Team Information

Team name: PlanTech
Team number: 1
Members:

- Brent Pereira (1673517)
- Hieu Dao Le Duc (1924891)
- Ishpal Mattu (1951633)

Project Description

The PlanTech application allows Fleet Managers and Farm Technicians to more easily control and regulate container farms. This is done through the use of dashboards that are personalized for each role. With these dashboards, users can read data that is being retrieved from sensors and control devices that are present in the containers. The hardware used in the containers includes a chainable RGB LED, a fan, a GPS, an accelerometer to calculate the containers angles and vibration, and buzzer to notify workers if the container is in an alarming state, a magnetic door switch, a micro servo to lock the door, as well as a water, moisture, temperature and humidity, motion, and sound sensor.

IoT System

Sensors

- D5 = Fan
- D16 = Door
- D22 = Motion Sensor
- D24 = RGB Led
- D26 = Temperature & Humidity Sensor
- A0 = Liquid Sensor
- A2 = Moisture Sensor
- A4 = Sound Sensor
- PWM = Mirco-Servo
- UART = GPS

Device-to-Cloud

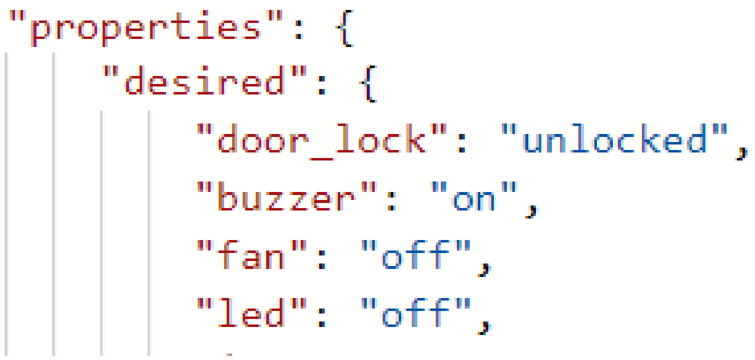
Communication method of choice: **Device Twin**

By using Device Twin, we can persist the requests from the backend app and the responses from the device even when one of the two is disconnected from the Internet. On the other hand, we cannot do the same thing using direct methods. Moreover, we already set up the usage of device twin for telemetry interval so we would like to keep the communication between the device and the app consistent.

Device Twin keys & values

- “telemetry_interval”: number
- “led”: “on/off”
- “fan”: “on/off”
- “buzzer”: “on/off”
- “door_lock”: “unlocked/locked”

Example of using Azure Portal to set desired properties:



```
"properties": {
    "desired": {
        "door_lock": "unlocked",
        "buzzer": "on",
        "fan": "off",
        "led": "off",
```

Contributions

Whole team	● Set up format of telemetry message ● Send telemetry messages ● Respond to desired properties
Brent	● Read/Set fan state ● Read temperature and humidity ● Read pitch and roll angles ● Read vibration level ● Motion detection ● Read noise level
Hieu	● Read/Send GPS data ● Read/Send/Control buzzer state ● Read/Send door state ● Read/Send luminosity level ● Send device id with telemetry message
Ishpal	● All hardware implementation for water level sensor, soil moisture sensor, servo (imitating door lock) and rgb led. ● Read water level data ● Read soil moisture data ● Read/Set door lock state ● Read/Set light state

Getting Started

Clone the repository

```
git clone https://github.com/jac-final-project-w22/course-project-plantech.git
```

Install grove.py

```
https://github.com/Seeed-Studio/grove.py
```

Install PySerial & pynmea2

```
pip install pyserial
pip install pynmea2
```

Install geopy

```
pip install geopy
```

Install Seeed Python ReTerminal

```
sudo pip3 install seeed-python-reterminal
```

Install GPIO Zero

```
sudo apt update
sudo apt install python3-gpiozero
```

Run the container file

```
/bin/python3 /home/pi/Documents/course-project-plantech/Hardware/container.py
```

In `course-project-plantech` directory, create a file called "**.env**". This file will contain all Azure IoT Hub device related connection strings. For more details, please look inside **.env.example** located in the same directory.

Keys needed inside **.env** file:

```
      IOTHUB_DEVICE_CONNECTION_STRING
      IOTHUB_DEVICE_ID
```

Mobile App

App Purpose

The purpose of the application is to facilitate the jobs of both Fleet Managers and Farm Technicians. This is done by allowing Fleet Managers to monitor the conditions and states of all of their deployed containers from a remote location, and by providing Farm Technicians the ability to control devices that are present in the containers. This ultimately ensures that every plant, no matter their needs, is grown to their full potential.

