# **Lab 3**

# **Hands-On Lab: How to Create a Watson AI Application**

In this lab, you will learn how to create a cloud-based application that will perform natural language processing of unstructured data - the corpus of domain knowledge as well as user-provided questions - and use multiple Watson APIs in a web-based architecture (HTML, CSS, NodeJS) in order to provide a user-interactive cognitive experience.

In order to allow students to fully concentrate on the exercise of binding different Watson services together in a web-based solution and deploying it to the cloud, all front-end and back-end application code will be provided beforehand. The lab will illustrate how to modify a project environment file with the credentials of instantiated microservices before showing how to run, build and deploy the entire application to the IBM Cloud.

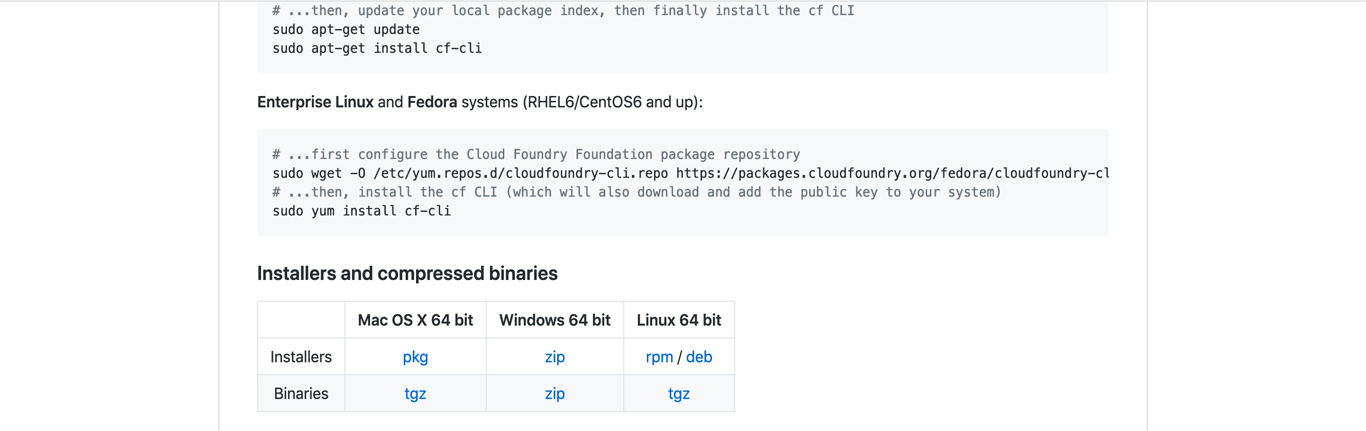
**Prerequisites**

* IBM Cloud Account (Use <https://console.bluemix.net> to sign up for a new account before starting the labs)
* Completed Lab 1 – Natural Language Processing with Watson Discovery
* Completed Lab 2 – Watson Assistant
* Access to lab and PoT assets at <https://github.com/cerebralace/WatsonPoT>

**Step 1: Download Cloud Foundry CLI**

Since we will be using the Cloud Foundry Command Line Interface (CLI) in order to run and deploy our completed Watson weather application, we will first need to download and install the Cloud Foundry CLI to our computer.

1. Go to <https://github.com/cloudfoundry/cli#downloads>
2. Scroll down the page to the **Installers and compressed binaries** section and click on the first **zip** link underneath **Windows 64 bit** to download the Cloud Foundry CLI installer to your Windows computer, preferably to the Desktop. If you use a Mac laptop in order to run this lab, then you would instead click on the first pkg link underneath Mac OS X 64 bit.

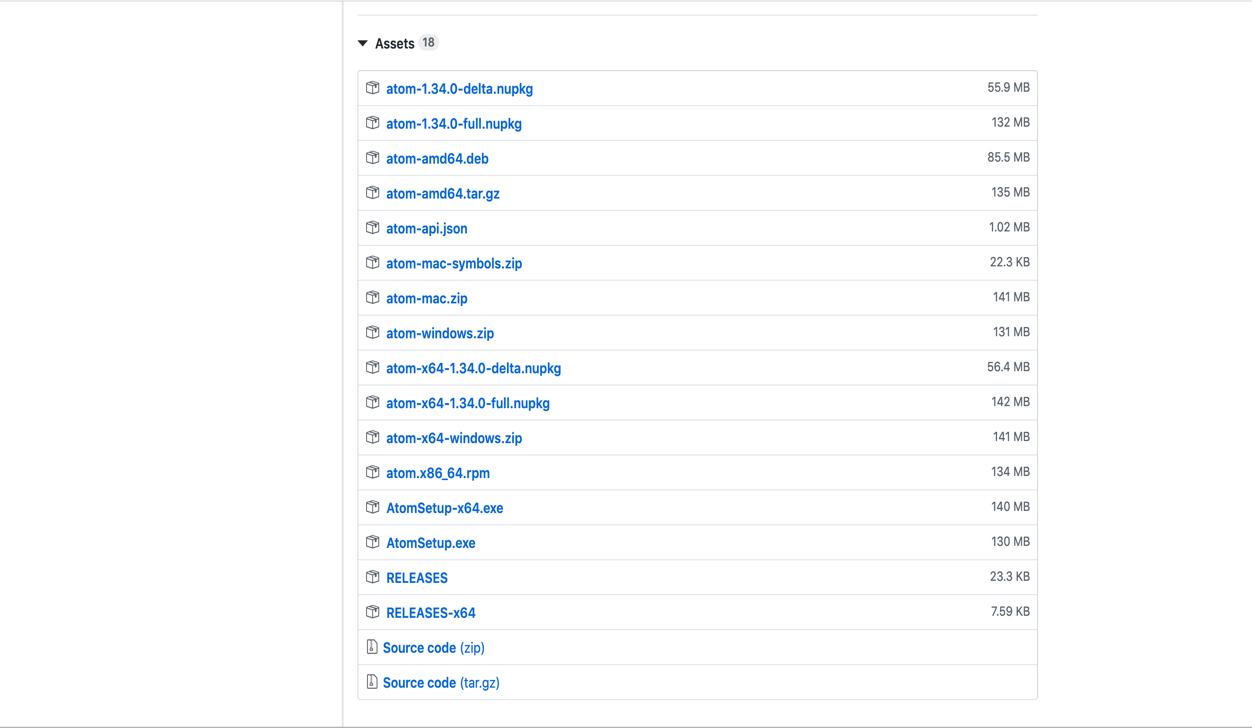


1. Once you have downloaded the Windows 64 bit installer, unzip the **cf-cli-installer\_6.43.0\_winx64** folder in order to open it and double-click the **cf\_installer.exe** file to begin installation.
2. Upon completion of installation, you can now use the Cloud Foundry CLI for running and deploying your app to the IBM Cloud.

**Step 2: Download the Atom code editor**

Before starting development, we will need a code editor in which to write and test the code for our Watson weather application. The Atom code editor is a robust tool which allows developers to write code in a variety of programming languages such as Javascript, Python, C++ and more in order to create different applications. Let’s download and install the Atom code editor.

1. Go to <https://github.com/atom/atom/releases/tag/v1.34.0>

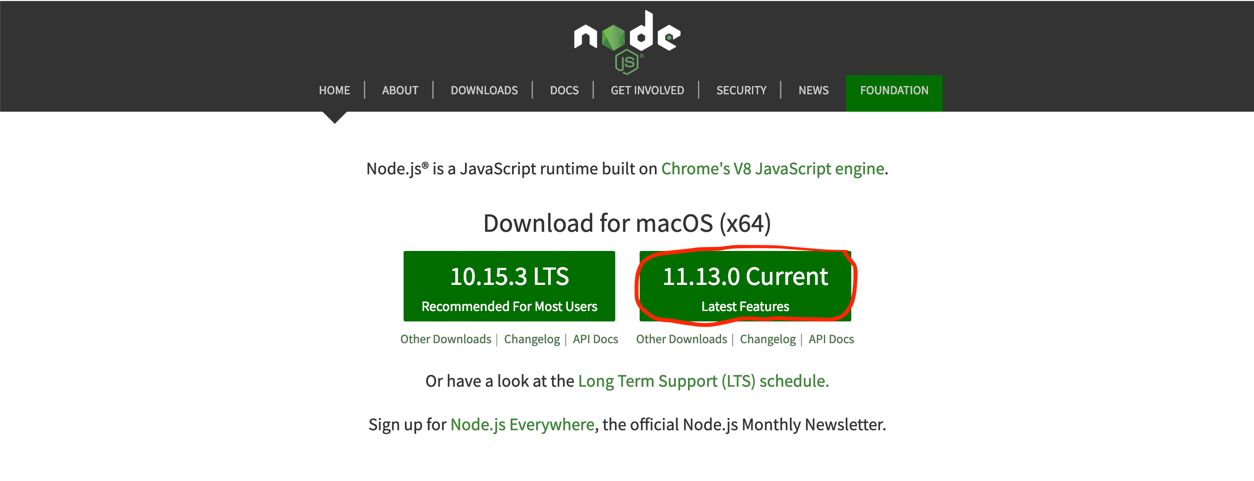


1. Scroll down to **Assets** and click on the **atom-windows.zip** link to download the Atom code editor to your Windows computer.
2. After downloading the zip file, unzip and open the **atom-windows** folder. Select and open the **Atom** folder. Locate the **atom.exe** file and double-click it in order to install it to your computer. As in the previous step, it is recommended to have all of these programs (Cloud Foundry CLI and Atom code editor) downloaded and installed to the Desktop.
3. You should now have the Atom code editor installed and ready to use for development in the rest of the lab.

