DWC
Smart-Vegetation System.

Supervised By

Assoc. Prof. Zaki Bassiony

Prepared By:
DoAa Atia Ahmed Ezzet
Mahmoud Ahmed Abd El-Moneem
Mohab Mohamed Mostafa
Omar Sayed Mostafa
Youssef sherif Mohamed

Cairo, Egypt
2020 - 2021

كلية الهندسة جامعه حلوان

جامعة حلوان

DWC
Smart-Vegetation System.

Supervised By

Assoc. Prof. Zaki Bassiony

Prepared By:

DoAa Atia Ahmed Ezzet

Mahmoud Ahmed Abd El-Moneem

Mohab Mohamed Mostafa

Omar Sayed Mostafa

Youssef Sherif Mohamed

Sponsored By

iPDSC Innovation Center

Cairo, Egypt

2020 - 2021

Acknowledgement

First and foremost, we are deeply grateful to Allah for giving us the health, patience, strength, and knowledge to complete this work, we must say that without blessing of Allah this project would not have been possible with this way.

We also thank our families for this outstanding effort with their love, supporting, patience, and encouragement.

Special thanks to our graduation project supervisor **Assoc. Prof. Zaki Bassiony** for his great efforts to provide us with all useful information, patience, guidance along the semesters and making the path clear for us. Moreover, it is our duty to thank all the professors for their generous discussions and encouragement.

It has been a great opportunity to gain lots of experience in real time project, followed by the knowledge of how to design and analyze it. For that we want to thank our college for providing us with a **IPDSC** lab sponsoring.

Abstract

Providing a good foods is one of the important life's necessities for all people everywhere so our project is about Hydroponics and how to replace the soil and sun as they cause a lot of problems, where hydroponics can overcome these problems with ease and find a solution for every user even make it easier by automating it, fully monitored, where you only need to put the seed and leave it to become a plant with no intervention in a very small area or even big as it's needed according to its size.
It only needs water, and the rest is on us.

It aims to make the agriculture easier, faster, more efficient, and even with minimum cost to meet every one's requirement. Fully monitored offline or online with the web application, any user can do it with no knowledge about anything as we deal with the plant with deep learning algorithms and Embedded system control.
The user will need to make some interventions but after a very long time with separated long periods with our reminders and easy steps to follow.

Contents

Table Of Figures

1. Chapter| Introduction

❖ Introduction

1.1. History

The DWC method was originally used to determine what exactly nutrients the plants need for healthy growth. It's still used today for doing plant research to reach for best achievements and quality.

In the 1850s and 1860s, two botanists from Germany developed a water culture technique which involved bubbling air through a nutrient-rich solution of water. These researchers, Julius von Sachs, and Wilhelm Knop were number of the first known people who grow plants like this way – a method we would now consider to be DWC (deep water culture). Growers today still refer to hydroponic growing as a solution culture because of the specially prepared solution of water and minerals were used to provide nutrients to the plants. But it wasn't until the 1900s for people to begin realizing the power of hydroponic growing to achieve larger and faster growth.

we finally began to understand which nutrients a plant needs to grow and thrive, we discovered that growers who provide the right conditions at the roots can achieve faster rates of growth than traditional methods achieve. one of the hydroponic growing advantages that it allows growers to control almost all variables for the plants, allowing them to provide exactly what plants need at the right time and for each phase.

1.2. Motivation

As our team believes that helping people and having a sense of mission in your pursuit of a greater life will make a substantial difference to your mindset and your work ethic, that brings us to the purpose of our project for helping people everywhere to grow their food easily with better quality and cheaper cost specially for people living in conditions of poor soil and little light, e.g. Finland to grow fresh food all year round, so we can clearly say that having a clear vision will always keep us on track in pursuing our mission and keeping us focused to achieve our desired goal.

1.3. Problem Statement

The end goal of the project is to design and build a hydroponic system functional, and more reliable to make hydroponic food growing easier and cheaper based on that Plants do not need soil to photo-synthesize. we can say A general overview of the problem and how to solve it. As food gets more expensive, and there are a lot of areas where soil, light or temperature can be problematic for growing, there are a lot of soil-borne pests and disease. Soil Agriculture use large amounts of pesticides which cause air and water pollution so hydroponic system will be especially useful now that food prices are increasing, and since it is an established branch of agronomy. Progress has been rapid, and results obtained in various countries have proved it to be thoroughly practical and to have very definite advantages over traditional methods of horticulture.

Hydroponic system will allow people to grow their own food more cheaply with higher quality, and by using of appropriate lighting it would make it possible for people living in conditions of poor soil and little light, e.g., Finland to grow fresh food all year round too and it may potentially produce much higher crop yields. plants flourish under the careful regimen of hydroponics. Using smaller space and 90% less water than traditional agriculture, and ingenious design, hydroponic gardens grow beautiful fruits and flowers in half the time.

Figure 1: DWC Hydroponic system agriculture VS traditional agriculture

The hydroponic food needs the soil to supply them with water and nutrients. When nutrients are dissolved in water, they can be applied directly to the plant's root system by flooding, misting, or immersion. Hydroponic innovations have proven direct exposure to nutrient-filled water can be a more effective method of growth than traditional irrigation. This achieved by monitoring and controlling the key environmental ingredients needed for successful hydroponic growing, chiefly the pH, EC and water levels and controlling the external HW, such as lights and water pumps. This Software design and software patterns were taken into use to make the SW more expandable and helpful for user. This book details the HW and SW and why certain design decisions were made. The design of the SW parts is also described as well as the overall functioning of the SW. An explanation of how to build the system is also given.

1.4. What is the DWC hydroponic system?

DWC system is considered as one of the most common hydroponic systems used today. Typically, DWC is used to grow short-term, non-fruiting crops such as leafy greens and herbs. The large volume of water helps to reduce rapid changes in temperature, pH, electrical conductivity (EC), and nutrient solution composition.

This method is called Deep Water Culture for two reasons. One, you typically grow with a reservoir that can hold a decent amount of water. More water means more stability in the nutrient solution, which means less manual monitoring and maintenance. The second reason is because of how much of the root mass you submerge in the water. Other methods expose your plant's root zone to air and put them in water just a few times a day (ebb and flow systems are a good example of this). In deep water culture, most of your plant's root system is submerged 24/7 – hence the name!

Figure 2: DWC Hydroponic system.

1.5. How does hydroponics work?

Hydroponic systems work by allowing minute control over environmental conditions like temperature and pH balance and maximized exposure to nutrients and water. Hydroponics operates under a very simple principle: provide plants exactly what they need when they need it. Hydroponics administer nutrient solutions suitable to the needs of the particular plant being grown. They allow you to control exactly how much light the plants receive and for how long. PH levels can be monitored and adjusted. In a highly customized and controlled environment, plant growth accelerates. By controlling the environment of the plant, many risk factors are reduced.

Plants grown in gardens and fields have many variables that negatively impact their health and growth. Fungus in the soil can spread diseases to plants. Wildlife like rabbits can plunder ripening vegetables from your garden. Pests like locusts can descend on crops and obliterate them. Hydroponic systems end the unpredictability of growing plants outdoors and in the earth. Without the mechanical resistance of the soil, seedlings can mature much faster. By get rid of pesticides, hydroponics produces much healthier and high-quality fruits and vegetables. Without obstacles, plants are free to grow vigorously and rapidly.

