ECE 196 FINAL PROJECT PROPOSAL

Ahmad Milad

06/08/2020

Smart Farmer

Introduction / Motivation / Project Description

I have been always fascinated by plants, so for my final project I decided to build an automated device, using Raspberry Pi, that will look after my indoor plants. As we know, growth of a plant is directly affected by its surrounding environment. By manipulating that environment, we can grow healthier plants. For instance, by maintaining a sustainable relative humidity level, temperature level, water cycle, and light hours we can maximize the health of our plant.

My project is intended to help individuals who love growing plants but luck the necessary experience to properly look after their plants. However, in the broader perspective, my project has the potential to be used in places where growing plants in a normal manure is not possible. For instance, a concept based on my project design can be used in the space station, or in a future loner mission. Working on this project will help me reflect on the concepts that I have learned during this quarter. Most importantly, this project will improve my hands-on software and hardware engineering skills.

This project device is basically a smart former. It will constantly monitor and adjust the humidity level, the temperature level, light hours, and water cycle for the plant. My project device will take a photo of the plan every 72 hours and then process the image to determine the growth stage and apply the necessary settings, automatically, to achieve the maximum result. For instance, if the plant is in the first growing stage, my device will keep the humidity level between 50 to 60 percent, keep the LED lights on for 16 hours every day, and water the plant every 24 hours. By modifying the device, we can also add nutrition; for this project I'll not add this feature.

Project Learning Objectives

To turn this project into reality, the developer(s) need to have a basic understanding of deep learning, python coding, and Raspberry Pi. For instance, the developer(s) need to be able to setup Raspberry Pi, know how to turn off/on a GPIO pin using python, and know how to work with calcification algorithms such as k nearest neighbors (KNN) to classify the growth stage of a plant based on the training data and input images.

Engineering Originality:

My intuition in designing this machine is to fill the gap for a fully automated plant caretaker. The engineering applied to this project is purely based on the concepts we have learned so far. For instance, we use the deep learning techniques to determine the growth stage of the plant in the following manner:

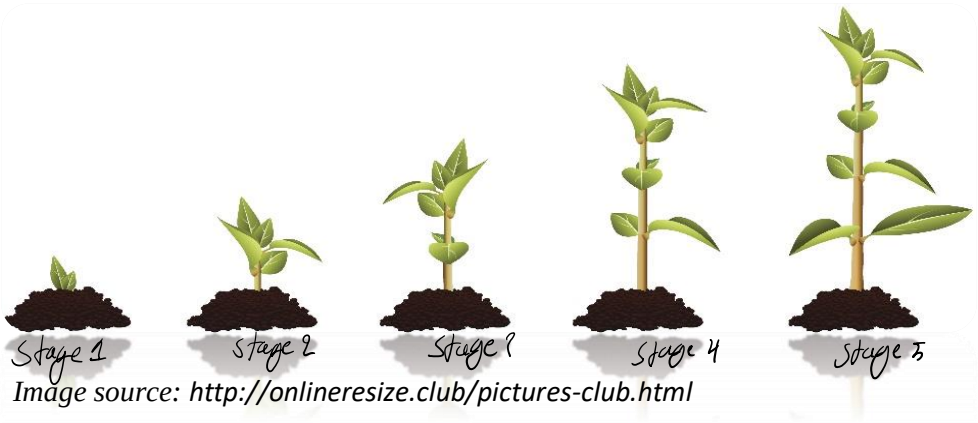
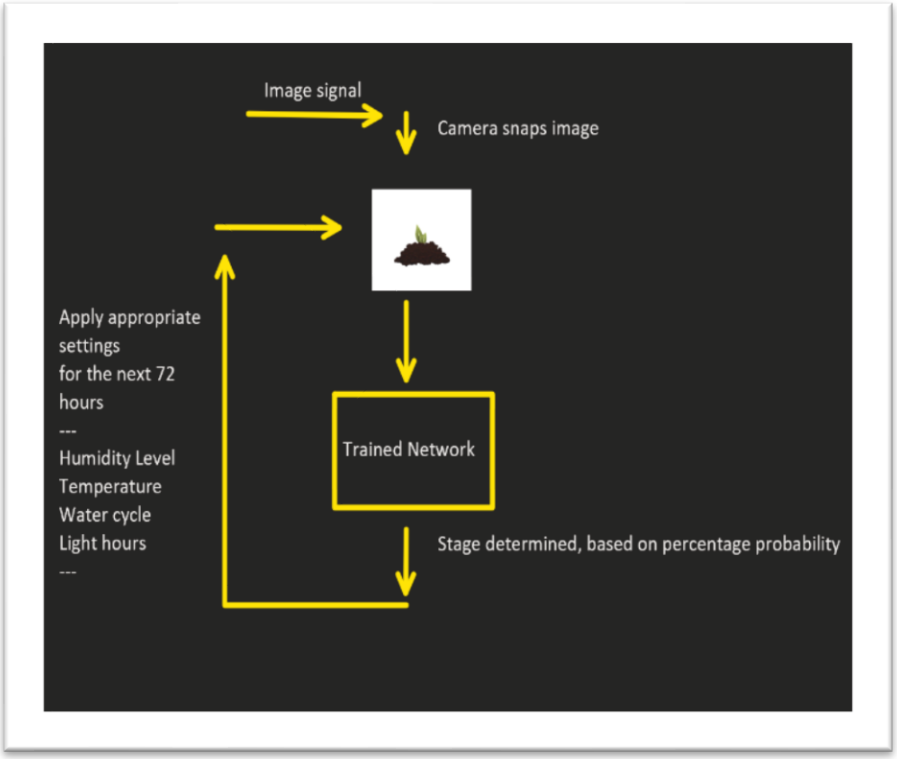


