LAMINA STUDIOS, LLC
DATA ANALYTICS INTERNSHIP

H2O Wave: System Monitor Tutorial Activity Documentation

Jewel Anne A. Reyes
BS Computer Science
Polytechnic University of the Philippines

September 1, 2023

Table of Contents

Task Overview

H2O Wave is a software stack for building beautiful, low-latency, realtime, browser-based applications and dashboards entirely in Python without using HTML, Javascript, or CSS.

It excels at capturing data, visualizations, and graphics from multiple sources and broadcasting them live over the web.

H2O Wave gives your Python programs the ability to push content to connected clients as it happens, in realtime. In other words, it lets your program display up-to-date information without asking your users to hit their browser's reload button.You can use H2O Wave for:

- Dashboards and visualizations for live monitoring.
- Live information displays: news, tickers, health, or performance data.
- Apps that require instant notifications, updates, events, or alerts.
- Apps that involve messaging: chat, bots, etc.
- Collaborative apps: whiteboards, sharing, etc.

You can also use H2O Wave when you find yourself reaching for a web application framework - it can handle regular (non-realtime) apps just fine.

In this task, we are required to create a program using H2O Wave and Python to run a simple system monitoring tool that displays the CPU, memory and network stats on a web page.

This task will also introduce a new concept, called data buffers, which allows us to use the Wave server to store rows (also called tuples or records) of information - much like how you would use tables in a database, or dataframes in Python or R - to deal with structured data.

Codes

The following are the steps with codes that was used for this task.

Step 1: Open a terminal and start the Wave Server

```
cd wave
waved.exe
```

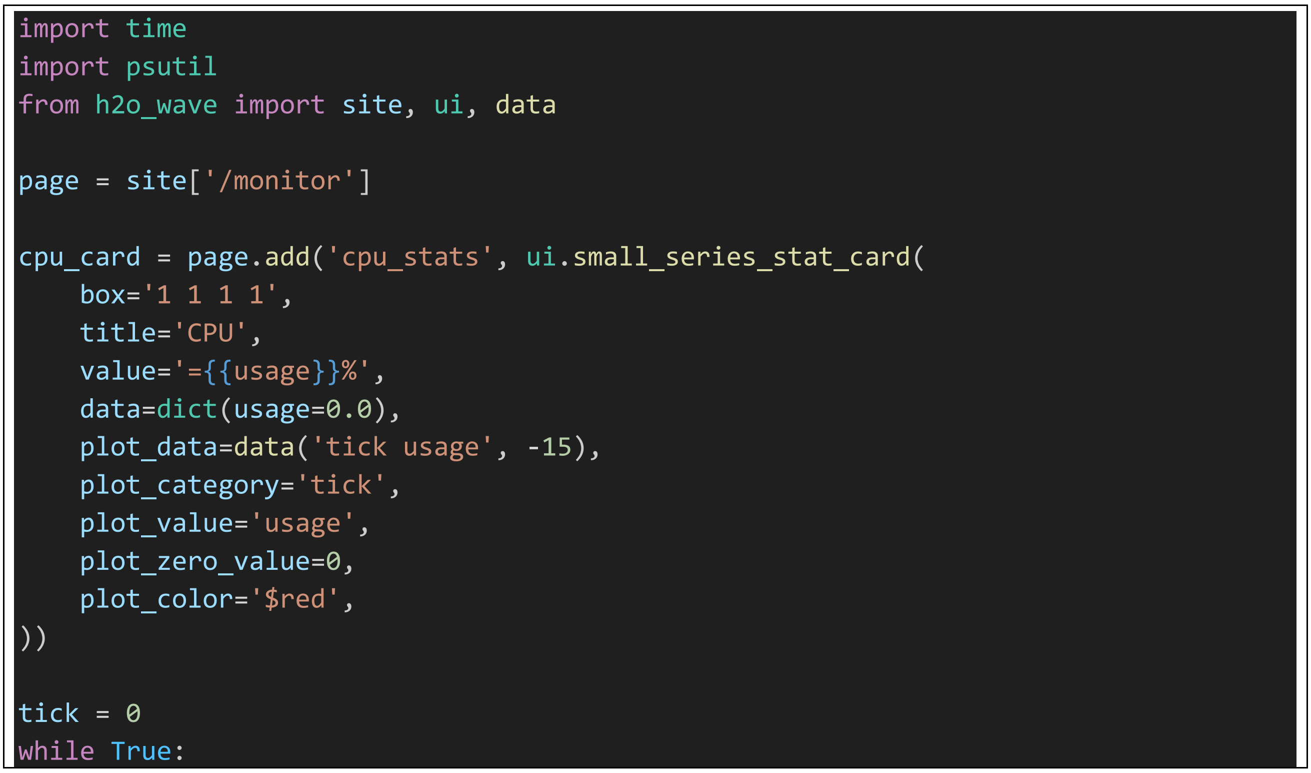
Step 2: Set up a virtual environment

```
python -m venv venv
.\venv\Scripts\activate
```

