



# Lab 1: Debugging a Raspberry Pi Internet of Things Flask Application

Author	<a href="#">Dave Glover</a> , Microsoft Cloud Developer Advocate
Platforms	Linux, macOS, Windows, Raspbian Buster
Tools	<a href="#">Visual Studio Code</a>
Hardware	<a href="#">Raspberry Pi 4. 4GB</a> model required for 20 Users. Raspberry Pi <a href="#">Sense HAT</a> , Optional: Raspberry Pi <a href="#">case</a> , <a href="#">active cooling fan</a>
<b>USB3 SSD Drive</b>	To support up to 20 users per Raspberry Pi you need a <b>fast</b> USB3 SSD Drive to run Raspbian Buster Linux on. A 120 USB3 SSD drive will be more than sufficient. These are readily available from online stores.
Language	Python
Date	As of September, 2019

Follow me on Twitter [@dglover](#)

## PDF Lab Guide

You may find it easier to download and follow the PDF version of the [Debugging Raspberry Pi Internet of Things Flask App](#) hands-on lab guide.

## Introduction

In this hands-on lab, you will learn how to create and debug a Python web application on a Raspberry Pi with [Visual Studio Code](#) and the [Remote SSH](#) extension. The web app will read the temperature, humidity, and air pressure telemetry from a sensor connected to the Raspberry Pi.



## Software Installation



This hands-on lab uses Visual Studio Code. Visual Studio Code is a code editor and is one of the most popular **Open Source** projects on [GitHub](https://github.com). It runs on Linux, macOS, and Windows.

## Install Visual Studio Code

1. Install [Visual Studio Code](#)

## Visual Studio Code Extensions

The features that Visual Studio Code includes out-of-the-box are just the start. VS Code

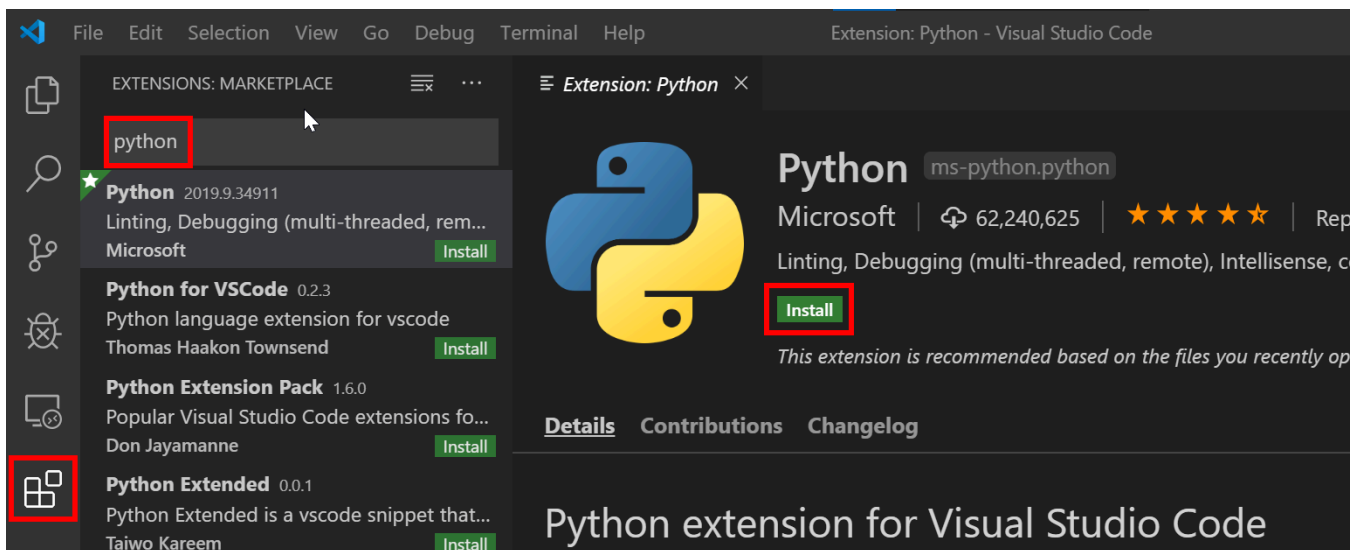
extensions let you add languages, debuggers, and tools to your installation to support your development workflow.

## Browse for extensions

You can search and install extensions from within Visual Studio Code. Open the Extensions view from the Visual Studio Code main menu, select **View > Extensions** or by clicking on the Extensions icon in the **Activity Bar** on the side of Visual Studio Code.



This will show you a list of the most popular VS Code extensions on the [VS Code Marketplace](#).



