<This document is ready to be used>

Current version: **10-23-20**

Final Project: End-to-End IoT System

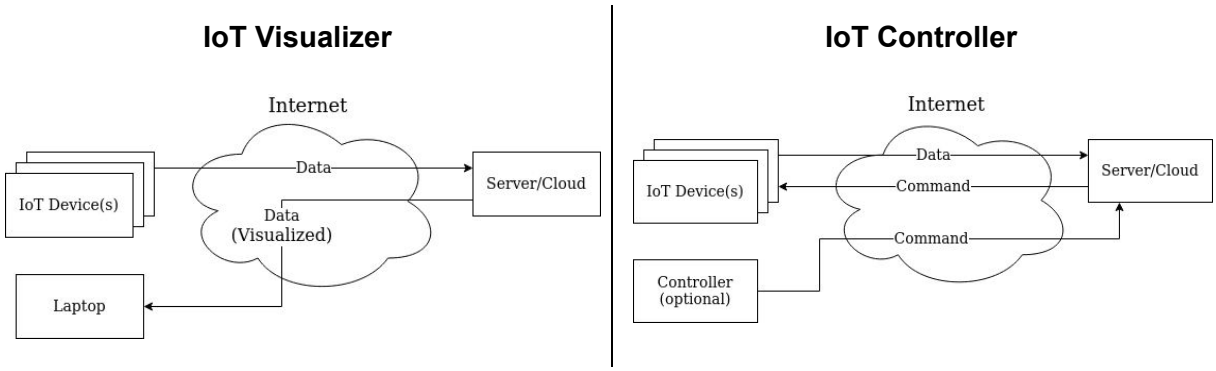
Start Date: **Oct 27, 2020**
Due Date: Code + Video Submission + Writeup Before 11:59 PM on **Nov. 13, 2020**
Teams of 1 or 2 are allowed for this project

Introduction

Over the course of EE-250 we've taught various concepts such as signal processing, TCP/IP, socket programming, and MQTT. In order to solidify your understanding of these topics, we use labs to apply these concepts into some concrete artifact. To this end, it is fitting that the final project incorporates a little of everything. This will also be beneficial as a review of the concepts that we have taught and serve as a lesson on how they connect together.

Objective

The objective of this lab is for you to build your own end-to-end IoT system using the tools and concepts we've taught you over the course of this project. We define an end-to-end IoT system to be one or more physical nodes connected to a central node for data collection/processing and visualization (i.e. at least 2 nodes must be involved). An alternative implementation can be one or more physical nodes which transmit data to and are controlled by a central node. The nodes may be your laptop, raspberry pi, arduino, or even your smartphone. You're welcome to get resourceful and play around with any hardware you may have laying around at home or purchase other components as desired, though that is not necessary to achieve a full score.



Above we illustrate the two different implementations of an end-to-end IoT system. The first being a visualizer where the connected IoT device(s) send data to a server. From there a user on a laptop can access the data and visualize it on their laptop. This should sound very similar to what you did in the monitoring and visualization lab. The main difference is that you will need to configure it to generate your own data from your IoT device(s), handle real time data, and potentially do some signal processing!

The second type is an IoT controller where data is transmitted from the IoT device(s) to a server and the server may or may not respond back with a command. Optionally, a controller may connect to it and command it to do something. If you recall, this is similar to what you did with MQTT lab.

It's worthwhile to note that your server can vary significantly from others depending on your application. It could be a remote desktop running a python TCP server, or it could be an AWS EC2 instance running Influxdb. It is up to you to decide what technologies are most applicable to your IoT system.

We will leave it up to you which implementation you go with and what it will do. Part of the grading rubric will be on originality so get creative and see what you come up with. *

* Due to the current situation with COVID-19, we understand that some of you will have restricted access to hardware resources such as sensors and breadboards. If you are in this position, we will allow you to use your laptop as an IoT device. You will also be allowed to use data collected from online sources in place of sensor data.

Requirements

1. Two or more physical nodes (such as laptop, rpi and cloud)
2. Data collection
    - Can be from actual sensor or from a 'virtual' sensor
    - A 'virtual' sensor could generate simulated data or use a public data API
3. Signal or data processing (in real time).

    - While our lecture focused on conversion to the frequency domain, it is likely easiest to do some filtering or event detection in the time domain.
4. Node-to-node communication
    - Data must be transferred between the nodes (meaning you can't do everything on a virtual machine)
    - You can use any cloud components, such as AWS IoT, EC2 etc.
5. Visualization and/or control element

If you have any concern about your choices meeting these requirements, just run it by the TA: Xiangchen or Sampad.

Helpful Resources (For reference only)

To help you get started with this project, we've compiled a list of some resources that we feel you might find useful:

- Turn your android into an IoT device
- Use python to get sensor data from your laptop
- Connecting your raspberry pi to your home network
- Expose your local device to the internet
- Mosquitto, MQTT broker
- AWS IoT core (communication)
- AWS Lambda (processing, function as a service)
- AWS S3 (storage)
- AWS EC2 (general computing)
- Web Scraping for data
- Using RESTful APIs

Additionally, to get you in the right mindset for this project here are some examples that we have come up with for this project. Please do not copy from these examples, they should only serve as a reference. Remember, we will be counting you on originality!

- An IoT sensor system where audio data is recorded, processed using FFT, and then sent to server to be visualized
- An IoT system where a RPi is controlled to set the lighting in the house based on time of day and ambient light
- An IoT system where keystrokes on a laptop are logged to a server and then a graph displaying word frequency is tweeted out

Final Deliverables

All submissions will be done through Vocareum. **Please ensure you clearly address the items in the Grading Rubric presented in the following section.**

Here is what required:

- Source code(s) - **11:59 PM, Nov. 13**
  - May be written in multiple programming languages as needed
- README.txt **(Video Demonstration Link)** - **11:59 PM, Nov. 13**
  - Team member names
  - Link to view demo online
  - Instructions on how to compile/execute program(s)
  - List of any external libraries that were used
- Writeup PDF - **11:59 PM, Nov. 13**
  - 2 pages (at most)
  - Also include team member names
  - A short description of your project and its functionality
  - A block diagram that shows interaction between components
  - Briefly describes components, protocols used, key processing techniques, etc. along with important implementation/design choices.
  - Reflection on limitations and their causes as well as lessons learned.

Grading Rubric

All files are to be submitted via Vocareum in teams.

Points	**Description**
Demo	
2	Clearly identifies each node in the network.
2	Explains how data is collected and sent to the server node. Mentions specific protocols, sensors, or data sources that are used.
2	Describes how the data is processed (either on server or client). Mentions a specific signal/data processing technique.
4	Demonstrates correct operation of project features (i.e. assess the level to which the project works correctly and achieves the intended behavior)
Code	
3	Code correctness (no syntax errors/code compiles, sufficiently bug-free for the assignment, etc.)

2	README.txt file contains team member names, link to video demo, instructions to compile/execute your code, and list of any external libraries used
Writeup	
2	Clear description of what your IoT system is trying to achieve
1	Block diagram clearly shows relevant components and interactions
2	Description of components, platforms, protocols used, and processing/visualization techniques
2	Reflection, discussion of limitations that demonstrates insights to their cause and possible remediation, lessons learned
3	X-Factor: Project originality and/or difficulty

Total: 10 + 5 + 10 = 25