Our project aims to find suitable testing approaches for verifying specifics within Carrier’s IoT (Internet of Things) systems and to explore new ways of testing IoT products, where the IoT infrastructure allows for advanced services that can access and analyze remote data, mobile applications, and web-based user interfaces.

In order to develop a testing methodology for IoT systems, we need to design and build a simple IoT system to run tests on. For our system, we utilize ThingSpeak as our cloud platform due to its integration with MATLAB. Data from the sensors is captured from the circuit and sent to ThingSpeak. An Android application reads and displays these values in real-time, and can control the actuators on the circuit through the cloud. With this system, we are able to perform various performance, stability, and usability testing/analysis using MATLAB and other software tools, with an end goal of having an established verification methodology for IoT systems in general.

**Objective**

Identify the goals of IoT verification, explore new ways of testing IoT products, and synthesize a process (with the necessary tools) to be applied to IoT system verification.

**Performance**
- Measure delay of sensor values updating from circuit to ThingSpeak
- Time for actuators to turn on/off after sending the signal from the app

**Stability**
- How much of the data is being sent to ThingSpeak?
- Testing under conditions where sensors and internet signal could potentially fail
- Can the system operate with many different sensors sending data simultaneously?

**Interoperability**
- Checking for proper interactions between the circuit, ThingSpeak, and the app

**Usability**
- Are commands functioning properly? Is the sensor data being displayed correctly?
- Is the app convenient for the user?

---

**Results (1 hour)**
- Python script replaces Android app
- Creates a request to the ThingSpeak server with our sensor data by simulating a certain number of users
- Final data is written to CSV file for analysis
- Analysis methods include:
  - Performing statistical analysis on the historical data with visualizers showing current statistics on the sensors and expected output over time
  - Out of range errors are displayed when at least one sensor value is out of the accepted range
  - Time for actuators to turn on/off after sending the signal from the app
  - Comparison of performance with less users

**Results (250 users)**
- Out of range errors are displayed when at least one sensor value is out of the accepted range
- Average response time was 5680.85 ms. The error rate and response time is much lower when tested with less users.

**Final data**
- Performance and stability testing on the ThingSpeak cloud service
- Creates a request to the ThingSpeak server with our sensor data by simulating a certain number of users
- Results (250 users): Load test was run for roughly 2250 samples (9 times) for the sensors where about 11.03% was the error rate. The average response time was 5680.85 ms. The error rate and response time is much lower when tested with less users.

---

**Discussion of Tools and Results**

- **Python**
  - Python script replaces Android app
  - Access circuit and ThingSpeak API simultaneously
  - Compare values in real-time to calculate latency and detect errors
  - Final data is written to CSV file for analysis
  - Results (1 hour): Latency ranges from 1.36s to 3.09s. Average is 1.56s. Rate of missing data (error rate) is 0.92%

- **MATLAB**
  - Analysis is limited to the sensor data on ThingSpeak
  - Analysis methods include:
    - Having an error message sent out when sensor data is out of a certain range
    - Performing statistical analysis on the historical data with visualizers showing current statistics on the sensors and expected output over time
  - Results: Statistics such as min, max, median, mean, and standard deviation are displayed.

- **Apache JMeter**
  - Performance and stability testing on the ThingSpeak cloud service
  - Creates a request to the ThingSpeak server with our sensor data by simulating a certain number of users
  - Results (250 users): Load test was run for roughly 2250 samples (9 times) for the sensors where about 11.03% was the error rate. The average response time was 5680.85 ms. The error rate and response time is much lower when tested with less users.