About CS5394 Project
Smart PyControl

Overview

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.

System Requirement Specifications

Statement of Functional Requirements

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

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

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

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

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

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.

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.

Non-functional Requirements

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.

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.

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.

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

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.

APIs

We created our own API to interact with our MongoDB instance on IBM Cloud. Full specifications for the API may be found here: https://pycontrolapi.us-south.cf.appdomain.cloud/api-docs#/

Algorithms

References

https://www.dataquest.io/blog/python-api-tutorial/
https://realpython.com/python-gui-tkinter/
https://api.mongodb.com/python/
https://anthscomputercave.com/tutorials/ifttt/
using_ifttt_web_request_email.html
https://www.sciencedirect.com/topics/engineering/finite-state-machine
https://www.tutorialspoint.com/digital_circuits/
digital_circuits_finite_state_machines.htm
https://stackabuse.com/theory-of-computation-finite-state-machines/
https://www.jetbrains.com/pycharm/
https://www.waterprogramming.wordpress.com/2015/07/29/pycharm-as-a-python-ide-for-generating-uml-diagrams/
https://www.lucidchart.com/pages/uml-state-machine-diagram

Project Development Team

Morgan Langlais

GUI framework, integration testing, and project manager

Carla Zacarias

Navigation menu, logs page, unit test cases, and SRS requirements

Philip Fitzgerald

Button actions, sequences, SRS requirements, and project description

Anisa Yniesta

Device status displays, code refactoring, testing, and SRS requirements

Stephen C Gross

References, Class diagram, State Chart, About Page

Technology Resources

Coding Languages

Python

The main application was written in Python using TKinter.

Node.js

Framework for the API was created using Node.js, Express, and MongoDB.

Databases

MongoDb

Store device statuses, user information, and logs.

Version Control

GitHub

Serve as code storage and branching for our project.

Communication

GroupMe

Share daily status updates.

Project Evaluation

Successful Processes

We were able to successfully control simulated smart devices.

Processes to Improve

Improvements can be made by connecting to actual smart devices.

Future Development

Features to Add Next Increment

Add Text

Code & Design Reuse Potential

Add Text

Team Professional Development

This is a nice way of saying what skills we could work on as engineers.