## Install the Python and Remote SSH Extensions

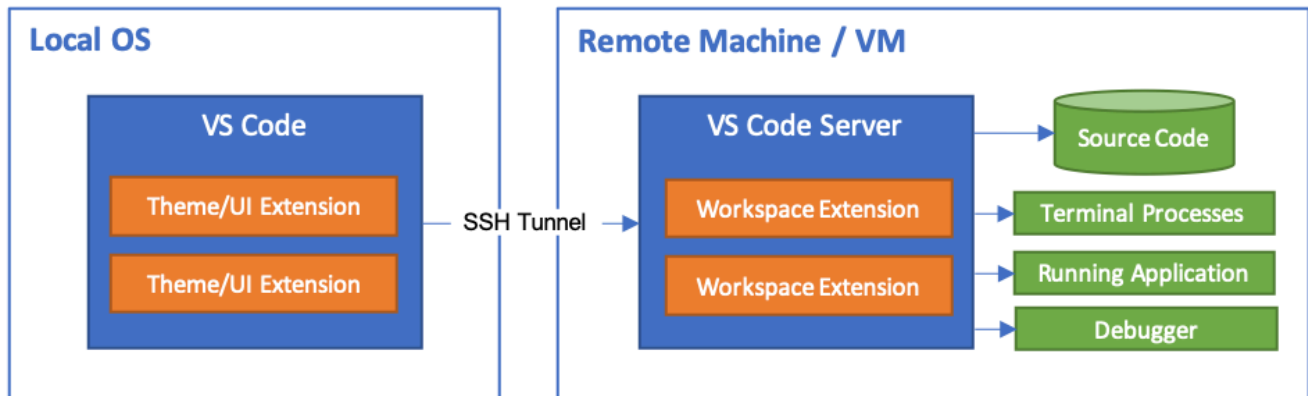
Search and install the following two Visual Studio Code Extensions published by Microsoft.

1. [Python](#)
2. [Remote - SSH](#)

## Remote SSH Development

The Visual Studio Code Remote - SSH extension allows you to open a remote folder on any

remote machine, virtual machine, or container with a running SSH server and take full advantage of Visual Studio Code.



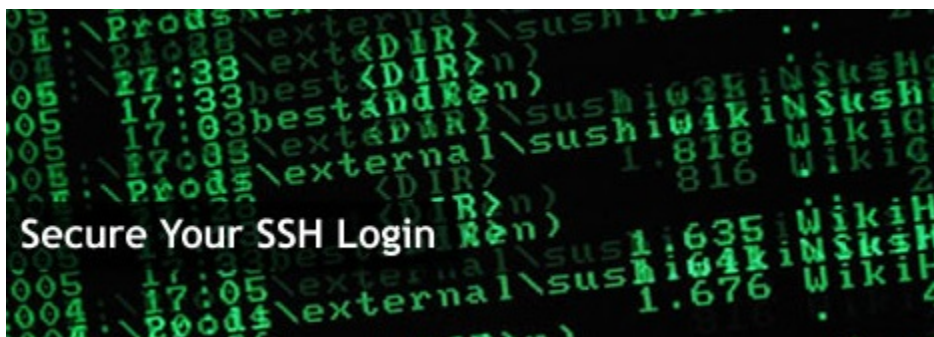
## Raspberry Pi Hardware

If you are attending a workshop, then you can use a shared network-connected Raspberry Pi. You can also use your own network-connected Raspberry Pi for this hands-on lab.

You will need the following information from the lab instructor.

1. The **Network IP Address** of the Raspberry Pi
2. Your assigned **login name** and **password**.

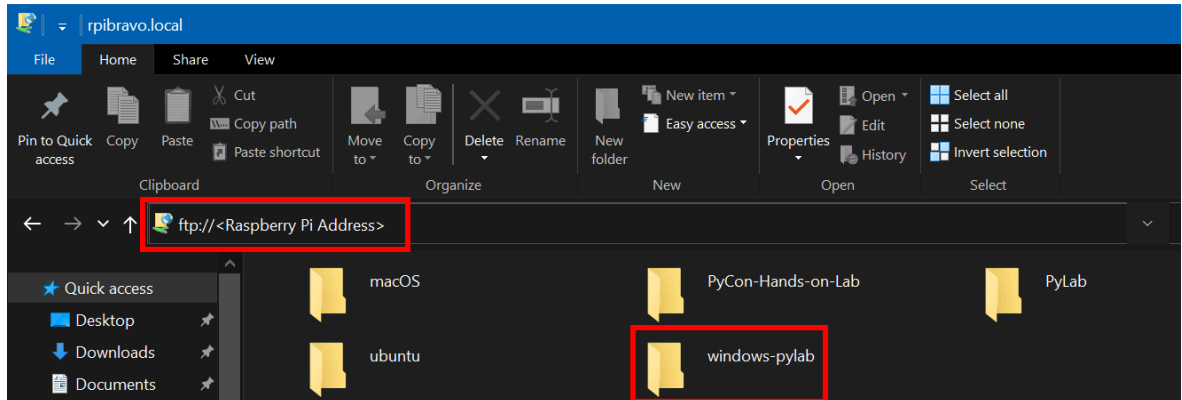
## SSH Authentication with private/public keys



Setting up a public/private key pair for [SSH](#) authentication is a secure and fast way to authenticate from your computer to the Raspberry Pi. This is recommended for this hands-on lab.

# SSH Set up for Windows Users

1. From Windows File Explorer, open **ftp://<Raspberry Pi Address>**



2. Copy the **windows-pylab** directory to your **desktop**
3. Open the **windows-pylab** folder you copied to your **desktop**
4. Double click **windows-ssh.cmd**

You will be guided through the process of setting up an SSH key and copying the SSH public key to the Raspberry Pi.

