Design Review

Team E4: Occupancy Monkey
Jake Cerwin, Ryan Huang, Angela Zhang

The Use Case:

Currently there is no efficient way to find an open seat in the CMU University Center 2nd floor, especially during peak hours.

Our Solution:

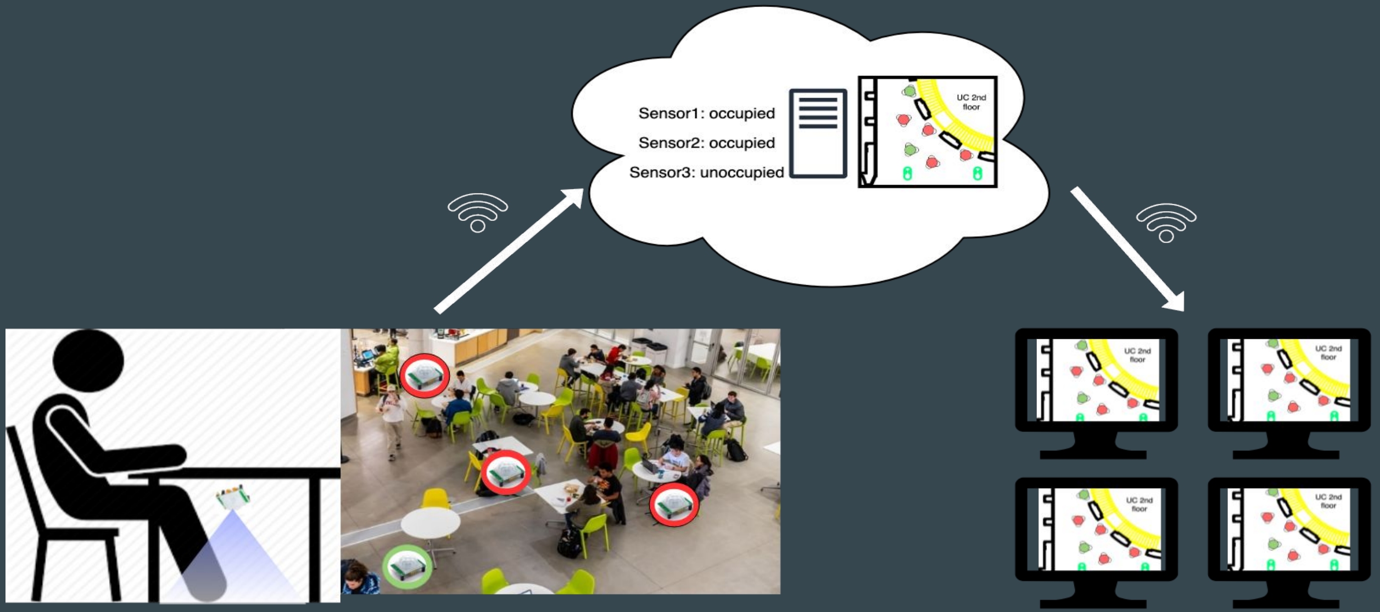
Track table occupancy in real time and display current occupancy on a web application.

Quantitative Use-Case Requirements

- 55+ hour active battery life and 52+ hour deep sleep battery life for hardware system
- Occupancy status is accurately updated on web app within a minute
- Hardware will never go more than 30 minutes without sending information
- Web App Front end will support 25 concurrent users
- Cloud Deployment can support 80 hardware devices at once

Solution Approach: Overview

Each sensor's occupancy data is uploaded to the cloud and made accessible to the web app



Sensor1: occupied

Sensor2: occupied

Sensor3: unoccupied

Fleet of occupancy sensors are deployed under every table and detect whether tables are open