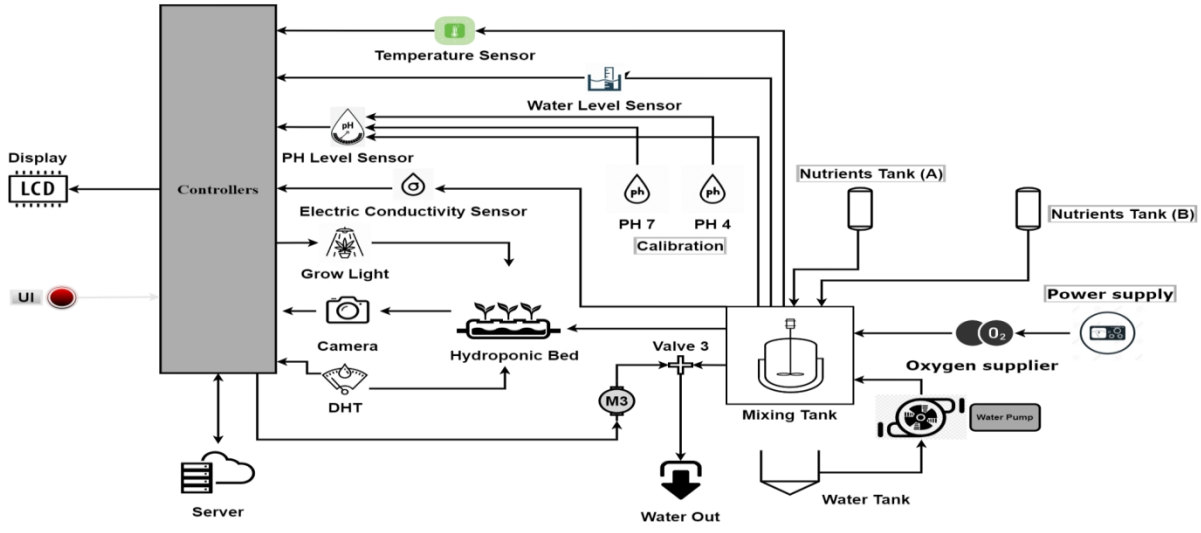


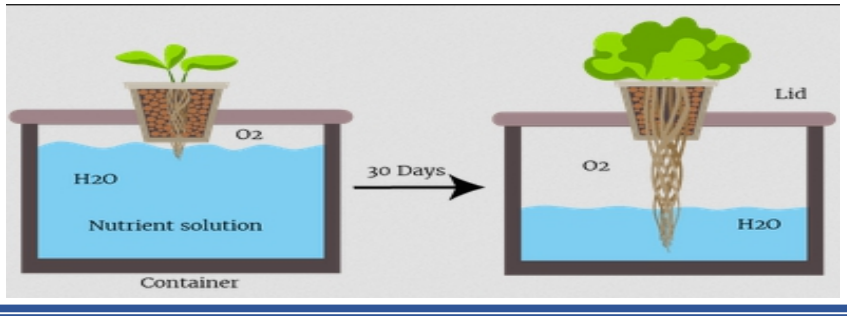
Figure 3 :Design prototype of the DWC system.

1.6. DWC Hydroponics system features

A general overview of hydroponics and its advantages over soil-based planting we can talk about. As food gets more expensive, it will allow people to grow their own food more cheaply especially in areas where soil, light or temperature can be problematic for growing.

The advantages of hydroponics are the following.

- No soil is needed for hydroponics
- Stable and high yields so Plants grow healthier
- Ease of harvesting
- No pesticide damages
- It is better for consumption
- Very low maintenance once you set it up
- Very little moving parts and assembly.
- Pests and diseases are easier to get rid of than in soil because of the container's mobility
- No nutrition pollution is released into the environment because of the controlled system
- Extremely fast-growing time compared to soil (lettuce grown takes 30 days to harvest instead of 60 in soil)



1.7. Project flow

- Transplant the seed in a Rock-wool by adding amount of water and lighting. the CNN model compares current plant picture with dataset to decide if our plant ready to start its program or not.
- When it's ready, the embedded system receives message from machine model to start DWC system's program, in parallel send message and pictures of the plant's current state to the web application for informing it that the program will start now.
- Embedded system starts it's program by many instructions as measuring water level by ultrasonic sensor after adding the required nutrients, measuring PH level and electrical conductivity (EC) of mixed solution by PH, EC sensors Respectively, adjusting room and solution temperature by DHT11, waterproof temperature sensor Respectively, control RGB luminance to be suitable for plant phase, and send these readings to the web page in parallel.
- Web application stores sensor reading, provides useful information as measuring level, and notify the user with the plants state and system updates.
- Deep learning model analyzing the picture of plant to determine the plant state and sends this analyzed picture to the web application.

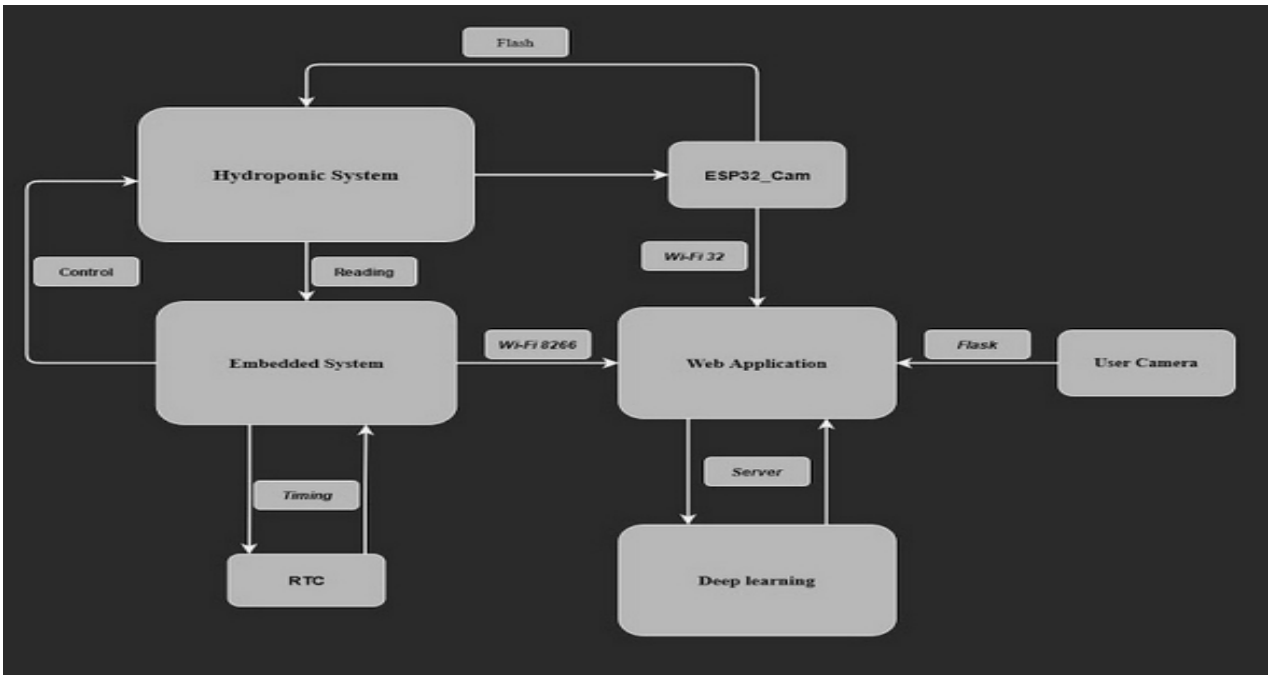


Figure 4: Project flow.

2. Chapter| Embedded System

❖ Embedded System

An embedded system is a computer system—a combination of a computer processor, computer memory, and input/output peripheral devices—that has a dedicated function within a larger mechanical or electronic system. It is embedded as part of a complete device often including electrical or electronic hardware and mechanical parts. Because an embedded system typically controls physical operations of the machine that it is embedded within, it often has real-time computing constraints. Embedded systems control many devices in common use today. In 2009 it was estimated that 98% of all microprocessors manufactured were used in embedded system.

This part is Based on 3 important phases such as programming, electronic, and mechanical parts which is controlled by the programming phase. We used the Software environment called "**Arduino IDE**" for programming and controlling our system, where the ESP will be compatible with it.

The currently required components:

- ESP 8266 (controller)
- ESP32 CAM (controller)
- Digital temperature and humidity sensor (DHT).
- Electrical conductivity sensor (EC).
- Temperature sensor.
- Ultrasonic sensor.
- PH sensor.
- Medium sized pump (Water).
- Mini pump (Solutions).
- Real time clock (RTC).
- Oxygen supplier.
- LCD 20x4.
- I2C module and LED strip.

2.1. Basic Structure of an Embedded System

The basic structure of an embedded system includes the following components:

- **Sensor:** It measures the quantities that are physical and converts it to an electrical signal which may be read by an observer or through any electronic tool like an A-D converter. A sensor shops the measured amount to the memory
- **AD Converter:** An analogue-to-digital converter converts the analogue signal sent by the sensor into a digital signal.

  Processor & ASICs: Processors assess the data to measure the output and store it to the memory.

- **Memory:** it is used to store the data that the processor receives and uses to respond to instructions that it's been programmed to carry out. A Micro controller has two main memory types:

  ●**Program memory:** which stores long-term information about the instructions that the CPU carries out. Program memory is non-volatile memory, meaning it holds information over time without needing a power source.

