# **Prerequisites**

* [**Visual Studio Code**](https://www.visualstudio.com/downloads)
  + **Azure App Service extension (instructions below)**
* [**Node.js**](https://nodejs.org/)
* [**Bot Framework Emulator**](https://aka.ms/bot-framework-emulator-readme)
* **Azure Subscription**
* [**git**](https://git-scm.com/) **(optional)**

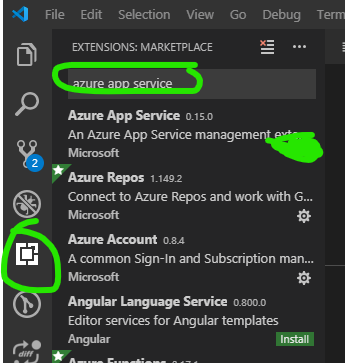
# **Download Workshop code**

* **Check out or download code from the following repository:**
  + [**https://github.com/jakeatmsft/voya\_hack\_bot**](https://github.com/jakeatmsft/voya_hack_bot)

# **Install Azure App Service Extension**

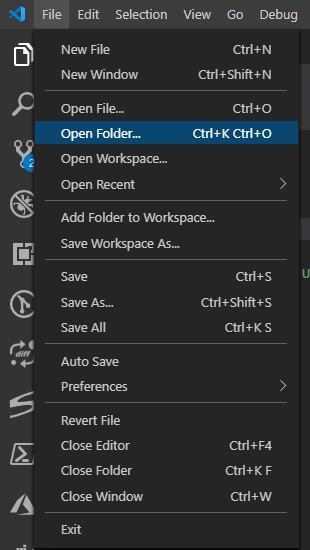
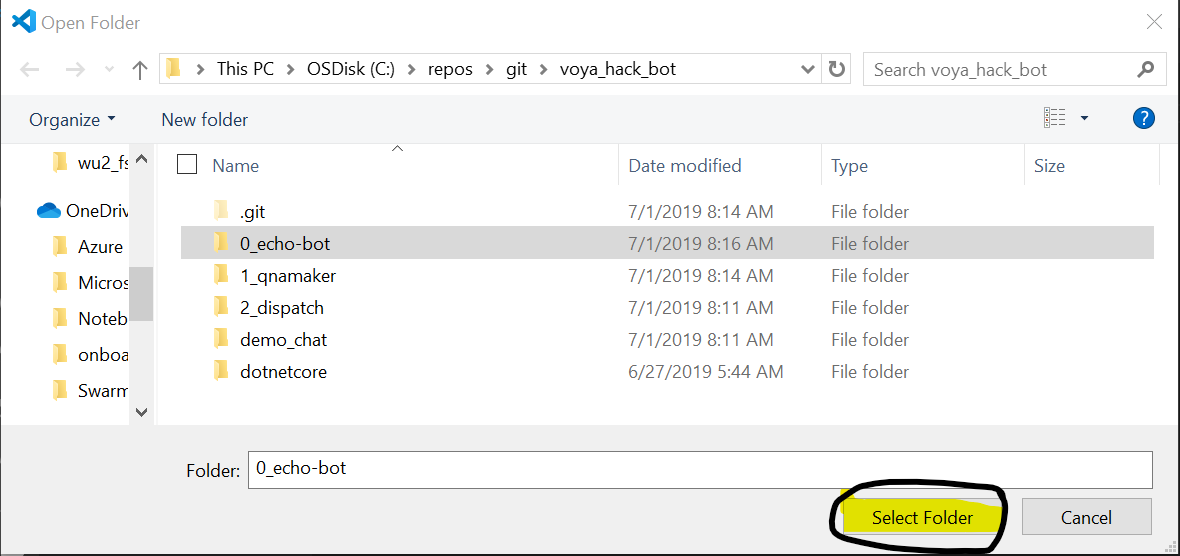
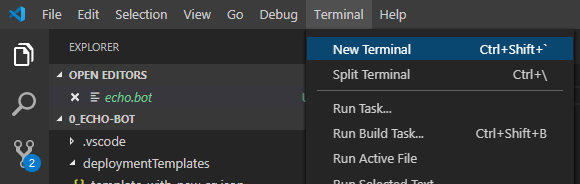
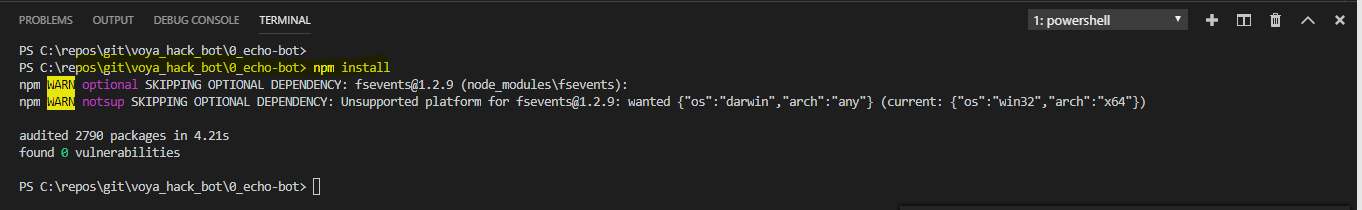
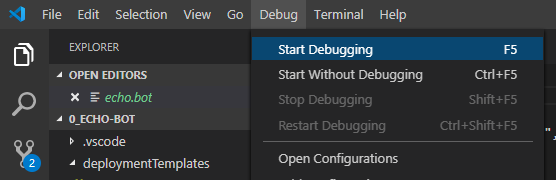
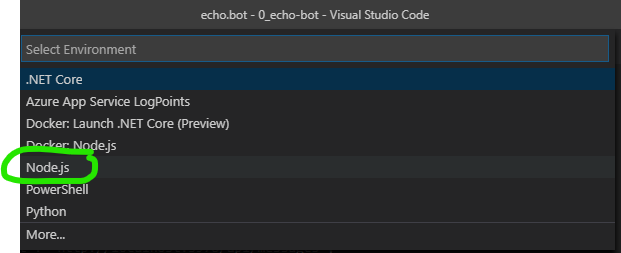
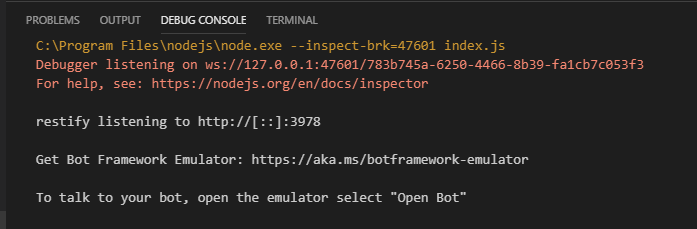
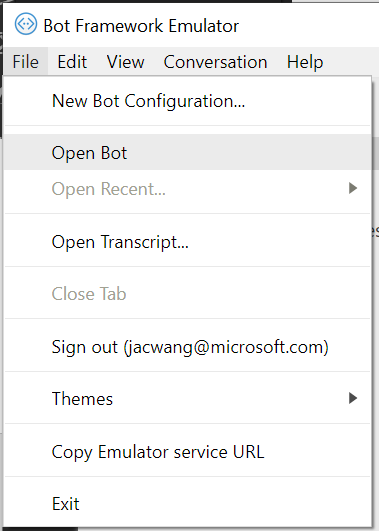
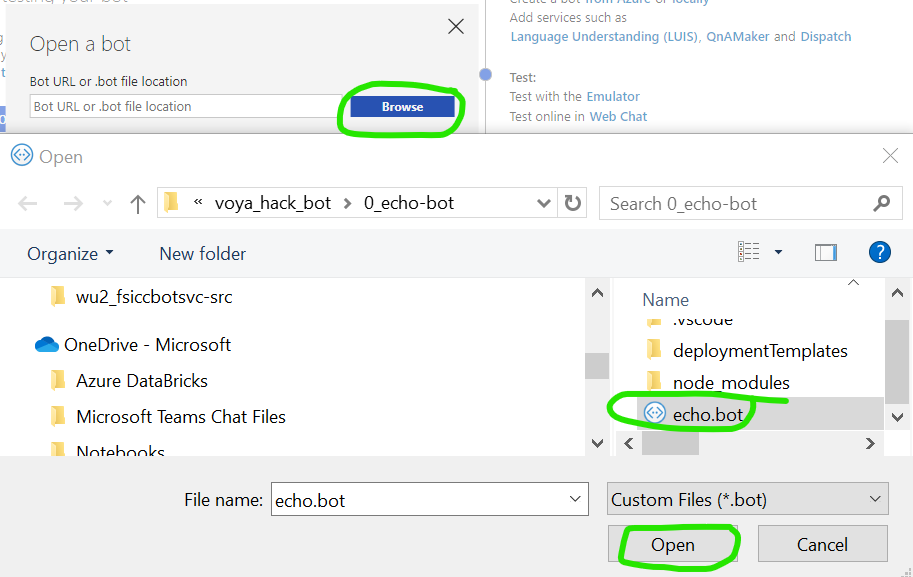
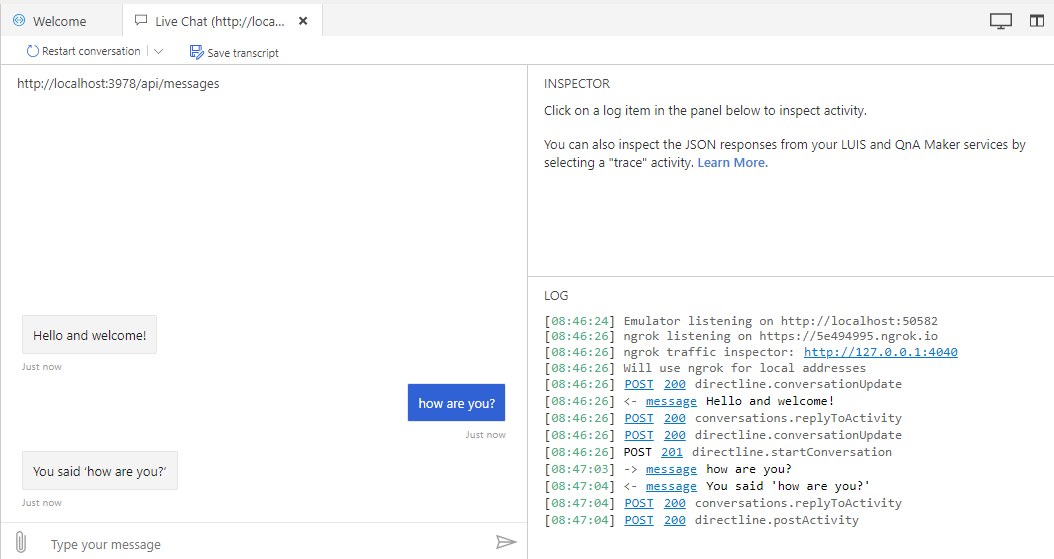
**Open VSCode**

