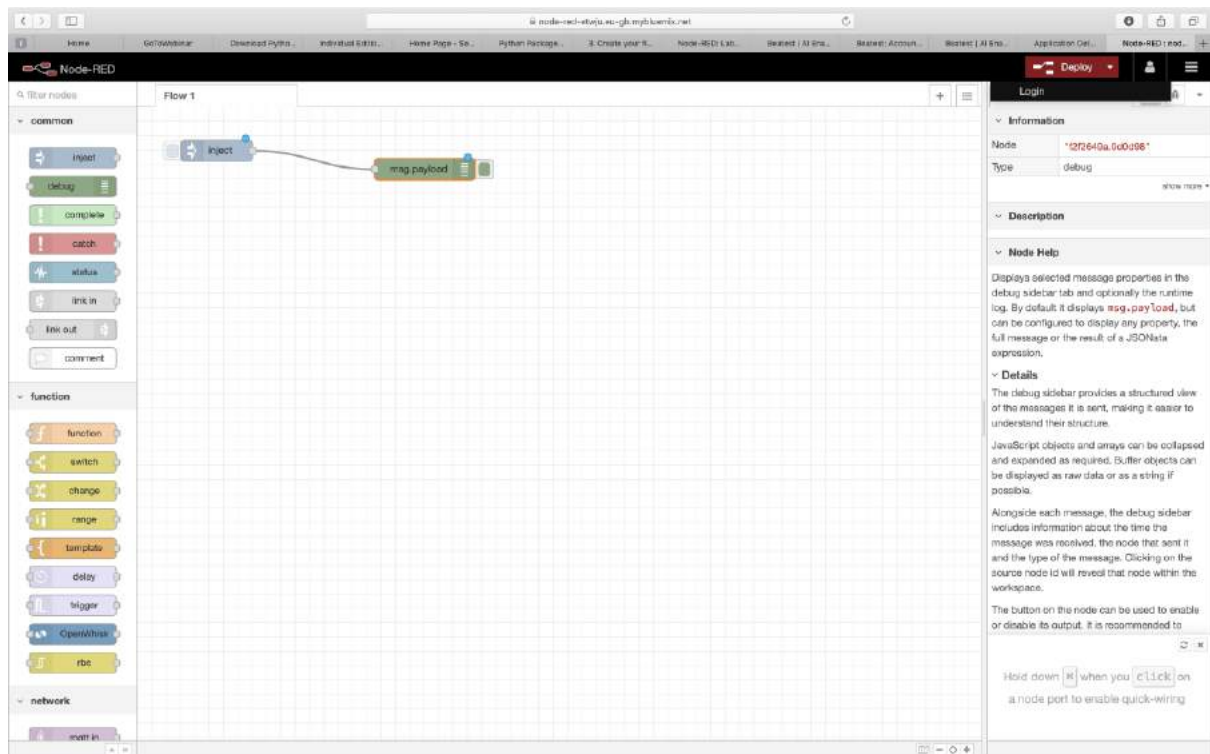
39. Now click on Deploy on top (Blue circles on nodes indicates your app is not deployed)



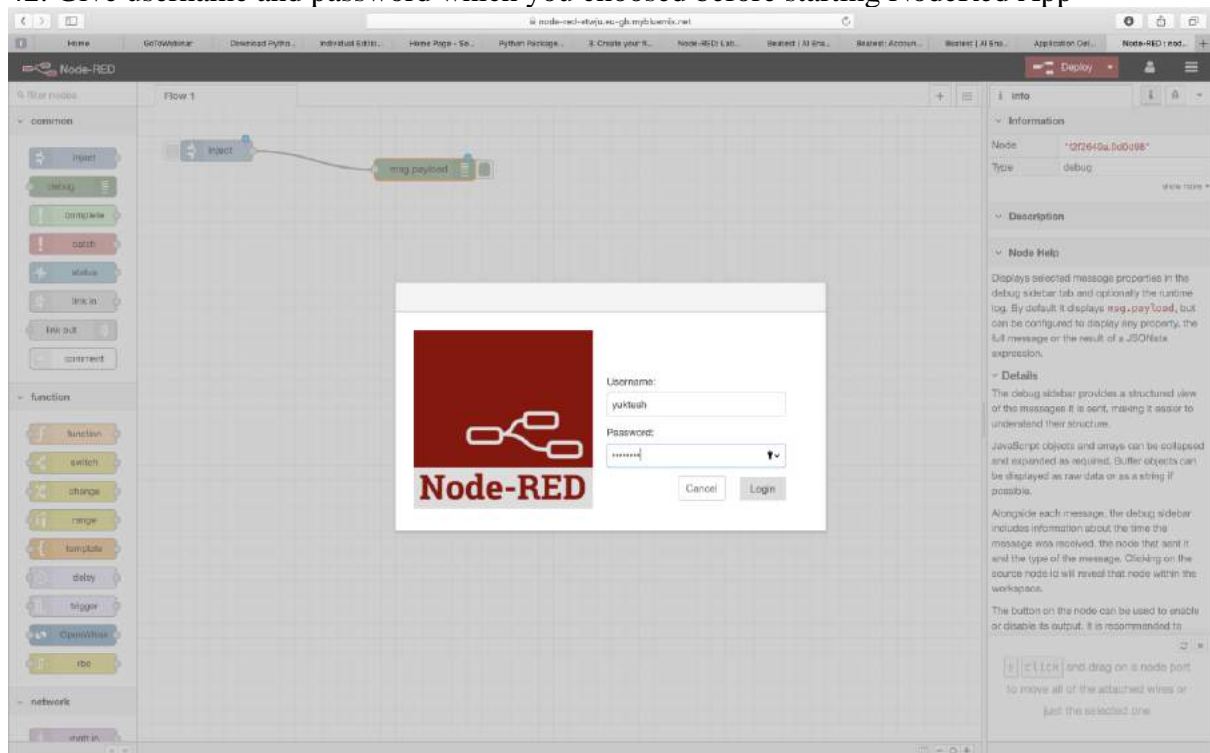
40. If it show like below error you need to login