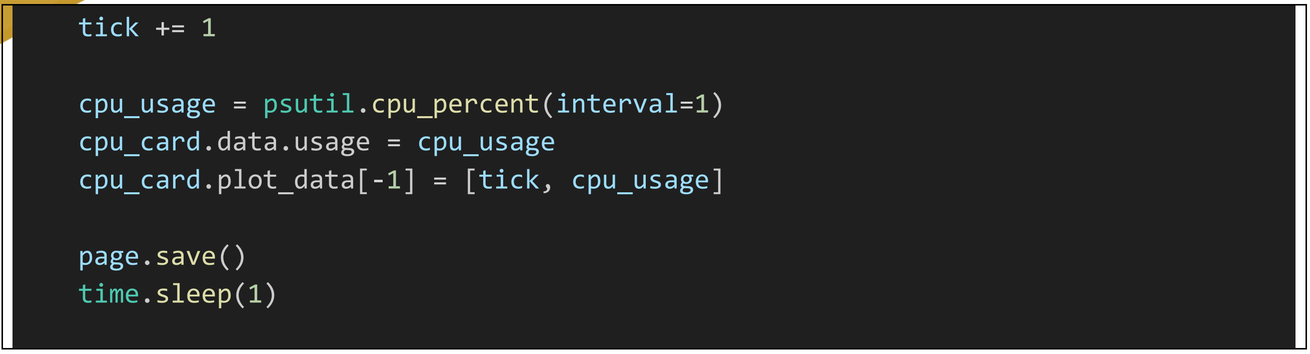
Step 3: Install dependencies specifically the psutil package to read system stats

```
pip install psutil
```

Step 4: Write the program for monitoring CPU usage



```python
import time
import psutil
from h2o_wave import site, ui, data

page = site['/monitor']

cpu_card = page.add('cpu_stats', ui.small_series_stat_card(
    box='1 1 1 1',
    title='CPU',
    value='={{usage}}%',
    data=dict(usage=0.0),
    plot_data=data('tick usage', -15),
    plot_category='tick',
    plot_value='usage',
    plot_zero_value=0,
    plot_color='$red',
))

tick = 0
while True:
```

