1. **Getting info in Python:**

Python is equipped with libraries like psutil and platform that provide access to system hardware details.

**Steps for Python:**

**Step 1: Install psutil Library**

The psutil library provides functions to fetch details about system memory, CPU, and more.

pip install psutil

**Step 2: Fetch Processor Information**

Use the platform module to retrieve the processor name and the psutil library for additional CPU details.

import platform

import psutil

# Processor name

processor\_name = platform.processor()

# CPU details

cpu\_count = psutil.cpu\_count(logical=True) # Logical cores

physical\_cores = psutil.cpu\_count(logical=False) # Physical cores

cpu\_freq = psutil.cpu\_freq().current # Current CPU frequency in MHz

print(f"Processor: {processor\_name}")

print(f"Logical Cores: {cpu\_count}")

print(f"Physical Cores: {physical\_cores}")

print(f"CPU Frequency: {cpu\_freq} MHz")

**Step 3: Fetch RAM Information**

Retrieve total RAM size using psutil.

# Total RAM in GB

ram\_size = round(psutil.virtual\_memory().total / (1024 \*\* 3), 2)

print(f"Total RAM: {ram\_size} GB")

**Resources:**

* Official psutil documentation: [psutil](https://psutil.readthedocs.io/en/latest/): <https://psutil.readthedocs.io/en/latest/>
* Python platform module: [platform](https://docs.python.org/3/library/platform.html): <https://docs.python.org/3/library/platform.html>

**2. Retrieving System Information in Bash**

In Linux or macOS environments, Bash provides built-in commands that can be used to extract hardware information like processor details, RAM size, and the number of threads. Here are the steps.

**Steps for Bash:**

**Step 1: Fetch Processor Information**

The /proc/cpuinfo file contains detailed CPU information on Linux systems.

# Fetch CPU model name

grep 'model name' /proc/cpuinfo | uniq

# Count logical cores

grep -c ^processor /proc/cpuinfo

**Step 2: Fetch RAM Information**

Use the free or /proc/meminfo commands to get RAM details.

# Total RAM in GB

free -h | grep "Mem" | awk '{print $2}'

# Alternatively, using /proc/meminfo

grep MemTotal /proc/meminfo

**Step 3: Fetch Thread Count**

To get the number of threads (logical cores), the following command can be used:

nproc --all

**Resources:**

* proc file system overview: The Linux Documentation Project: <https://tldp.org/>
* nproc and free documentation: GNU coreutils: <https://www.gnu.org/software/coreutils/>

**3. Retrieving System Information in JavaScript (Node.js)**

JavaScript running in a browser doesn't have access to hardware information for security reasons. However, in a Node.js environment, the built-in os module provides system details like CPU information, total memory, and thread count.

**Steps for Node.js:**

**Step 1: Install Node.js**

Ensure Node.js is installed on your system. We can download it from <https://nodejs.org/en> .

**Step 2: Retrieve Processor Information**

Use the os.cpus() method to retrieve CPU model and core details.

const os = require('os');

// CPU details

const cpuInfo = os.cpus();

const processorName = cpuInfo[0].model;

const logicalCores = cpuInfo.length;

console.log(`Processor: ${processorName}`);

console.log(`Logical Cores: ${logicalCores}`);

**Step 3: Retrieve RAM Information**

The os.totalmem() function returns total RAM in bytes.

// Total RAM in GB

const totalRAM = (os.totalmem() / (1024 \*\* 3)).toFixed(2);

console.log(`Total RAM: ${totalRAM} GB`);

**Step 4: Retrieve Thread Count**

The os.cpus().length property gives the number of logical processors (threads).