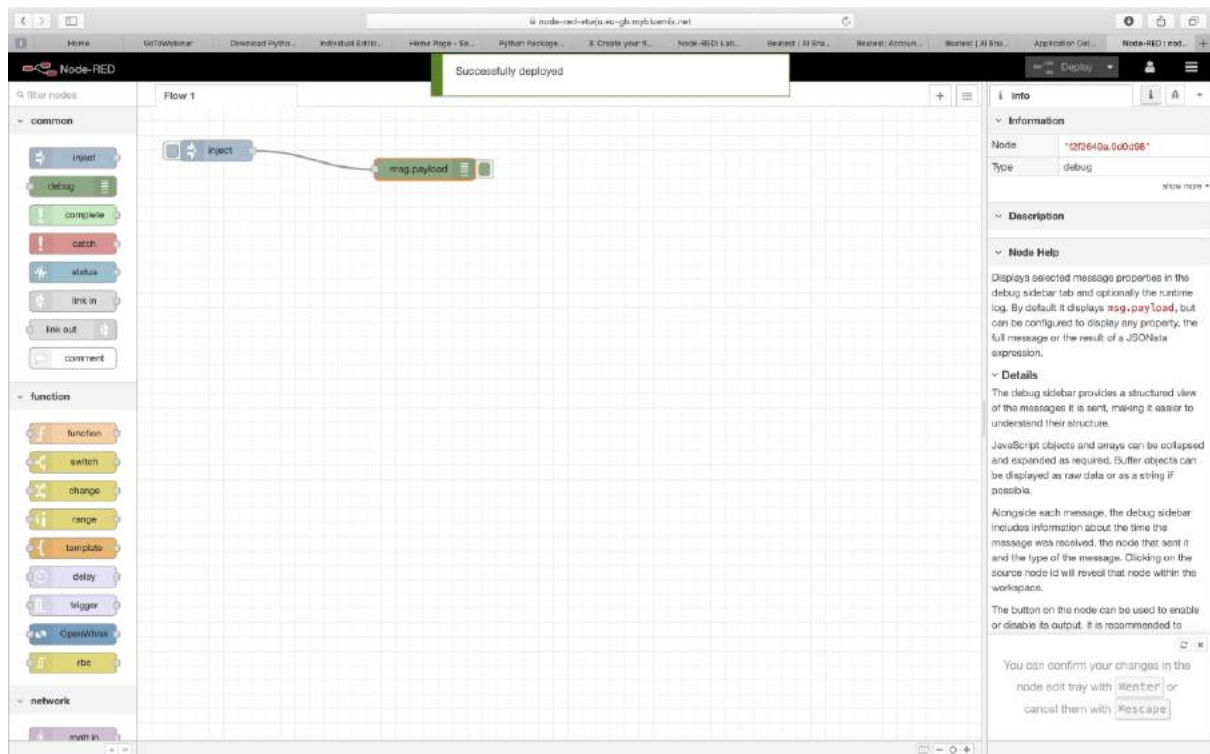
41. Click on human kind symbol and click on login