## SSH for Linux and macOS Users

From a Linux or macOS **Terminal Console** or from **git bash** in windows run the following commands:

1. Create your key. This is typically a one-time operation. **Take the default options.**

```
ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa_python_lab
```

2. Copy the public key to the Raspberry Pi.

```
ssh-copy-id -i ~/.ssh/id_rsa_python_lab <login@Raspberry IP Address>
```

For example:

```
ssh-copy-id -i ~/.ssh/id_rsa_python_lab dev99@192.168.1.200
```

3. Test the SSH Authentication Key

```
ssh -i ~/.ssh/id_rsa_python_lab <login@Raspberry IP Address>
```

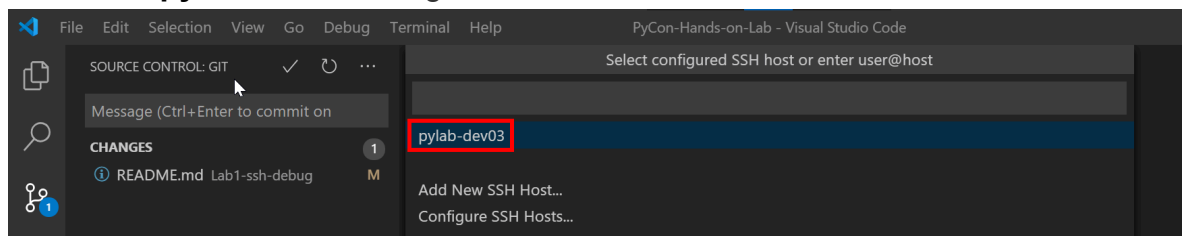
A new SSH session will start. You should now be connected to the Raspberry Pi

**without** being prompted for the password.

4. Close the SSH session. In the SSH terminal, type `exit`, followed by `ENTER`.

## Start a Visual Studio Code Remote SSH Connection

1. **Start Visual Studio Code**
2. Press **F1** to open the Command Palette, type **ssh connect** and select **Remote-SSH: Connect to Host**
3. Select the **pylab-devnn** configuration



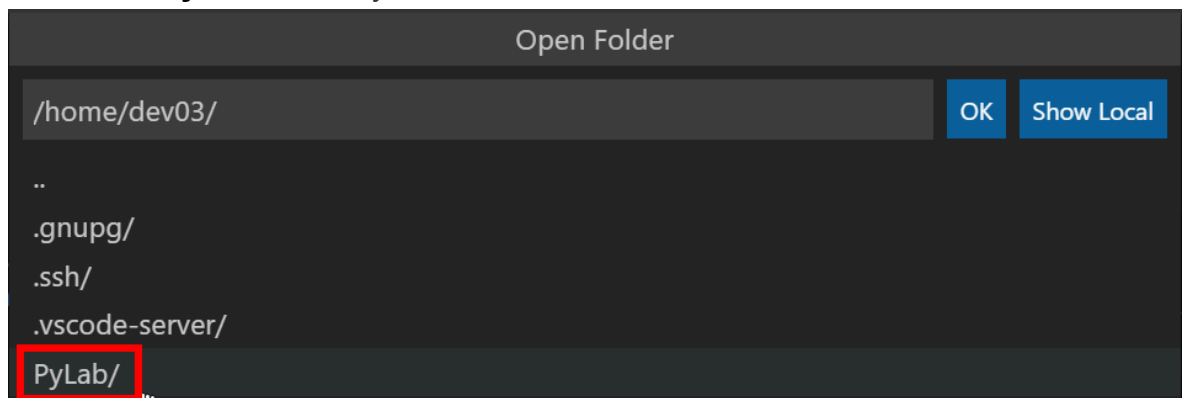
It will take a moment to connect to the Raspberry Pi.

## Open Lab 1 SSH Debug Project

### Python Flask Web Apps

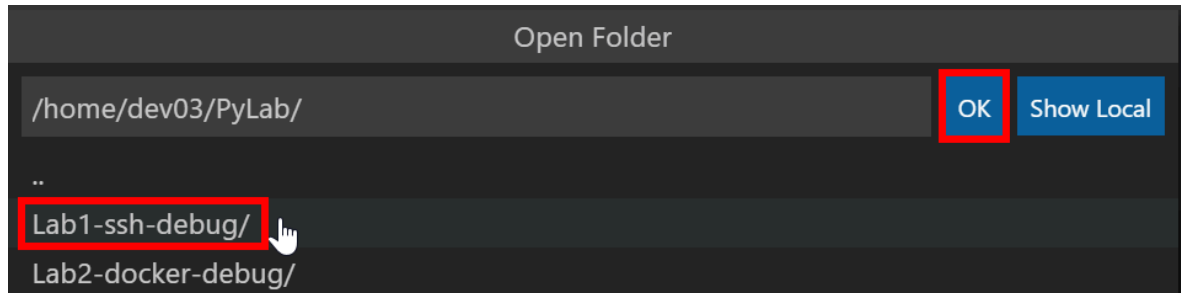
In this lab we are going to start and debug a [Flask](#) app that reads a sensor attached to the Raspberry Pi. Flask is a popular Python Web Framework, powerful, but also easy for beginners.