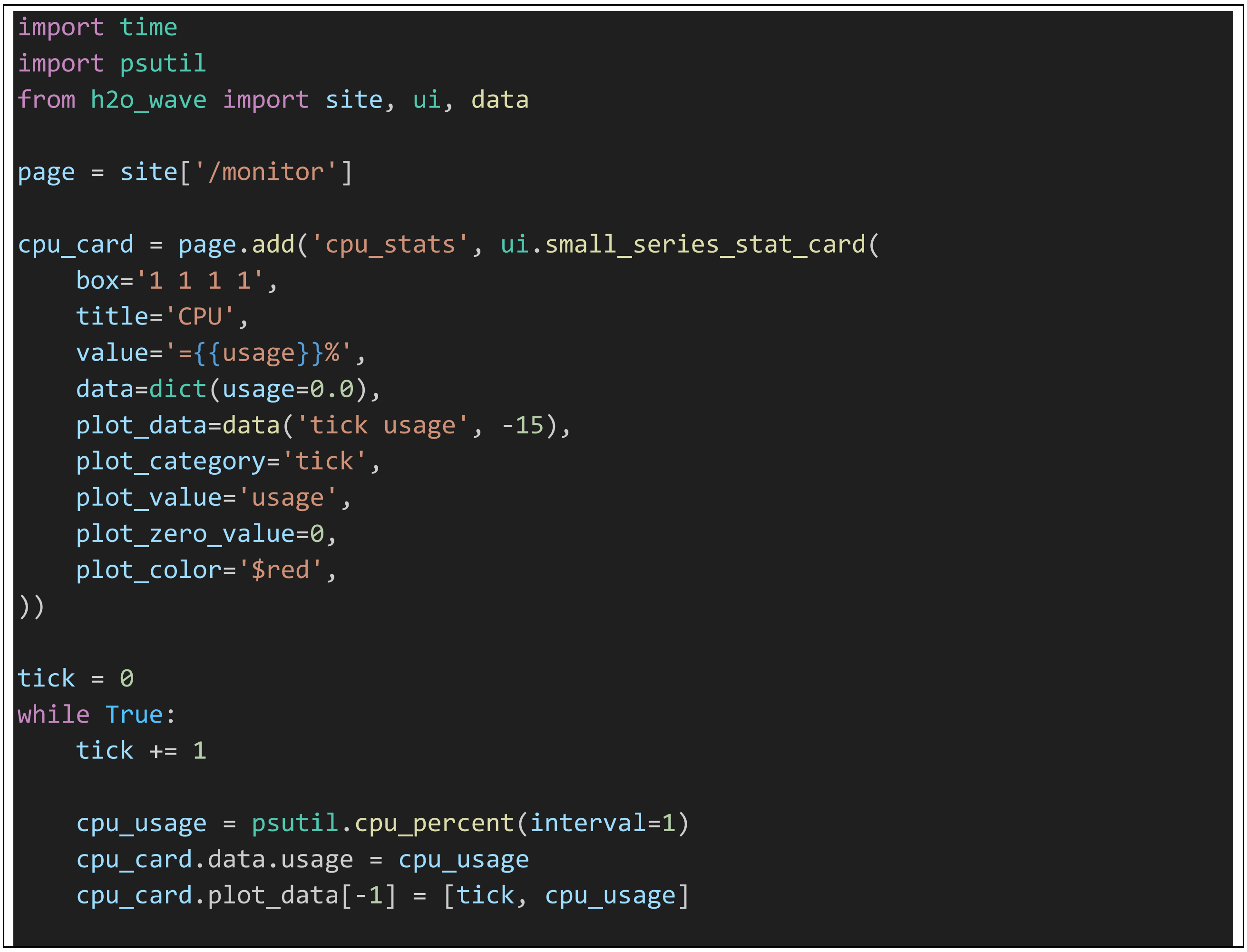


```
    tick += 1

    cpu_usage = psutil.cpu_percent(interval=1)
    cpu_card.data.usage = cpu_usage
    cpu_card.plot_data[-1] = [tick, cpu_usage]

    page.save()
    time.sleep(1)
```