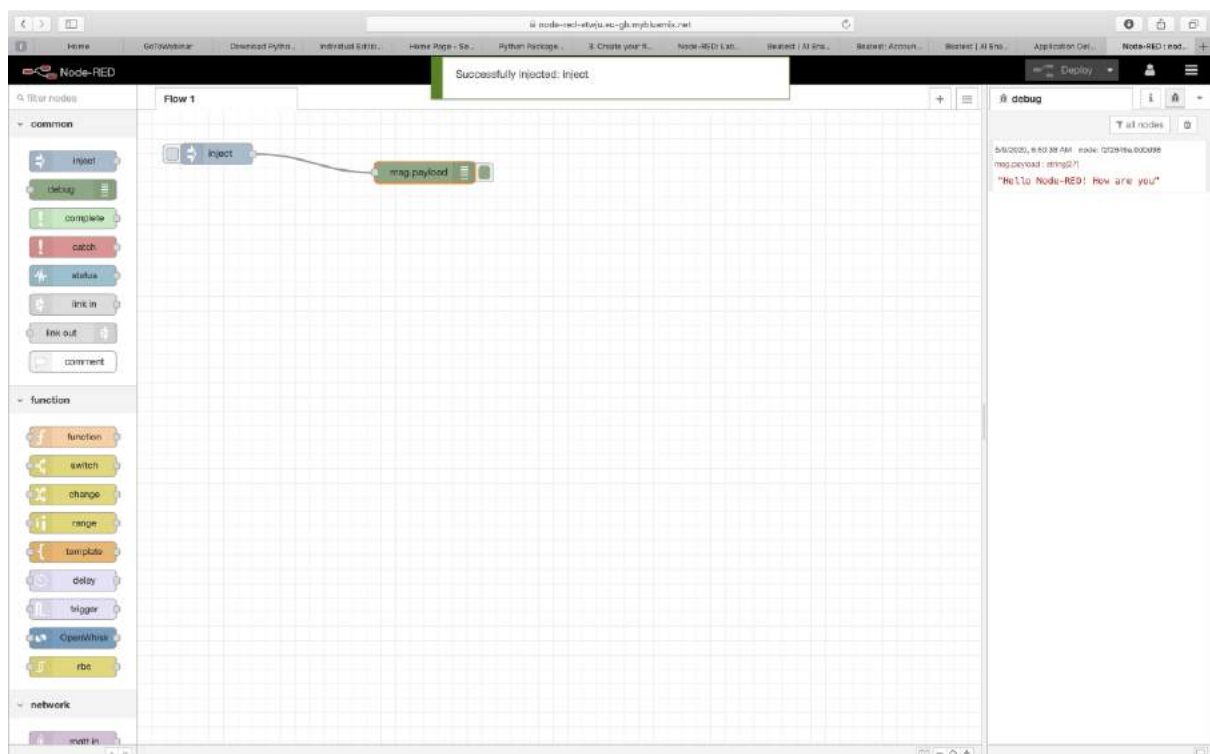
42. Give username and password which you choosed before starting NodeRed App



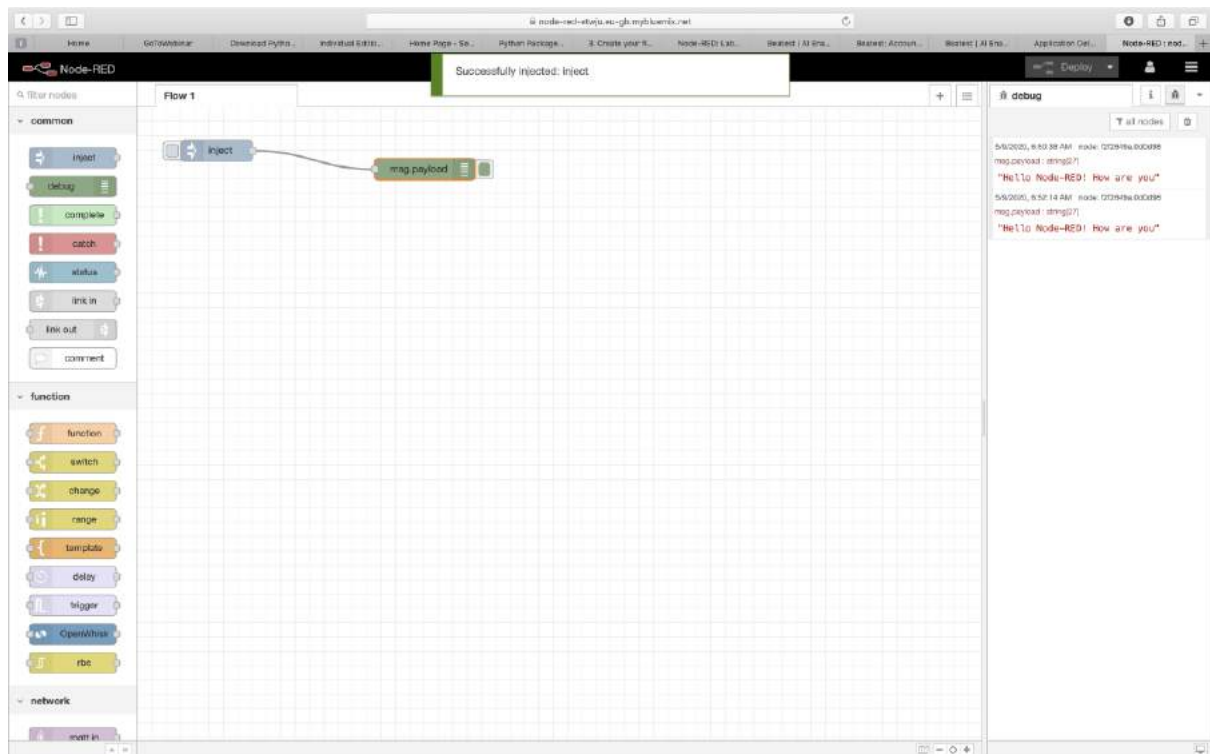
43. Now click on Deploy again it should show successfully deployed