Web App visualizes occupancy information allowing users to view it from anywhere

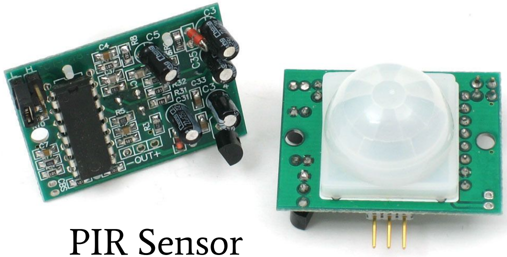
Solution Approach - Hardware

- Battery: 3.7V 6000mAh Lithium Ion, rechargeable
  - High-level power analysis concluded 20J needed (around 3.7V 5400mAh)
- Sensors: PIR and thermal
- MCU with WiFi Communication Module

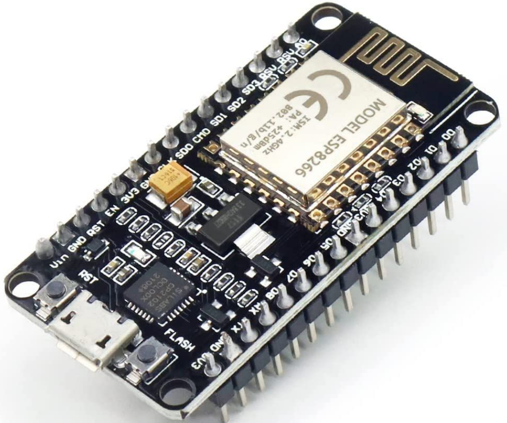
Thermal Sensor

Battery

PIR Sensor



MCU w/ WiFi Module

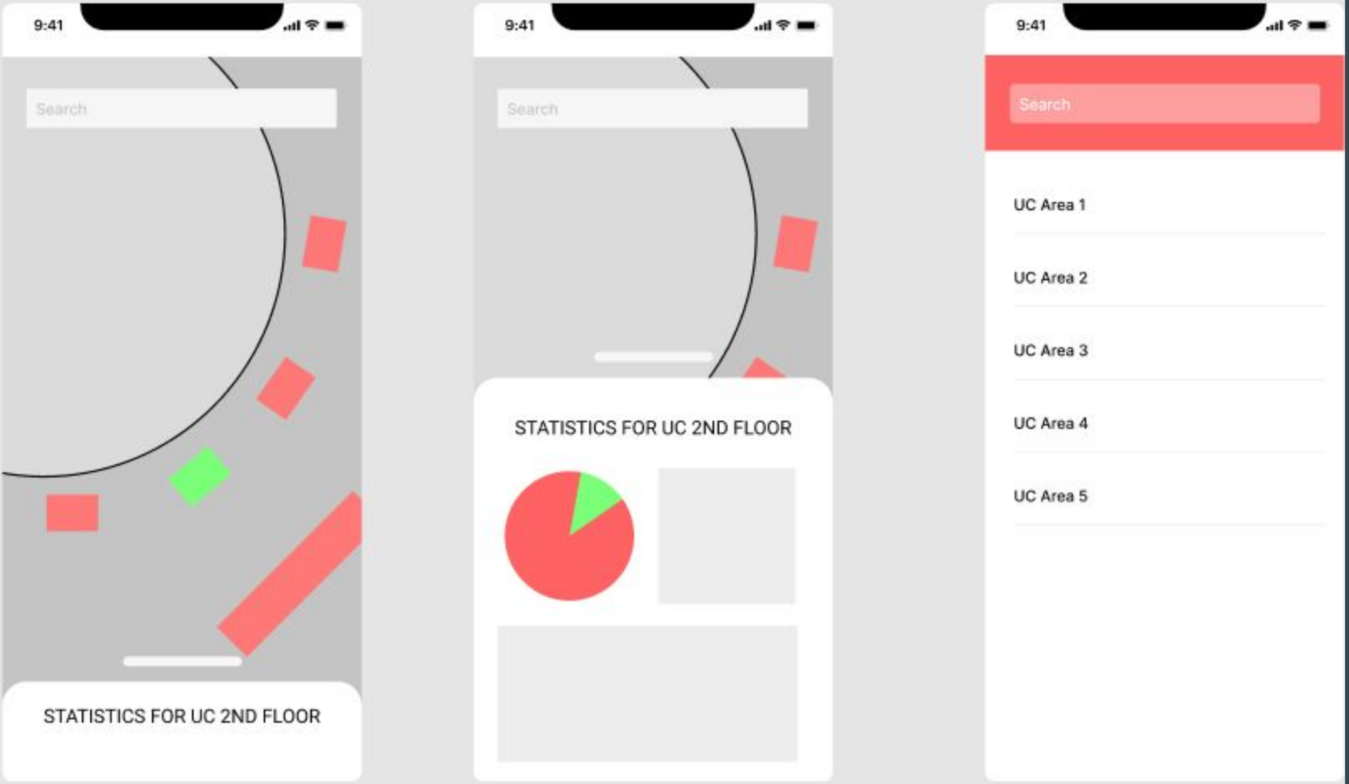


Solution Approach - HW/SW Communication

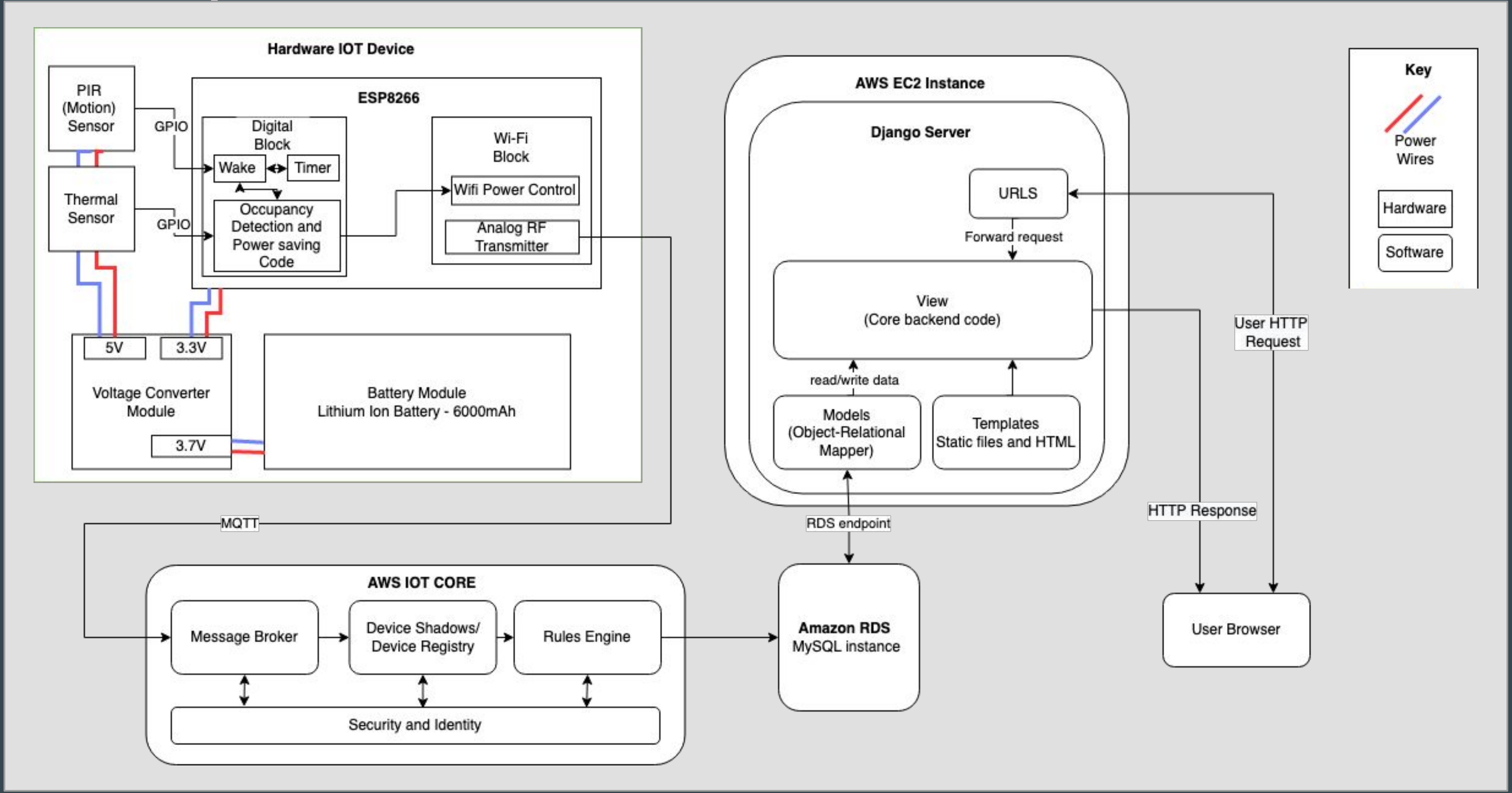
Three modules under AWS :