Step 5: Run the program

```
python system_monitor.py
```

Step 6: Add the program for monitoring memory usage



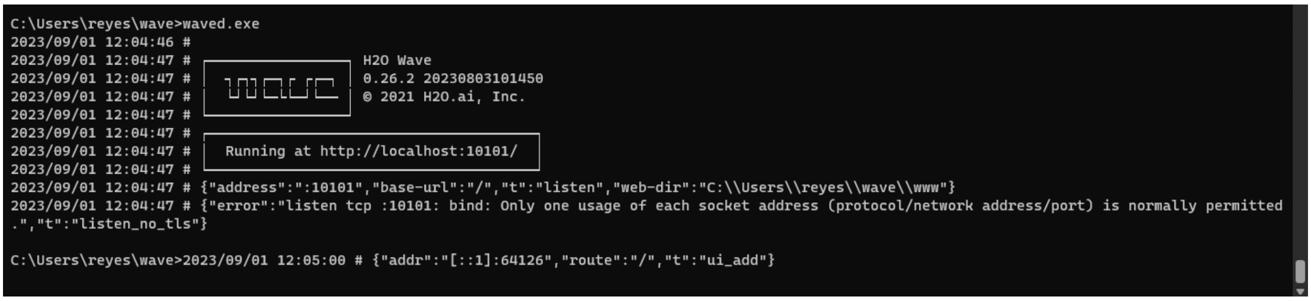
```
import time
import psutil
from h2o_wave import site, ui, data

page = site['/monitor']

cpu_card = page.add('cpu_stats', ui.small_series_stat_card(
    box='1 1 1 1',
    title='CPU',
    value='={{usage}}%',
    data=dict(usage=0.0),
    plot_data=data('tick usage', -15),
    plot_category='tick',
    plot_value='usage',
    plot_zero_value=0,
    plot_color='$red',
))

tick = 0
while True:
    tick += 1

    cpu_usage = psutil.cpu_percent(interval=1)
    cpu_card.data.usage = cpu_usage
    cpu_card.plot_data[-1] = [tick, cpu_usage]
    page.save()
    time.sleep(1)
```

Step 7: Run the program

```
python system_monitor_with_memory.py
```

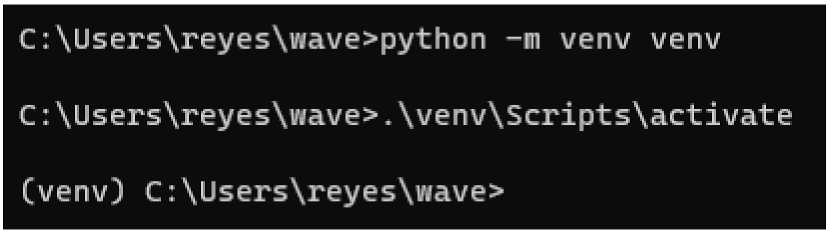
Screenshots of Work

Figure 1. Start the wave server using waved.exe command



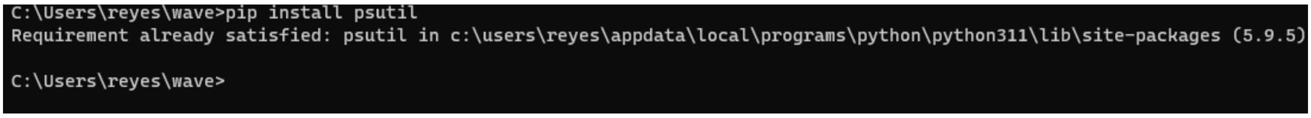
```
C:\Users\reyes\wave>waved.exe
2023/09/01 12:04:46 #
2023/09/01 12:04:47 #                  H2O Wave
2023/09/01 12:04:47 #                  0.26.2 20230803101450
2023/09/01 12:04:47 #                  © 2021 H2O.ai, Inc.
2023/09/01 12:04:47 #
2023/09/01 12:04:47 #
2023/09/01 12:04:47 #   Running at http://localhost:10101/
2023/09/01 12:04:47 #
2023/09/01 12:04:47 # {"address":":10101","base-url":"/","t":"listen","web-dir":"C:\\Users\\reyes\\wave\\www"}
2023/09/01 12:04:47 # {"error":"listen tcp :10101: bind: Only one usage of each socket address (protocol/network address/port) is normally permitted
.","t":"listen_no_tls"}

C:\Users\reyes\wave>2023/09/01 12:05:00 # {"addr":"[::1]:64126","route":"/","t":"ui_add"}
```