**Step 3: Download and Install NodeJS**

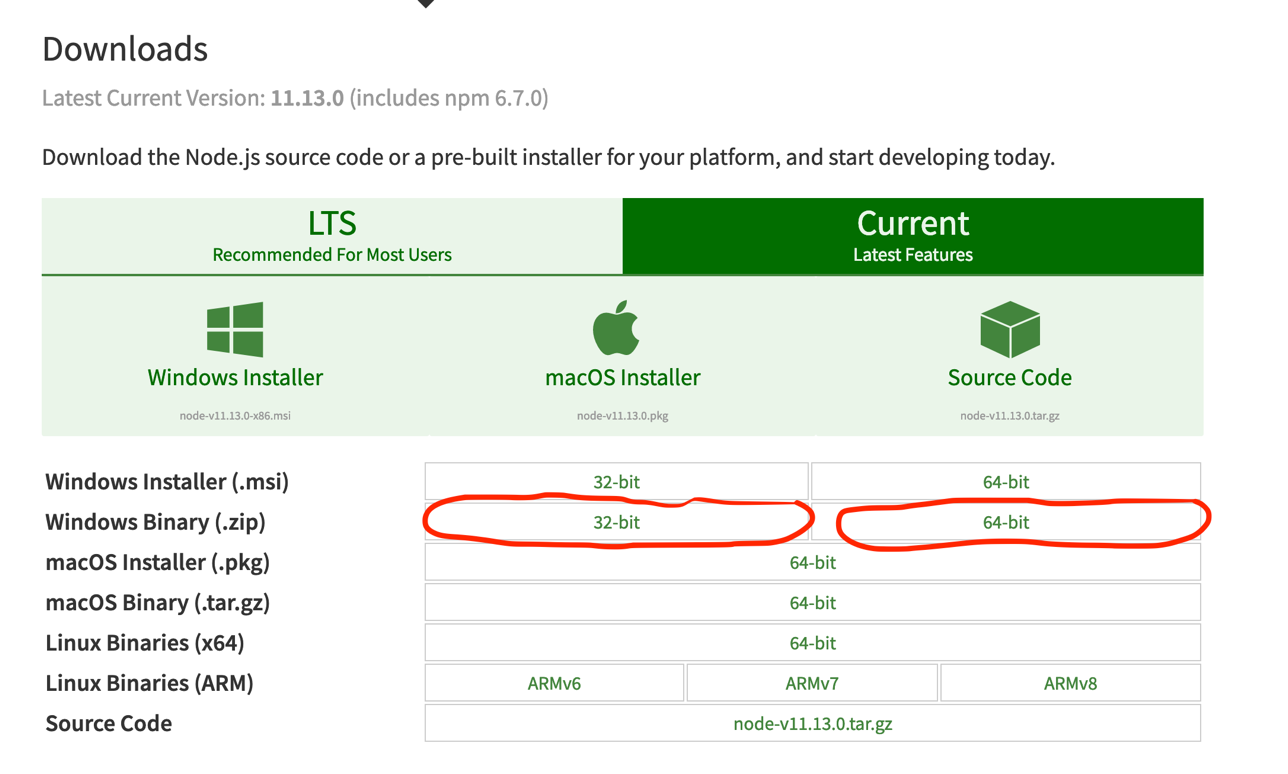
In order to properly compile the application code using various npm commands, we will also need to download and install the latest version of NodeJS to your computer.

1. Go to <https://nodejs.org/en/>
2. If you have a Mac laptop, please click the green button marked **11.13.0 Current** to download the latest version of NodeJS to your computer.



Once the node-v11.13.0.pkg is downloaded to your computer, double-click it to install NodeJS to your computer.

1. If you have a Windows computer, go to <https://nodejs.org/en/download/current/> and scroll to **Windows Binary (.zip)**. Double-click the **32-bit link for a Windows x86** or **64-bit link for a Windows x64**.



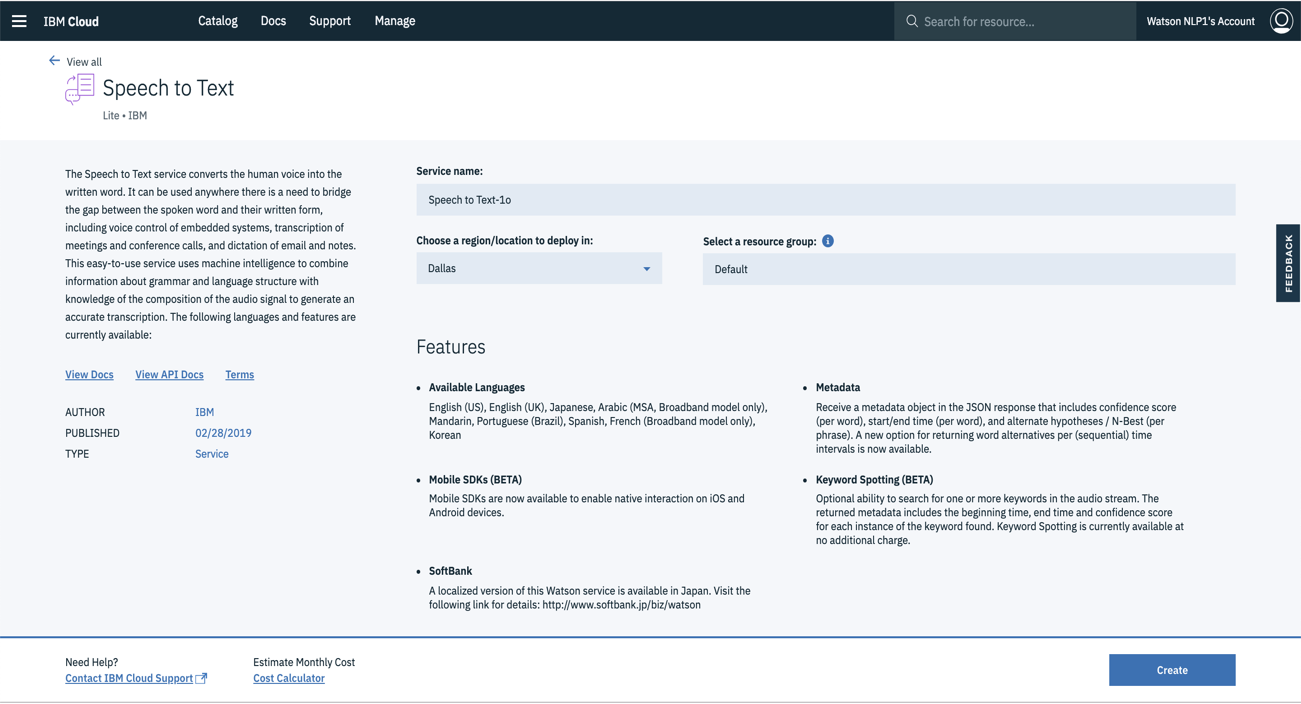
Once the zip file is downloaded to your computer, you can unzip it in order to install NodeJS to your computer.

1. Upon completion of installation, you should have the latest version of NodeJS running on your computer, which will allow you to use npm in order to compile and run your application code prior to deployment.