Image source: http://onlineresize.club/pictures-club.html

Every 72 hours the device will send a signal to activate the camera, take a picture, and then use that picture as an input for the, already, trained network to process and classify the growth stage of the plant (I'll describe the camera activation process in more details in the *Project Build Steps*).



To achieve a high level of accuracy in classifying the input image, we can use the K-Nearest Neighbors classification algorithm and a good set of training data. The reason I chose K-Nearest Neighbors classification is because it is the best and simplest way to process our data given the fact that the camera is placed in a constant position. We can determine the growth stage simply by training our algorithm to compare the cluster of green pixels in the input image to our training data, and then classify the growth stage. To obtain the training data, we can use search engines to look for the most relevant images for each stage.

From an engineering standpoint, this project should work flawlessly because to build a functional device we use Raspberry Pi, python, an input sensor(DHT22), output listeners (heater, water pump, spray, LED plant lights), and basic logical procedures (I'll provide more details in the *project build steps* part of this journal). For instance, once the growth stage of the plant is classified by our network, the algorithm calls the appropriate class of settings with an *if else*, or a *switch* statement; this class of settings will include instructions. For example:

If the K-Nearest- Neighbors output = stage 1, call class x,

Class x:

> Read the input form the DHT22 sensor every 10 minutes and check if the temperature is less than 15 C, turn on the heater, if it's greater than 25C turn off the heater (I'll describe the circuits in the Project schematics part), or based on the input from the sensor if humidity is < 50% turn on the humidifier and the heater, and if it's greater than 60% turn off the humidifier and keep the heater off until we reach <=60% humidity. Likewise, turn on the water pump for 10 minutes every 24 hours, and let the LED panel on for 16 hours every 24 hours.

I believe the most challenging part in building this device is the deep learning portion of the project because we need to build and train a network that can provide a very accurate result. On the other hand, it's also challenging but critical to consider the safety aspects of our device.

