Smart House with home assistant

An IoT project

Bijie Qiu, Manvitha Challagundla, Raghvendra Dixit, Shilakha Dawar
Department of Computer Engineering
San Jose State University
San Jose, USA

***Abstract*— We built an app that functions as a central console to manage all devices in a home. We are able to work with devices with different protocols from different companies. This report will explain what technologies and devices being used and how we implemented them.**

I. Introduction

Currently in the market, there are apps for each smart home device, for example, Alexa has one, Google home has one, Philips has one etc. It is inconvenient for customers to manage them. They need to open one app to turn on the light, then if they want to turn on the TV, they need to open another app. To make things easier for the customers, we decided to make one app that can manage and monitor all devices inside one home unit.

Furthermore, most traditional devices at home are not smart, and therefore impossible to be managed through any app. We added smart control to them so that even those devices can be monitored and managed from our app.

We also added three sensors to our project, for the purpose of reading status of our home and running automation of some of the devices such as garage light. The features that make home assistant special and unique is that it allows you to have data without running into the toil of making api calls from multiple sources. It does the work for you keeping all that information and one sensor and maintaining everything just under one yaml file for you to be informed about everything inside and around your home..

II. Software used

Home Assistant:

We used home assistant platform which runs on Python 3 to develop our app. Home assistant can be used to manage, control, monitor, and track all the devices at home.

Arduino Genuino:

We also used Arduino Genuino to code the program for NodeMCU (ESP8266) in C. We also used it for setting up and controlling all of the sensors and LED lights we used in this project.

AppDaemon :

We used HADashboard, which is a dashboard for Home Assistant that is intended to be wall mounted, and is optimized for distance viewing. HADashboard is dependent upon AppDaemon. AppDaemon is not meant to replace Home Assistant Automations and Scripts, rather complement them. For a lot of things, automations work well and can be very succinct. However, there are certain complex automations for which they become harder to use, and appdaemon then comes into its own - Ease of use, Dynamic, power of Python, Reuse.

Since AppDaemon under the covers uses the exact same APIs as the frontend UI, you see it react at about the same time to a given event, so both the interfaces are interchangeable. Calling back to Home Assistant is also pretty fast , specifically if they are running on the same machine. In action, observed latency above the built in automation component is usually more or less a sub-second.

Dashboards can be created in single files or made modular for reuse of blocks of widgets. Dashboards are configured using YAML.

Remotely Voice Control integration to home assistant:

Controlling smart devices through voice control is what's prevailing much. With Home Assistant we can control our smart and non smart devices anywhere whether inside or outside home. All that is required is to trick home assistant to see alexa as emulated hue(Philips bulb) and add it as one of its components and make all the components of home assistant visible to it. This is done by few changes in the configuration.yaml file of home assistant installed on pi. Whatsoever device or automation is there in home assistant,

including sensor, whether, scene or detection, accordingly alexa would control all of them and we can control the home remotely from anywhere.

Similarly, we have attempted for google home where it can control home assistant as well as controlled by home assistant to through other interface such as siri or alexa or the UI itself.

There are some smart devices such as philips hue lamps which can be automatically discovered by home assistant and those devices are integrated in our user interface along with all other devices.

III. Hardware used

We used Raspberry pi, NodeMCU(ESP8266), Regular bulbs (110V), relay switch, mini fan (5VDC), motion sensor, humidity sensor, temperature sensor, LDR sensor, Google Chromecast, Philips hue go, Philips hue lamp, Amazon Echo dot and Google home mini.

How those components are connected and utilized will be explained further in details under the section of architecture.

IV. Demonstration

The first link is a video which shows all of our features in action following the introduction of our project and the setup we built for this demonstration.

https://photos.app.goo.gl/rP7qYjhbAkSnuN722

Below is the link to show that after “motion sensor alert” is opted in, the user will get a text message whenever there is a motion detected in the living room. The user can also disable that function if he or she does not want to received messages, the motion detection can still be visible on the app.