Figure 2. Setting the virtual environment



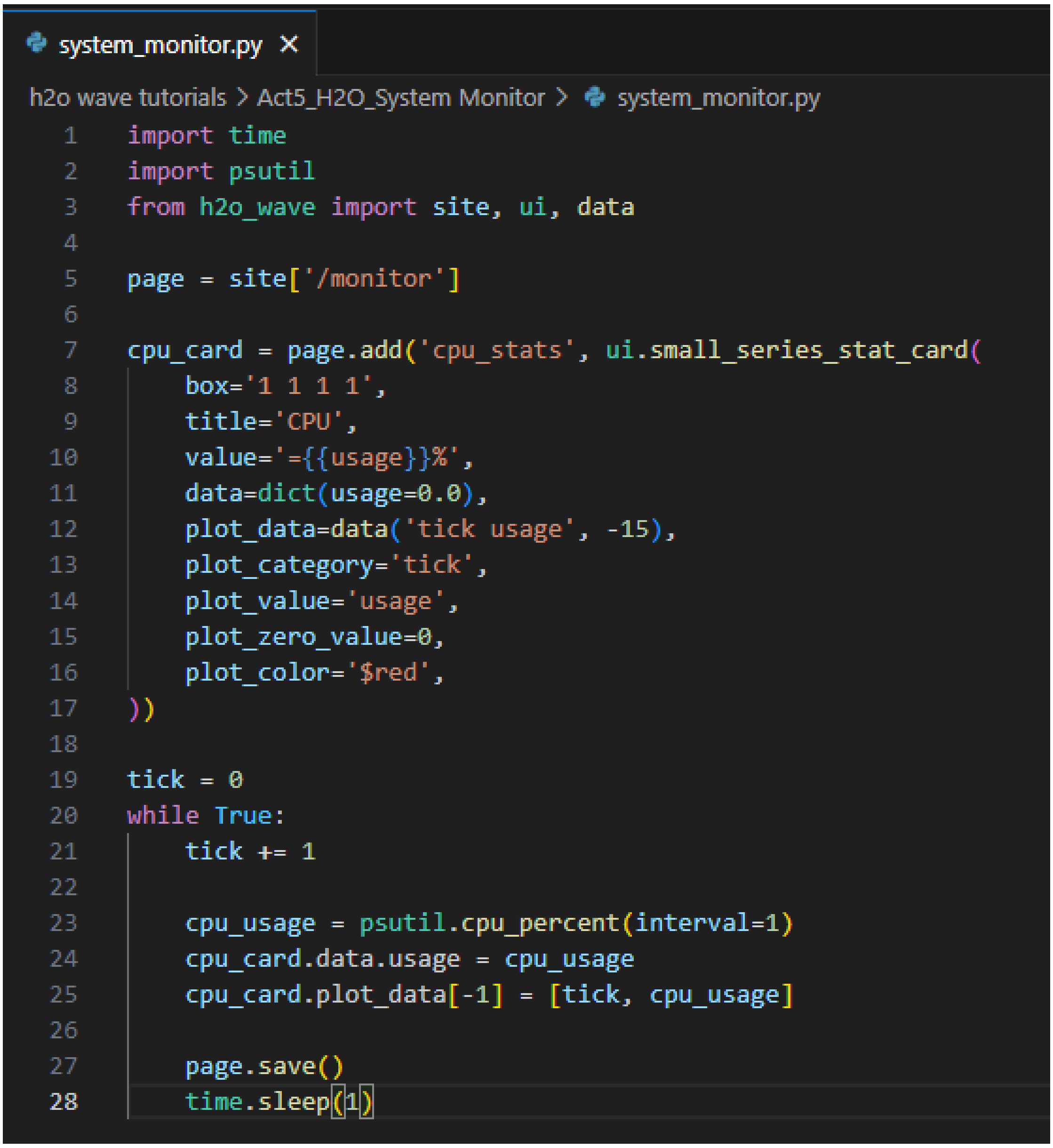
```
C:\Users\reyes\wave>python -m venv venv

C:\Users\reyes\wave>.\venv\Scripts\activate

(venv) C:\Users\reyes\wave>
```