  ● **Data memory:** which is required for temporary data storage while the instructions are being executed. Data memory is volatile, meaning the data it holds is temporary and is only

- **DA Converter:** A digital-to-analogue converter changes the digital data fed by the processor to analogue data
- **Actuator:** An actuator compares the output given by the D-A Converter to the actual output stored and stores the approved output.

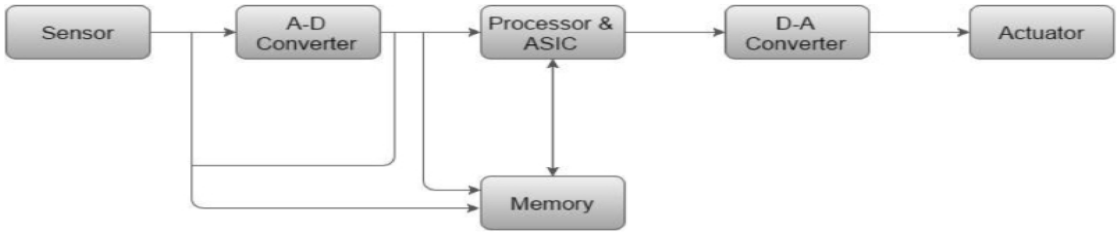


Figure 5: General structure of Embedded system.

2.2. Overview

The main goal of using the Embedded Systems in our project is to be able to analyze and process different signals and use them for useful purposes and controlling the flow of the project to meet the needs of plants, requiring nutrients in some specific times, refilling the water for mixing with nutrients, indicating the levels and the needed luminance for the plant. This calls for having Sensors, water pumps, nutrients pumps, and camera module in order to take in the signals which we want to process using the ESP8266 and ESP32-CAM micro controller chips.

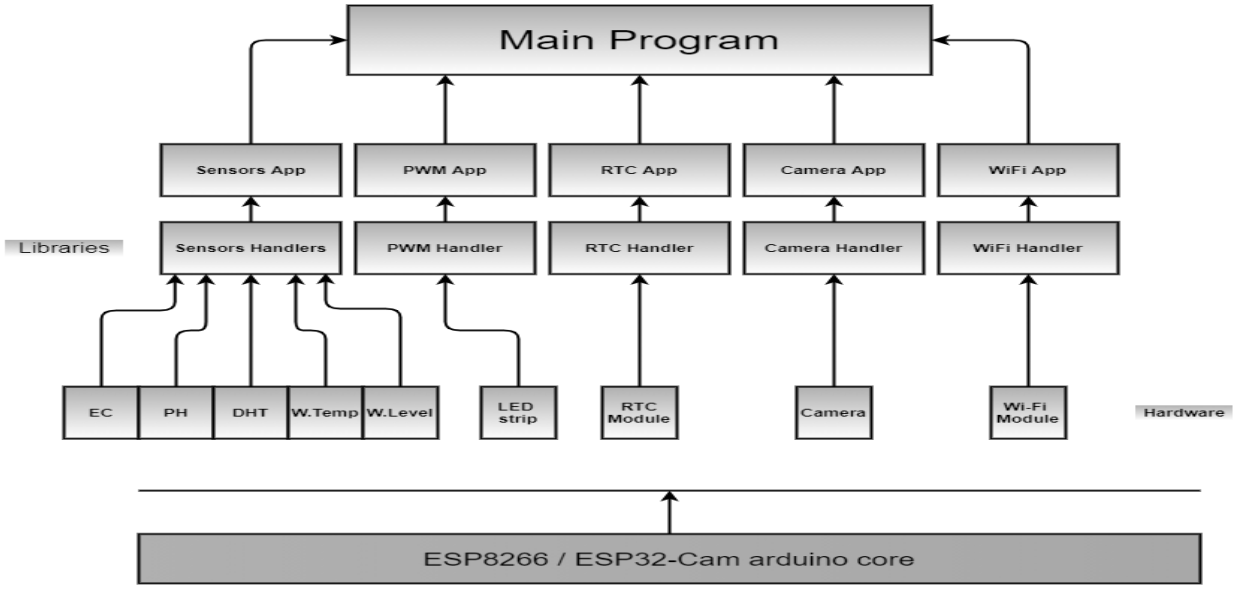


Figure 6: Software Architecture of Embedded part.

2.2.1. What is a Micro Controller?

A micro controller is a compact integrated circuit designed to govern a specific operation in an embedded system. A typical micro controller includes a processor, memory and input/output (I/O) peripherals on a single chip.

Sometimes referred to as an embedded controller or micro controller unit (MCU), micro controller is found in vehicles, robots, office machines, medical devices, mobile radio transceivers, vending machines and home appliances, among other devices. They are essentially simple miniature personal computers (PCs) designed to control small features of a larger component.

2.2.2. How do micro controllers work?

The micro controller is embedded inside of a system to control a singular function in a device. It does this by interpreting data it receives from its I/O peripherals using its central processor. The temporary information that the micro controller receives is stored in its data memory, where the processor accesses it and uses instructions stored in its program memory to decipher and apply the incoming data. It then uses its I/O peripherals to communicate and enact the appropriate action. micro controllers are used in a wide array of systems and devices. Devices often utilize multiple micro controllers that work together within the device to handle their respective tasks.

2.3. Steps

- Choosing a certain plant on the web application to load a certain program with the plant values on the ESP8266.
- User puts the seeds in their place with the Rock-wool and press the start program.
- Water begins to fill the upper container until it reaches the growing medium, where ultrasonic sensor tells when to stop pumping.
- Oxygen supplier starts to work for providing the Oxygen for seeds.
- Once every day the LED strip goes on to take a picture of the plant to know if it is ready to the next step or not.
- If plant is ready, the server sends a command to start adding the nutrients.
- First pump pushes a certain amount of 'A' solution to the water.
- Waiting 10 minutes with RTC then starting to push the same amount but from 'B' solution.
- PH and EC sensors tells if it is the right values or not and the ESP8266 takes the action, whether to add pure water (If PH is low) or add nutrients otherwise.
- The RTC also control when to interrupt the ESP32 CAM to take image of the plant in several phases for detecting the plant's condition and determine the phase of it to provide it with all what it needs.
- The values would be monitored daily according to the time of RTC communicating with the web application to upload them.

LED strip is controlled with RGB by the RTC timing to give the plant the right luminance and wavelength, 12 to 14 hours per day actively, 12 to 10 sleeping.