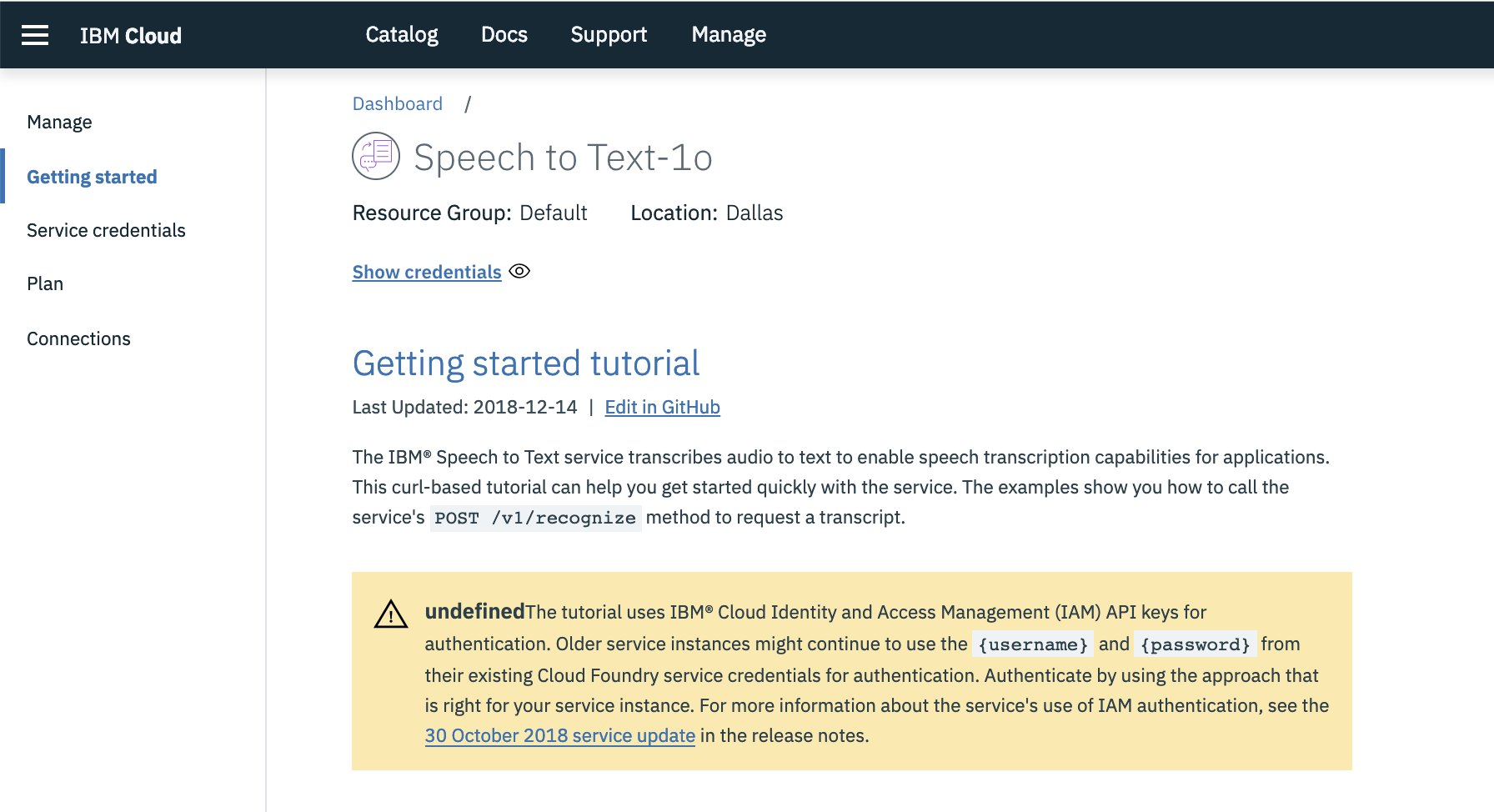
**Step 3: Create Speech to Text Instance**

We would like our Watson weather application to contain a chatbot capability that will not only prompt users to type in their questions, but also directly speak to the chatbot using their computer’s mic. Thus, we will need to add the Watson Speech to Text capability and pair this with our Watson Assistant instance. Let’s first create the Speech to Text Instance within our IBM Cloud account and copy its credentials.

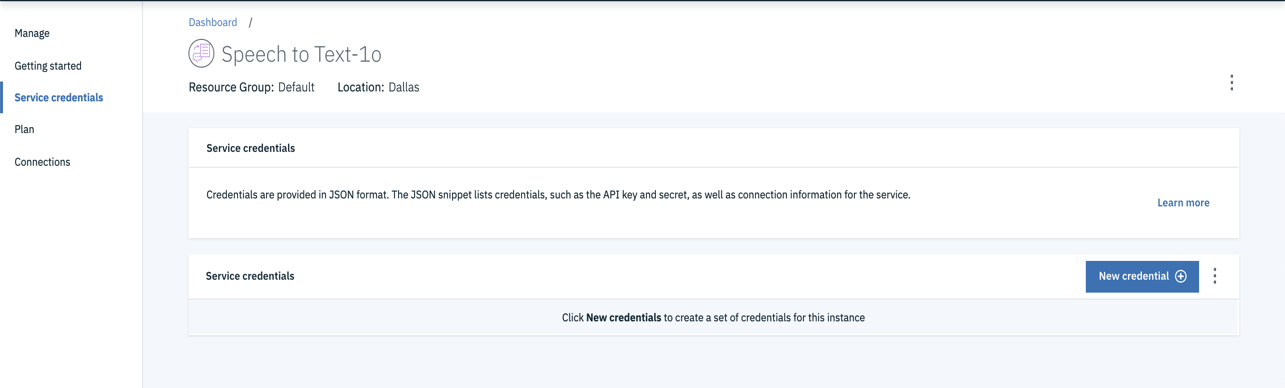
1. Go to the Speech to Text page on the IBM Cloud Catalog: <https://console.bluemix.net/catalog/services/speech-to-text>
2. Give your service a unique name (or keep the randomly generated name), leave the region as Dallas and the resource group as Default. Click **Create**.



1. Upon creation of your Speech to Text instance, you will be taken directly to the Speech to Text Get Started tutorial. On the left side of the page, click **Service credentials.**



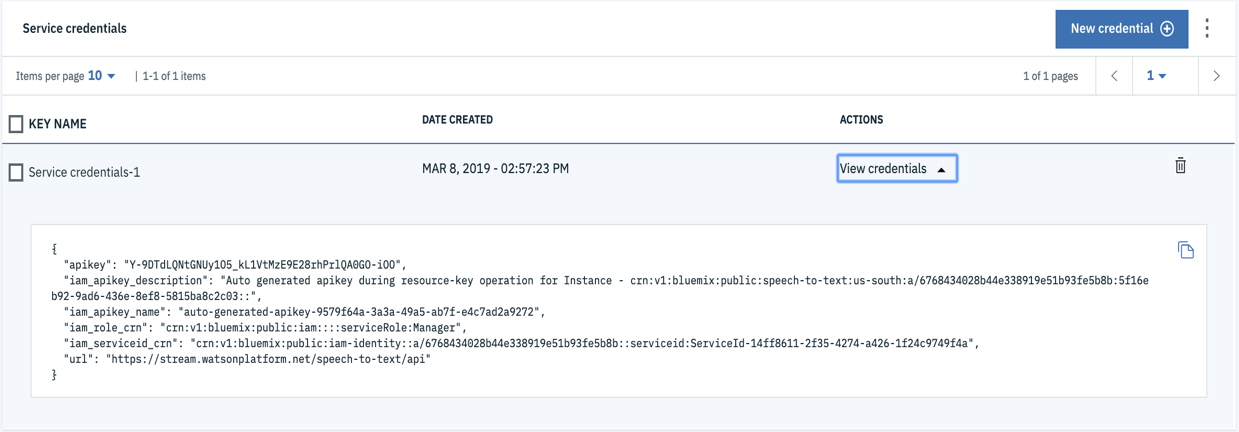
1. Create a new set of credentials for this instance by clicking on **New credential** inside of the second Service credentials table.



1. Leave all fields within the **Add new credential box** unchanged and click **Add.**



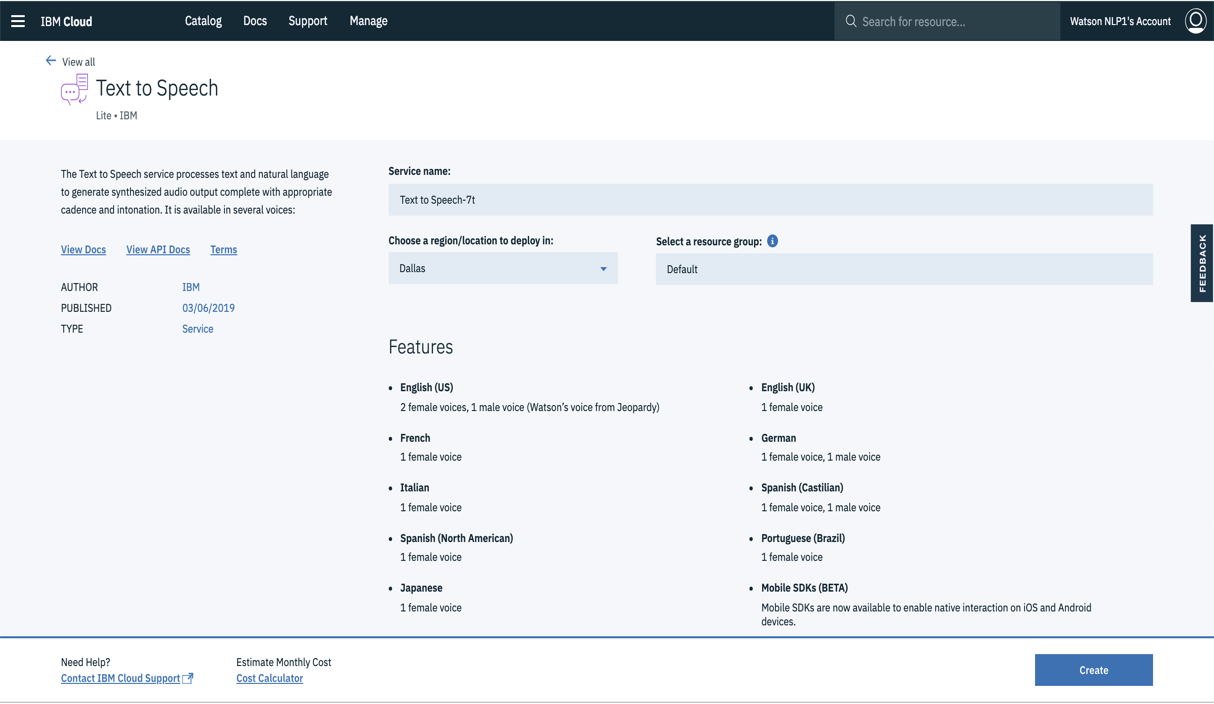
1. Underneath **KEY NAME**, you should see your created Service credentials-1 variable. Click on **View credentials** and please copy your **apikey** and **url**. These credentials will need to be specified in the environments file of your application, which we will cover later in the lab.