Figure 3. Installing the psutil package



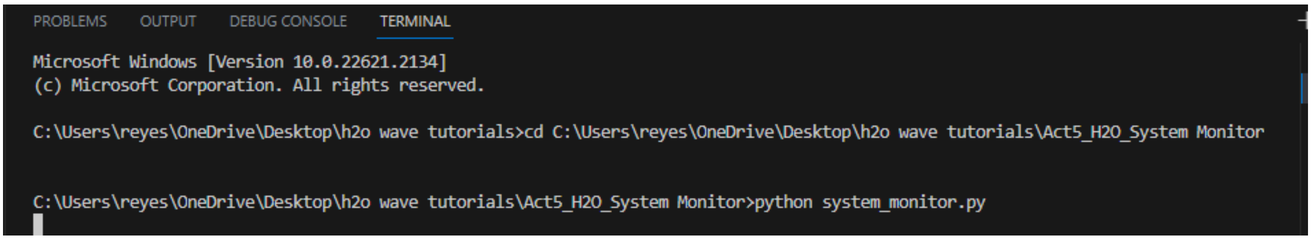
```
C:\Users\reyes\wave>pip install psutil
Requirement already satisfied: psutil in c:\users\reyes\appdata\local\programs\python\python311\lib\site-packages (5.9.5)

C:\Users\reyes\wave>
```

Figure 4. Source code for monitoring CPU usage in Visual Studio Code


system_monitor.py ×
h2o wave tutorials > Act5_H2O_System Monitor > system_monitor.py

```python
import time
import psutil
from h2o_wave import site, ui, data

page = site['/monitor']

cpu_card = page.add('cpu_stats', ui.small_series_stat_card(
    box='1 1 1 1',
    title='CPU',
    value='={{usage}}%',
    data=dict(usage=0.0),
    plot_data=data('tick usage', -15),
    plot_category='tick',
    plot_value='usage',
    plot_zero_value=0,
    plot_color='$red',
))

tick = 0
while True:
    tick += 1

    cpu_usage = psutil.cpu_percent(interval=1)
    cpu_card.data.usage = cpu_usage
    cpu_card.plot_data[-1] = [tick, cpu_usage]

    page.save()
    time.sleep(1)
```

Figure 5. Running the code in a terminal



```
Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\reyes\OneDrive\Desktop\h2o wave tutorials>cd C:\Users\reyes\OneDrive\Desktop\h2o wave tutorials\Act5_H2O_System Monitor


C:\Users\reyes\OneDrive\Desktop\h2o wave tutorials\Act5_H2O_System Monitor>python system_monitor.py
```