Engineering Calculations, Assumptions, Risk Reduction:

This project is divided into two main parts: the software part, and the hardware part. For this device to operate as intended, both parts should work with each other flawlessly. For instance, to make sure that the hardware part of the project is sound, I had to consider the voltage requirement for each listener device (water pump, heater, LED panel, humidifier) because on Raspberry Pi the GPIO pins are supplied with a 3.3 volte electricity, and are limited to only 50 mA output, which is not sufficient for powering the heater, the water pump, or the LED light panel. Therefore, I use a 5v 8 channel Relay Shield, control jumper, board which supports loads up to 10A 250V AC or 10A 30V DC to power my high-voltage listener devices. The usage of a Relay also reduces the risk of circuit shortage. Thus, understanding the voltage requirement for each listener device is crucial for the project because it will grantee the safety and reliability of our machine. On the other hand, for the software part of this project, I assume that the developer(s) is/are familiar with python coding, deep learning, and Raspberry Pi. Therefore, I only provide a high-level description for the algorithms that are needed for this project. Overall, my proposal for this project is practical, and I believe it can be built without any major obstacles because it's all based on the simple concepts that we have learned during this quarter. However, I'm assuming that the Raspberry Pi microcomputer have the sufficient computing capacity to power our deep learning algorithm, handle the controller classes, and the hardware for this project.

Project Cost & Budget Justification:

Major component	Cost	E-shop
Raspberry Pi 3 Model B+	44.99	https://www.amazon.com/LoveRPi-Raspberry-Model-Computer-Heatsinks/dp/B07WFWR5RF/ref=sr_1_18?dchild=1&keywords=Raspberry+Pi&qid=1590733646&sr=8-18
OV5647 Webcam	$3.69	https://usa.banggood.com/CSI-Interface-Camera-Module-5-Million-Pixel-with-15cm-Flex-Cable-1080p-720p-5MP-Webcam-Video-Camera-p-1457351.html?utm_source=googleshopping&utm_medium=cpc_organic&gpla=1&gmcCountry=US&utm_content=shopping&utm_campaign=usg-pc&currency=USD&createTmp=1&utm_source=googleshopping&utm_medium=cpc_bgcs&utm_content=frank&utm_campaign=pla-usg-rm-all-purchase-pc&gclid=Cj0KCQjwwr32BRD4ARIsAAJNf_1Rw_DKjYPDML2aygcRaFuMtoEj7acd3hKKaoYCmTnDnu1Mi3LjfqAaAmcIEALw_wcB&cur_warehouse=CN#jsReviewsWrap
5V 8 Channel Relay Shield AC250V 10A ; DC30V 10A	8.79	https://www.amazon.com/SunFounder-Channel-Shield-Raspberry-Arduino/dp/B00DR9SE4A/ref=sr_1_7?dchild=1&keywords=8+Channel+Relay+Module+Board+for+raspberry+pi&qid=1590735674&sr=8-7
DHT22/AM2302 Temperature Humidity Sensor	4.93	https://usa.banggood.com/Wholesale-DHT22-AM2302-Digital-Temperature-Humidity-Sensor-Replace-SHT11-SHT15-Logger-p-47240.html?rmmds=search&cur_warehouse=CN
Heater: MET safety certified seedling heat mat	15.99	https://www.amazon.com/certified-Seedfactor-Waterproof-Germination-Hydroponic/dp/B074753J5V/ref=sr_1_7?dchild=1&keywords=plant+heater&qid=1590732943&sr=8-7
50 GPH 3W Mini submersible water pump and watering pip	11.99	https://www.amazon.com/PULACO-Submersible-Aquariums-Fountain-Hydroponics/dp/B07SJGKFT7/ref=sr_1_1?dchild=1&keywords=mini+water+pump&qid=1590733133&sr=8-1