1. From Visual Studio Code main menu: **File > Open Folder**
2. Select the **PyLab** directory

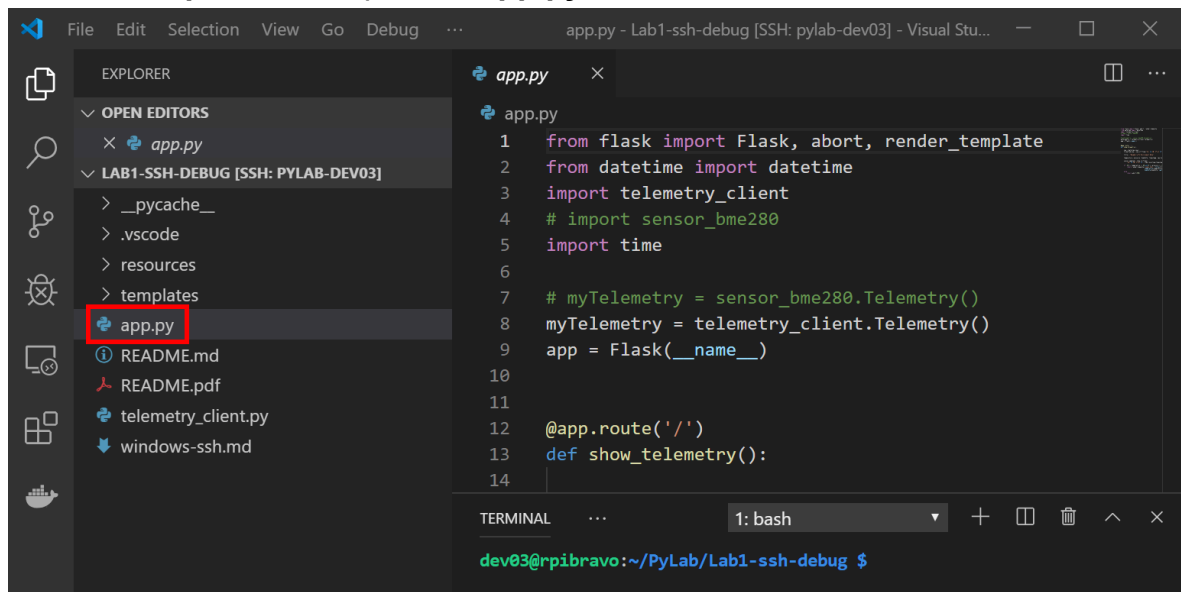




- Next select the **Lab1-ssh-debug** directory

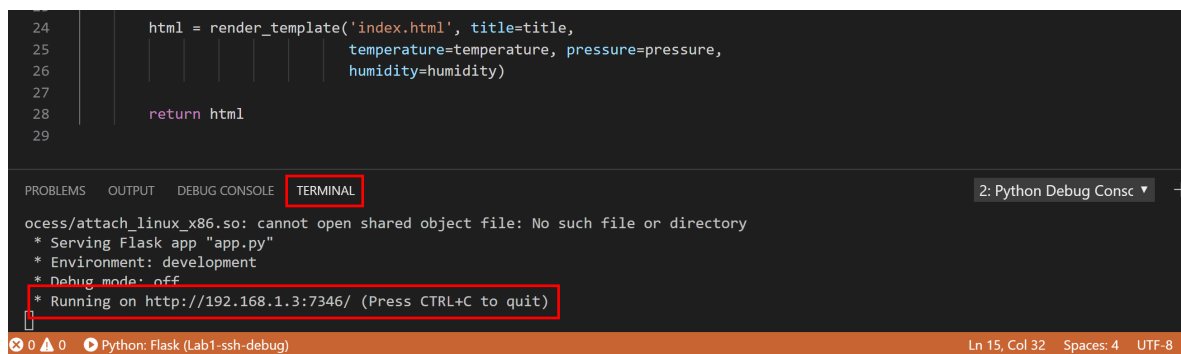


- Click **OK** to Open the directory
- From the **Explorer** bar, open the **app.py** file and review the contents



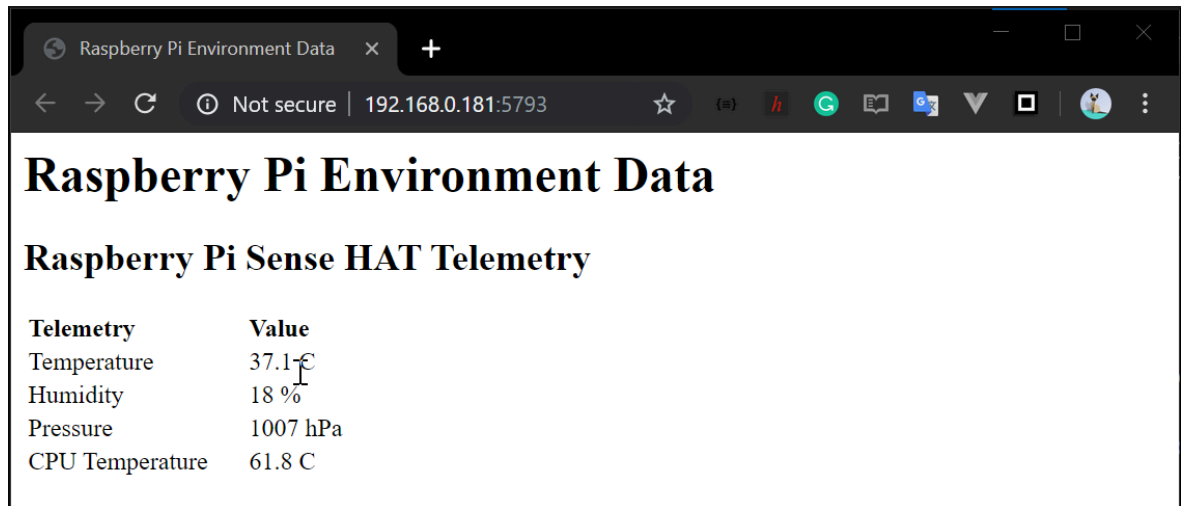
## Start the Python Flask App

- Press **F5** to start the Python Flask app.
- From the Visual Studio Code **Terminal Window**, click the **running on http://...** web link.



