Sensors Options:

-Nitrogen, Phosphorus, Potassium

-Calcium, Magnesium, Sulfur\*\*\*

-Moisture

-pH

-Temperature

-Humidity

-Light sensor (outside soil)

-Temp and Humidity (outside soil)

Nitrogen, Phosphorus, and Potassium:

<https://www.amazon.com/gp/product/B08MXXSP59/ref=as_li_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=B08MXXSP59&linkCode=as2&tag=shophow2elect-20&linkId=791f967ac266cf14f2c69f3f299278b4>

Actuators:

-Watering device

-Fertilizer dispenser

Custom PCB Components:

(No RasPi)

-MCU

-Wifi Module

-Connectors for probe sensors that go directly into soil.

(General)

-Voltage regulators (resistor voltage dividers?)

Project Outcome #2:

-”Information collected from these sensors will be transmitted to a central IoT platform, where advanced algorithms will analyze it to ascertain the plants’ precise needs. Based on this analysis, the system will automatically execute tasks such as watering and fertilizing, using actuators and control mechanisms.”

* To me (Chase), this sounds like the actual processing of data and determination of what the device needs to do will be happening on the server, rather than a RasPi.
* We need to determine as a group how this will be split up. What tasks will be delegated to the device vs. the server? Should the server just relay data to the app? Should the server also handle and process the data, or would that be better to do on the device?
* Hardware: making the hardware cheaper would be better / generally desired. We are already going to have a server, so having a cheaper processor would make the individual device cheaper -> able to sell more (hypothetically).

Potential PCB Implementations:

* Carrier board for RasPi
  + With all these sensors and actuators, we might not have enough pins directly on the RasPi for all the devices.
  + Can provide a separate battery (or AC adapter) power and provide to all devices, sensors, and RasPi.
  + Will need to have voltage regulators for the different devices. Highly unlikely they all take 3.3V or 5V.
    - Determine whether a voltage regulator is necessary or if resistor voltage dividers would get the job done. Research pros / cons to each. (Resistor voltage divider would definitely be cheaper)
  + Utilizing a RasPi would probably vastly simplify the debugging process. We would know that the RasPi works correctly, so hardware issues would be easier to identify / narrow down for specific sensors / actuators. With a custom setup, there would be the potential for much larger bugs / issues.
* RasPi + Arduino(s)
* Custom Dedicated PCB with MCU
  + Closer to a real world product. A final product would not have a RasPi or Arduino inside. Would be designed to only include what is necessary, as opposed to having many unused features of the RasPi.
  + Would provide the EEs with important PCB design experience.
  + Ability to use cheaper sensors.

-EEs could do the low-level MCU programming if the PCB is custom to take load off of CprEs.

-The server will be doing the data analysis and algorithms.

-The device will only need to take in commands from the server and send data, allowing for a potentially cheaper MCU.

-A custom PCB is required at least for powering the actuators. This could be used with RasPi, or custom selection.

-Individual / separate fertilizers for each nutrient as opposed to a single main one.

-RasPi carrier board first semester. 2nd semester custom PCB using no RasPi, only take functions that were needed on RasPi. Make it a seamless transition. Only use device for data transmission and receiving data, so that the higher levels of this project are unaffected. This would still allow us to fall back on the RasPi application of the fully custom PCB fails.

-**Tell team we only need one microcontroller, so nothing like CAN bus or EtherCAT.**

**-Only application possibly for sending info to server. Would CAN bus be used for this.**

**-This would be delegated to the CprE team.**

-Research voltage regulators vs. resistor voltage dividers.

-Always do fertilizing before watering to allow pelletized fertilizer to dissolve.

-CAN bus used for communicating between multiple microcontrollers.

-Single “backbone” wire to all microcontrollers, takes specific sequence of data for individual microcontroller, so wires from microcontroller to microcontroller isn’t needed.

-Alternatively, similar to how EtherCAT can be used.

<https://www.amazon.com/dp/B08M5PHQZ7/ref=sspa_dk_detail_2?psc=1&pd_rd_i=B08M5PHQZ7&pd_rd_w=48h4S&content-id=amzn1.sym.f734d1a2-0bf9-4a26-ad34-2e1b969a5a75&pf_rd_p=f734d1a2-0bf9-4a26-ad34-2e1b969a5a75&pf_rd_r=03GAKMMFYRZ4SYQ0R79H&pd_rd_wg=D5ZVD&pd_rd_r=13b662dc-ac4b-4644-8ebb-25654b1fc289&s=lawn-garden&sp_csd=d2lkZ2V0TmFtZT1zcF9kZXRhaWw>