https://drive.google.com/open?id=1kKM1aOlTtyfyVTN8ZzgFQZ5LApBwZAef

V. Approaches

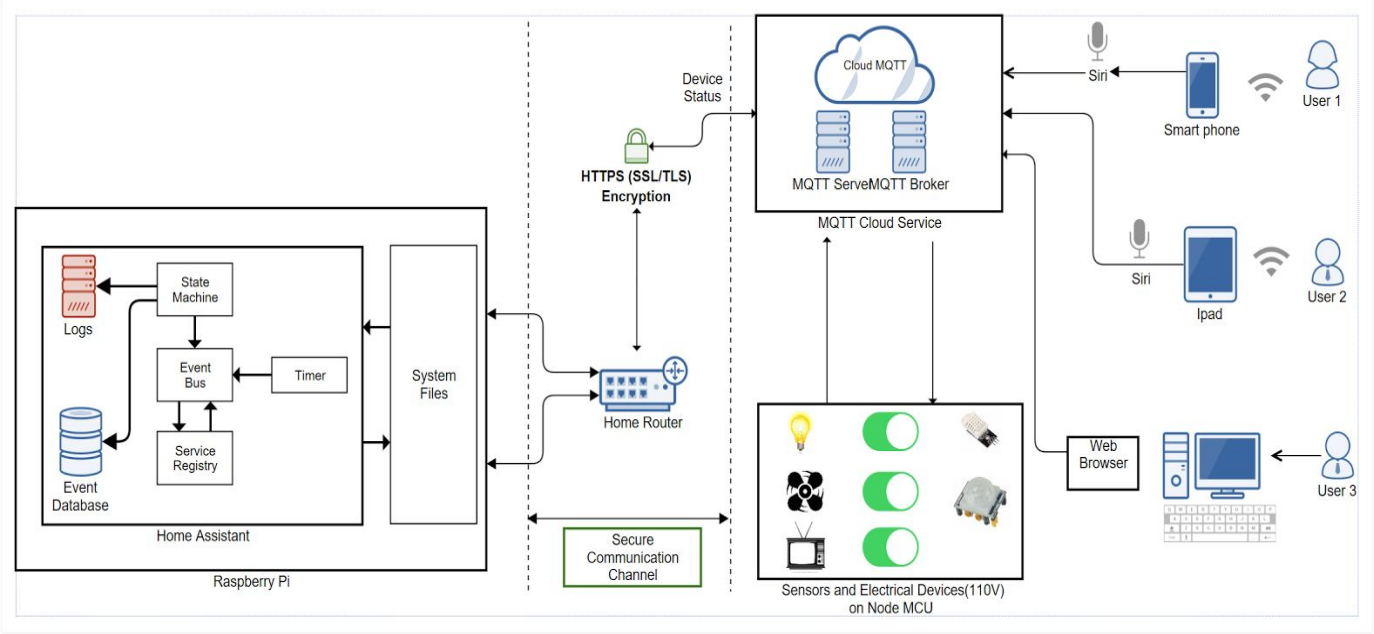
We started with openHAB as our platform. We made an app that is able to control Samsung TV, Chromecast, smart phones, and also get weather reports from Yahoo. The app controls includes volume settings, channel selections, program selections for both TV and Chromecast. We can also monitor the status and track the location of mobile devices. To conclude, using openHAB, we can bind any device which are supported by openHAB, then create an UI element in the app and we can control those devices through the app with any available settings provided by the companies who manufactured the devices.

However, we could not make any progress on relay switch and other sensors. Then we found there is another platform which is similar to openHAB but has more components and more tutorials. We tried on home assistant platform then we found it is easier to work with NodeMcu and other sensors. Therefore we decided to start over and switch to the home assistant platform.

We started with installing home assistant on raspberry pi, then we used NodeMcu and LED, we were able to switch on and of the LED through home assistant. We moved on to smart devices. We tested with fitbit and we were able to retrieve data from fitbit on home assistant.After we included relay switch, which can operate on 110 voltage, we included a mini fan and three light bulbs to our system. We assigned the light bulbs to stand for different devices. We can trigger them from the home assistant app.

We added cell phones, so the app will know where the device is at, and we can get the status of user home or user away from home. We then added multiple sensors. We can capture a signal if there is any movement from the motion sensor. We can read the current temperature and humidity inside the house. For LDR sensor, we can read the light intensity that falls upon on it. After we bring the data available to home assistant, we can run automation for the devices based on the value we received from sensors. For example, we can trigger a lamp to be on when the LDR sensor reaches a threshold value.

VI. Architecture



Above is the architecture diagram of our system. On the left side is the raspberry pi used as the central processing unit. Inside raspberry pi, there is home assistant, which is the platform we used to develop the app. It will also keep the errors in the logs and all the events, which include the history and status of all the devices we configured, into the event database. In the center, we connect raspberry pi via wifi to home router. We also used Duck DNS to obtain a free domain name, then we performed port forwarding on the router, from port 443 to port 8123, in order to have accessibility to internet. Next connected our sensors and electronic devices to the NodeMCU through GPIO pins. In order to communicate from Raspberry pi to NodeMCU, we used cloudMQTT. Therefore if we place NodeMCU in any place that has internet access, and we will be able to control the attached device from anywhere thanks to the cloud service. Lastly on the right side, authorized users can access our app from mobile devices and computers using browsers. In addition, siri and alexa can also use voice command to control all the devices configured in the app.

VII. FEATURES

Crime Data of nearby location:
The crimereport sensor allows one to track reported incidents occurring in a given area. Incidents include anything reported to https://www.crimereports.com/ Your regional emergency services may or may not report data. The sensor only counts incidents from the current day.

Lyft Sensor:
The lyft sensor gives you time and price estimates for all available Lyft products at the given start altitude and start longitude.The attributes are used to provide extra information about products, such as vehicle capacity and fare rates. If an end latitude and end longitude are specified, a price estimate will also be provided. One sensor will be created for each product at the given start location, for pickup time. A second sensor for each product, for estimated price, will be created if a destination is specified. The sensor is powered by the official Lyft API.