44. Goto debug (tortoise symbol on right) and click on inject node so that you can see output in debug window

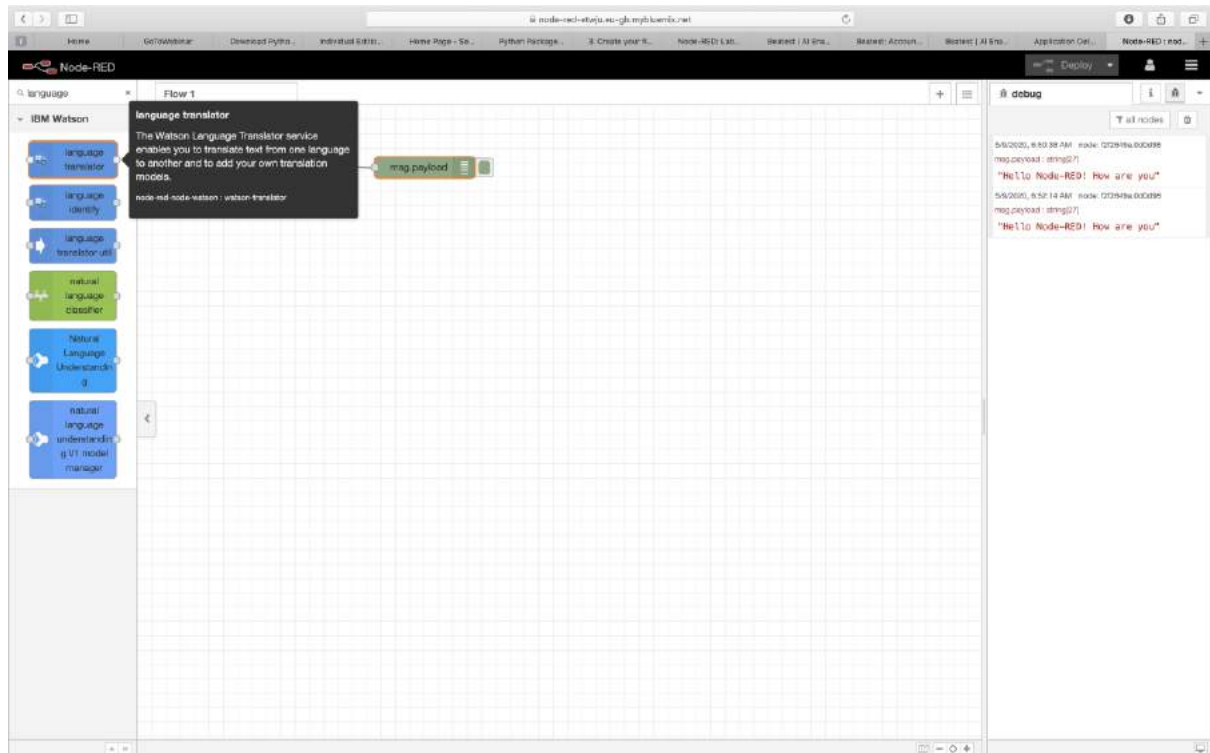


45. Click again and inject again you should see output in debug like below