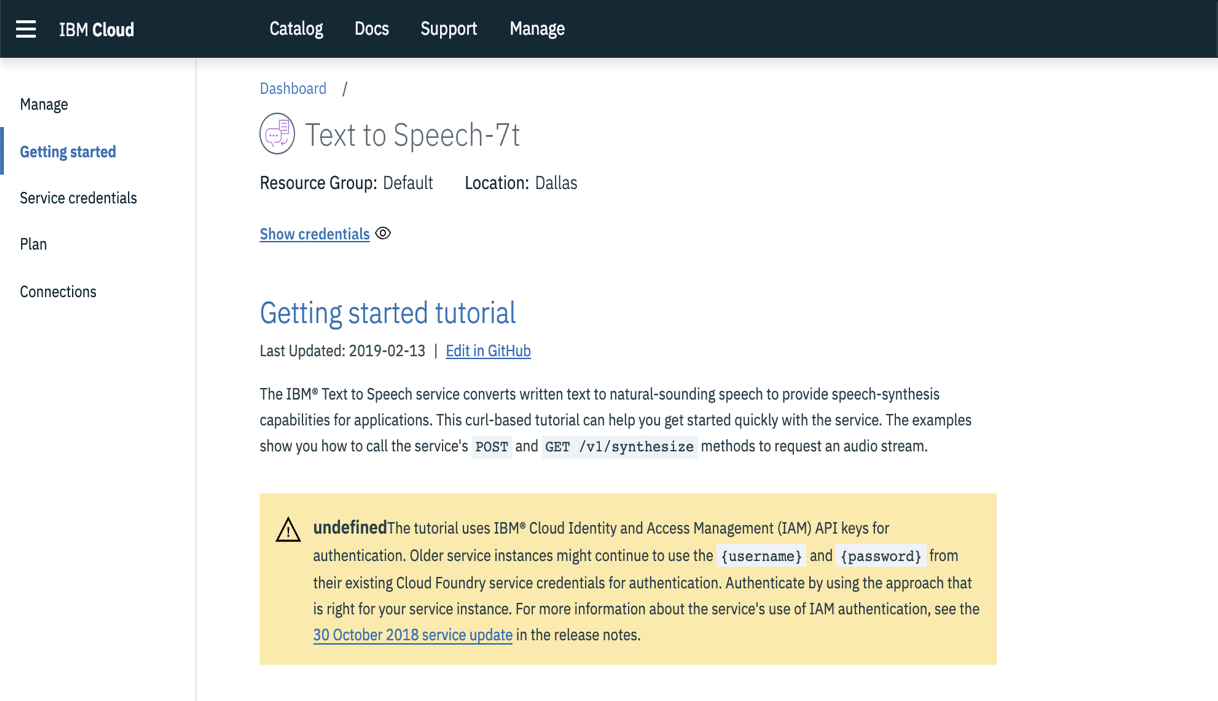
**Step 4: Create Text to Speech Instance**

Now that we have created a Speech to Text service to incorporate in our Watson weather app, we will also create a Watson Text to Speech instance in order to generate synthesized audio output for the Watson chatbot within the application. In creating a new instance of Text to Speech, we will also need to copy its credentials for later use.

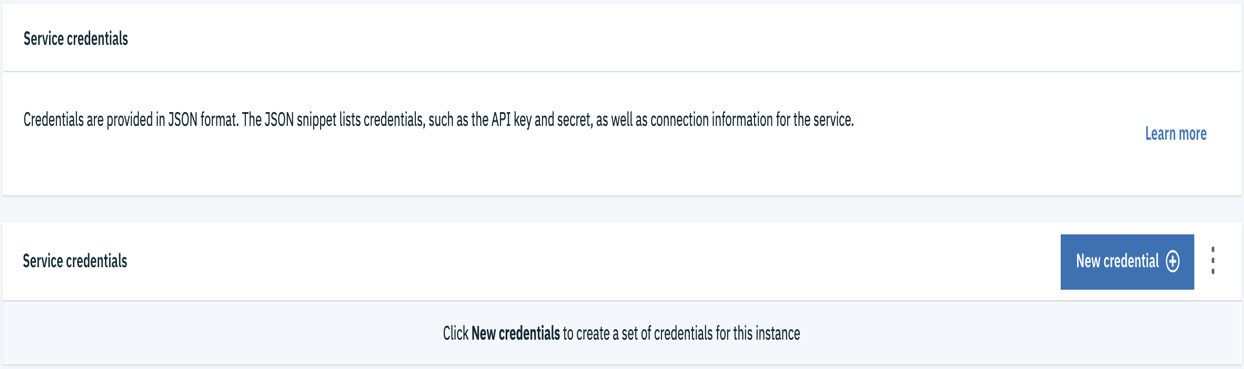
1. Go to the Text to Speech Catalog page on IBM Cloud: <https://console.bluemix.net/catalog/services/text-to-speech>
2. Keep the service name unchanged (or give it a unique name), leave the region as Dallas and the resource group as Default. Click **Create**.



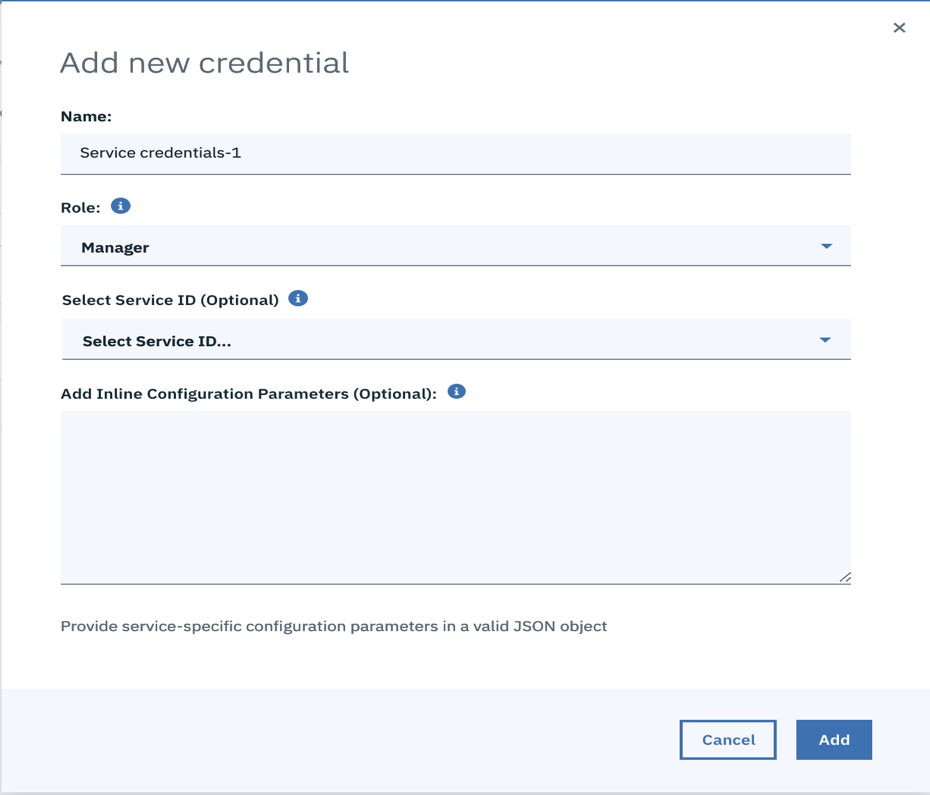
1. Upon creation of the Text to Speech service, you should be automatically taken to the **Getting started tutorial** page. On the left side of the page, click **Service credentials**.