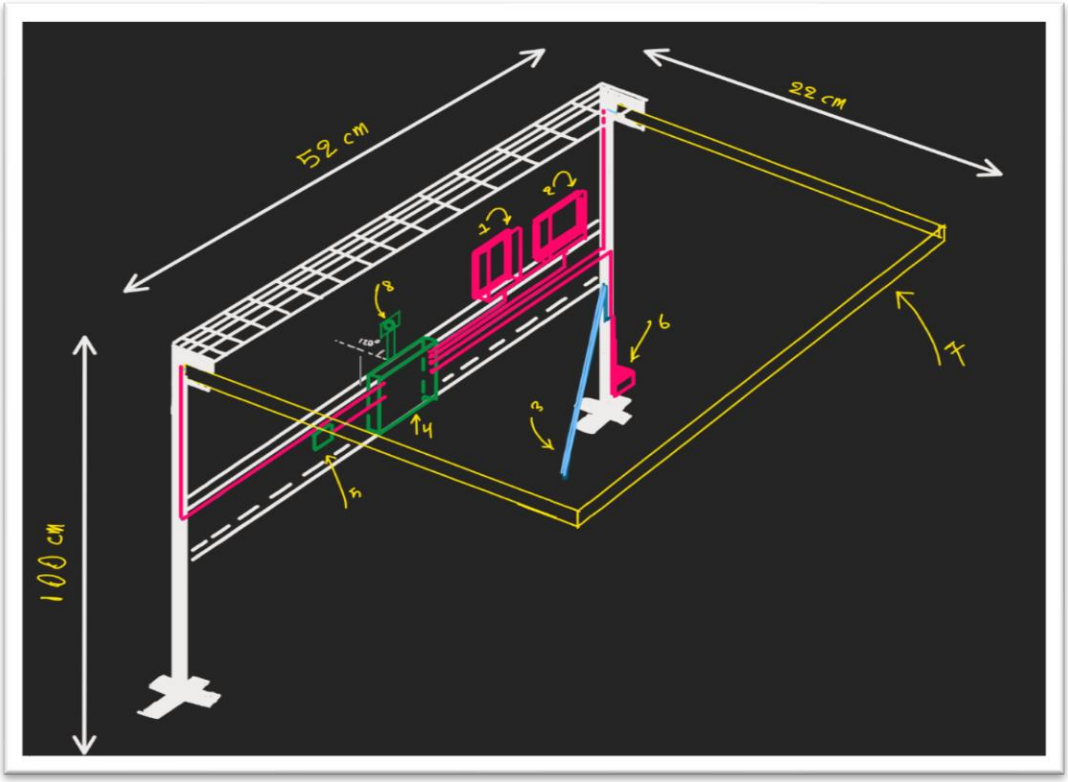
LED Grow Light, 20W Thin Panel, Full Spectrum	17.96	https://www.amazon.com/dp/B07T28YHTY/ref=sspa_dk_detail_4?psc=1&pd_rd_i=B07T28YHTY&pd_rd_w=kKcO2&pf_rd_p=48d372c1-f7e1-4b8b-9d02-4bd86f5158c5&pd_rd_wg=aZ3i0&pf_rd_r=8AQ6HJNX85PBACDXG7EJ&pd_rd_r=287a5578-991e-40e9-b8de-f488145cf551&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUEyVVpRUlpNUklGWjRHJmVuY3J5cHRlZElkPUEwMzQ5NTgzMVVaMzdPS0pSQjhYQiZlbmNyeXB0ZWRBZElkPUEwMzQzNzcxWUdXVTlCWVNMNUFKJndpZGdldE5hbWU9c3BfZGV0YWlsJmFjdGlvbj1jbGlja1JlZGlyZWN0JmRvTm90TG9nQ2xpY2s9dHJ1ZQ==
WHDTS 20mm 113KHz Ultrasonic Atomization Maker Mist Atomizer DIY Humidifier	10.99	https://www.amazon.com/WHDTS-Ultrasonic-Atomization-Atomizer-Humidifier/dp/B073322QF8/ref=sr_1_22?dchild=1&keywords=greenhouse+humidifier&qid=1590736046&sr=8-22
Total	119.33	

The total cost of this project is below 120 dollars but depending on the status of the market this price might slightly change.

Project Build steps:

Project layout:

The frame structure for this project can be designed using solid works and 3d printed. The figure below is a general description of how the final product will look like.