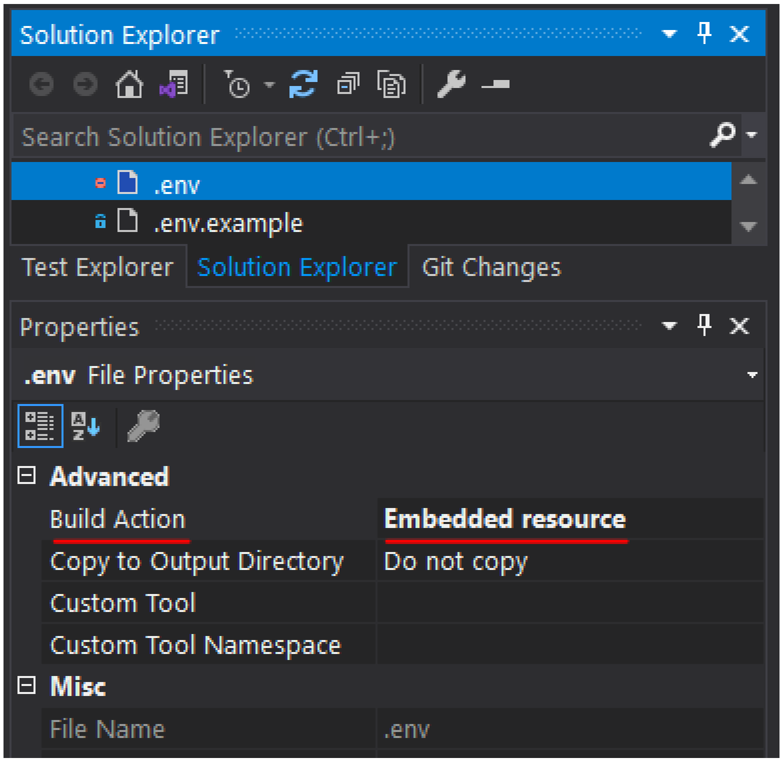
Getting Started

In `course-project-plantech/Mobile_App/PlanTech/PlanTech` directory, create a file called "**.env**". This file will contain all Azure IoT Hub connection strings. For more details, please follow the guide in **.env.example** located in the same directory.

Keys needed inside **.env** file:

```
    EVENT_HUB_COMPATIBLE_NAME
    EVENT_HUB_CONNECTION_STRING
    EVENT_HUB_COMPATIBLE_ENDPOINT
    STORAGE_CONNECTION_STRING
    BLOB_CONTAINER_NAME
```

The created **.env** file should be built with the project as an **Embedded Resource**.



Advanced

Build Action	Embedded resource
Copy to Output Directory	Do not copy
Custom Tool	
Custom Tool Namespace	

Misc

File Name	.env

App Functionality

Role Selection

Upon opening the application, the user is presented with the role selection page where they can choose whether they are a Fleet Manager or a Farm Technician. After making the choice, they are then navigated to the container list page where they can view all of the containers that are assigned to their chosen role.

Container List

The container list page allows the user to view all of their containers in one place, as well as some basic information about each container. This includes the container's name, location, deployment state, and whether it’s in an alarming state or not. The user can then select any of the containers in the list to navigate the dashboard page for that specific container.

Dashboard

On the dashboard page the user is presented with tiles that represent each of the container hardware devices that are relevant to the users position. On each of these tiles, the user can view the current value that is being retrieved by the device's sensor, and they turn on and off the devices that have the ability to be controlled. Upon clicking on any of the tiles, the user will be navigated to the chart page of the respective sensor.

Charts

The chart pages are updated in real-time and they are designed to allow the user to view and track data changes over time for each container sensor. From the container list page, the user also has the option of accessing the comparison chart page through the side menu. On this page the user can compare all sensor data in real-time across all of their assigned containers.