1. AWS IoT Core
    a. Websocket connection with esp8266
    b. Fleet focus, allows us secure and scalable routing of messages
2. MySQL instance - Amazon RDS
    a. MySQL- Scalable, Quick, Easily integrated with Django DOM
    b. Amazon RDS hosting of our database frees up EC2 server
3. Django Server
    a. EC2 instance hosting a Django Server
    b. Interacts with Amazon RDS for dynamic data on status
    c. Holds all static files and backend code

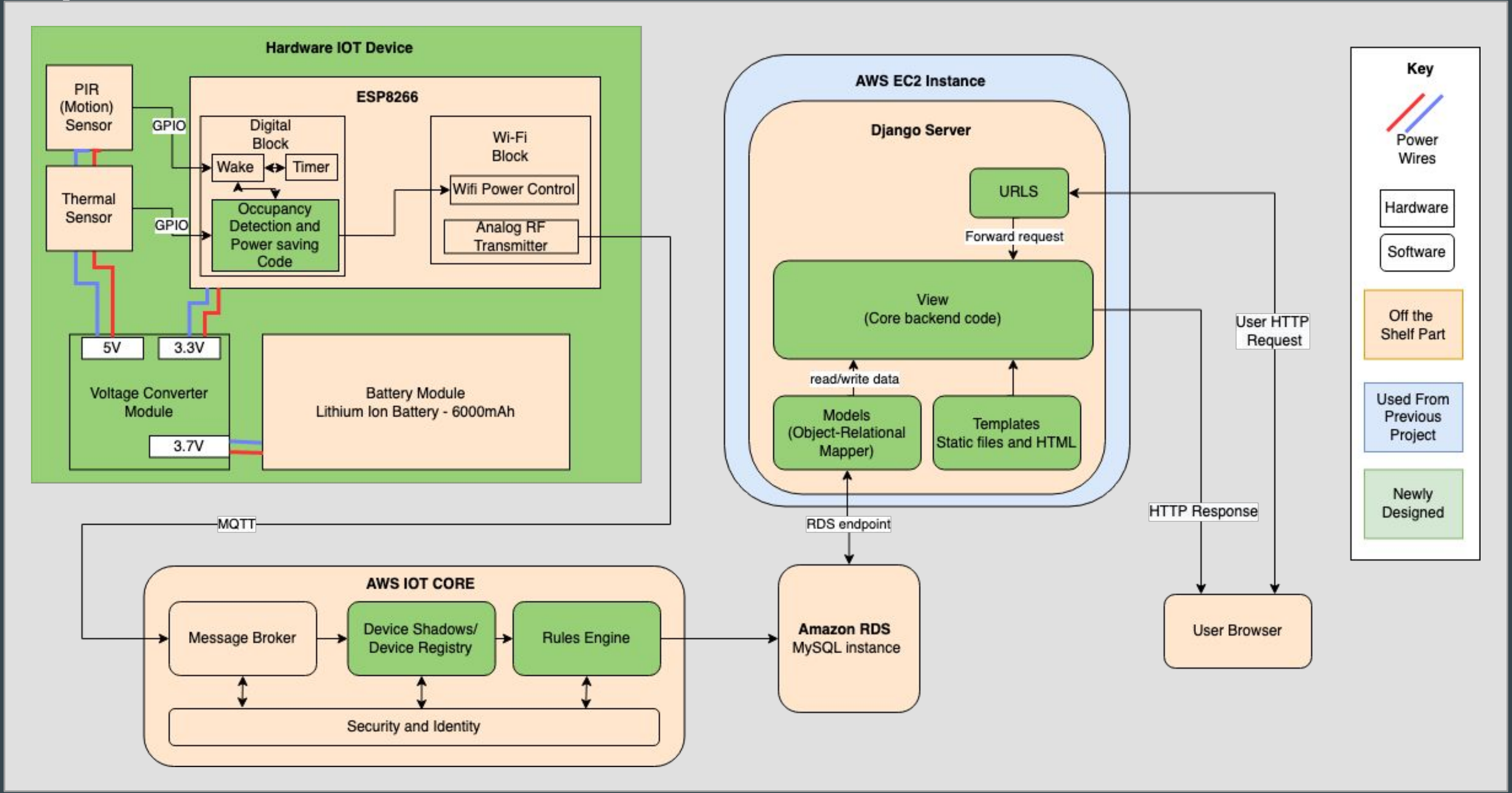
Solution Approach - Web Application



System Specification / Block Diagram



Implementation Plan



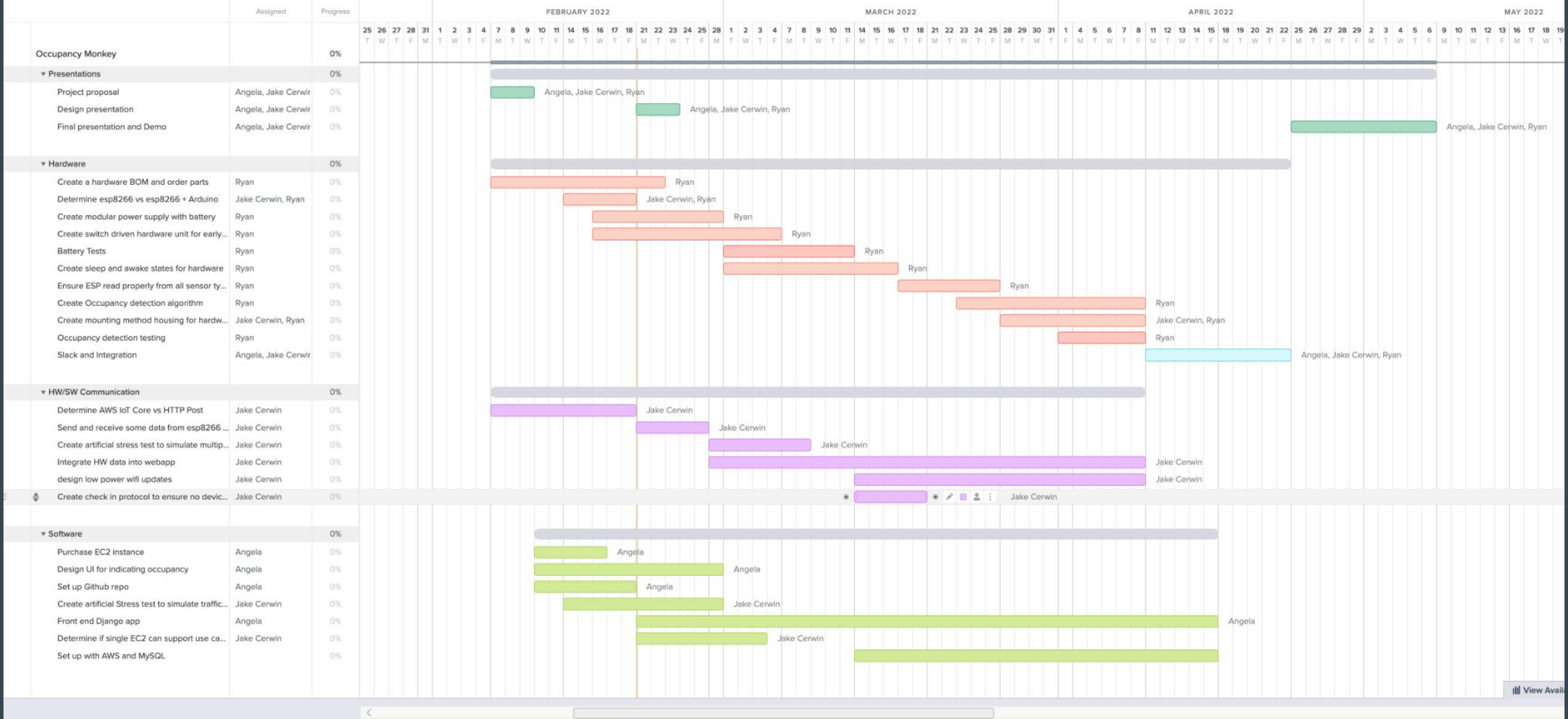
Test, Verification and Validation

Requirement	**Test**	**Passing Metric**	**Risk Mitigation**
Battery Capacity	Run hardware device for a week, simulating real environment	After 55 active hours and 52 deep sleep hours, battery has **>0% capacity**	Reduce active time of PIR sensor (heaviest power consumer)
Component Power Consumption	Measure current into all components during operation	Current draw is **less than or equal to** datasheet numbers	Reduce active time of PIR sensor (heaviest power consumer)