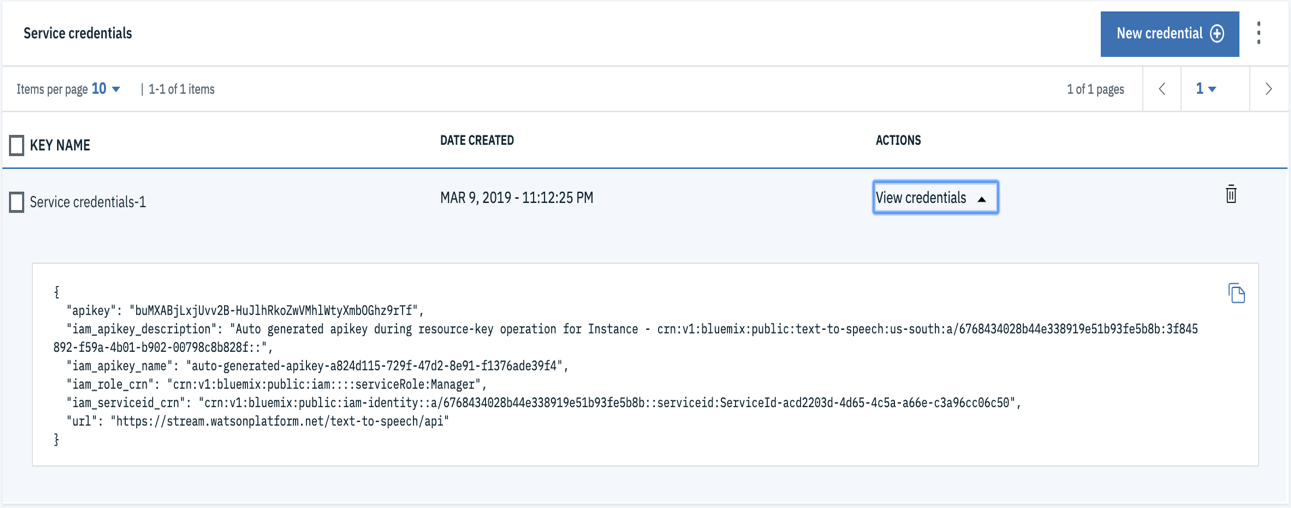
1. Create a new set of credentials for this instance by clicking on **New credential** inside of the second Service credentials table.



1. Leave all fields within the **Add new credential box** unchanged and click **Add.**



1. Underneath **KEY NAME**, you should see your created Service credentials-1 variable. Click on **View credentials** and please copy your **apikey** and **url**. These credentials will need to be specified in the environments file of your application, which we will cover later in the lab.

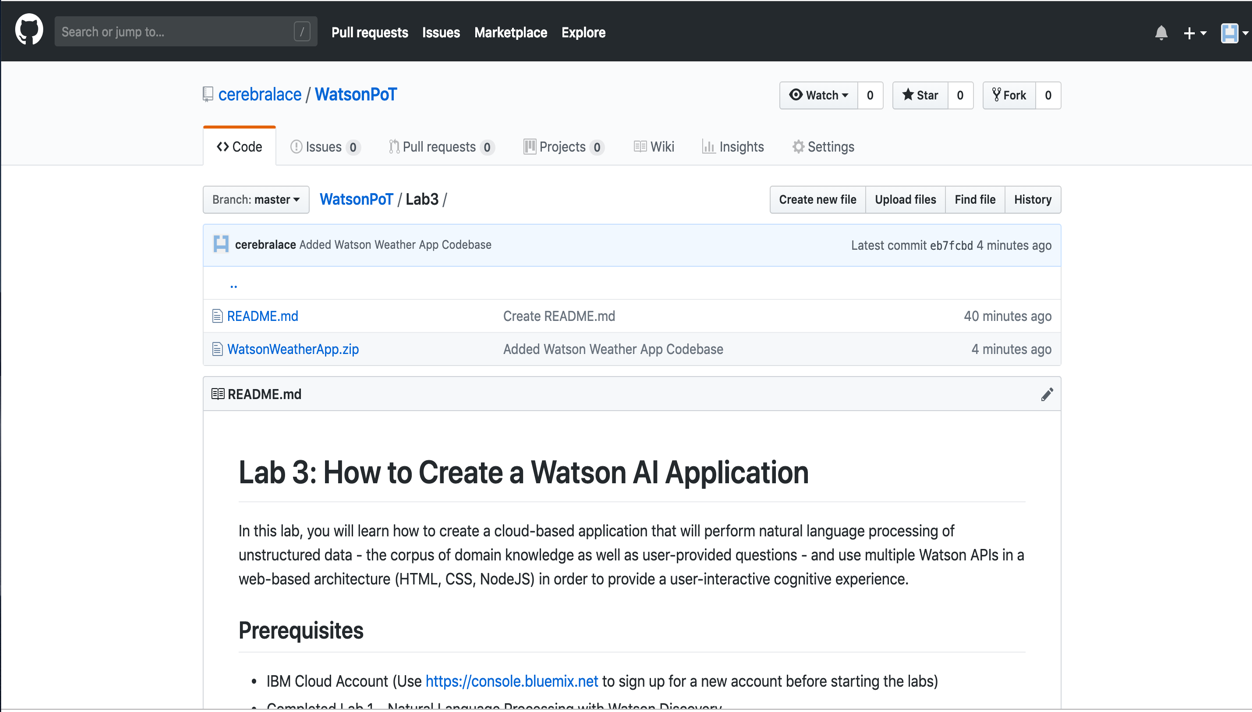


**Step 5: Download Application Codebase**

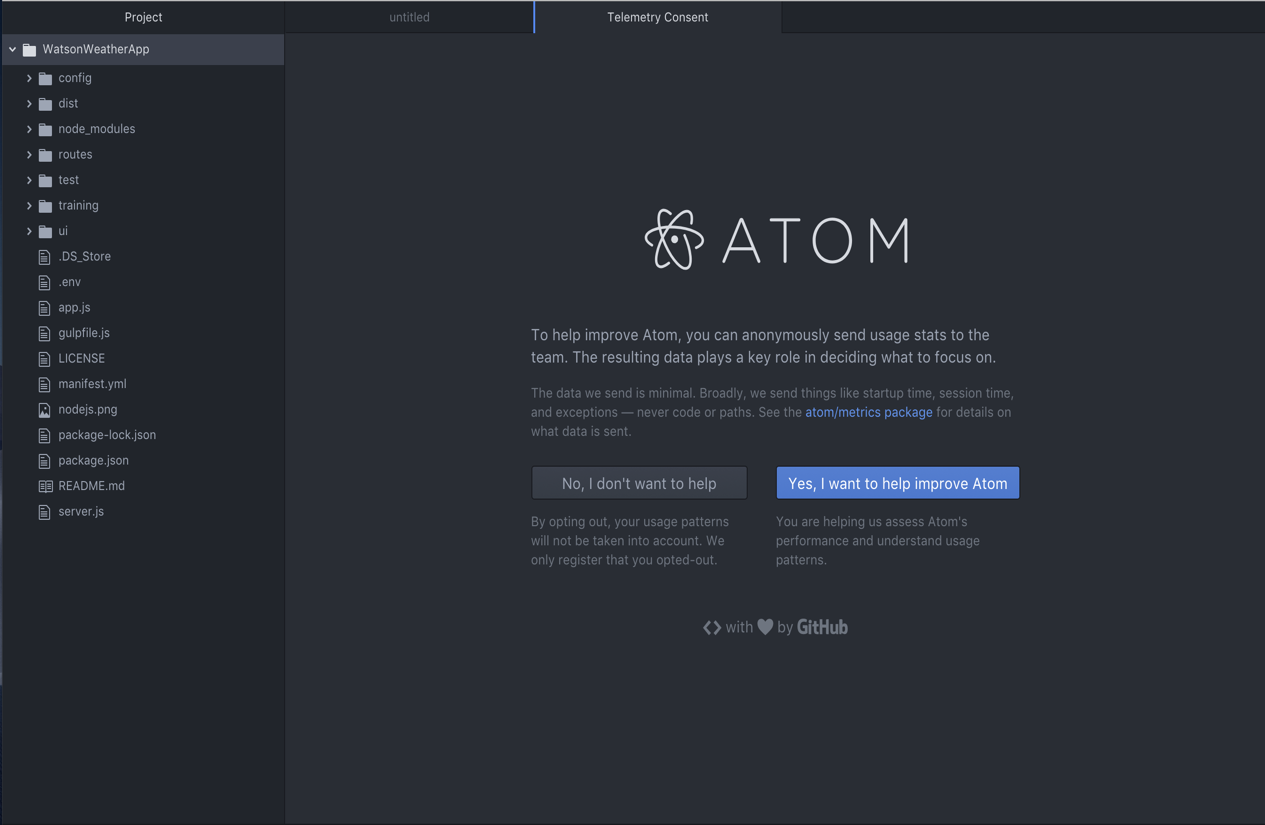
At this point, you should have four Watson services in your IBM Cloud account: Watson Discovery, Watson Assistant, Watson Speech to Text and Watson Text to Speech. With the exception of Discovery, whose query results have already been integrated into Assistant, we will be embedding these services within a web-based application.