Test Payload

```
{"geo_location": {"address": "John Abbott College, Sainte-Anne-de-Bellevue",
"angles": {"pitch": 0.0, "roll": 0.0}, "vibration": 1166}, "security":
{"luminosity": 400, "motion": True, "sound": 15, "door_is_closed": True},
"plant": {"water_level": 300, "soil_moisture": 500, "humidity": 40.33,
"temperature": 20.45}}
```

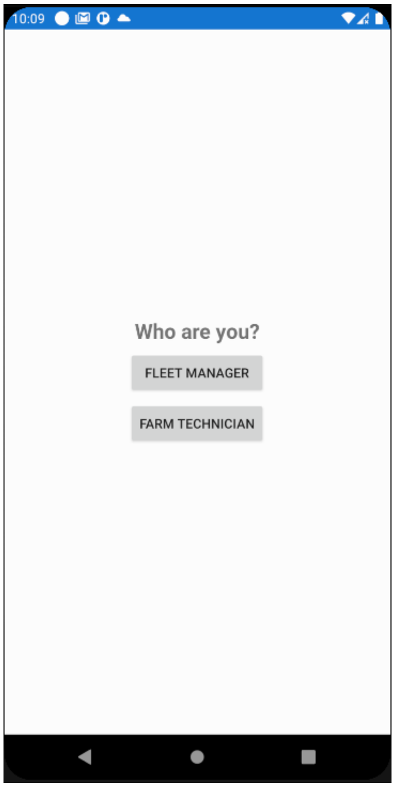
```
message.address = "John Abbott College, Sainte-Anne-de-Bellevue"
message.angles = {'pitch': 0.0, 'roll': 0.0}
message.vibration = 1166
message.temperature = 20.45
message.humidity = 40.33
message.water_level = 300
message.soil_moisture = 500
message.luminosity = 400
message.motion = True
message.sound = 15
message.door_is_closed = True
```

Test Reported Properties

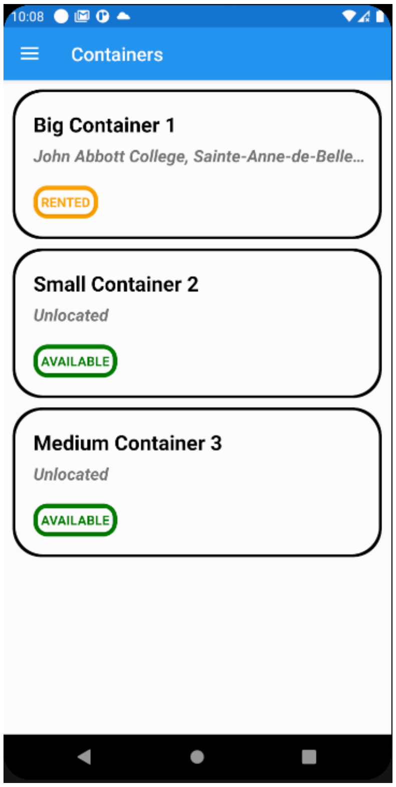
```
patch = {}
patch[self.__TELEMETRY_INTERVAL_KEY] = 10
patch[self.__LED_KEY] = "on"
patch[self.__FAN_KEY] = "on"
```

```
patch[self.__BUZZER_KEY] = "on"
patch[self.__DOORLOCK_KEY] = "locked"
```

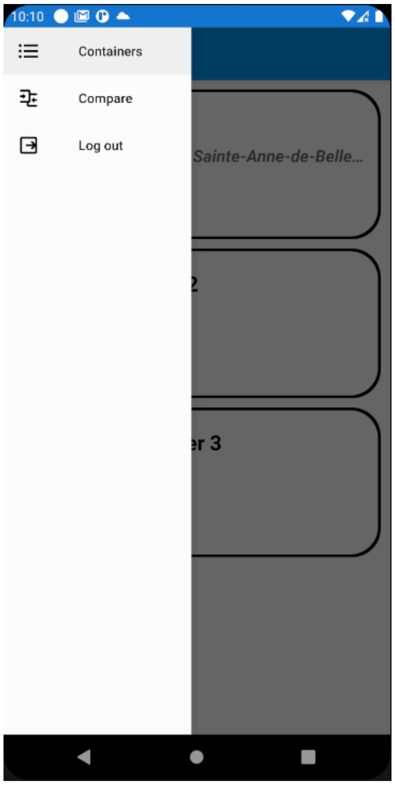
App Snapshots



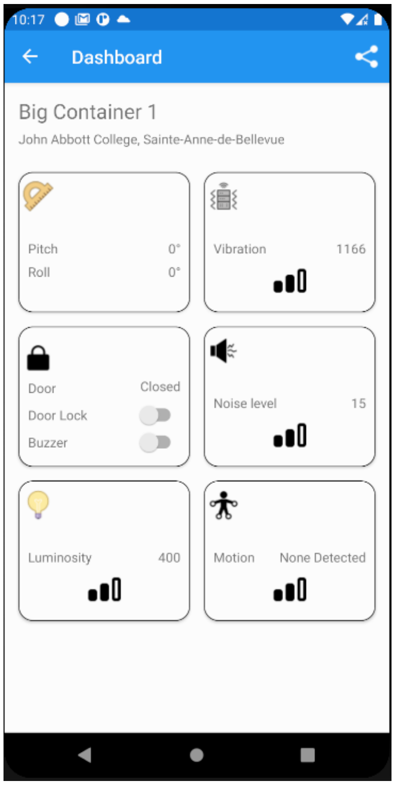
Startup page where the user can select their role