Weather Details:
The darksky platform uses the DarkSky web service as a source for meteorological data for your location. The location is based on the longitude and latitude coordinates configured in your configuration.yaml file. The coordinates are auto-detected but to take advantage of the hyper-local weather reported by Dark Sky, you can refine them down to your exact home address. GPS coordinates can be found by using Google Maps and clicking on your home or Openstreetmap.

Notification:
If any action in the living room is detected through the motion sensor, our app can send user a text message if the setting for notification is turned on. We have integrated this with Twilio sms service.

Automation:
When the user gets home, all devices in the living room will be automatically turned on. This event is triggered based on the location data received from the user's cell phone.
When the sun sets, all the lights will be automatically turned on. To know the real time when the place is dark, we integrated yahoo to our app, it can also tell the user about current weather, wind speed, humidity, pressure based on current geographic location.

Switches and Scenes:
We have enabled scenes that capture the states you want certain entities to be. For example a scene can specify that light A should be turned on and light B should be bright red. Switch component Keeps track which switches are in your environment, their state and allows you to control them.

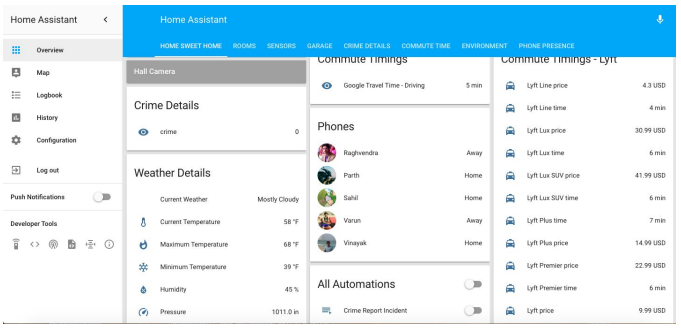
Device tracking:
We have also enabled device tracking for Home Automation. We would need to add the ips of the phones into the configuration file. The status of the phone can be dynamically mapped in real-time. Whenever the phone connects to home network, "Home" state will be mapped, and whenever the phone is disconnected from the network, "Away" status will be mapped.

Maps:
This component offers a map on the Home Assistant to display the location of tracked devices. Zones are the components that allow you to specify certain regions on earth (for now). When a device tracker sees a device to be within a zone, the state will take the name from the zone. Zones can also be used as a trigger or condition inside automation setups.

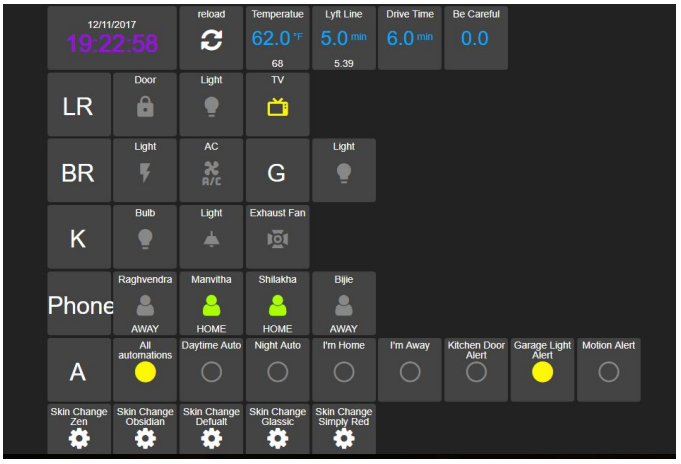
LetsEncrypt: Exposing your Home Assistant instance outside of your network always has been tricky. You have to setup port forwarding on your router and most likely add a dynamic DNS service to work around your ISP changing your IP. After this you would be able to use Home Assistant from anywhere
Lets Encrypt is a free, automated, and open certificate authority (CA). We will use this to acquire a certificate that can be used to encrypt your connection with Home Assistant. It helps using Home Assistant from anywhere.

Brute Force handling:
We have also disabled the brute force possibility by banning the IP to access the URL if it exceeds the allowed number of attempts for login.

User Interfaces:
Below is the user interface from home assistant:



Below is the user interface from Dashboard:



VIII. FUTURE WORK

We should make the hardware part more compact so that we can sell them. We could also put the NodeMCU part in a container so that the user can easily plug it to any of their existing non-smart devices and control them through our app. We could also make an UI that allows the user to customize their console and add any new device they want to include.

REFERENCES

[1] https://home-assistant.io/
[2] https://www.duckdns.org/
[3] http://appdaemon.readthedocs.io/en/latest/TUTORIAL.html.
[4] https://www.modmypi.com/blog/how-to-give-your-raspberry-pi-a-static-ip-address-update
[5] https://www.splitbrain.org/blog/2016-05/14-simple_letsencrypt_on_debian_apache
[6] https://materialdesignicons.com/.
[7] https://mosquitto.org/man/mosquitto-conf-5.html
[8] https://github.com/home-assistant/homebridge-homeassistant