1: Heater
2: Humidifier
3: Watering pip
4: Raspberry Pi
5: DHT22 sensor
6: Water pump
7: LED LIGTH panel holder
8: OV5647 Webcam

To carry out this project I would like to start with connecting the project components to Raspberry Pi.

First, we connect the OV5647 Webcam to 2-lane MIPI CSI camera port on raspberry pi:

2-lane MIPI CSI camera port

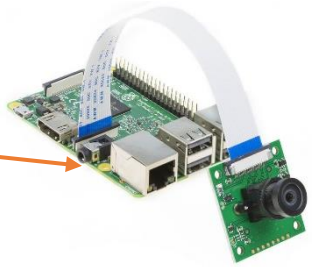


Image source: https://sg.cytron.io/p-8mp-sony-imx219-camera-module-for-raspberry-pi

Camera setup code:

-from picamera import PiCamera

-Import time

-webcam = PiCamera()

-webcam.start_preview ()

-webcam.capture(home/RpPrjoect/image.jpg)

-webcam.stop_preview()

Keep in mind that the camera is placed in a constant position, and we want to capture an image every 72 hours. To do that we need to import **datetime** and **time** classes in our code. I'll leave the rest for the diveloper because the code above provides the foundation for webcam usage.

Our program should be running in a loop, every 72 hours (we need a counter to keep truck of time) it should do the following: activate the camera with code above, capture an image (save the image to the project direcotry), run K-Nearest Nighbours on the input image (access the image from the project directory), determing the growth stage, and apply the necessary settings.

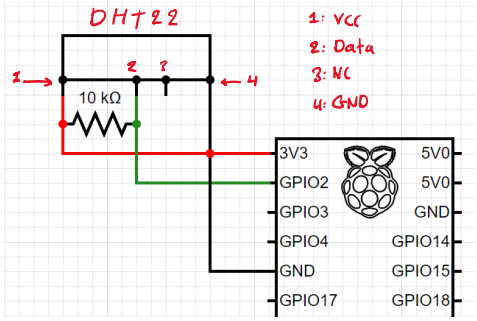
To get the local time we can use the following code:

localTime = datetime.datetime.now().time()

Note: At the beginning of each itteration, we need to erease the saved images from the "project directory ./savedimage.jpg" before capturing a new image to provide a clear path for our network input. This can be done by using the **os class**:

- improt os
- os.remove ("home/RpPrjoect/image.jpg")

Next we connect the DHT22 tempreture and humidy sensor:



Setup code for DHT22:

-Import Adafruit_DHT

-DHT_SENSOR = Adaruit_DHT.DHT22

-DHT_PIN = 2

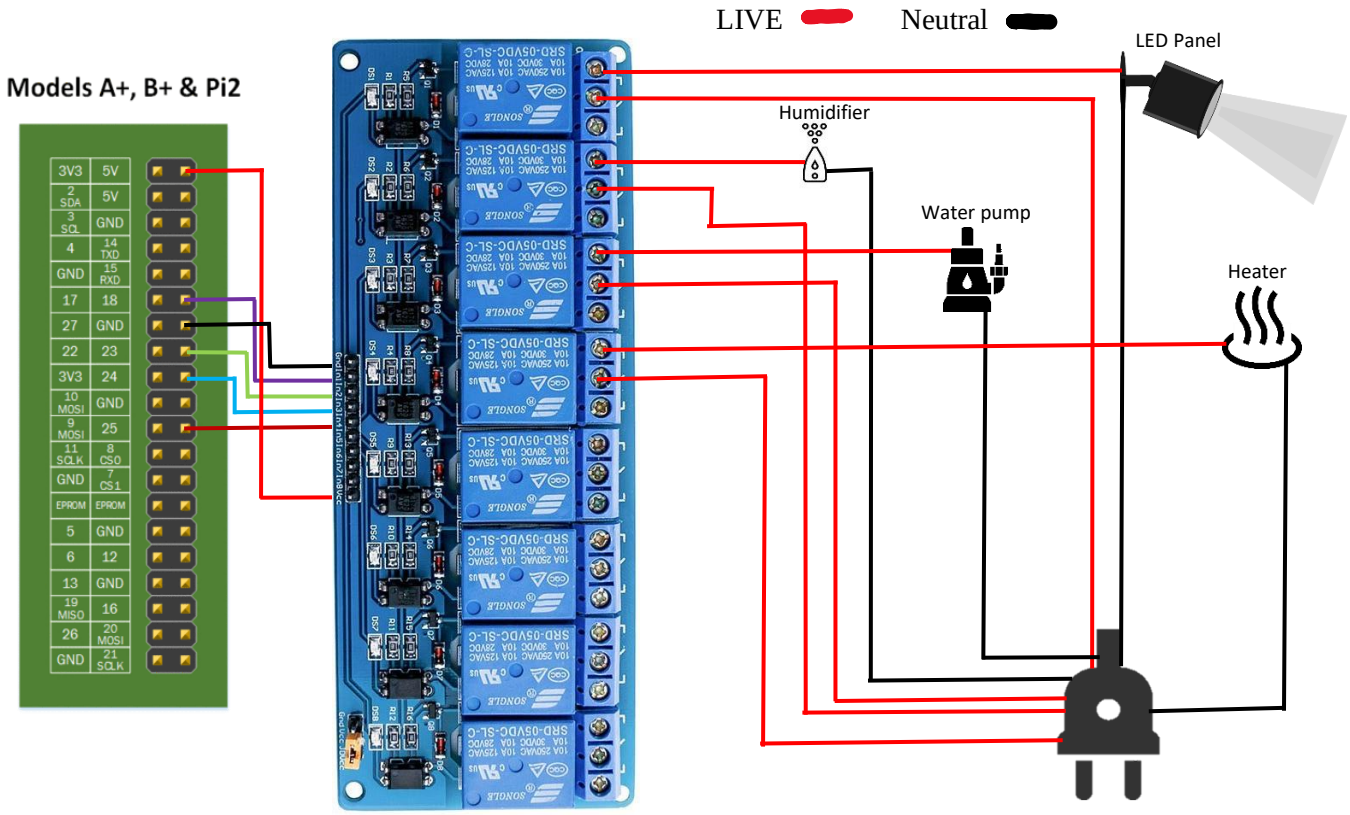
-humidity, tempreture =Adaruit_DHT.read_retry (DHT_

SENSOR, DHT,PIN)

Here we assume that RPi.GPIO is already imported, and GPIO.setmode(GPIO.BCM) already exist in our code. From this point on, we can use the variables “humidity” and “tempreture” to control the hardware components. For instance, we can use an if statement to look for humidity level; if the humidity is greater than 60% turn of the humidifier and the heater until humidity reaches =<60%, if it’s less than 50% turn on the humidiferthe and the heater. By following these steps we make sure that the humidty stays between 50 – 60 %, but keep in mind that each stage have a different setting, and the heat is also controlled by another if statement type algorithm.

Note: the way we turn on and off the humidifier and the heater is relevent to the fact that heat and humidity are related to each other: the higher the air tempruater is the more water molecules it can hold.

Finally, we connet the LED light panel, humidifier, heater, and the water pump to our Raspberry Pi:
For this part of the project to meet the power requirements, we need to use a 5V 8 Channel Relay Shield 10A ; DC30V 10A.
Schematic:



Raspberry Pi to Relay Connections:
Connect 5v to VCC via
Connect GND to GND via
Connect pin 18 to IND1 via
Connect pin 23 to IND2 via
Connect pin 24 to IND3 via
Connect pin 25 to IND4 via

Setup code for Led panel, humidifer, waterpump, and heater (assiming that all the necessary classes are already imported):

```
GPIO.setmode (GPIO.BCM)
GPIO.setup(18, GPIO.out)
GPIO.setup(23, GPIO.out)
GPIO.setup(24, GPIO.out)
GPIO.setup(25, GPIO.out)
ledPanel =  18                  //varibale for LED panel controls
Humidfier = 23                  //varibale for Humidfier controls
waterPump=24                    //variable for water pump controls
heater= 25                      //variable for  heater contorls
```