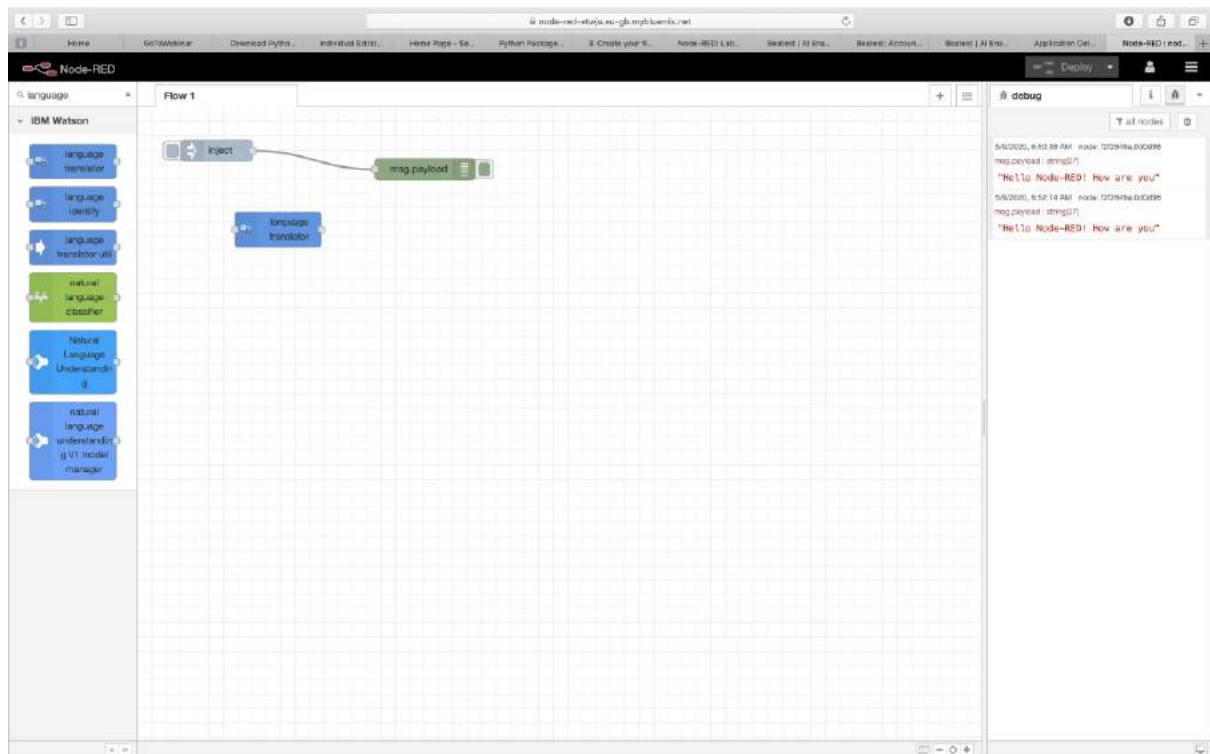


Now lets add Language translator service

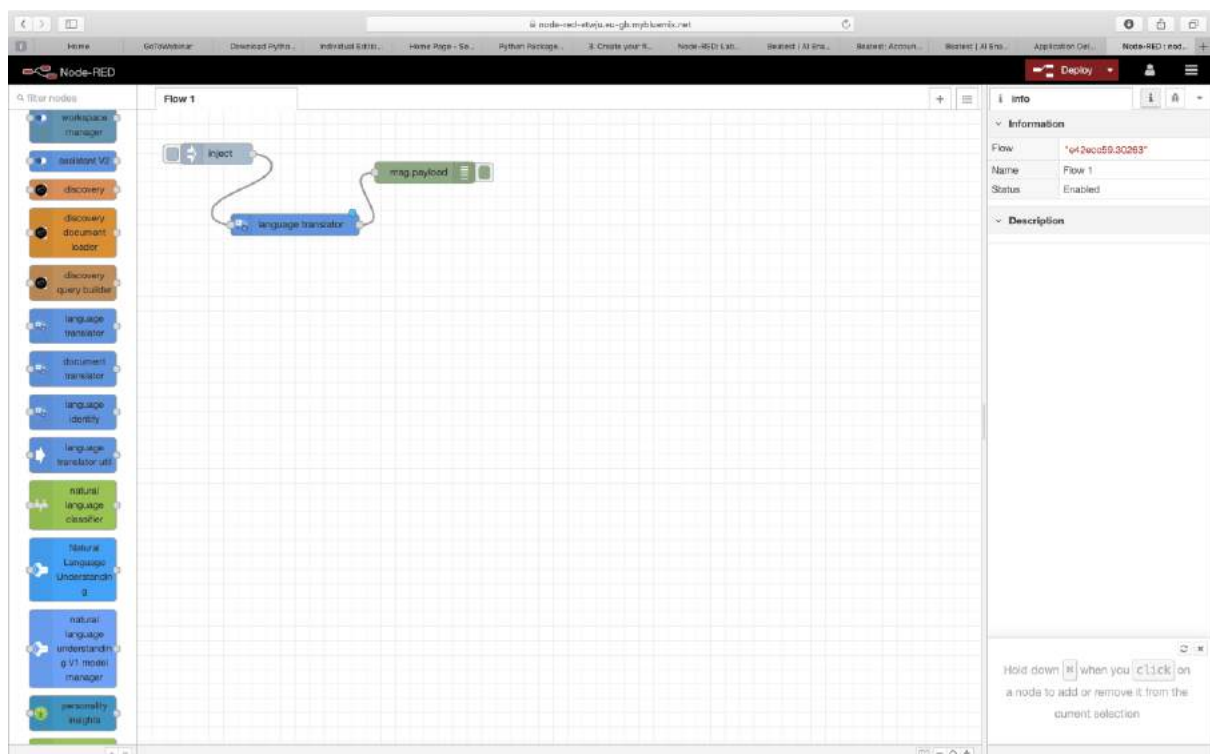
46. In the left top search for language and you should see language translator node



