# Gateway Device Application (Connected Devices)

## Lab Module 02

Be sure to implement all the PIOT-GDA-\* issues (requirements) listed at [PIOT-INF-02-001 - Lab Module 02](https://github.com/orgs/programming-the-iot/projects/1#column-9974938).

### Description

NOTE: Include two full paragraphs describing your implementation approach by answering the questions listed below.

What does your implementation do?

? GatewayDeviceApp Class: This class serves as the main entry point for your application. It manages the lifecycle of the application, including starting and stopping the SystemPerformanceManager.

? SystemPerformanceManager Module: This module monitors system performance metrics like CPU and memory utilization. It collects this data at scheduled intervals and makes it accessible through the application.

? BaseSystemUtilTask Class: This class serves as a base for utility tasks that retrieve system metrics. It provides a structure for the specific tasks related to CPU and memory utilization.

? SystemCpuUtilTask Module: This module retrieves the CPU utilization of the system. It implements a method to access the CPU metrics.

? SystemMemUtilTask Module: This module retrieves the JVM memory utilization. It includes functionality to monitor memory usage effectively.

? Scheduled Telemetry Handling: Within the SystemPerformanceManager, thereÆs a method (handleTelemetry()) that utilizes scheduled threads to invoke the telemetry value methods from SystemCpuUtilTask and SystemMemUtilTask, allowing periodic collection of performance data.

How does your implementation work?

1.

Application Initialization:

o The GatewayDeviceApp class is instantiated, initializing the SystemPerformanceManager.

o The application starts the SystemPerformanceManager, which begins monitoring system performance.

2. Performance Monitoring:

o The SystemPerformanceManager schedules a task (using a scheduled thread) that periodically calls the handleTelemetry() method.

o Inside handleTelemetry(), the getTelemetryValue() methods from both SystemCpuUtilTask and SystemMemUtilTask are called to retrieve current CPU and memory usage metrics.

3. Task Implementation:

o BaseSystemUtilTask provides a foundation for SystemCpuUtilTask and SystemMemUtilTask, ensuring they share common properties or methods (if any).

o Each specific task retrieves its respective data (CPU or memory) using system calls or Java APIs and returns the values to the SystemPerformanceManager.

4. Data Handling:

o The data collected by the performance tasks can be logged, displayed, or processed further, depending on the requirements of your application.

5. Graceful Shutdown:

o When the application is stopped, GatewayDeviceApp ensures that the SystemPerformanceManager is properly shut down, halting any ongoing monitoring tasks.

This implementation structure ensures modularity, making it easier to manage and extend the application as needed, while providing real-time insights into system performance metrics.

### Code Repository and Branch

NOTE: Be sure to include the branch (e.g. https://github.com/programming-the-iot/python-components/tree/alpha001).

URL: https://github.com/zo1235/connectedDevice/tree/lab02

### UML Design Diagram(s)

A screenshot of a computer

Description automatically generated

### Unit Tests Executed

NOTE: TA's will execute your unit tests. You only need to list each test case below

(e.g. ConfigUtilTest, DataUtilTest, etc). Be sure to include all previous tests, too,

since you need to ensure you haven't introduced regressions.

- ConfigUtilTest

- SystemCpuUtilTaskTest

- SystemMemUtilTaskTest

### Integration Tests Executed

NOTE: TA's will execute most of your integration tests using their own environment, with

some exceptions (such as your cloud connectivity tests). In such cases, they'll review

your code to ensure it's correct. As for the tests you execute, you only need to list each

test case below (e.g. SensorSimAdapterManagerTest, DeviceDataManagerTest, etc.)

- GatewayDeviceAppTest

- SystemPerformanceManagerTest

-

EOF.