**Go to the Extension menu and search form “Azure App Service”, click install in the lower right hand corner to install. If no button appears, the extension is already installed.**



# **Lab 0 – Echo Bot**

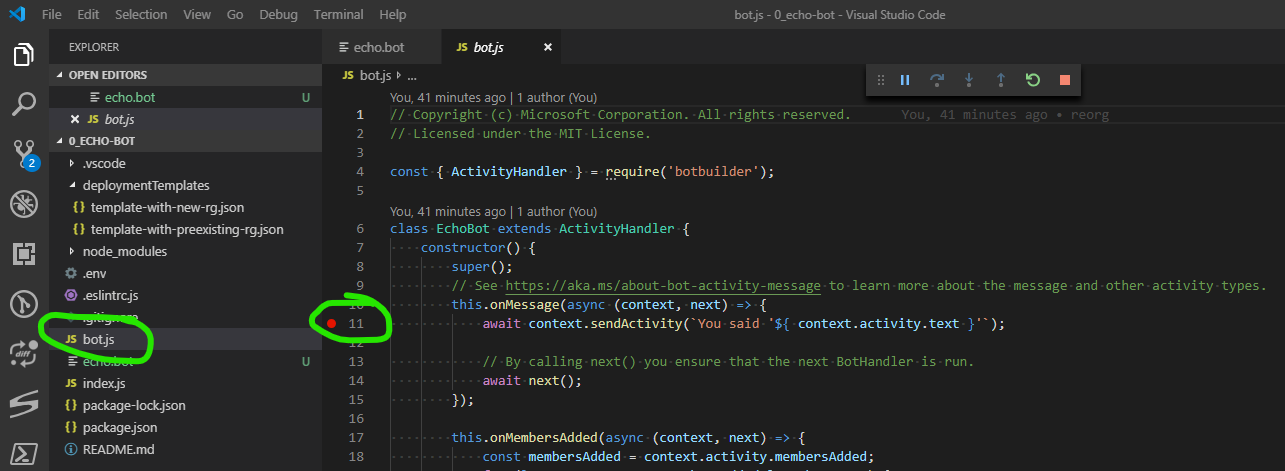
This first example shows a simple bot that will repeat back messages that are sent to it. Let begin:

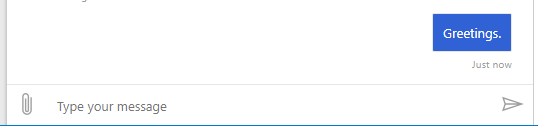
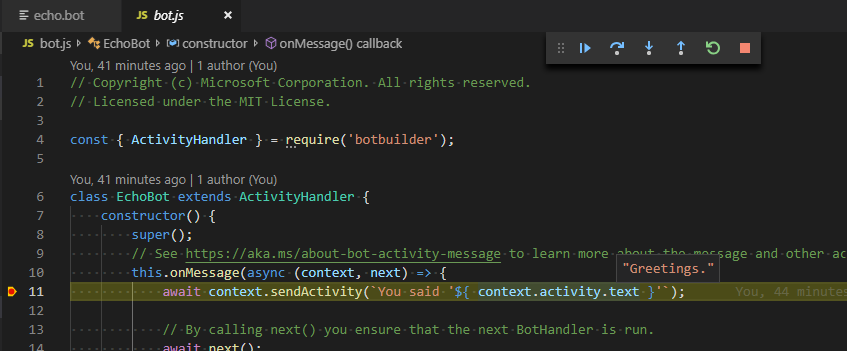
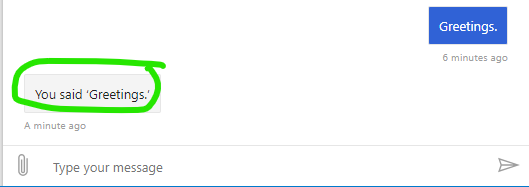
* **Open VSCode.**
* **Open the “echo-bot” folder:**
* ****
* ****
* **Navigate to a new Terminal window**
* ****
* **Run command: npm install**
* ****
* **Now go to Debug menu to Start Debugging**
* ****
* **Select “node.js” environment.**
* ****
* **Verify bot is running in “Debug Console”**
* ****
* **Next open Bot Emulator to connect to bot.**
* **Open the “echo.bot” file to connect to running bot.**
* ****
* ****
* **“Connect”**
* **You should now be connected to your bot and be able to type messages in the chat window.**
* ****

# **Setting Breakpoints**

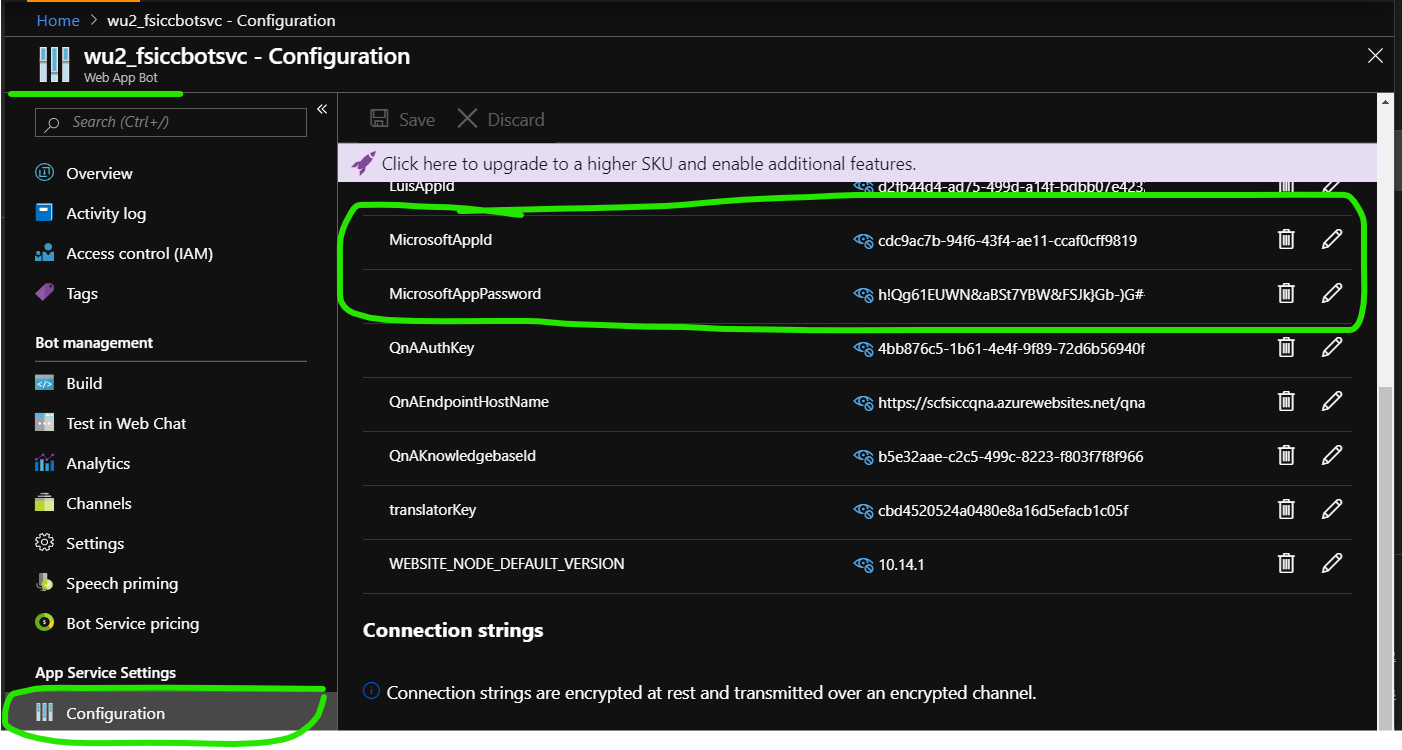
Now that you are connected to the bot service and are running it locally, we can set breakpoints in the bot and see the code behind the interactions.