Device updates status on web app within a minute	Link a single hardware device to web app and spoof a status update	Time delay from status change to website update is **within 1 minute**	Cut out the intermediate AWS service that is causing the greatest slow down

Test, Verification and Validation

Requirement	Test	Passing Metric	Risk Mitigation
80 hardware device can be supported simultaneously	Hardware can successfully update cloud DB	A Hardware sent message propagates to DB in less than a minute	Move DB to EC2 and send signals directly from esp8266
80 hardware device can be supported simultaneously	Create a test program to send several devices worth of information	Record number of simulated nodes at which data is first lost	Add an intermediary device to aggregate status updates of a portion of IoT fleet then send combined data to cloud
25 users can be supported on web app concurrently	Create a test program to simulate web traffic	Record EC2 CPU credit balance loss	Restructure backend to minimize unnecessary computing

Project Management



	Assigned	Progress
Occupancy Monkey		0%
▾ Presentations		0%
Project proposal	Angela, Jake Cerwir	0%
Design presentation	Angela, Jake Cerwir	0%
Final presentation and Demo	Angela, Jake Cerwir	0%
▾ Hardware		0%
Create a hardware BOM and order parts	Ryan	0%
Determine esp8266 vs esp8266 + Arduino	Jake Cerwin, Ryan	0%
Create modular power supply with battery	Ryan	0%
Create switch driven hardware unit for early...	Ryan	0%
Battery Tests	Ryan	0%
Create sleep and awake states for hardware	Ryan	0%
Ensure ESP read properly from all sensor ty...	Ryan	0%
Create Occupancy detection algorithm	Ryan	0%
Create mounting method housing for hardw...	Jake Cerwin, Ryan	0%
Occupancy detection testing	Ryan	0%
Slack and Integration	Angela, Jake Cerwir	0%
▾ HW/SW Communication		0%
Determine AWS IoT Core vs HTTP Post	Jake Cerwin	0%
Send and receive some data from esp8266 ...	Jake Cerwin	0%
Create artificial stress test to simulate multip...	Jake Cerwin	0%
Integrate HW data into webapp	Jake Cerwin	0%
design low power wifi updates	Jake Cerwin	0%
Create check in protocol to ensure no devic...	Jake Cerwin	0%
▾ Software		0%
Purchase EC2 instance	Angela	0%
Design UI for indicating occupancy	Angela	0%
Set up Github repo	Angela	0%
Create artificial Stress test to simulate traffic...	Jake Cerwin	0%
Front end Django app	Angela	0%
Determine if single EC2 can support use ca...	Jake Cerwin	0%
Set up with AWS and MySQL		0%