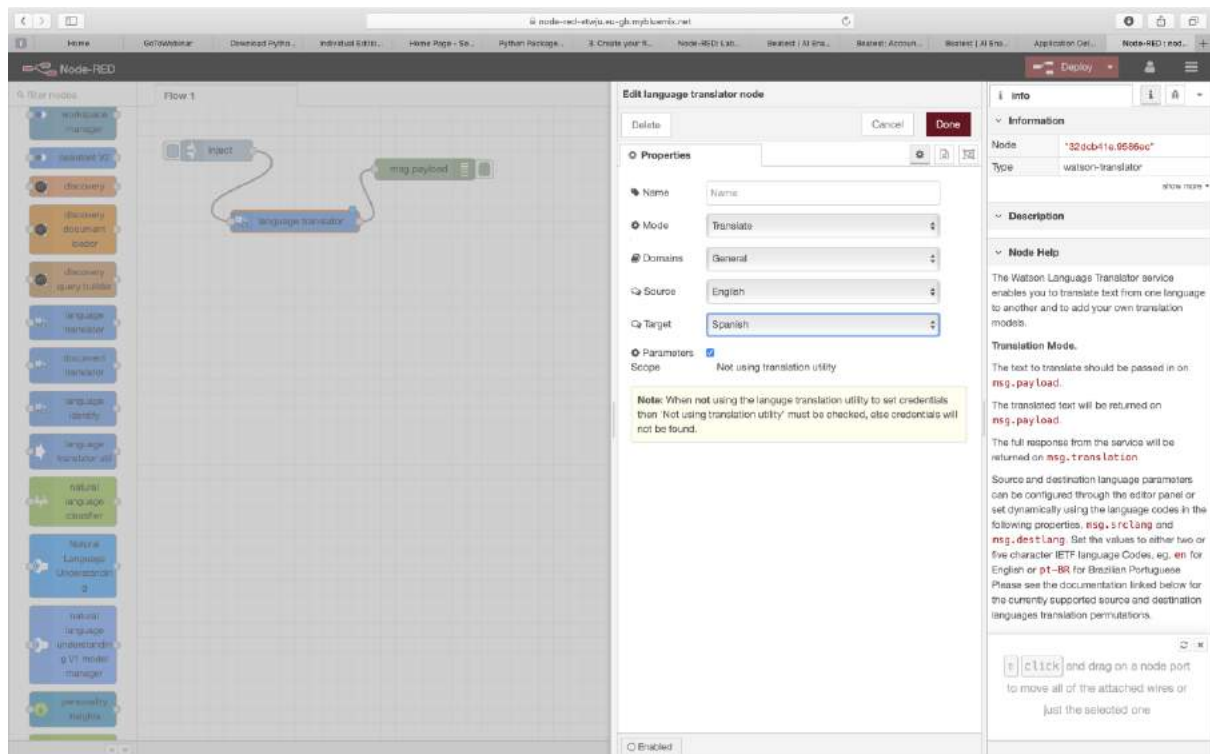
47. Drag and drop on the middle window



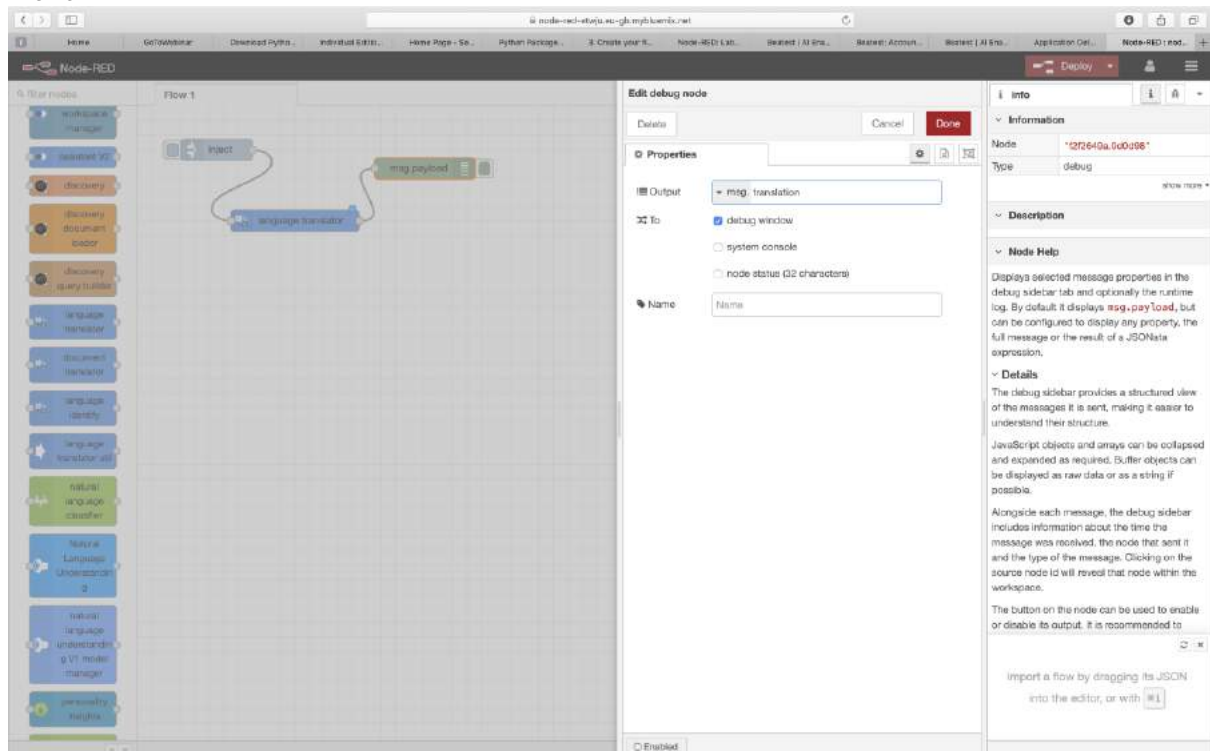
48. Connect (drag and drop) language translator to above 2 nodes and