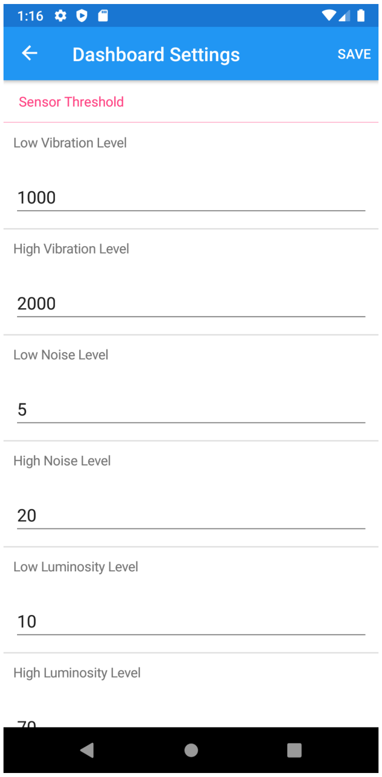
Container list page to view all containers



Side menu to access the compare page and to be able to logout



Dashboard to view and control devices and sensors



Low Vibration Level

1000

High Vibration Level

2000

Low Noise Level

5

High Noise Level

20

Low Luminosity Level

10

High Luminosity Level

70

Settings page to set container thresholds

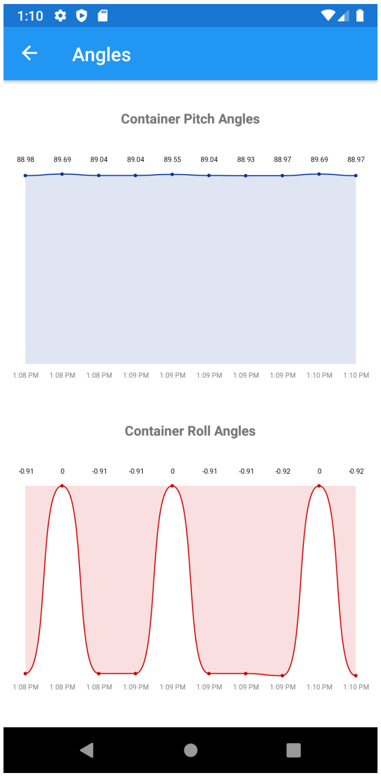


Chart page to view device data over time

Future Work

We are excited by the result of the PlanTech application, however moving forward there are still additional features that we would like to implement in order to further augment the user experience and facilitate their work. We would like to secure users accounts by adding authentication, make it easier to know exactly where all deployed containers are by integrating a map page, and implement 3D models of the containers themselves to give users a better idea of each container's current angles.

Contributions

Whole team	● Set up Azure IoT connection ● Receive telemetry messages ● Retrieve/Update device twin
Brent	● Implemented the applications side menu ● Create the chart pages for each device and allowed for it to be updated in real-time ● Create the comparison page and allowed for it to be updated in real-time ● Stored telemetry data in database and created repository methods for accessing data
Hieu	● Import .env ● Create role selection page ● Create containers list page for fleet manager and farm technician where alarming/warning state is updated in real-time ● Implement “No Internet Connection” banner ● Notifications for alarming/warning states
Ishpal	● Created fleet manager & farm technician dashboard page and binded telemetry data to it. ● Setup database ● Implemented share dashboard feature for fleet manager & farm technician ● Created dashboard settings page with ability to set custom threshold values ● Implemented “signal icons” on some dashboard tiles to use the threshold values & update icon in real time.

Work was divided at the beginning of each sprint. This was done by using planning poker to assign each story a number of points and a priority number. Each team member was then allocated the same number of story points to ensure that work was divided evenly for each sprint. As the sprints went on, team members worked on stories that revolved around that prior tasks, as well as stories that complimented their skillset the most.