❖ ESP8266 flow Stage 1

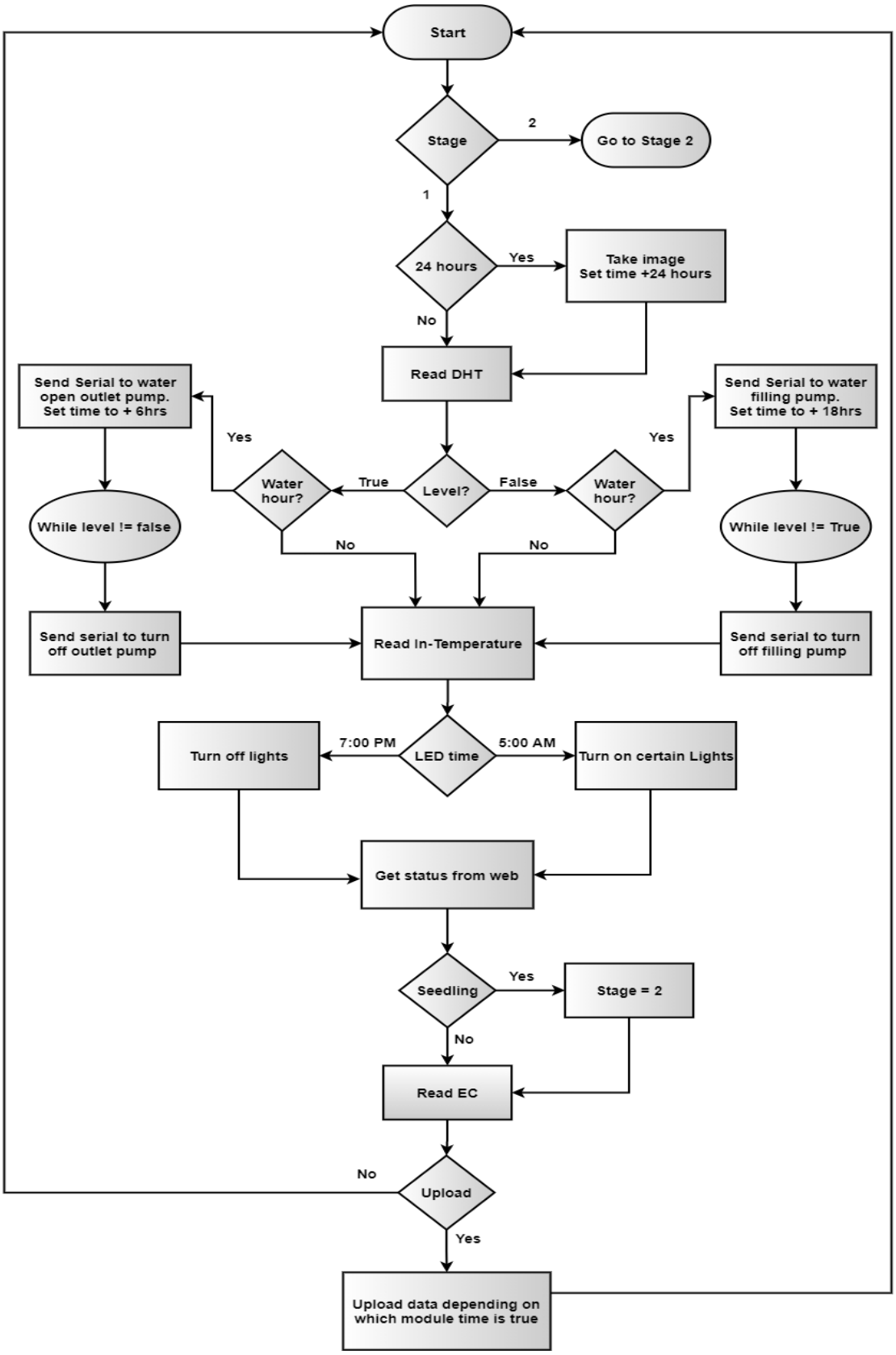
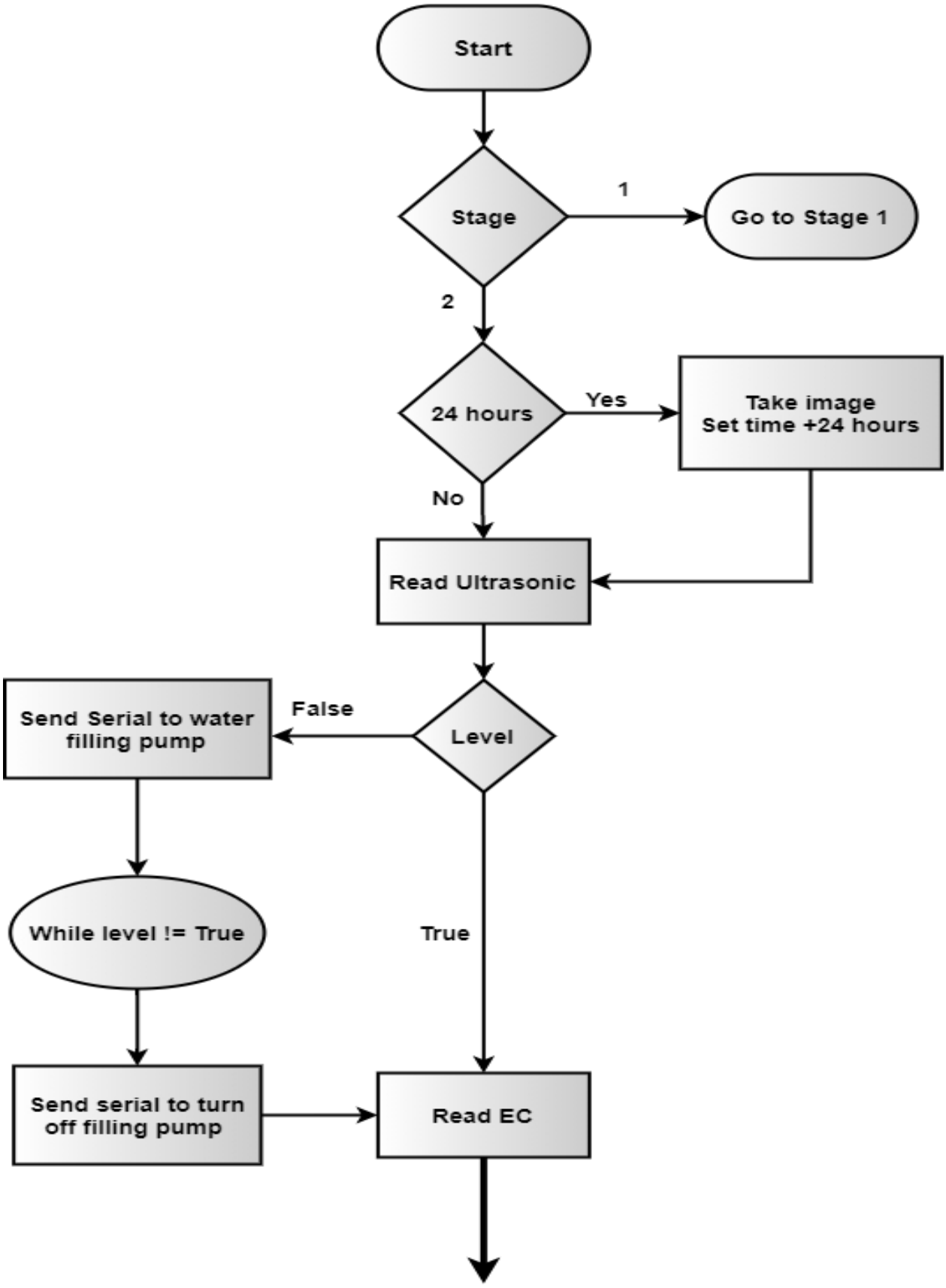


Figure 7: Stage1 flow.

