Setting up your door lock

To get started, pull our repository here.

Flask server install

```
pip install Flask
```

RPi Cam Install

1. Attach picamera and enable the camera in the RPi Configuration menu
    a. Raspberry Pi Configuration → Interfaces → Camera: Enable
2. Install RPi Cam
    a. Update the Raspberry Pi
    ```
    sudo apt-get update
    sudo apt-get dist-upgrade
    ```
    b. Clone the RPi Cam git repository and install
    ```
    git clone https://github.com/silvanmelchior/RPi_Cam_Web_Interface.git
    cd RPi_Cam_Web_Interface
    chmod u+x *.sh
    ./install.sh
    ```
    c. The camera web interface can now be viewed at the pi's ip address and the port specified during installation.
3. Configuration
    a. Automatically configured run on boot
    b. Enable motion detection on boot
        i. Edit the configuration file `/etc/raspimjpeg`
        ii. Change `motion_detection` from false to true.
    c. Configure RPi Cam to notify Flask server when motion is detected
        i. Go to web interface. This can be accessed at the pi's ip address at the port specified when installing Rpi Cam *<RPi_ip_address>:<port>*
        ii. Click *Edit motion settings*
        iii. Enter the following in the field *on_motion_detected*:
        `curl --data " http://127.0.0.1:5000/motion`

Setting up the Webserver

Setting up the webserver involves building the website, setting up various command line variables, and starting the node server.

1. Building the website.

    a. To build the website, first install node.js. You can do this by downloading the latest gziped source code from the Node.js download page to your ec2. Unzip the source code and install node.js by running "./configure; make; sudo make install".
    b. Enter into the app/ec2 directory. Install the project dependencies by running "npm install".
    c. Finally, build the website by running "npm run build". After the build, a "dist" directory should appear in your current directory. This is proof that the project has been built without any problems.
2. Setting up command line variables.
    a. We have provided a default ec2.env for your modification. You should register an AWS IOT device to serve as your server, and attach certificates to that thing. Ec2.env should be filled with the real value of the various properties it attempts to export. When done, running "./ec2.env" will set up those properties as environment variables.
3. Starting the node server.
    a. At this point, the node server is ready to run. You can run it by typing "node api.js". You can use tools like forever to daemonize this operation.

Set up a ssh tunnel between your pi and the ec2

The node.js server expects the raspberry pi to set up an ssh tunnel on port 8081 of the ec2 to port 8081 of the Raspberry Pi (the same port as the pi cam server). This is done by running "ssh -R 8081:localhost:8081 <user>@<ec2 ip>" from the Raspberry pi. We have created a hodoor.service file that does this for you. Modify this file so that the ssh tunnel it creates is to your ec2. Once that is done, copy the modified hodoor.service to /etc/systemd/system. Then restart the Raspberry Pi. On startup a tunnel will be created for you.