Python Assignment 1
Smart Home IoT Automation Simulator

📑 Description:

In this assignment, you will need to develop a Python-based IoT simulator for a smart home automation system. The simulator should emulate the behaviour of various IoT devices commonly found in a smart home, such as smart lights, thermostats, and security cameras. You will also create a central automation system that manages these devices and build a monitoring dashboard to visualise and control the smart home. This assignment will help you apply your Python programming skills, including OOP, data handling, real-time data monitoring, and graphical user interfaces (GUIs).

🎯 Objective:

- Develop a Python-based IoT device simulator.
- Implement basic data analytics and processing.
- Create an eye-catching monitoring dashboard to display sensor data and analytics results.
- Use Python data structures, classes, OOP objects, instance and class variables, OOP methods, pip, static.
- methods, files & standard library, exception handling, modules, and packages.

Part 1: IoT Device Emulation (25 %)

- Device Classes: Create Python classes for each type of IoT device you want to simulate, such as SmartLight, Thermostat, and SecurityCamera. Each class should have attributes like device ID, status (on/off), and relevant properties (e.g., temperature for thermostats, brightness for lights and security status for cameras).
- Device Behavior: Implement methods for each device class that allow for turning devices on/off and changing their properties. Simulate realistic behavior, such as gradual dimming for lights or setting temperature ranges for thermostats.
- Randomization: Include a randomization mechanism to simulate changing device states and properties over time.

Part 2: Central Automation System (25 %)

- Automation System Class: Create a central automation system class, e.g., AutomationSystem, responsible for managing and controlling all devices. It should provide methods for discovering devices, adding them to the system, and executing automation tasks.
- Simulation Loop: Implement a simulation loop that runs periodically (e.g., every few seconds) to trigger automation rules, update device states, and simulate device behaviors.

Part 3: Documentation (30%)

- Documentation: Provide clear documentation for your code, including class descriptions, method explanations, and instructions on how to run the simulation and use the dashboard.
- Develop test cases to ensure that the simulator and automation system behave as expected. Test various scenarios, such as different automation rules and user interactions.

Part 4: Monitoring Dashboard (20%)

- Graphical User Interface (GUI): Create a GUI for monitoring and controlling the smart home system. You can use Python GUI libraries like Tkinter. The GUI should display the status and properties of each device, provide controls to interact with them, and visualize data.
- Real-time Data Monitoring: Display real-time data from the simulated devices on the dashboard. This includes temperature graphs for thermostats, motion detection status for cameras, and brightness levels for lights.
- User Interaction: Allow users to interact with devices through the GUI, such as toggling lights on/off, adjusting thermostat settings, and arming/disarming security cameras.

💡 Hints:

- Use Python's random library for generating sensor data.
- Utilize the statistics module for data analysis.
- Use a simulation loop for rule execution.
- Timestamp sensor data with Python's datetime module.
- Design a GUI (try to be creative) using Tkinter.
- Document class attributes and methods.
- Feel free to use ChatGPT, but make sure to be able uderstand and explain your solution!
- Do not hesitate to ask your instructor for help and asking for the template for the GUI!

Sample GUI:

Smart Home IoT Simulator

Living Room Light: SmartLight Status: On
Living Room Thermostat: Thermostat Status: On
Front Door Camera: SecurityCamera Status: On

Living Room Light Brightness

47

Toggle ON/OFF

Living Room Light – 47%

Living Room Thermostat Temperature

17

Toggle ON/OFF

Living Room Thermostat – 17C

Front Door Camera Motion Detection

Random Detect Motion

Toggle ON/OFF

Front Door Camera – Motion: NO

Automation Rule: Turn on lights when motion is detected