❖ ESP8266 flow Stage 2



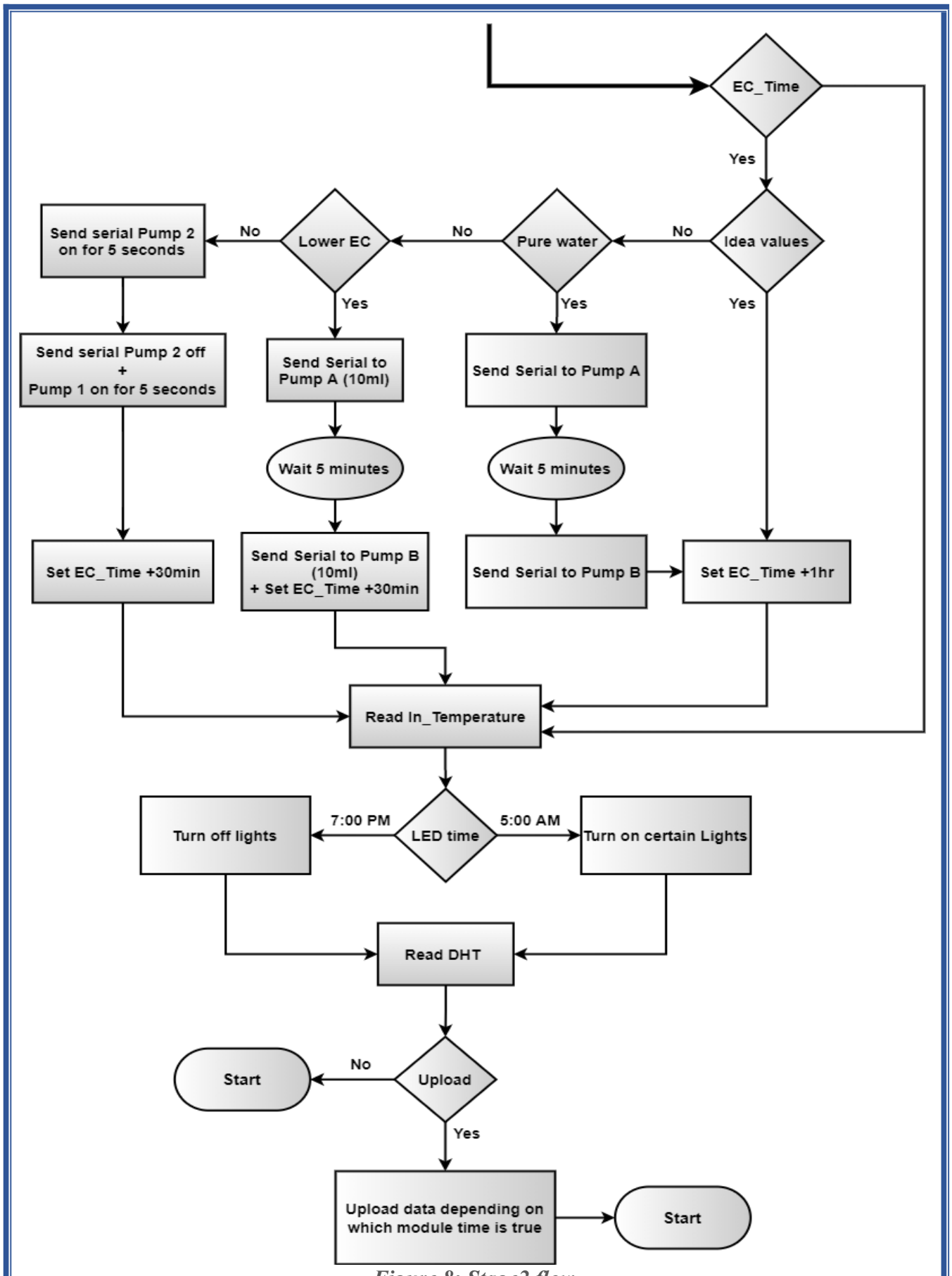


Figure 8: Stage2 flow.

2.4. The ESP8266 Micro Controller

It is one of the main executioners for the project due to its high processing power, and it has many PWM pins which will be used for controlling the LED strip brightness according to the plant growing phase.

ESP8266 board control and connected with the Ultrasonic sensor, DHT sensor, Temperature sensor, LCD 20x4, PWM pins, and the analogue multiplexer which is connecting the PH and EC sensors to our micro controller.

Also, we used the Real Time Clock (RTC) which is the clock of the whole things and used to track the current time and date, that tells when to put each nutrient, when to fill the water or stop filling, even the time stamp of the uploads to the web application. Then the control signals to the mechanical components when needed pumps.

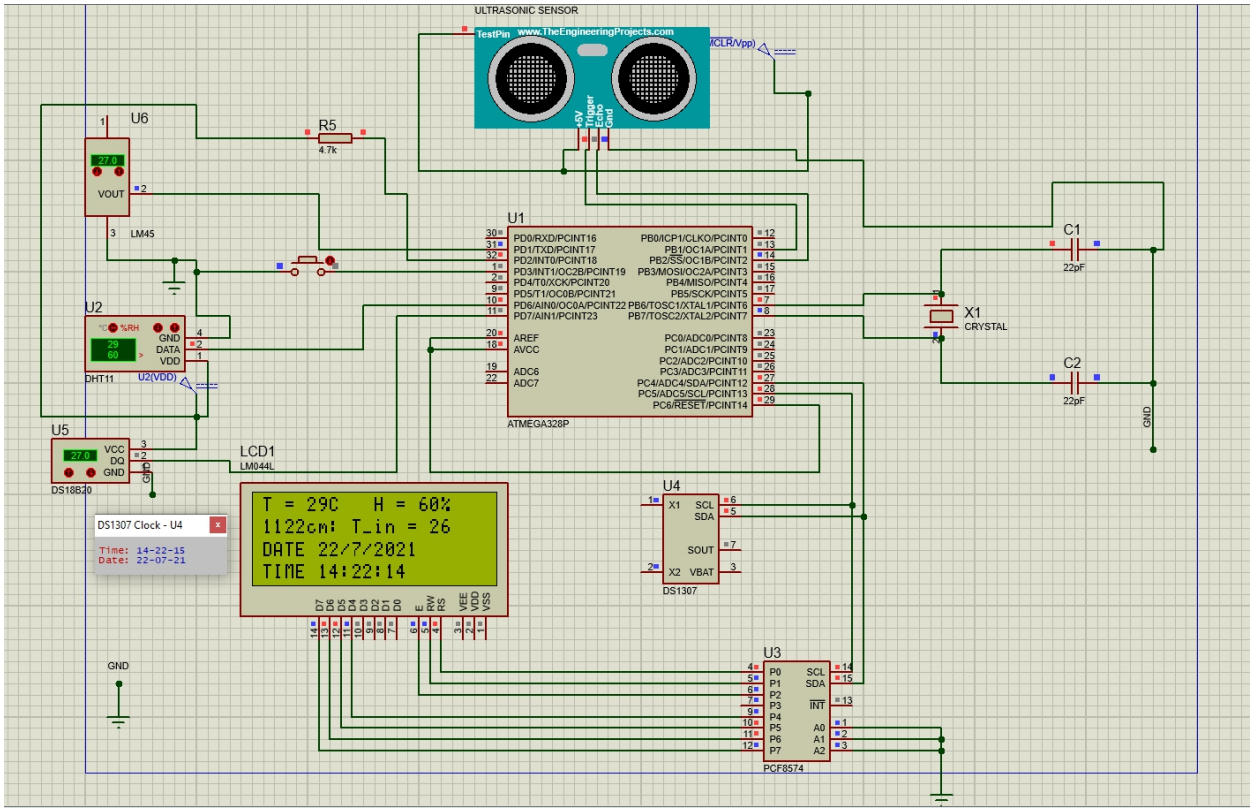


Figure 9: Connections between ESP8266 and other I/O.

ESP8266 micro controller helps to monitor the readings of the Ultrasonic to know the water level and decide whether it is needed to manually refill the container or not and taking readings from DHT (Humidity and temperature of the room) to ensure the environment of the plants is suitable or not. And check the temperature sensor readings for the solution which should be less the 35 degrees. Also, the ESP8266 has a Wi-Fi module, so it can reach the web application to upload the readings if needed and receive any critical command from the server.

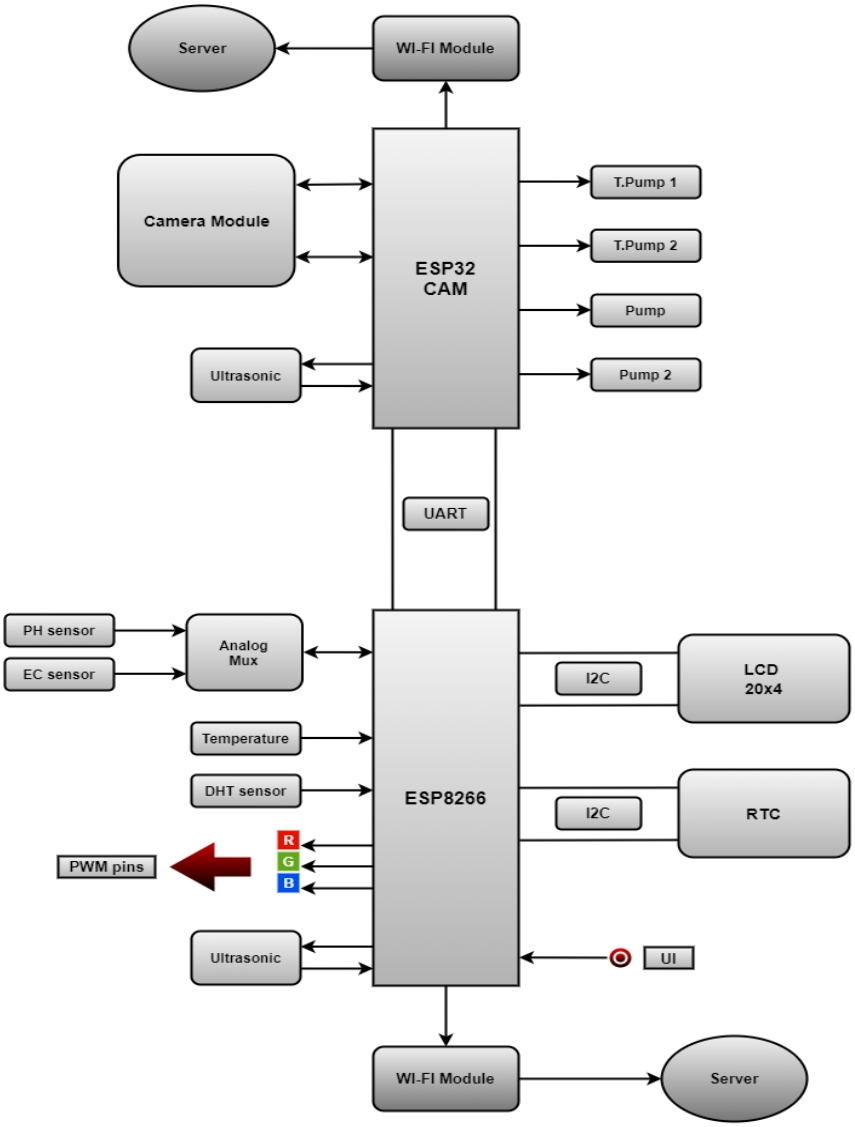


Figure 10: Diagram of all the components connections and serial comm. Protocols.

