B. User Documentation

Indoor Positioning System

In the current form, there are many requirements that the user will need to accomplish before running the code, this will change and get simpler as we further develop.

-User will need two Debian Based Linux Distributions. Both need to be connected to the same local area network.
-Compile “gcc server.c -o server” on the stationary machine.
-Run “./server” on the stationary machine.
-The client.c source code will need to be edited depending on user specification.
    -There is a portion of the code where the ip address of the remote server needs to be entered(“remote_server.sin_addr.s_addr = inet_addr("127.0.0.1")”), enter the ip address of the computer running server.c.
    -The user must find the name of the interface for his/her computer(that he/she will be walking around with) wireless card using. The user will enter the name in the first system function in the getSignal function(“system("iwconfig wlo1>netinfo.txt")”).
-Compile “gcc client.c -o server” on mobile machine
-Run “./client” on the mobile machine.

Android App

Schedule:

The user must first open the Schedule application. The user must initialize the application by inputting an alarm time and amount of time the user takes on average to get ready. Tap the Map button to let the App find your location. The application interface opens and the user must enter a start location and an end location, or the user can long tap on the map to drop a pin to the destination. Then tap back, then tap prediction. After clicking the submission button, a request will be sent to the Google Maps server. Google will return a JSON file which contains the estimated time of arrival. The implementation of parsing the file and returning the estimated arrival time to the application interface has not yet been implemented.

HVAC:

In the HVAC system the main functionality being worked on now is time to temperature which is based on Nest’s functionality and name. Currently, the android application is not connected to a temperature sensor so the app has a static current temperature of 69 degrees. You can increase or decrease the preferred temperature on the application and press set. The algorithm based on Newton’s Law of Cooling makes a prediction and displays it to the user.

Database

The user should not have to interact with user documentation.

Alexa

Prerequisites for running Alexa on Raspberry Pi:

- Raspberry Pi
  - Pi 3 is recommended, however a Pi 2 is also compatible with an internet connection
  - The Raspberry Pi must have Raspbian installed.
- Power cable for Raspberry Pi
- A MicroSD card
- A USB microphone
- Speakers that have an auxiliary port
- A keyboard and mouse for the Raspberry Pi for setup

Steps to create and use Alexa on Raspberry Pi:

1: Create an Amazon Developer Account
  - To create a developer account, visit developer.amazon.com and then create an account.
  - Navigate to the Alexa tab and select Register. Select Device for the product type.
  - Name the device anything you choose. Then click next.
  - In the Security Profile section, click "Create a new profile". Under the General tab, name your profile and click next.
  - Make a note of the Product ID, Client ID, and Client Secret that the website generates for you.
  - Navigate to the Web settings tab and click the edit button to edit the settings.
  - Click "Add Another" next to Allowed origins and add "https://localhost:3000"
  - Click "Add Another" next to Allowed return URLs and add "https://localhost:3000/authresponse". Then click next.
  - Next, fill out the Device Details tab, it doesn't matter what you put, as long as the required fields are satisfied. Then click next.
  - Finally, select "No" for Amazon Music as it is not compatible with the Raspberry Pi.
2: Clone and Install Alexa
  - Open terminal on the Raspberry Pi and type "cd Desktop".
  - Type "git clone https://github.com/alexa/alexa-avs-sample-app.git" and click enter.
  - Type "cd ~/Desktop/alexa-avs-sample-app" and click enter.
  - Type "nano automated_install.sh" and click enter. This will open the text editor. You will need to enter your ProductID, ClientID, and ClientServer as seen in the Amazon Developer Site.
  - Then enter CTRL+X to save and exit.
  - Now back to the command terminal, type in "cd ~/Desktop/alexa-avs-sample-app" and press enter.
  - Type ". automated_install.sh" and click enter.
  - When prompted, press Y for the various questions, and answer the questions as you see fit. This will install the supplemental software which is necessary for Alexa to run.
3: Run the Alexa Web Service
  - Type in "cd ~/Desktop/alexa-avs-sample-app/samples" and press enter.
  - Type in "cd companionService && npm start" and press enter.

- Doing this will start the Alexa Companion Service and opens a port on your computer to communicate with Amazon. Leave the terminal window open.

4: Run the Sample app and confirm the account

- Now either create a new tab, or open a new terminal window.
- Type in "cd ~/Desktop/alexa-avs-sample-app/samples" and press enter.
- Type in "cd javaclient && mvn exec:exec" and press enter.
- Now a new window will open to ask to authenticate your device. Click yes. A second pop-up will appear in the Java app, which will ask you to click OK. Do not click this yet.
- The browser will open to a page where you login to your amazon developer account. Then click okay and the site will say "device tokens ready".
- You can now click ok in the popup on the Java app.

5: Start your Wake up word engine

- Open a new tab or terminal window and type "cd ~/Desktop/alexa-avs-sample-app/samples" and press enter.
- Type "cd wakeWordAgent/src && ./wakeWordAgent -e kitt_ai" and press enter.
- This will start the wake up work recognition. Now you can say the wake up word, "Alexa", and should hear a beep of acknowledgement through the speaker. Now the Echo is fully functioning and you can ask it anything you'd like.

6: Add Airplay Support

- Open a new tab, or terminal window.
- Type "sudo apt-get install git libao-dev libssl-dev libcrypt-openssl-rsa-perl libio-socket-inet6-perl libwww-perl avahi-utils libmodule-build-perl" and click enter.
- The above command will install the prerequisite software needed for the airplay protocol.
- Press Y when prompted and then wait for the installation to complete.
- Type "git clone https://github.com/njh/perl-net-sdp.git" and press enter.
- Type "cd perl-net-sdp" and press enter.
- Type "perl Build.PL" and press enter.
- Type "sudo ./Build" and press enter.
- Type "sudo ./Build test" and press enter.
- Type "cd .." and press enter.
- Type "git clone https://github.com/hendrikw82/shairport.git" and press enter.
- Type "cd shairport" and press enter.
- Type "make" and press enter.
- Type "./shairport.pl -a AlexaPi" and press enter. This will start the AirPlay service for the Pi and you can stream music to it.

Webapp

1. open bash terminal (requires python 2.7.1+ , and pip)
2. move into project directory/1_code/WebApp/SmartHomeSystemWebApp_Old
3. vitrualenv venv
4. ./venv/bin/activation
5. pip install -r requires.txt
6. python __init__.py

7. Go to localhost:5000