// Thread count

const threadCount = os.cpus().length;

console.log(`Total Threads: ${threadCount}`);

**Resources:**

* <https://nodejs.org/api/os.html>

**Summary Comparison:**

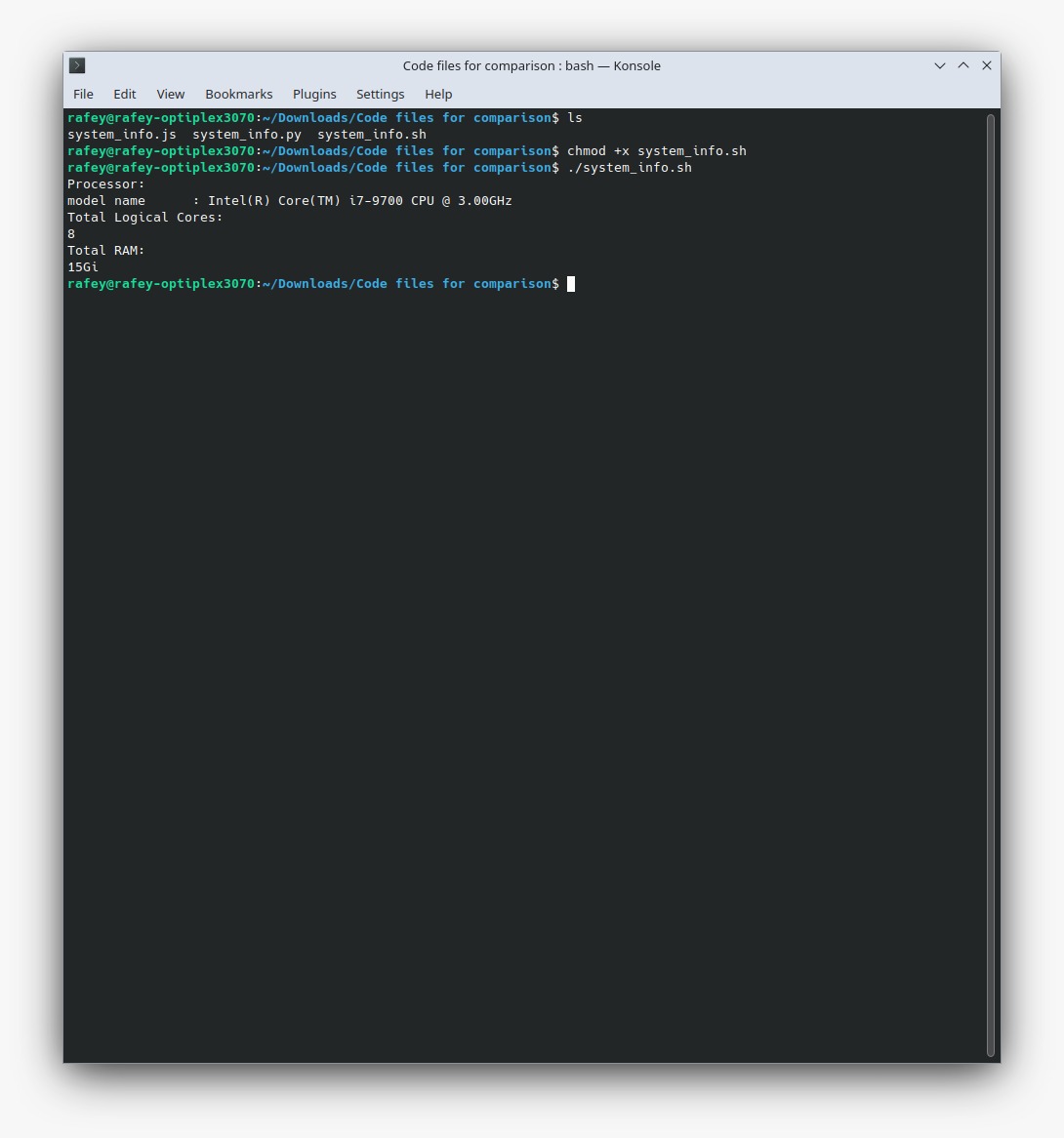
|  |  |  |  |
| --- | --- | --- | --- |
| Aspect | Python | Bash | JavaScript (Node.js) |
| Processor Info | platform.processor() & psutil.cpu\_\* | /proc/cpuinfo, grep | os.cpus() |
| RAM Info | psutil.virtual\_memory().total | free -h or /proc/meminfo | os.totalmem() |
| Thread Count | psutil.cpu\_count(logical=True) | nproc --all | os.cpus().length |

