Project 4: Server Backend for IoT Hub

I.

Overview

In this project, we will build the server backend for our IoT hub as a Spring Boot application. The server backend will provide RESTful services to the frontend web application that we will build in Project 6. To control smart plugs and obtain their state updates, the server backend will communicate with one or more IoT simulators via an MQTT broker.

User stories are described in the next section. While we are going to discuss the possible class design and implementations in the lecture, you are free to choose designs and implementations that you are comfortable with.

You should be able to follow the red-green cycle to add unit tests and implement your classes by now. For convenience, you may need to utilize MQTT and HTTP communications to test some of your classes. Please refer to GradeP3.java and GradeP4.java for useful codes.

II.

User Stories

1.

State of a Single Plug

As an end-user, I want to query the state of the plug “plugName” via a GET request to /api/plugs/plugName, so that I can obtain the state of individual plugs in a web application. The response should be a JSON object in the format, e.g.

{"name":"plugName", "state":"on", "power":100}.

The value for “state” could also be “off”.

2.

States of All Plugs

As an end-user, I want to query the states of all plugs via a GET request to /api/plugs, so that I can obtain all of them at once in a web application. The response should be a JSON array of objects, where each represents the state of a single plug.

3.

Control a Single Plug

As an end-user, I want to switch on/off or toggle the plug “plugName” via a GET request to /api/plugs/plugName with a query string, so that I can control it in a web application. To toggle the plug, the query string is action=toggle. To switch the plug on, the query string is action=on. To switch the plug off, the query string is action=off.

III.

Testing Procedures

The testing procedures are implemented in ece448.grading.GradeP4. 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.