Image sorce of raspberry pi pin layout: https://thepihut.com/blogs/raspberry-pi-tutorials/27968772-turning-on-an-led-with-your-raspberry-pis-gpio-pins
Image source of 8 chennal raley board: https://www.elegoo.com/product/elegoo-8-channel-dc-5v-relay-module-with-optocoupler/
Image source of light clip art: https://www.pinterest.com/pin/510314201510754597/
Image source of waterpump clipart: https://www.pinclipart.com/maxpin/iixiomi/
Image source of heater clipart: https://www.pinterest.ca/pin/757167756073211112/
Image source of humidifier clipart: https://www.123rf.com/clipart-vector/humidifier.html?sti=lyaslenrhf7nelfyo9|
Image source of plug clipart: https://www.clipartkey.com/view/hmwoTT_plug-clipart-adapter-png-download/

After connecting the project components as described above, we should implement the required algorithms. First, we should build our deeplearning algorithm, K-Nearest Nighbours (I described the implementaion of this algorithm in the *Engineering Originality*). Then we use the following growth stage discriptions:

Stage 1: waterpump activation = 10 minutes/24 hours; LED panel activation 16 hours/24hours; Humidty level = 95%; Tempreture = 23 degrees celcius
Stage 2: waterpump activation = 12 minutes/24 hours; LED panel activation 15hours/24hours; Humidty level = 60-70%; Tempreture = 24 degrees celcius
Stage 3: waterpump activation = 14 minutes/24 hours; LED panel activation 14 hours/24hours; Humidty level = 50-60%; Tempreture = 24 degrees celcius
Stage 4: waterpump activation = 16 minutes/24 hours; LED panel activation 12 hours/24hours; Humidty level = 50-60%; Tempreture = 25 degrees celcius
Stage 5: waterpump activation = 18 minutes/24 hours; LED panel activation 12 hours/24hours; Humidty level = 50-60%; Tempreture = 25 degrees celcius

To implement 5 saperate class that includes setting code for each of the five coresponding stages (we use the basic logical procedures to implament these classes: to turn on the heater simply use GPIO.output(heater, GPIO.HIGH), heater variable is defined in the setup code for the heater) . The code flow should be as following:

- Place the overall code in a 72 our loop cycle using **datetime** and **time** classes
- At the bening of each loop, erease the existing image in the specified path using **os class**, capture a new image using **picamera class**, and save it to our specified project path.
- Run the K-Nearest Knighbours on the captured image (acces the image from the project specified path) and classify the growth stage.