**Conclusion:**

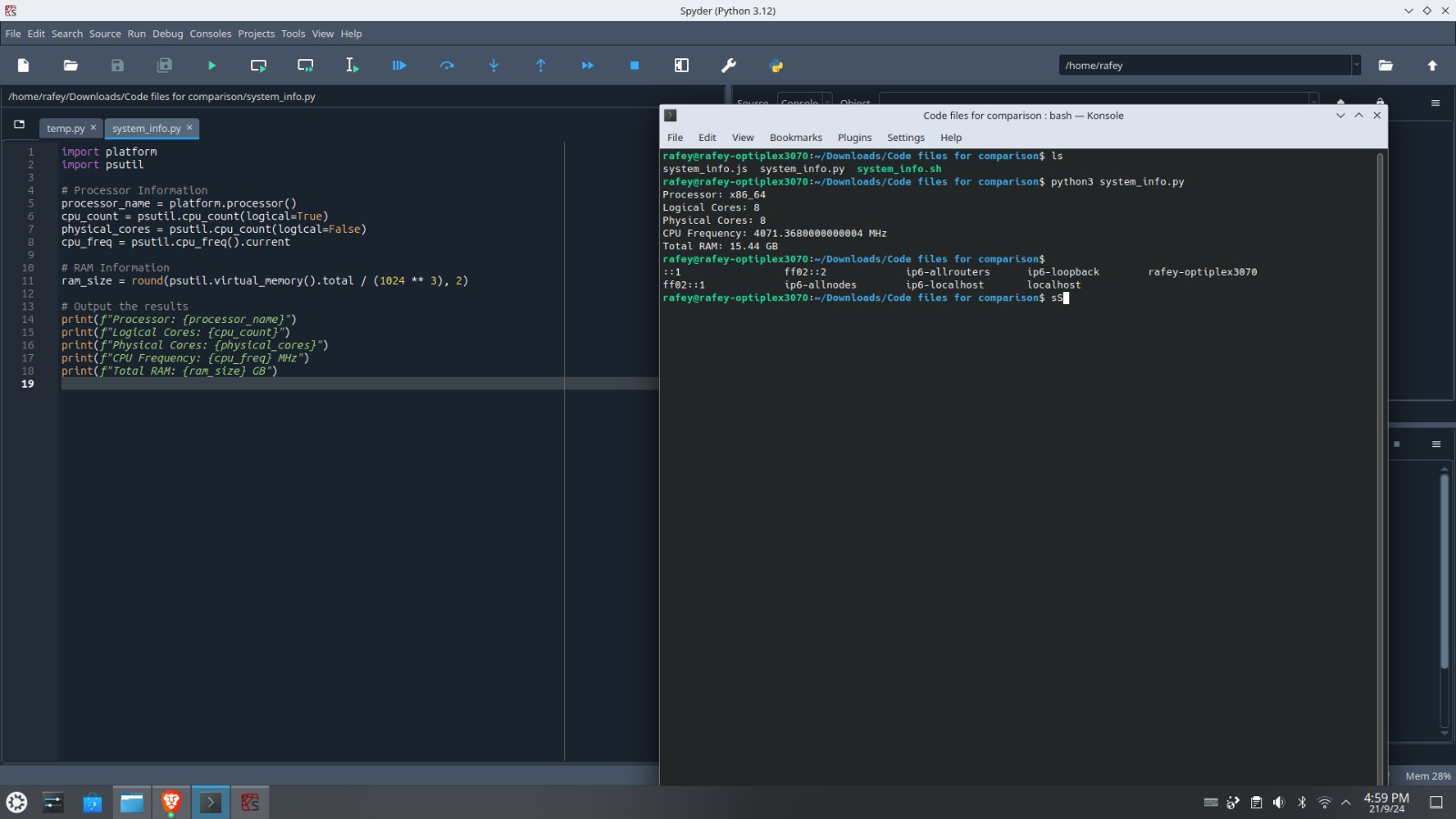
* Python provides high-level libraries that simplify access to system hardware information.
* Bash is ideal for scripting system details on Linux-based systems through commands like grep and nproc.
* JavaScript (Node.js) offers an efficient method to retrieve system information, making it convenient for server-side applications.

**Runtime Comparison**

* .sh file:



* .py file



* .js file

A screenshot of a computer

Description automatically generated