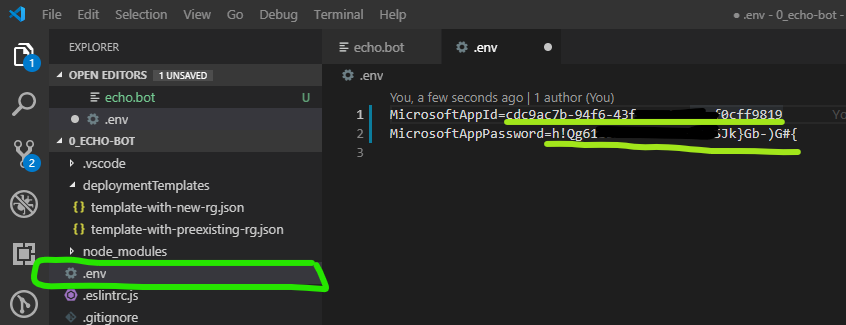
* **Set a breakpoint in the bot.js file at line: 11,** you can set the breakpoint by clicking in the margin to the left of the line number. You will see a red circle appear where the breakpoint is set.

****

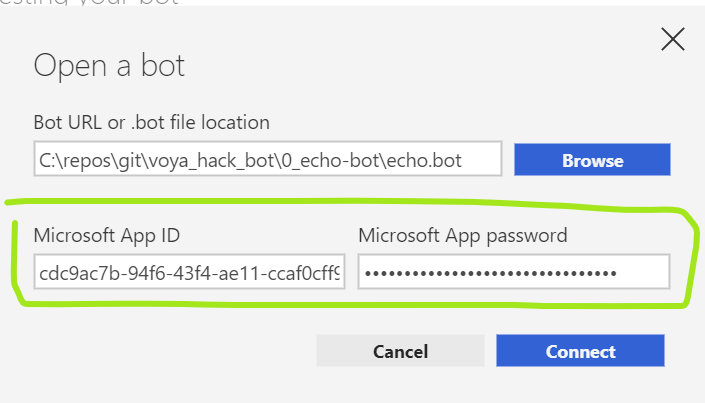
* **Now go back to the Bot Emulator and type a message. You should see the line highlighted once the bot executes that line of code.**
* ****
* **Here you can view the current context by hovering over the code.**
* ****
* **To release control of the breakpoint you can click the “Continue” button in the debug controls.**
* ****
* **Alternatively, you can “Step Into” or “Step Over” lines of code to see the function execution.**
* **The bot will now continue execution and reply within the emulator:**
* ****

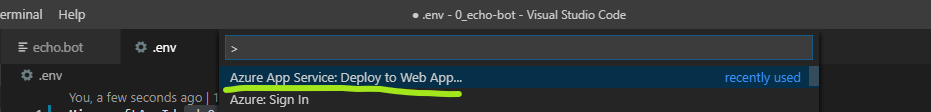
# **Bot Deployment**

* **To deploy the bot, begin by creating a bot service in the Azure Portal.**
* <https://docs.microsoft.com/en-us/azure/bot-service/bot-service-quickstart?view=azure-bot-service-4.0>
* **Retrieve, the MicrosoftAppID and MicrosoftAppPassword from the “Configuration” blade:**
* ****
* **Paste the “MicrosoftAppID” and “MicrosoftAppPassword” into the .env file in your bot project:**

****

**(Note: Once you have the “AppID and Password” defined in your Bot solution if you want to connect through the emulator you will need to reconnect with the AppID and Password in the “Open” prompt.**

**)**

* **Once the “.env” file is updated you can deploy using the App Service Extension, by opening the command menu (Control – Shift – P) and using the “Azure App Service: Deploy to Web App” command.**
* ****
* **Select the current folder.**
* **Select the target bot App Service (created in step 1).**
* **Verify the deployment.**