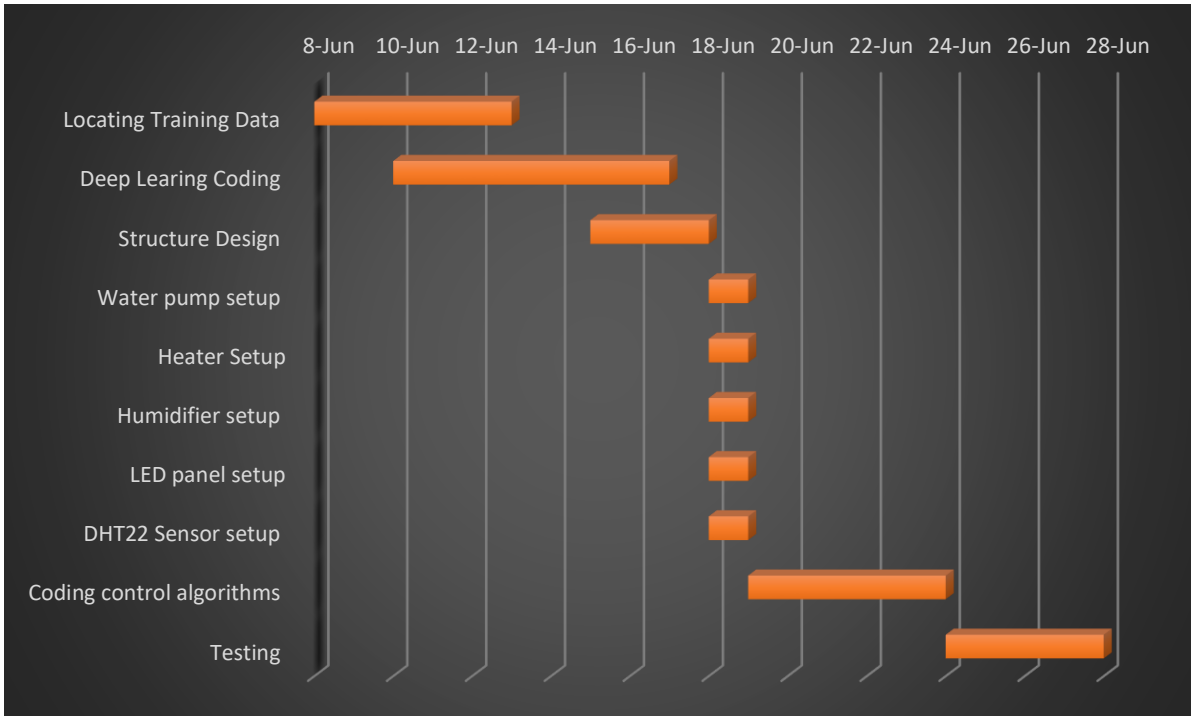
- Based on the K-Nearest Knighbours output, call the opproperiate class to apply the control settings

Each class should contian all the necessary settings to control the heater, humidifier, water pump, and LED panel based on the DHT22 inptut. For instance, we can use the following code to monitor the humidity and temperature level in a constant 1 minutes loop:

```
GPIO.setmode (GPIO.BCM)
GPIO.setup(18, GPIO.out)
GPIO.setup(23, GPIO.out)
GPIO.setup(24, GPIO.out)
GPIO.setup(25, GPIO.out)
ledPanel =  18                  //varibale for LED panel controls
Humidfier = 23                  //varibale for Humidfier controls
waterPump =24                   //variable for water pump controls
heater = 25
DHT_PIN = 4

While true:                     //loop to monitor the humidiry and tempreture every 2 minutes
    Humidty,tempreture = Adafruit_DHT.read_retry (DHT_SENSOR, DHT_PIN)

if tempreture < 15:
    GPIO.output (heater, GPIO.HIGH)         //turn on the heater
if tempreture > 25:
    GPIO.output (heater, GPIO.LOW)          //turn on the heater
If humidity < 50 :
    GPIO.output (Humidfier, GPIO.HIGH)      //turn on the humidifer
    GPIO.output (heater, GPIO.HIGH)         //turn on the heater
if humidity > 60:
    GPIO.output(humidfier, GPIO.LOW)        //turn of the humidfier
    GPIO.output (heater, GPIO.LOW)          //turn on the heater

Sleep(60)                                   // wait for 1 mintues before the next iteration
```

Like wise we can use **datetime, and time** classes to control other aspects of our device such as when to start the water pump, for how long we should leave it on, and when to start the LED panel and for how long we should leave it on.

Project Timeline



Resouces

Developing KNN tutorials: https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/

Image classification using KNN tutorials: https://www.youtube.com/playlist?list=PLRqwX-V7Uu6YPSwT06y_AEYTqIwbeam3y

Raspberry Pi – Python tutorials: https://www.youtube.com/watch?v=RpseX2ylEuw&list=PLQVvvaa0QuDesV8WWHLLXW_avmTzHmJLv

Online circuit diagram editor: https://www.circuit-diagram.org/editor/

Soldering tips: https://www.makerspaces.com/how-to-solder/

Solidworks tutorials: https://www.youtube.com/watch?v=zwqucVAmitw&list=PLROUP1bV8RERrsOG6tE3kbRo7qU4gjJ7A

Tips on how to grow a healty plant: https://gardenerspath.com/how-to/beginners/growing-plants-101/