ECE 448/528 – Application Software Design
Spring 2025

Project 5: Group Management for IoT Hub

Due: Sunday, April 20th, 2025, midnight

IMPORTANT:	***You must sign and date below acknowledgment statement below on the title page of your report.*** **Failing to do so, or any violation of this rule will result in an automatic failure for this course.**
Acknowledgment:	I acknowledge all works including figures, codes, and writings belong to me and/or persons who are referenced. I understand if any similarity in the code, comments, customized program behavior, report writings, and/or figures are found, both the helper (original work) and the requestor (duplicated/modified work) will be called for academic disciplinary action.

I. Overview

In this project, we will improve the server backend for our IoT hub by introducing group management that allows users to define groups of smart plugs and control a group of smart plugs as a whole.

You should be able to find the user stories describing the requirements in the next section. While we are going to discuss possible class designs and implementations in the lectures, you are free to choose designs and implementations you are comfortable with.

You should be able to follow the red-green cycle to add unit tests and implement your classes now. For convenience, you may need to utilize MQTT and HTTP communications to test some of your classes. Please refer to GradeP5.java for useful codes.

II. User Stories

1. Create a Group

As an end-user, I want to create a group "groupName" of plugs via a POST request to /api/groups/groupName, so that I can manage multiple plugs as a whole. The body of the POST request is a JSON array of the names of the plugs to be included in the group. If the group already exists, all its members will be replaced. Note that a single plug is allowed to be assigned to multiple groups.

2. **Remove a Group**

As an end-user, I want to remove a group "groupName" of plugs via a DELETE request to /api/groups/groupName, so that I can remove the group in a web application.

3. **State of a Group**

As an end-user, I want to query the state of a group "groupName" as the states of its member plugs via a GET request to /api/groups/groupName, so that I can obtain their states in a web application. The response should be a JSON object with keys "name" for the name of the group, and "members" for a JSON array of objects, that each represents the state of a member plug.

4. **States of All Groups**

As an end-user, I want to query all the groups for the states of member plugs via a GET request to /api/groups, so that I can obtain everything together in a web application. The response should be a JSON array of objects, that each represents the state of a group.

5. **Control a Group**

As an end-user, I want to switch on/off or toggle all the plugs in a group "groupName" via a GET request to /api/groups/groupName with a query string, so that I can control plugs in a group together. The query string is the same as those to control a single plug.

III. Testing Procedures

The testing procedures are implemented in ece448.grading.GradeP5. It should be fairly straightforward to verify all user stories are covered. Note that we also check the IoT simulators and the MQTT broker to make sure everything works properly.

IV. Additional Grading Policy for Project 5

Your code should terminate gracefully upon completion of ece448.grading.GradeP5 execution by returning to the Bash or command prompt. If your program fails to do this, you will be deducted 10 points toward your Project 5 grade.