disconnect the first thread between inject and payload like below



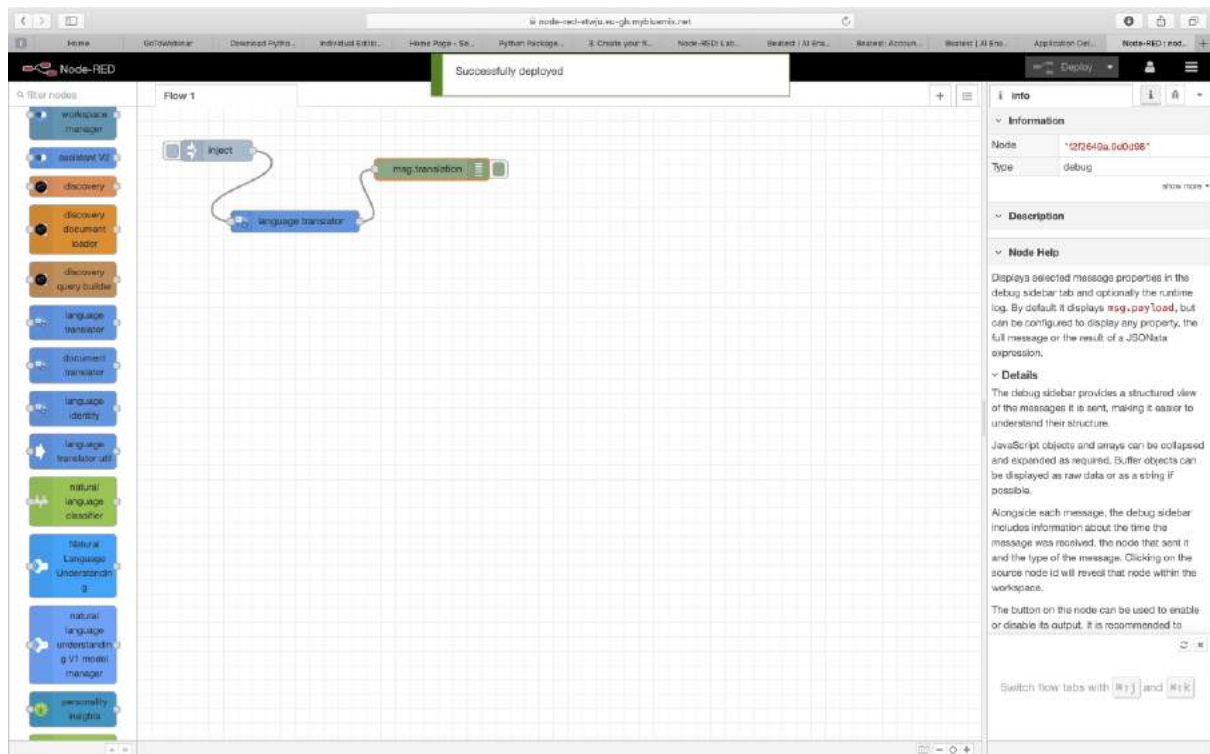
49. Double click on language translator change Target to Spanish and click on Deploy