Fortunately, you will not have to develop the front-end code (HTML, CSS and Javascript) for the application nor will you be responsible for writing the Javascript code for binding the back-end services. You will be provided with the entire application codebase and will only need to copy and save the credentials for each of your Watson services in the environment file and deploy it to the IBM Cloud using the Cloud Foundry CLI. Let’s first grab the application codebase from the GitHub page.

1. Go to <https://github.com/cerebralace/WatsonPoT/tree/master/Lab3>
2. Click on the WatsonWeatherApp.zip file and then select Download in order to save it to your Desktop.



1. Once you have downloaded this zip file, unzip it to access the WatsonWeatherApp folder.
2. Let’s open the folder within the Atom code editor. Find the Atom code editor startup link on your Desktop and double-click it to open.
3. Inside of the Atom code editor, select **File** and then **Open.** Find the WatsonWeatherApp folder on your Desktop and click **Open**.

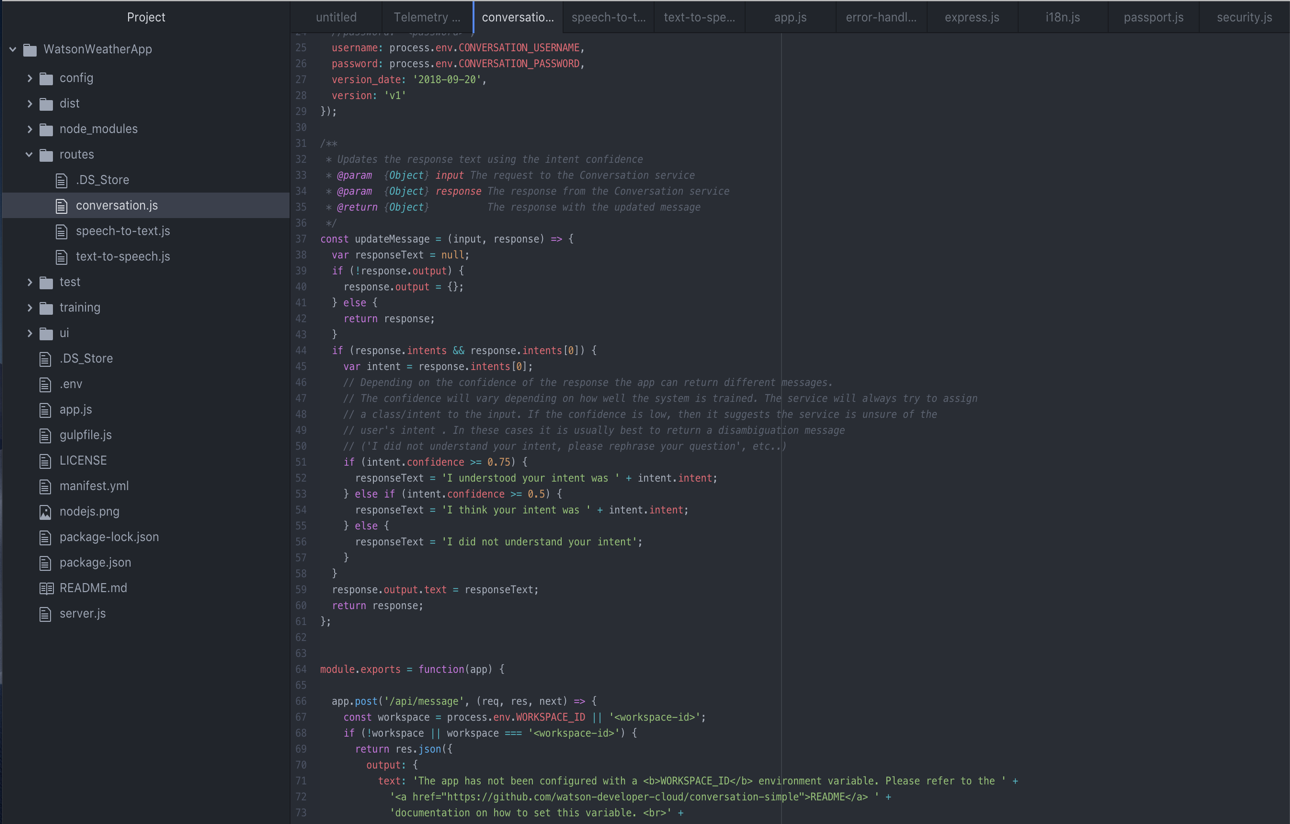


You should now be able to see the files and folders within the WatsonWeatherApp folder on the left side of the page. We will be editing some of the application files within Atom in the next section.

**Step 6: Modify Application Codebase**

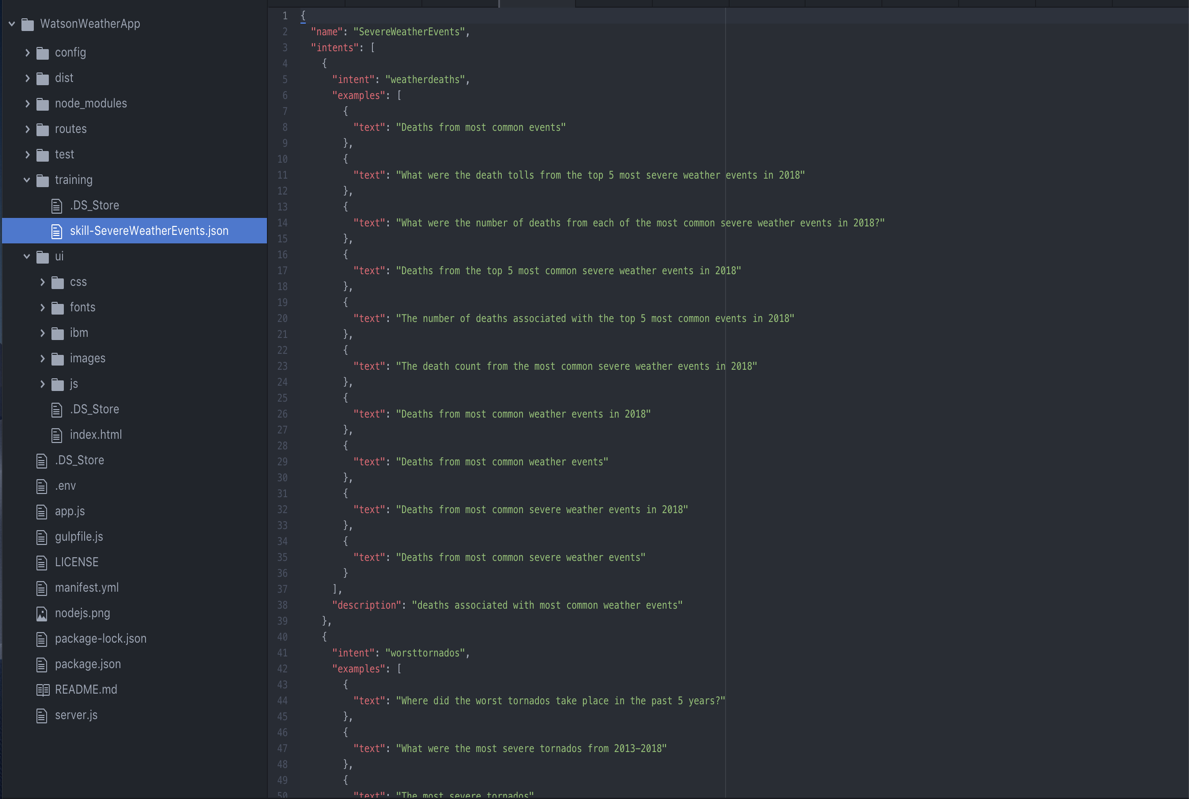
The WatsonWeatherApp folder contains all of the front-end and back-end code necessary for our application. Although we will only be making a few changes to some of the files via Atom, it is nonetheless useful to familiarize yourself with the general organization of the project folder.

1. Inside of the **routes** folder, you will find three Javascript files (.js) pertaining to the initialization and usage of the Watson Assistant, Speech to Text and Text to Speech services. Without making any changes, feel free to click on each file to notice that these Javascript files rely on the passage of credentials. For example, the **conversation.js** file, which is the Javascript program that initializes the Watson Assistant service in our application, requires the passage of a username, password and workspace ID.