2.5. Sensors

2.5.1. PH sensor

A pH sensor is one of the most essential tools that's typically used for water measurements. This type of sensor is able to measure the amount of alkalinity and acidity in water and other solutions. When used correctly, pH sensors are able to ensure the safety and quality of a product and the processes that occur within a wastewater or manufacturing plant.

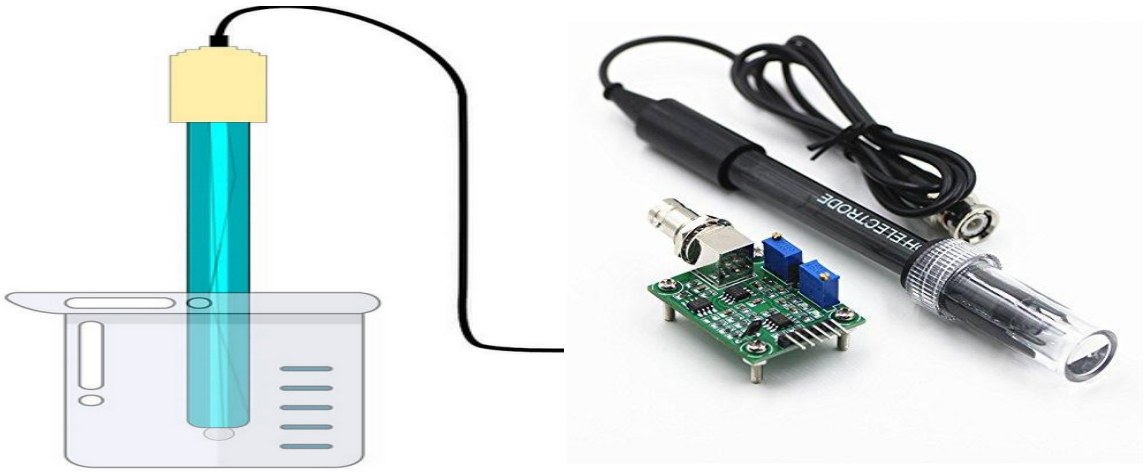


Figure 11: PH sensor.

in most cases, the standard pH scale is represented by a value that can range from (0-14) When a substance has a pH value of seven, this is neutral. Substances with a pH value above seven represent higher amounts of alkalinity whereas substances with a pH value that's lower than seven are believed to be more acidic. Fundamentally, a pH meter consists of a voltmeter attached to a pH-responsive electrode and a reference (unvarying) electrode. The pH-responsive electrode is usually glass, and the reference is usually a silver–silver chloride electrode, although a mercury–mercurous chloride (calomel) electrode is sometimes used. When the two electrodes are immersed in a solution, they act as a battery. The glass electrode develops an electric potential (charge) that is directly related to the hydrogen-ion activity in the solution (59.2 millivolts per pH unit at 25 °C [77 °F]), and the voltmeter measures the potential difference between the glass and reference electrodes.

pH electrodes are like batteries; they run down with time and use. As an electrode ages, its glass changes resistance. This resistance change alters the electrode potential. For this reason, electrodes need to be calibrated on a regular basis.

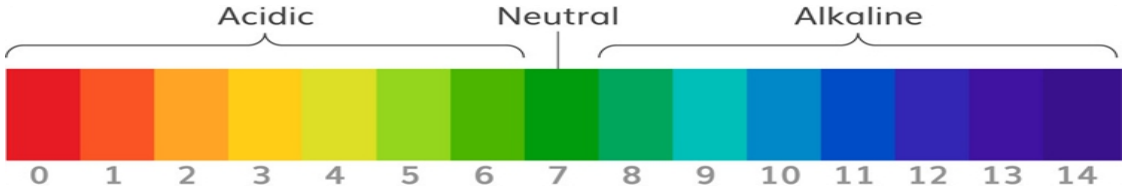


Figure 12: PH sensor scale.

❖ Calibration process

Calibration in pH buffer solution corrects for this change. Calibration of any pH e equipment should always begin with buffer 7.0 as this is the "zero point." The pH scale has an equivalent mV scale. The mV scale ranges from +420 to -420 mV. At a pH of 7.0 the mV value is 0. Each pH change corresponds to a change of ±60 mV. As pH values become more acidic the mV values become greater.

So, PH meter calibration is a necessary step of using a PH meter to get an accurate measurement because of how the electrode is affected over time. So, we can consider that the typical calibration procedure consisting of **the following steps:**

- At first take the sensor out of the system and clean its electrode.
- Move the sensor electrode in the PH sample (which equals to 4) and wait the reading to be stable, the code will optimize the readings to 4.
- Take the PH sensor and move its electrode in the PH sample (which equals to 7) and wait the reading to be stable, the code will adjust the sensor to 7.
- Clean the sensor and put it back in the system, calibration is done.

2.5.2. EC Sensor

The electrical conductivity sensor (EC sensor) measures the electrical conductivity in a solution which usually used for aquaculture and water quality testing. The Grove - Electrical Conductivity Sensor is specially designed for a low-cost system with a relatively high accuracy which can cover most applications. The Grove connector and BNC probe interface make it easy to use and very suitable for Arduino and Raspberry Pi project.

❖ Feature

- Widely used for most applications of aquaculture and water quality testing.
- Compact size for easy deployment.
- Support with both Arduino and Raspberry Pi.
- Cost-effective.

Figure 13: EC sensor.

2.5.3. Temperature-Humidity Sensor (DHT11) module

Humidity sensors are electronic devices that measure and refers to the moisture and air temperature of the surrounding environment where they are deployed e.g., in air, or confined spaces.

Humidity measurements indicate the concentration of water vapor presented in the air. They provide their measurements in the form of a proper electronic signal. Moreover, they also refer to relative humidity i.e., the ratio of moisture in the air to the maximum moisture at a given temperature. and these measurements can be critical since the higher the humidity, the warmer it may seem. humidity measurement is often important because it can affect the health of plants. As a result, temperature and humidity sensors are often quite important.

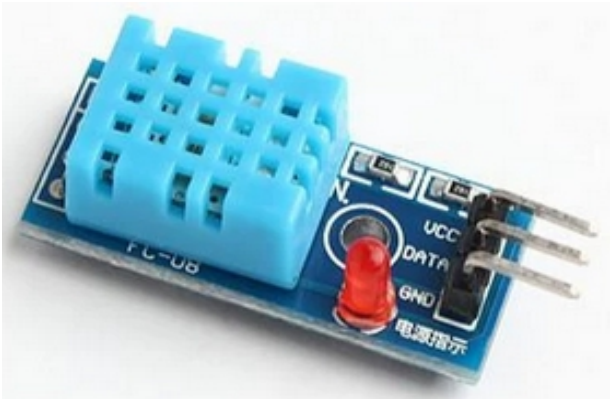


Figure 14: DHT sensor.

❖ Features

- Repeatability
- Accuracy
- Long-term stability
- Interchangeability
- Ability to recover from condensation
- Resistance to physical and chemical contaminants
- Packaging and suitable size.
- Cost effectiveness.

2.5.4. Waterproof Temperature Sensor

This digital sensor allows us to precisely measure the temperature of solution in the solutions tank to make sure that the solution temperature is suitable for the plants roots and don't be harmful for it .

Figure 15: Waterproof Temperature Sensor.

2.5.5. Ultrasonic Sensor

It used for measuring the level of water and solution tanks in our project, Ultrasonic sensors are reliable, cost-effective instruments for these applications. In operation, the sensor is mounted over the water. To determine the distance to the water, it transmits a sound pulse that reflects from the surface of the water and measures the time it takes for the echo to return.

