Smart PyControl System Requirements Specification

CS 5394/4398

Summer II 2020

Texas State University

Zacarias, Carla
Fitzgerald, Philip
Langlais, Morgan
Yniesta, Anisa
Gross, Stephen

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to describe Smart PyControl, a python based, desktop application that is intended to control a variety of smart home devices. Smart PyControl is intended to ease the use of smart devices for users by allowing the users to control all their smart home devices through one application on one controllable network for convenience. The Smart PyControl system will be based off a database system that will allow the users to view their logged data. Lighting, security systems, locks, thermostats, tvs, speakers laptops and smartphones can all be accessed and controlled under one network through our application.

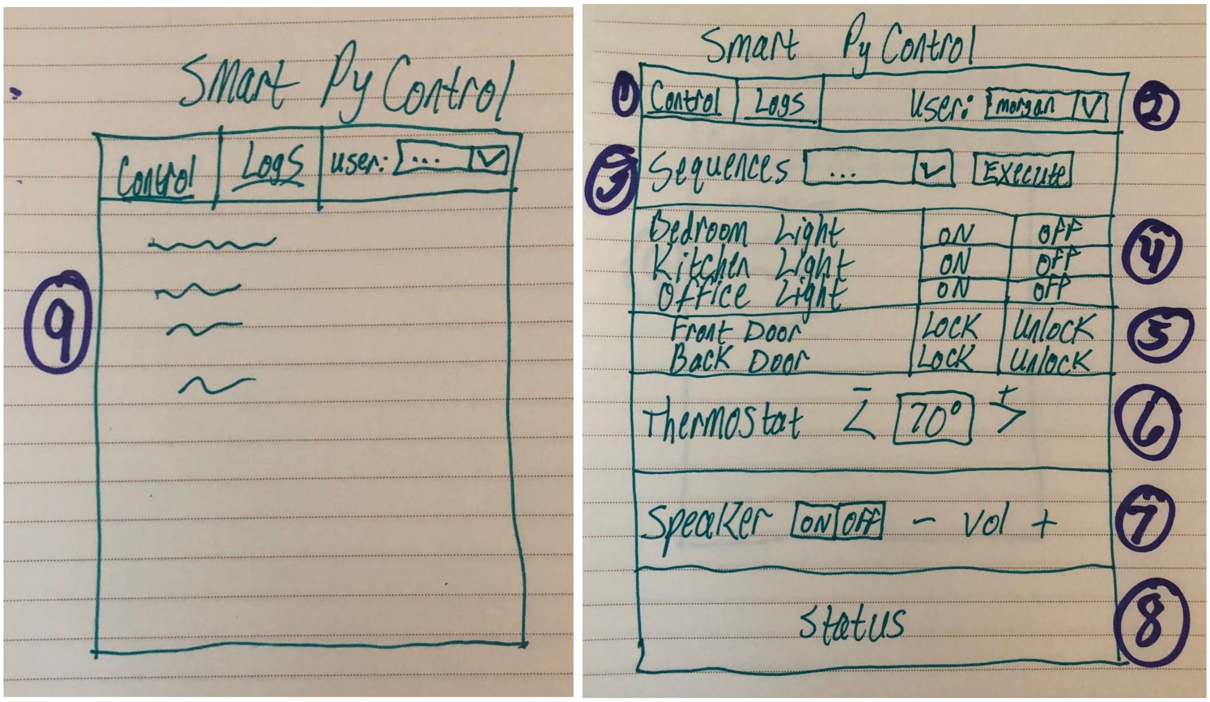
1.2 Scope

This requirements document is intended for users and programmers of this system. This System Requirements Document provides information about the Smart PyControl application.

1.3 Requirements

It is assumed that users will have their own smart home devices. We will be connecting to certain brands, but brands will not be mentioned in this System Requirements Specification document.

2. Statement of Functional Requirements



(1) Light control:
Users shall be able to turn lights off or on.

(2) Door control:
Doors shall be able to be either locked or unlocked by clicking the corresponding buttons.

(3) Thermostat control:
Users shall be able to control the temperature of the room by selecting the desired temperature using + and - buttons.

(4) Speaker control:
Users shall be able to turn connected speakers off or on.

(5)Volume control:
Users shall be able to control the volume of applicable connected devices.

(6) Status:
After an action is completed, a status will be displayed in the status section that will show whether the action completed successfully or not.

(7) Logging page:
The logging page will allow the user to select the user logs to view from the drop-down list. Then logs will be displayed in the window for the currently-selected user.

3. Non-functional Requirements

3.1 Reliability

The application shall document an error message in the case of the app shutting down. The application shall reboot in the case that it freezes for more than 10 seconds. The application shall leave all devices at their current settings in the case of the app losing power.

3.2 Robustness

In the case of error occurrence, the app shall take no longer than 10 seconds to reset. The application shall be able to support up to six connected devices.

3.3 Maintainability

Application updates shall take no longer than 30 minutes to download. No more than two updates per month will be pushed to the application.

3.4 Security

The application shall only have one user logged in at a time. The application shall not allow for remote control.

4. Design and Implementation Constraints

The user must have smart devices that can be connected to the application. The user's selected device to use the application must be connected to the internet.

5. References

5.1 Python APIs

Python Language
https://www.dataquest.io/blog/python-api-tutorial/
Sengled Python Client
https://pypi.org/project/sengled-client/
Python GUI Programming With Tkinter
https://realpython.com/python-gui-tkinter/
MongoDB API Docs for python
https://api.mongodb.com/python/
Python-kasa
https://github.com/python-kasa/python-kasa
IFTTT web requests using the SMTPLIB library
https://anthscomputercave.com/tutorials/ifttt/using_ifttt_web_request_email.html

5.2 Finite State Machines

Mealy State Machines
https://www.sciencedirect.com/topics/engineering/finite-state-machine
https://www.tutorialspoint.com/digital_circuits/digital_circuits_finite_state_machines.htm
FSMs and Python
https://stackabuse.com/theory-of-computation-finite-state-machines/