- This will launch your desktop Web Browser.
  - The Flask app will read the temperature, air pressure, humidity from

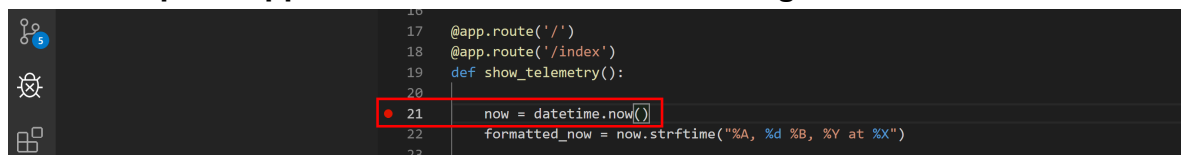
the **sensor** attached the Raspberry Pi and display the results in your web browser.



## Debugging with Breakpoints

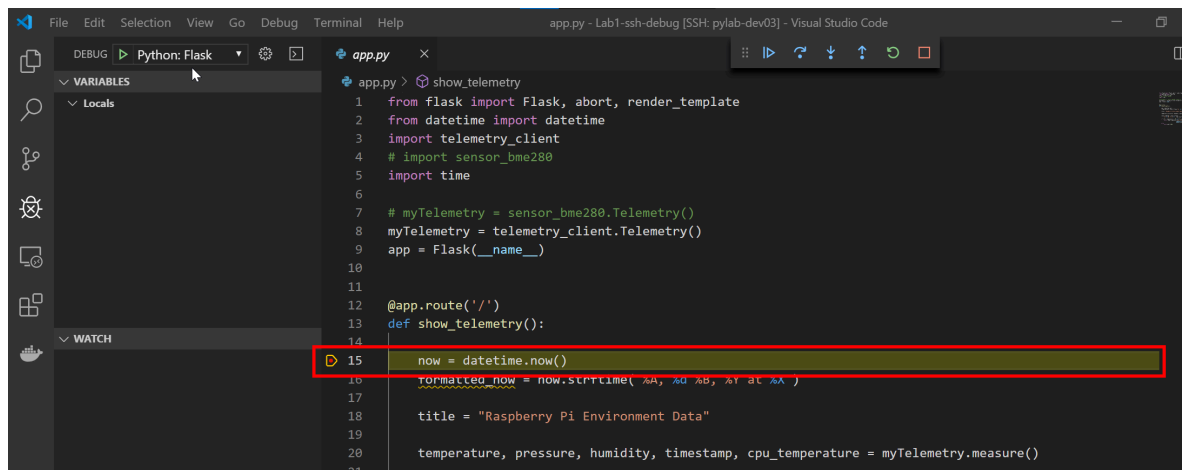
1. Switch back to Visual Studio Code and ensure the **app.py** file is open.
2. Put the cursor on the line that reads **now = datetime.now()**
3. Use one of the following methods to set a breakpoint.
  - Press **F9**
  - From the main menu, select **Debug > Toggle Breakpoint**
  - **Click directly** in the **margin to the left** of the line number (a faded red dot appears when hovering there)

The breakpoint appears as a red dot in the left margin



4. Switch back to the **Web Browser** and click **Refresh**. The web page will **not response** as the Python code has been stopped by the debugger.
5. Switch back to **Visual Studio Code**. You will see that the code has stopped running at the **breakpoint**.





## Debugger Toolbar Options

When a debug session starts, the **Debug toolbar** will appear at the top of the editor window.

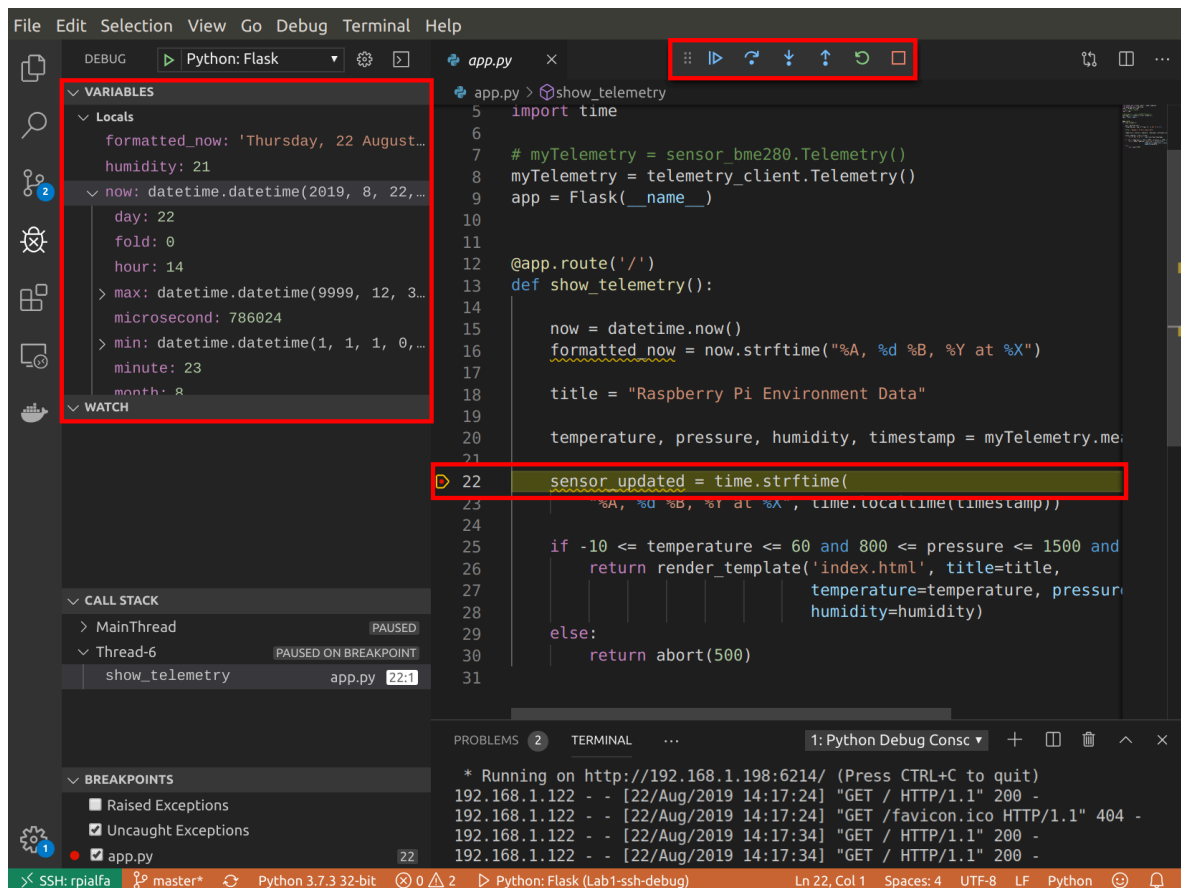
The debugging toolbar (shown below) will appear in Visual Studio Code. It has the following options:



1. Pause (or Continue, F5),
2. Step Over (F10)
3. Step Into (F11),
4. Step Out (Shift+F11),
5. Restart (Ctrl+Shift+F5),
6. and Stop (Shift+F5).

## Start Debugging

1. Step through the code by pressing (**F10**) or clicking **Step Over** on the debugging toolbar.
2. **Repeat** pressing **F10** until you reach the line that reads **if -40 <= temperature <= 60 and 0 <= pressure <= 1500 and 0 <= humidity <= 100:**
3. You will notice that Python variables are displayed in the **Variables Window**.

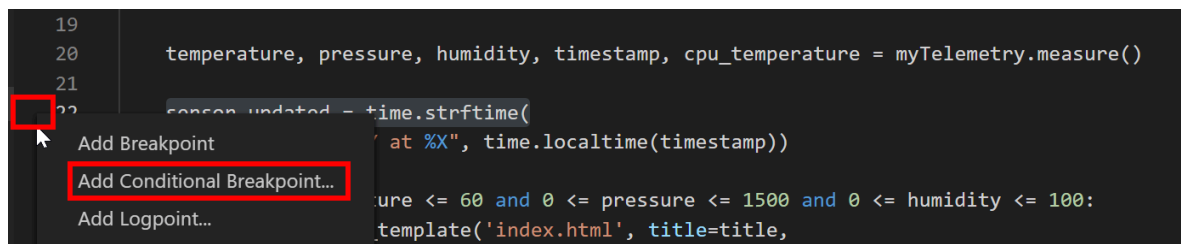


4. Try to change the **temperature** variable to **50**. Hint, **right mouse** click on the temperature variable and select **Set Value**, or double click on a **temperature** variable.
5. Press **F5** to resume the Flask App, then **switch back to your web browser** and you will see the temperature, humidity, and pressure Sensor data displayed on the web page.

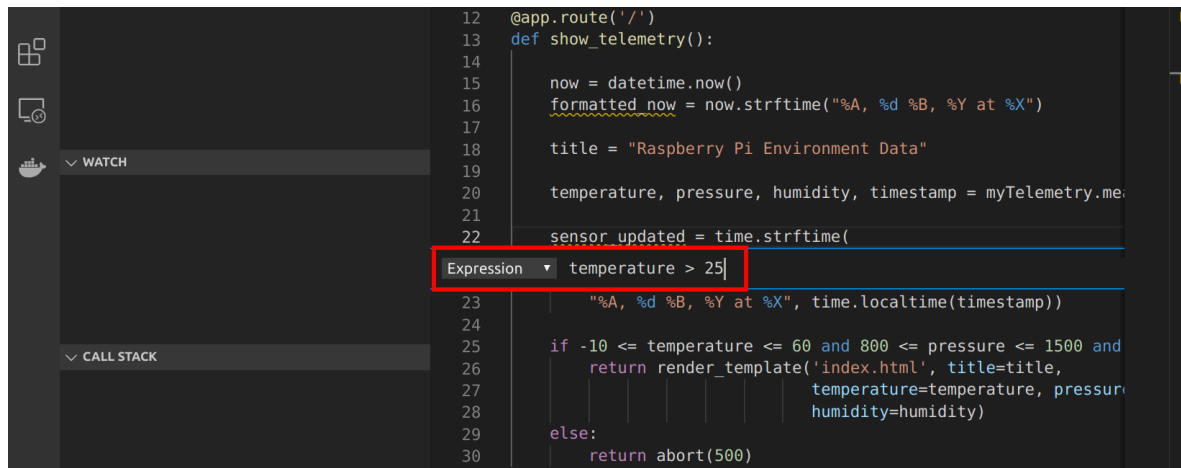
## Debugging with Conditional Breakpoints

Try setting a **conditional** breakpoint

1. Clear the existing breakpoints. From the main menu select **Debug > Remove all breakpoints**.
2. Ensure the **app.py** file open.
3. **Right mouse click** directly in the margin to the **left** of the line number **22**.