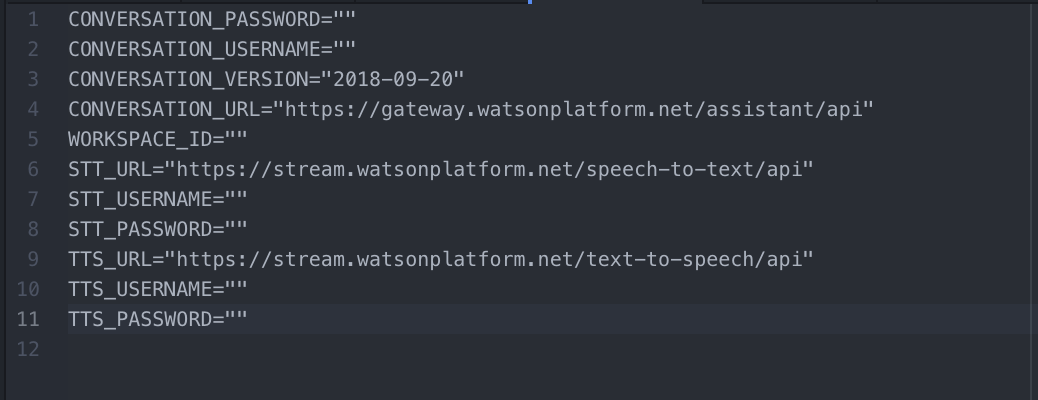
1. Inside of the **training** folder, we will need to add the workspace json file (corresponding to our Watson Assistant Skills instance) which we saved and downloaded in Lab 2. In fact, let’s do that right now.

* Locate the workspace json file that you downloaded to your Desktop at the end of Lab 2. If you forgot to download it, please open Watson Assistant and after clicking on **Launch tool**, navigate to the Skills page and find the SevereWeatherEvents skills instance. Download this as a workspace JSON file to your Desktop.
* While keeping your Atom window open, click and drag this file into the training folder of your project. If you did that correctly, you should now see the **skill-SevereWeatherEvents.json** file in the training folder.



1. Underneath the ui folder, you should be able to see the .env file. Upon clicking on it, you will notice that this environment file will contain the credentials for our Watson Assistant (formerly known as Conversation), Speech to Text and Text to Speech services as well as our workspace id. We will now enter the credentials for all the services that will be embedded in our application.

* Let’s start with the entities that will be the same for everyone. Enter **“2018-09-20”** for the **CONVERSATION\_VERSION** and **"https://gateway.watsonplatform.net/assistant/api"** for the **CONVERSATION\_URL**
* Enter **"https://stream.watsonplatform.net/speech-to-text/api"** for the **STT\_URL** and **"https://stream.watsonplatform.net/text-to-speech/api"** for the **TTS\_URL**.

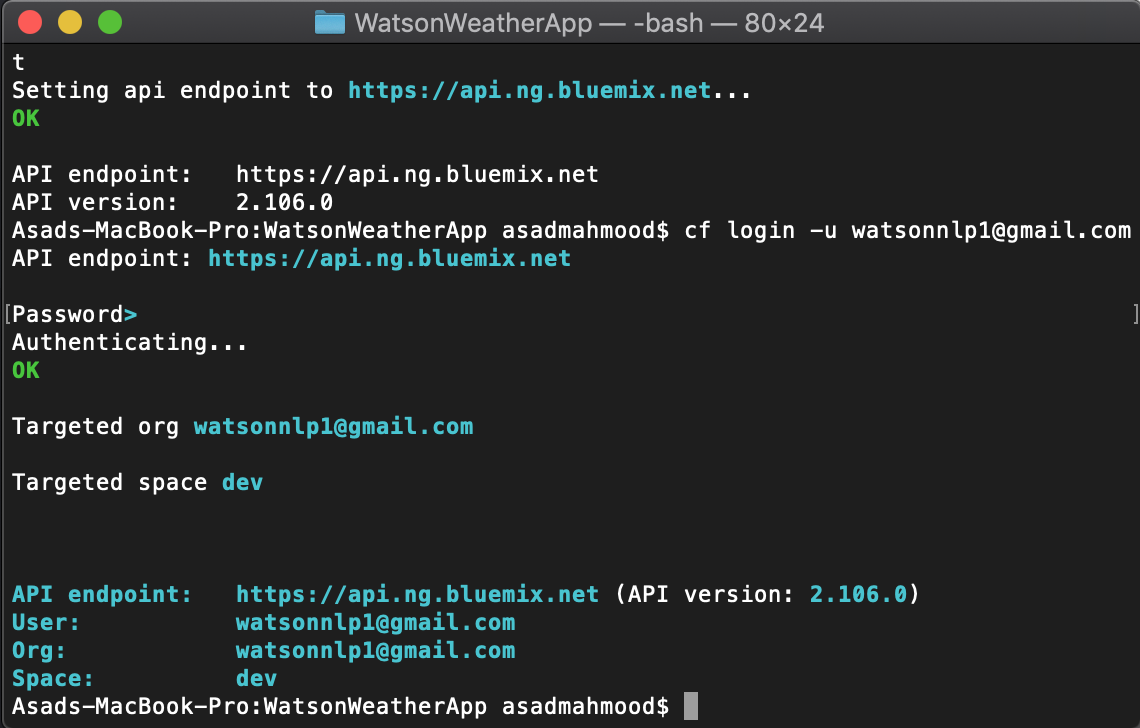


* You should already have the **WORKSPACE\_ID** saved from Lab 2 but just in case you misplaced it, it can be found within the Workspace JSON file that you just saved in the training folder. Click on the **skill-SevereWeatherEvents.json** file within the training folder to bring up the JSON contents on the right. Type **Ctrl/Command + F** and inside of the Find field, type in **workspace\_id**. Click **Find**. This should return the value of the workspace\_id variable, which you can copy and paste into the .env file for **WORKSPACE\_ID**.
* You should have already saved the Assistant username and password in Lab 2. These credentials can be retrieved at any time by opening Watson Assistant, navigating to the Skills home page and clicking on the vertical dots within the SevereWeatherEvents skill to select **View API Details**. Here, you can find the username and password underneath Service Credentials. Copy and paste these credentials for **CONVERSATION\_PASSWORD** and **CONVERSATION\_USERNAME**.
* Use the apikey credential that you copied in Step 3 of this lab to fill in the **STT\_USERNAME** and **STT\_PASSWORD**. The **STT\_USERNAME** will be **“apikey”** and the **STT\_PASSWORD** will be the value of the apikey variable that you copied in Step 3.
* Use the apikey credential that you copied in Step 4 of this lab to fill in the **TTS\_USERNAME** and **TTS\_PASSWORD**. The **TTS\_USERNAME** will be **“apikey”** and the **TTS\_PASSWORD** will be the value of the apikey variable that you copied in Step 4.
* Save your .env file by either selecting File->Save within the Atom editor or pressing CTRL/Command + S on your keyboard.

**Step 7: Run and Deploy the App to IBM Cloud**

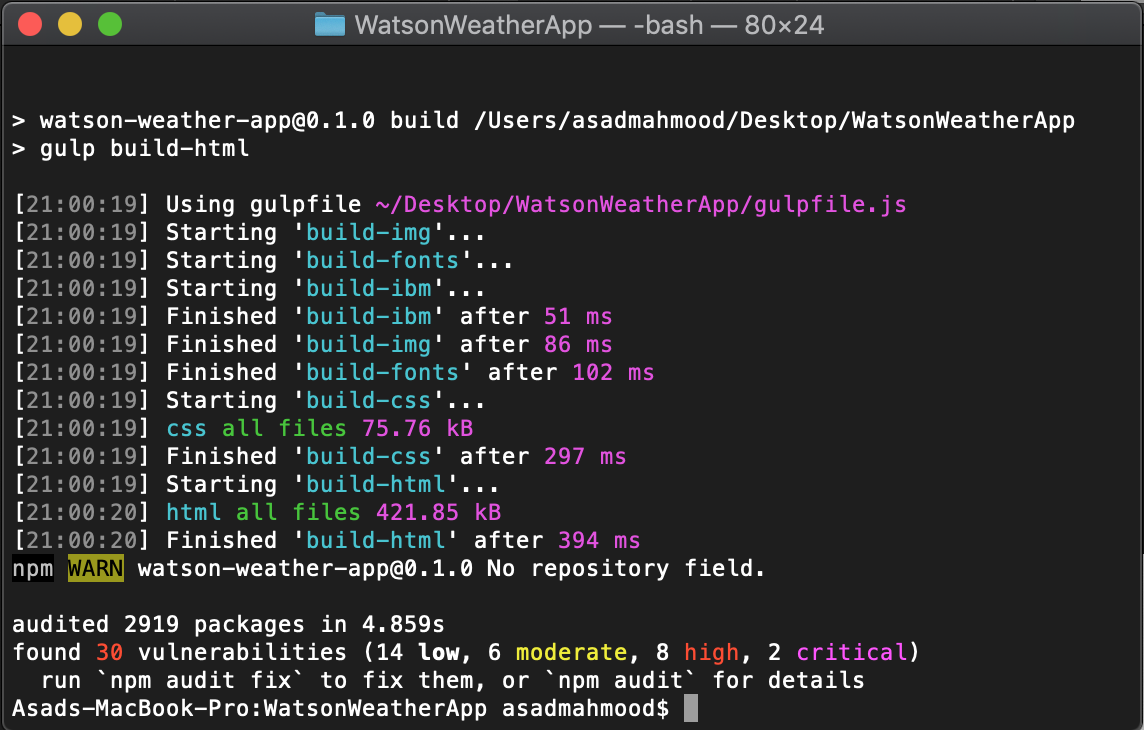
Now that we have added our Workspace JSON file and have edited the .env file with the credentials for all of the Watson services that will be employed, the Watson weather application is ready to be run and deployed to the IBM Cloud. We will be using the Cloud Foundry CLI which we downloaded and installed in Step 1 to deploy our app. Before running commands from the command line, please ensure that your WatsonWeatherApp folder is saved in the Desktop folder.

