

**Smart PyControl**

**System Requirements Specification**

CS 5394

Summer 2020

Texas State University

Zacarias, Carla

Fitzgerald, Philip

Langlais, Morgan

Yniesta, Anisa

Gross, Stephen

### Table of Contents

[Introduction 3](#_Toc45741388)

[1.1 Purpose 3](#_Toc45741389)

[1.2 Scope 3](#_Toc45741390)

[1.3 Requirements 3](#_Toc45741391)

[Statement of Functional Requirements 4](#_Toc45741392)

[2.1 UI Mock-up 4](#_Toc45741393)

[2.2 Descriptions 4](#_Toc45741394)

[Non-functional Requirements 6](#_Toc45741395)

[3.1 Reliability 6](#_Toc45741396)

[3.2 Robustness 6](#_Toc45741397)

[3.3 Maintainability 6](#_Toc45741398)

[3.4 Security 6](#_Toc45741399)

[References 7](#_Toc45741400)

[4.1 Python APIs 7](#_Toc45741401)

[4.2 Finite State Machines 7](#_Toc45741402)

# 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.

# Statement of Functional Requirements

### 2.1 UI Mock-up

A close up of text on a whiteboard

Description automatically generated A close up of text on a whiteboard

Description automatically generated

### 2.2 Descriptions

(1) Users will be able to switch between tabs in the application: the control page and the logs page. A user can single-click on either tab to be taken to that page.

(2) A dropdown to select the current user. The user will click the drop-down to be presented with a list of users then click the name of the user they would like to login as.

(3) Select a sequence of actions to be performed from a drop-down. Each sequence has a name and once the name is selected from the sequence drop-down menu the actions will be performed.

(4) Light control section. Lights can be turned on or off by clicking the corresponding buttons.

(5) Door control section. Doors can be either locked or unlocked by clicking the corresponding buttons.

(6) Thermostat control section. The thermostat can be adjusted up or down. To turn the thermostat up, the user will click the plus arrow until the temperature display reaches the desired temperature. The same applies for decreasing the temperature.

(7) Speaker control section. Each speaker can be turned on or off and have the volume adjusted by using the corresponding buttons or +/- buttons for the volume.

(8) Status section. After action a status is displayed in the status section that will show whether the action completed successfully or not. For example, if a user turns on a lightbulb, the message “User has turned on light bulb 18F” will appear in the status window.

(9) 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.

# 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.

# References

### 4.1 Python APIs

Python Language

[https://www.dataquest.io/blog/python‐api‐tutorial/](https://www.dataquest.io/blog/python%E2%80%90api%E2%80%90tutorial/)

Sengled Python Client

[https://pypi.org/project/sengled‐client/](https://pypi.org/project/sengled%E2%80%90client/)

Python GUI Programming With Tkinter

[https://realpython.com/python‐gui‐tkinter/](https://realpython.com/python%E2%80%90gui%E2%80%90tkinter/)

MongoDB API Docs for Python

<https://api.mongodb.com/python/>

Python‐kasa

[https://github.com/python‐kasa/python‐kasa](https://github.com/python%E2%80%90kasa/python%E2%80%90kasa)

IFTTT web requests using the SMTPLIB library

<https://anthscomputercave.com/tutorials/ifttt/using_ifttt_web_request_email.html>

### 4.2 Finite State Machines

Mealy State Machines

[https://www.sciencedirect.com/topics/engineering/finite‐state‐machine](https://www.sciencedirect.com/topics/engineering/finite%E2%80%90state%E2%80%90machine)

<https://www.tutorialspoint.com/digital_circuits/digital_circuits_finite_state_machines.htm>

FSMs and Python

[https://stackabuse.com/theory‐of‐computation‐finite‐state‐machines/](https://stackabuse.com/theory%E2%80%90of%E2%80%90computation%E2%80%90finite%E2%80%90state%E2%80%90machines/)