JJ Boolean
Joshua Calder V00849953
Julien Godding V00877834

Seng330 Assignment 4 Design Document

This assignment was completed using the Spring MVC framework and Gradle as the build tool. Our code solution is based off of Spring's *gs-serving-web-content* starter repo. It uses certain functions and classes from our own assignment 2 solutions as well as Professor Ernst's.

Spring MVC Processing Sequence

The application implements an HTML templating language, *JSP*, to render views to the client after the Dispatcher Servlet receives an HTTP request and has successfully invoked a handler. This handler then calls the specified controller, which communicates with their corresponding services (otherwise known as models) to access and update the device's behaviour and attributes. Since Spring provides the necessary handling and mapping of these requests, we had to implement the remaining Controller, Service (Model) and Data Access classes. Link 1 in the appendix shows a more in-depth look at the Spring MVC processing sequence.

Application Components and Functionality

The application is broken down into four main packages that contain: Controllers, Models, Services, and Views (HTML5 using the thymeleaf template engine). Of special note is the AbstractService.java which acts upon the database and the log files.
The application allows for user registration and secure login. Once logged in the user is taken to the hub page which contains all registered devices. If the user is an Administrator, they can view and the log of activity as well as act upon regular users. Devices can be controlled by clicking on the control button that returns that devices particular control page. Registered users and devices are stored in the database in order to be used upon the application startup.

Running the Application

To run the IOT application, `cd into the complete directory` and type `gradle bootRun`. After 'bootRun' is executed, the application will initialize and then begin serving on port 8080 (default) of localhost. If the server starts on another port, there will be a message displayed in the command line. After the application has successfully started, navigate to `localhost:8080` (or 127.0.0.1:8080) on the web browser to begin using the application. You may use the following login information or register your own information to test the application.

Default Login Information

Username: seng330 Password: <none>

Test Coverage

Assignment 4 improves upon the implementation of Assignment 3 by adding: persistent data storage, random event triggering, user registration, admin/regular user views, and video streaming. Running `gradle test` will show the testing results of every test case..

UML Class Diagram

Please see the JJBoolean_UML_deliverable.png in the repository to see a full-size of the UML Class diagram divided into Model-View-Controller sections.

Appendix

[1]https://terasolunaorg.github.io/guideline/1.0.1.RELEASE/en/Overview/SpringMVCOverview.html