1. Navigate to the command prompt on your computer. On a Mac computer, it can be immediately accessed by typing Terminal into Finder. On a Windows computer, you can quickly get to the command prompt by pressing the **Win and R keys** on your keyboard and then typing **cmd** and pressing **Enter**.
2. Inside of the command prompt window, navigate to the WatsonWeatherApp folder by typing: **cd Desktop/WatsonWeatherApp**
3. We are now going to log into the IBM Cloud using our log-in information. Type **cf api https://api.ng.bluemix.net**
4. Type **cf login -u <your user ID>** (where you can substitute your username for <your user ID>)
5. Enter the password for your IBM Cloud account

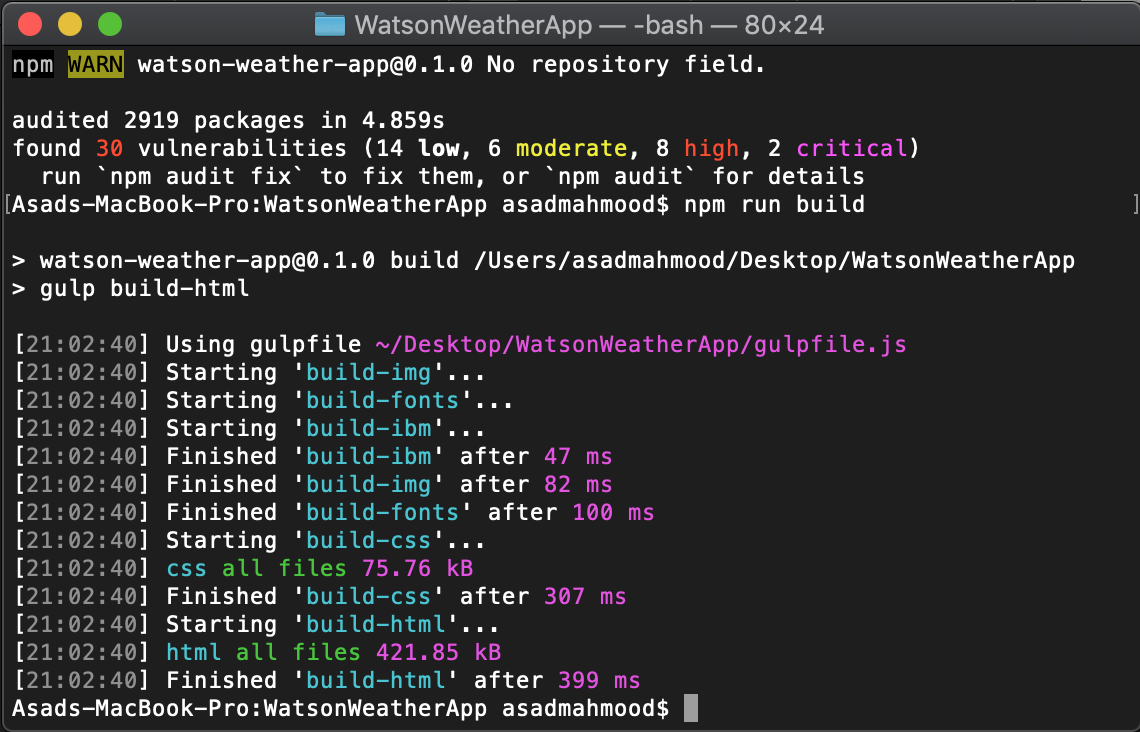


If there are any issues with the cf login command, please use the following alternative: **cf login -u <your user ID> -p <your password>**

1. Once you are authenticated, you can run and build the UI of your application. We will need to make sure everything compiles correctly before deploying the app to the IBM Cloud. Type **npm install**



1. Now that our application has compiled without any errors (ignore any warnings and vulnerabilities you might see), we can run and build the UI. Type **npm run build**

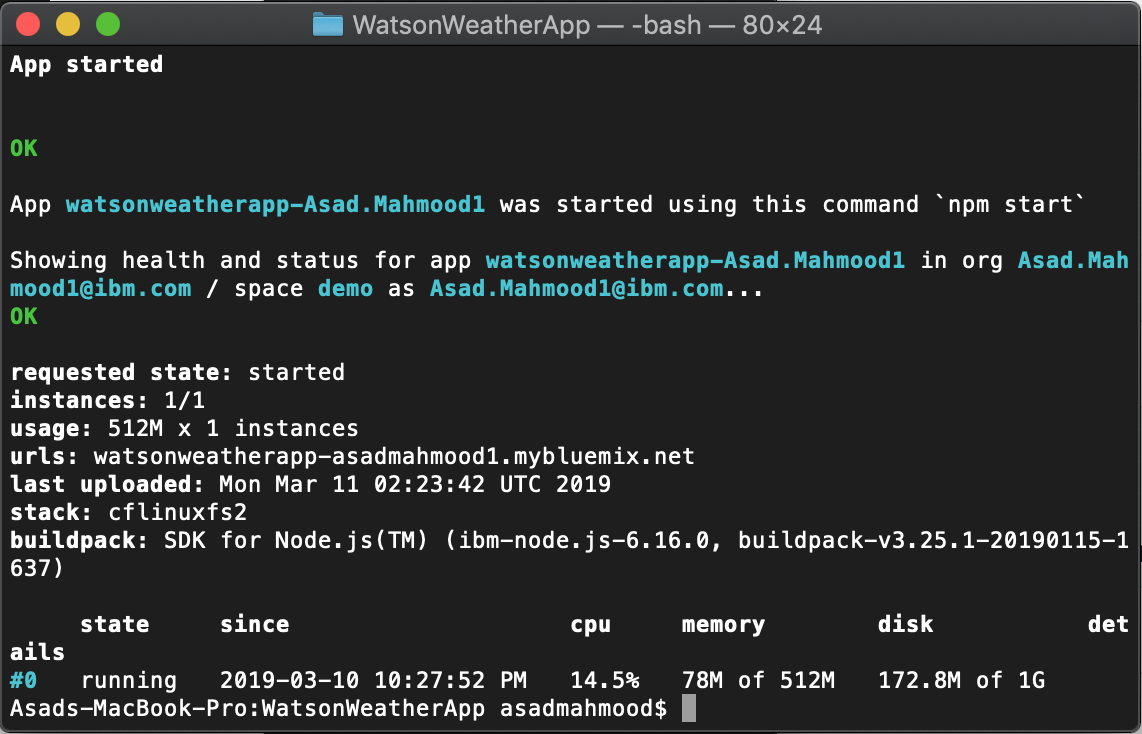


1. Finally let’s deploy our app to the IBM Cloud. All web-based IBM Cloud apps have a URL of the following form: https://<applicationName>.mybluemix.net. Since everyone completing this lab will be compiling the same application codebase to the cloud, you will have to provide your app with a unique name.

Create a unique application name using the following criteria: **applicationName = watsonweatherapp-(IBM Cloud username)**.

Thus if my IBM Cloud username was johnsmith, the application name will be watsonweatherapp-johnsmith and its IBM Cloud URL will be https://watsonweatherapp-johnsmith.mybluemix.net

1. Type the following into the command prompt in order to deploy your app to the IBM Cloud: **cf push <applicationName>**



1. After approximately 5-7 minutes, your app should be successfully deployed to the IBM Cloud. You can now access it live at https://<applicationName>.mybluemix.net. The URL for your app is also available on the command prompt next to **urls.**
2. You can now play around with your application in order to ensure that the Watson Weather Assistant can answer all the questions it has been trained on by typing in your questions, selecting the questions from **What can you ask me?** or speaking directly to Watson after turning on the mic at the bottom of the page.