Figure 6. Monitoring CPU Usage output on localhost:10101/monitor

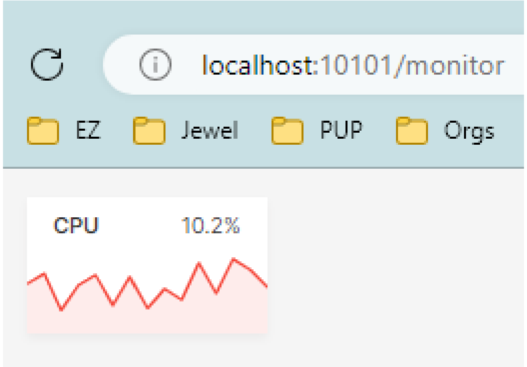
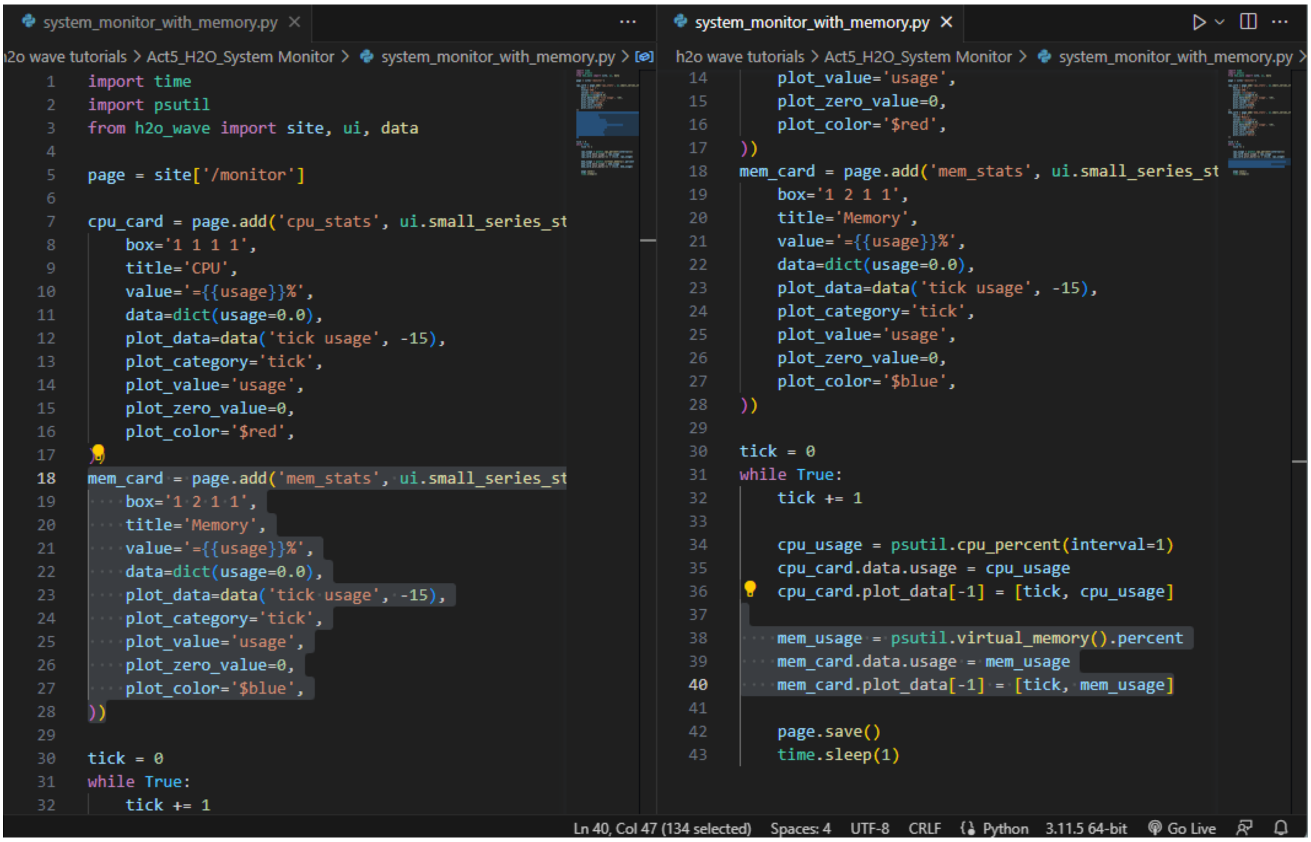


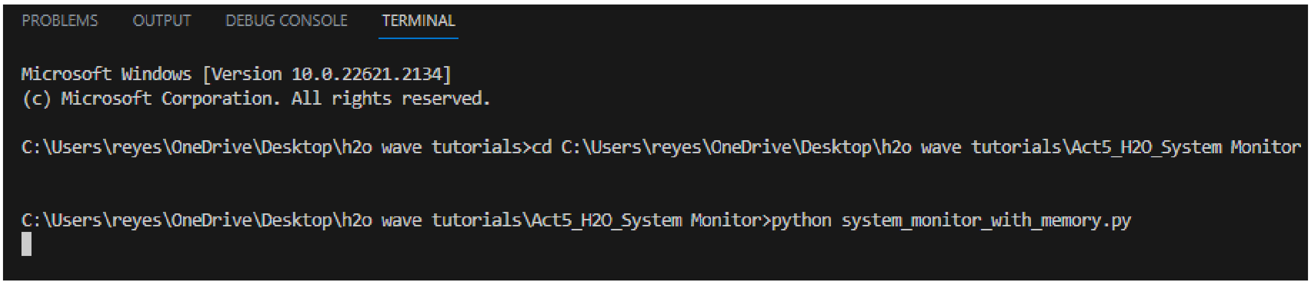
Figure 7. Add the source code for monitoring memory usage in Visual Studio Code