4. Select **Add Conditional Breakpoint...**
5. Set the condition to **temperature > 25**, then press **ENTER**



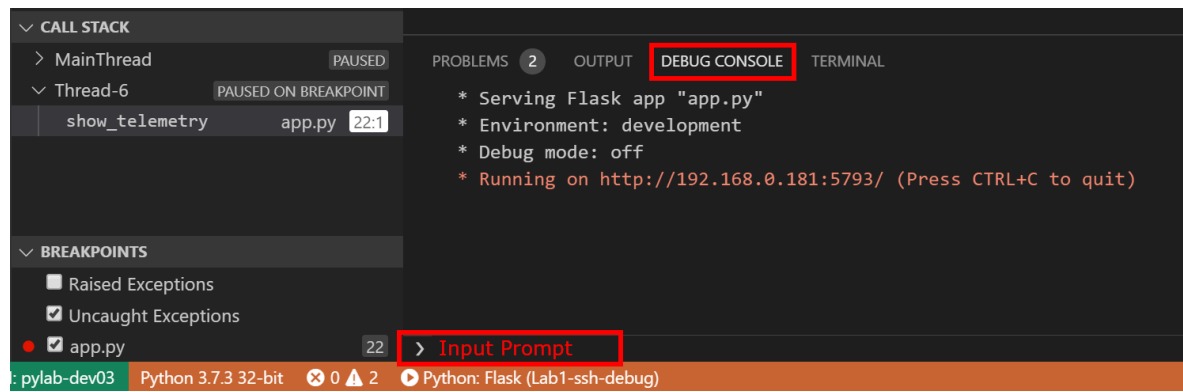
The breakpoint appears as a red dot with an equals sign in the middle:

6. **Switch** back to your **web browsers** and **click refresh**. The web page will **not response** as the Python code has been stopped by the debugger.
7. **Switch** back to **Visual Studio Code** and you will see the debugger has stopped at the **conditional breakpoint**.
8. Press **F5** to continue running the code
9. **Switch** back to your **web browser** to view the page.

## Interactive Debug Console

The Visual Studio Code **Debug Console** will give you access to the [Python REPL](#) (Read, Evaluate, Print Loop).

1. **Switch** back to your **web browser** and click refresh. The web page will **not response** as the Python code has been stopped by the debugger.
2. **Switch** back to **Visual Studio Code**
3. The code will have stopped at the conditional breakpoint you previously set.
4. Select the Visual Studio **Debug Console** window.



5. Type the following Python code into the Input Prompt >

```
print(temperature)
```

6. Press **Enter** to execute the Python code you typed.
7. Try running the following Python code snippets from the input prompt.

```
temperature = 24
import random
random.randrange(100, 1000)
```

8. Press **F5** to continue the execution of the Python code.
9. Switch back to your web browser to see the updated page.

## Lab Challenges

### Lab Challenge 1: Update the Flask Template

1. Update the Flask **index.html** template found in the **templates** folder to display the current date and time.
2. Rerun the Flask app.

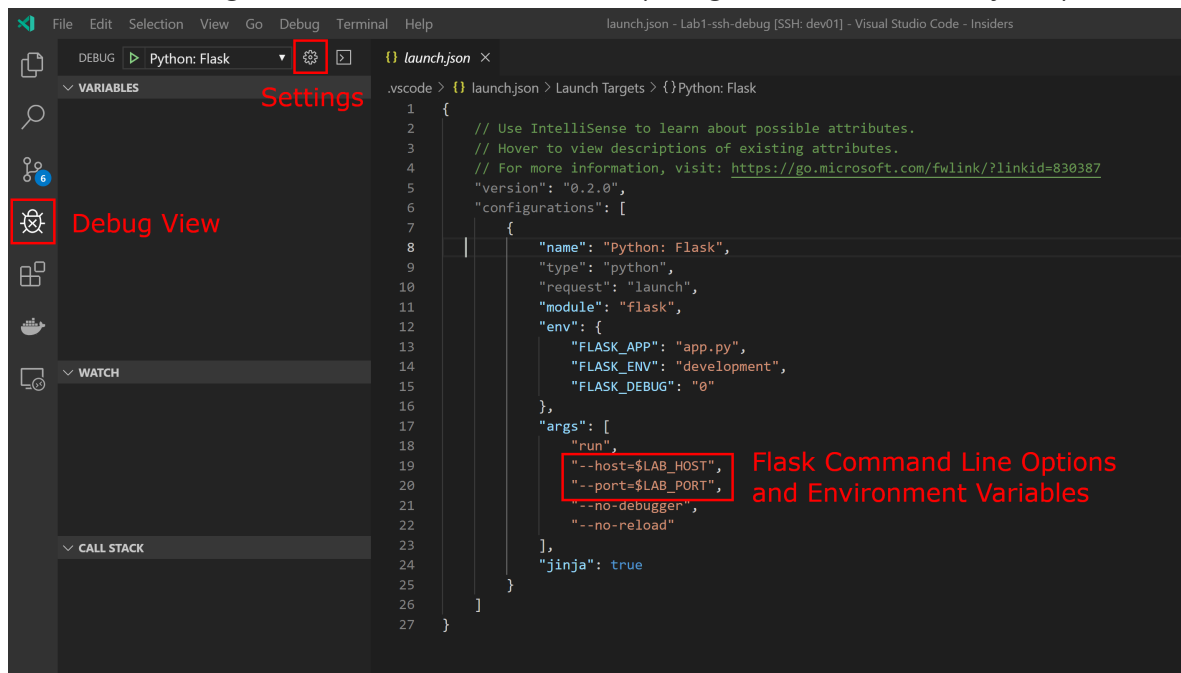
### Lab Challenge 2: Experiment with Debugger Options