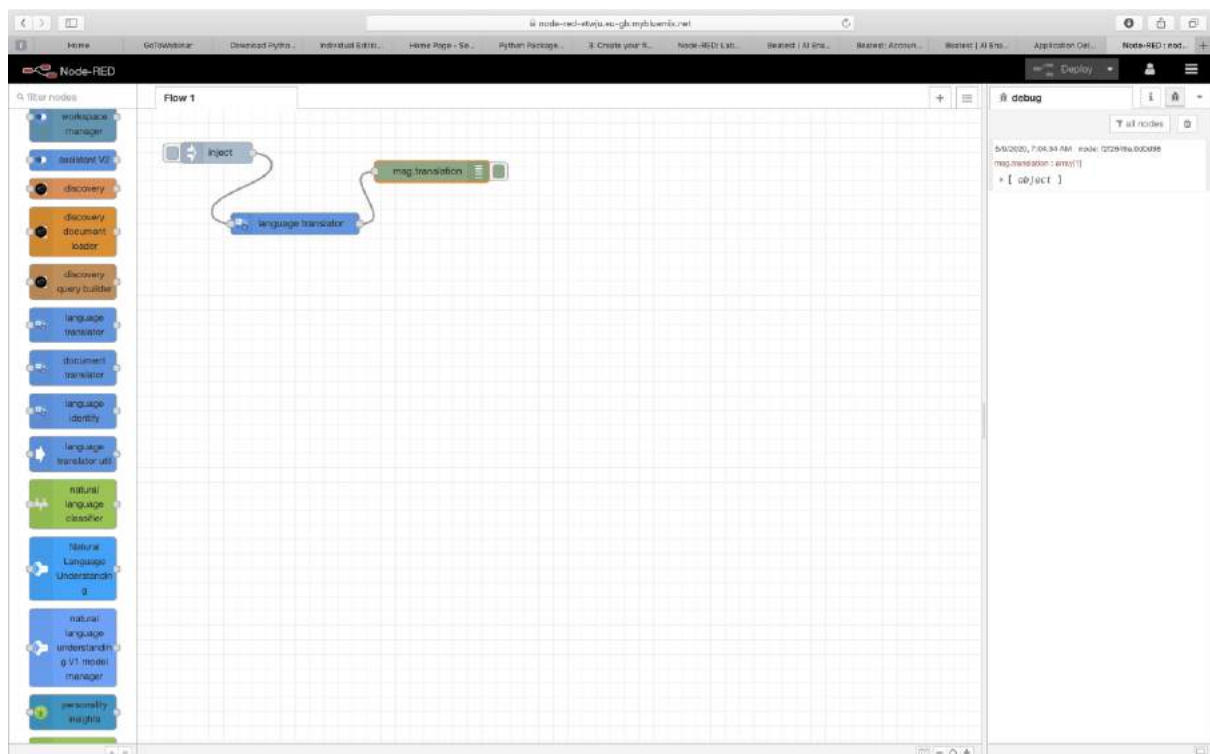
50. double click on debug/output node and change payload to msg.translator and click on Done



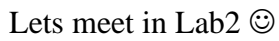
51. Now click on Deploy



52. Click on Debug on right side and now click on inject node you should see below output



53. Now explore the object by click on object and observe the output which got translated in Spanish



Lets meet in Lab2 ☺