system_monitor_with_memory.py
h2o wave tutorials > Act5_H2O_System Monitor > system_monitor_with_memory.py
system_monitor_with_memory.py
h2o wave tutorials > Act5_H2O_System Monitor > system_monitor_with_memory.py
Ln 40, Col 47 (134 selected)
Spaces: 4
UTF-8
CRLF
Python
3.11.5 64-bit
Go Live

```python
import time
import psutil
from h2o_wave import site, ui, data

page = site['/monitor']

cpu_card = page.add('cpu_stats', ui.small_series_st
    box='1 1 1 1',
    title='CPU',
    value='={{usage}}%',
    data=dict(usage=0.0),
    plot_data=data('tick usage', -15),
    plot_category='tick',
    plot_value='usage',
    plot_zero_value=0,
    plot_color='$red',
))
mem_card = page.add('mem_stats', ui.small_series_st
    box='1 2 1 1',
    title='Memory',
    value='={{usage}}%',
    data=dict(usage=0.0),
    plot_data=data('tick usage', -15),
    plot_category='tick',
    plot_value='usage',
    plot_zero_value=0,
    plot_color='$blue',
))

tick = 0
while True:
    tick += 1
```

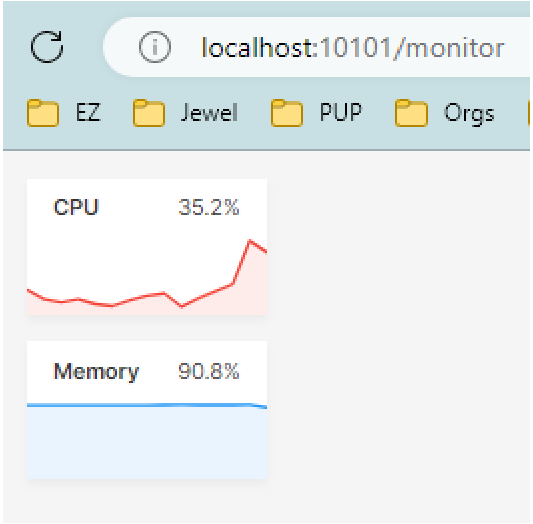
```python
    plot_value='usage',
    plot_zero_value=0,
    plot_color='$red',
))
mem_card = page.add('mem_stats', ui.small_series_st
    box='1 2 1 1',
    title='Memory',
    value='={{usage}}%',
    data=dict(usage=0.0),
    plot_data=data('tick usage', -15),
    plot_category='tick',
    plot_value='usage',
    plot_zero_value=0,
    plot_color='$blue',
))

tick = 0
while True:
    tick += 1

    cpu_usage = psutil.cpu_percent(interval=1)
    cpu_card.data.usage = cpu_usage
    cpu_card.plot_data[-1] = [tick, cpu_usage]

    mem_usage = psutil.virtual_memory().percent
    mem_card.data.usage = mem_usage
    mem_card.plot_data[-1] = [tick, mem_usage]

    page.save()
    time.sleep(1)
```

Figure 8. Running the code in a terminal



```
Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\reyes\OneDrive\Desktop\h2o wave tutorials>cd C:\Users\reyes\OneDrive\Desktop\h2o wave tutorials\Act5_H2O_System Monitor


C:\Users\reyes\OneDrive\Desktop\h2o wave tutorials\Act5_H2O_System Monitor>python system_monitor_with_memory.py
```

Figure 9. Monitoring CPU and Memory Usage output on localhost:10101/monitor



Jewel Anne A. Reyes
BS Computer Science
Polytechnic University of the Philippines