Things to try:

1. Review the [Visual Studio Code Python Tutorial](#)
2. Review the [Python Flask tutorial](#)
3. Review the [Visual Studio Code Debugging Tutorial](#)

# Review the debug options

1. Switch to Debug view in Visual Studio Code (using the left-side activity bar).



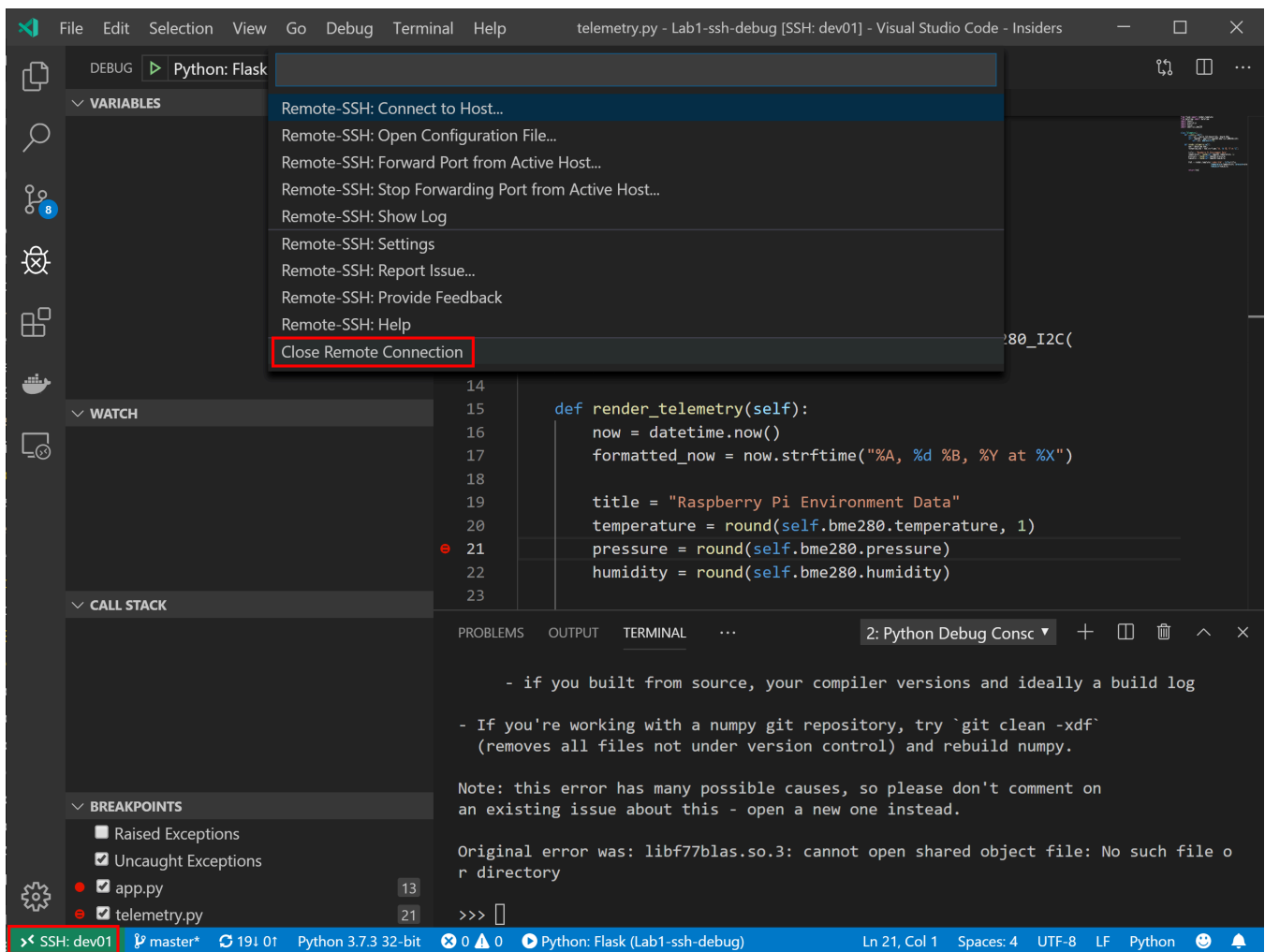
2. Click the **Settings** button which will open the **launch.json** file.
3. The **launch.json** file defines how the Flask app will start, and what **Flask Command Line** parameters to pass at startup.

There are two environment variables used in the launch.json file. These are **LAB\_HOST** (which is the IP Address of the Raspberry Pi), and **LAB\_PORT** (a random TCP/IP Port number between 5000 and 8000). These environment variables are set by the .bashrc script which runs when you connect to the Raspberry Pi with Visual Studio Remote SSH.

## Closing the Remote SSH Session

From Visual Studio Code, **Close Remote Connection**.

1. Click the **Remote SSH** button in the **bottom left-hand corner** and select **Close Remote Connection** from the dropdown list.



# Finished

finished

# References

- [Visual Studio Code](#)
- [Python](#)
- [Raspberry Pi](#)
- [Flask](#)

# Trouble Shooting SSH Client Installation

- [Remote Development using SSH](#)
- [Installing a supported SSH client](#)