Sometimes structural components, such as a small pipe, are located in the acoustic path between the ultrasonic sensor and the water. They can reflect a portion of the sound and produce a false echo that interferes with the ability of the sensor to properly detect the echo from the surface of the water. Advanced ultrasonic sensors can be adjusted to ignore these false echoes, which therefore enables them to provide accurate water level measurements.

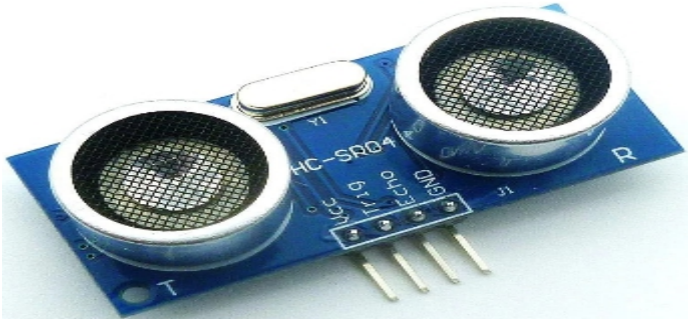


Figure 16: Ultrasonic Sensor.

2.5.6. The PCF8574 I2C

This chip is used to convert I2C serial data to parallel data for the LCD 20x4 to display all the possible readings of sensors, so we make a benefit of this to decrease the required pins from 6 to 2 pins.

Also, the push buttons to handle the LCD and user interface. The ESP8266 has a Wi-Fi module, so it can reach the web application to upload the readings if needed and receive any critical command from the server.

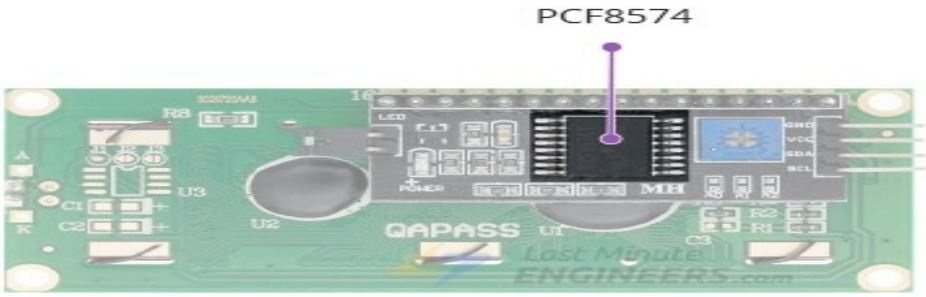


Figure 17: I2C Module connected to LCD.

❖ Features:-

- Operating Voltage: 5V
- Serial I2C control of LCD display using PCF8574
- Come with 2 IIC interface, which can be connected by Dupont Line or IIC dedicated cable
- Compatible for 16x2 or 20x4 LCD
- This is another great IIC/I2C/TWI/SPI Serial Interface
- With this I2C interface module, you will be able to realize data display via only 2 wires.

2.6. Real Time Clock (RTC) DS1307 module

It is an electronic device IC with an 8-pin using an I2C interface that keeps track of the current time and can be used in any device which needs to keep accurate time, this feature is critical because in many cases CPU is operating some delicate tasks like receiving sensors data. It also manages all timekeeping functions and features a simple two-wire I2C interface which can be easily interfaced with any micro-controller.

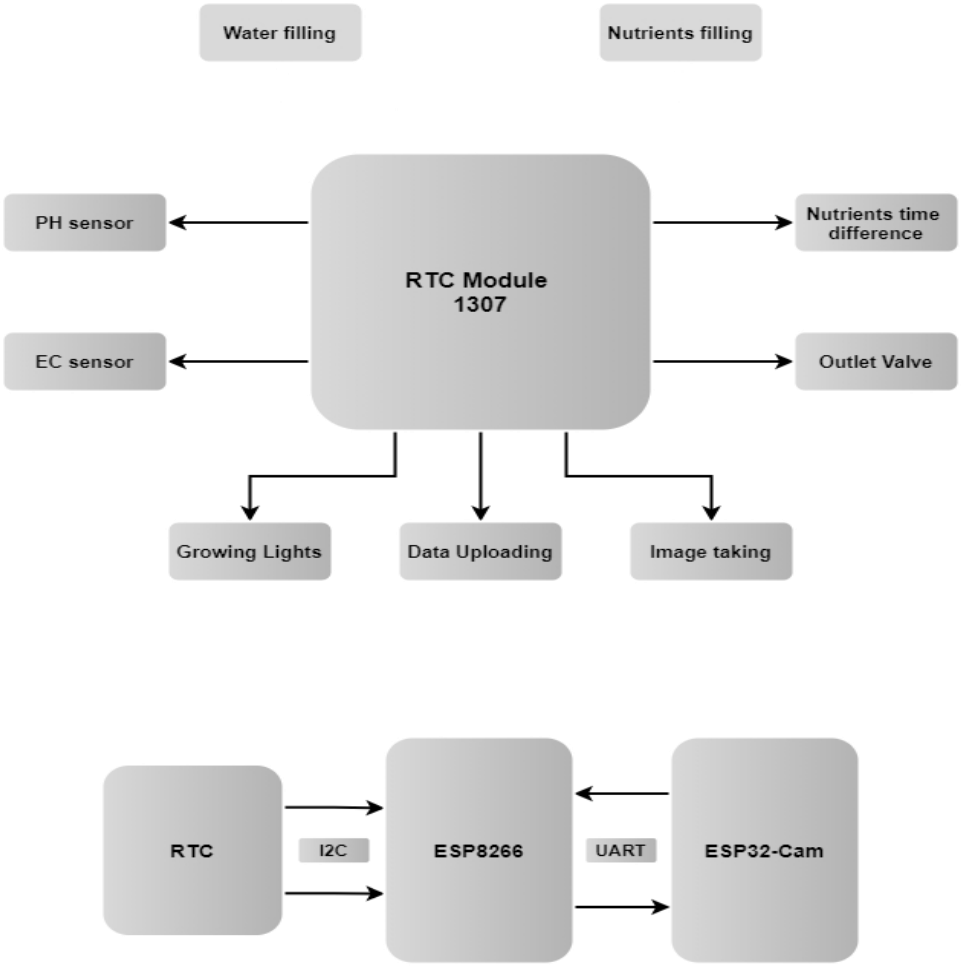


Figure 18 :RTC connections diagram.

The chip used to maintain seconds, minutes, hours, day, date, month, and year information and display it on the LCD by converting the bits into decimal by the following **binary-coded decimal** (**BCD**) **equation** which is a class of binary encodings of decimal numbers where each digit is represented by a fixed number of bits, usually four or eight. Sometimes, special bit patterns are used for a sign or other indications (e.g., error or overflow).

Second converting equation for example:

$$second = (second >> 4) * 10 + (second \ \& \ 0x0F)$$

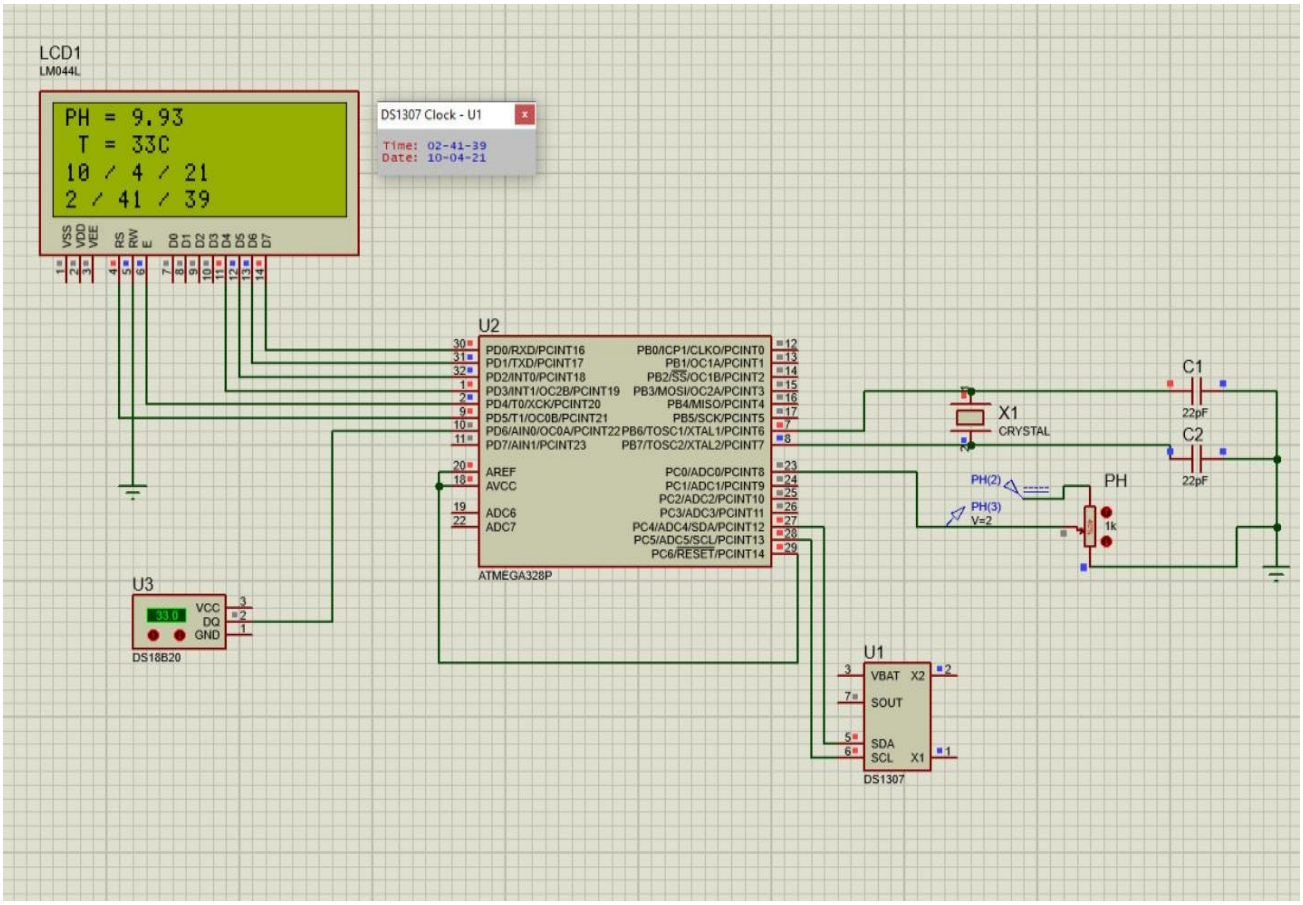
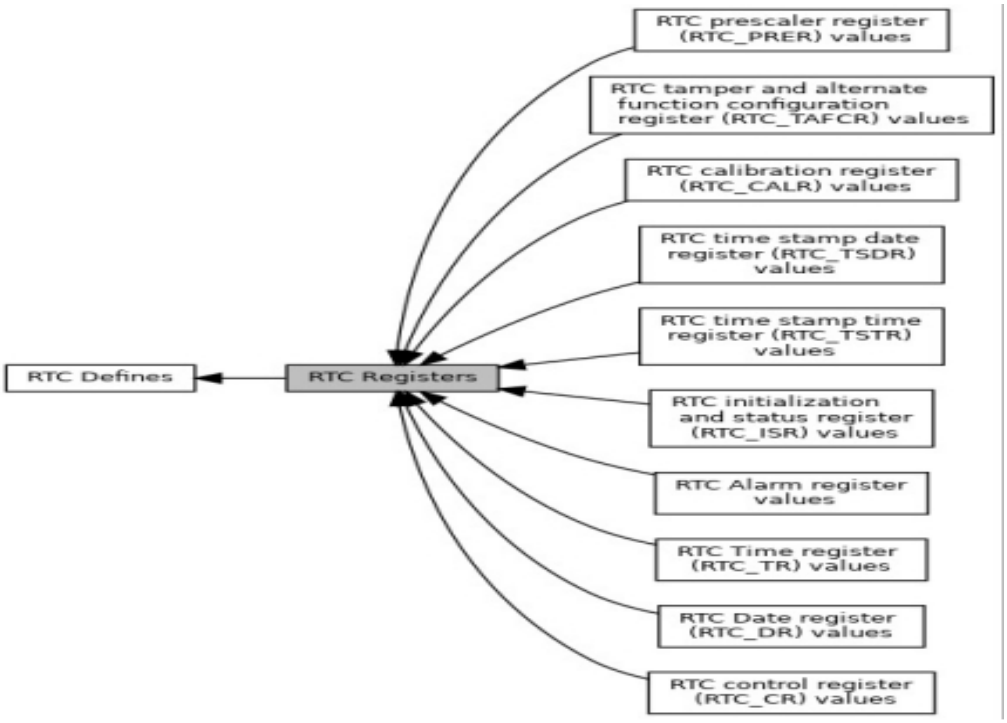


Figure 19: Connections between ATmega328 and other I/O.

2.6.1. Steps for Using RTC with its registers



1. Initialize RTC

- Disable RTC clock using CCR (Clock Control Register)
- Reset clock using CCR register
- Enable RTC calibration in RTC Calibration Register
- Enable the clock for RTC using CCR register

2. Set Date and Time

As we saw previously, there are separate Time Counter registers for each time parameter hour, Min, sec and same is the case date. We just need to copy the required values to these registers.

3. Read Date and Time

The values of date and time can be read from associated Time Counter registers. Alternately, date and time can also be read from Consolidated Time registers.

We will use Time Counter registers for both reading and writing.

2.6.2. Pin Description of RTC DS1307

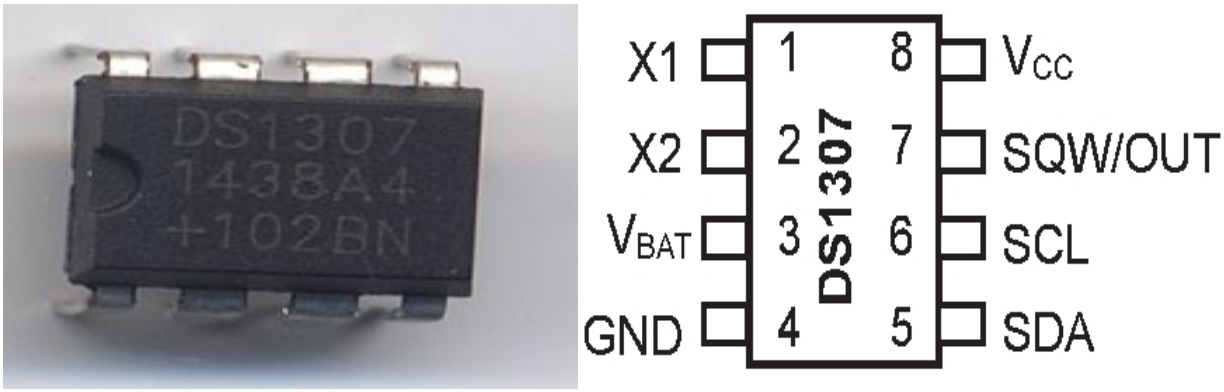


Figure 20: RTC pin configuration.

- Pin 1, 2: Connections for standard 32.768 kHz quartz crystal. The internal oscillator circuitry is intended for operation with a crystal having a specified load capacitance of 12.5pF. X1 is the input to the oscillator and can alternatively be connected to an external 32.768 kHz oscillator. The output of the internal oscillator, X2 is drifted if an external oscillator is connected to X1.
- **Pin 3:** Battery input for any standard 3V lithium cell or other energy source. Battery voltage should be between 2V and 3.5V for suitable operation. The nominal write protect trip point voltage at which access to the RTC, and user RAM is denied is set by the internal circuitry as 1.25 x VBAT nominal. A lithium battery with 48mAhr or greater will back it up for more than 10 years in the absence of power at 25ºC. UL recognized to ensure against reverse charging current when utilized as a part of conjunction with a lithium battery.
- **Pin 4:** Ground.
- **Pin 5:** Serial data input/output. The input/output for the I2C serial interface is the SDA, which is open drain and requires a pull up resistor, allowing a pull up voltage up to 5.5V. Regardless of the voltage on VCC.
- **Pin 6:** Serial clock input. It is the I2C interface clock input and is used in data synchronization.

- **Pin 7:** Square wave/output driver. When enabled, the SQWE bit set to 1, the SQW/OUT pin outputs one of four square-wave frequencies (1Hz, 4 kHz, 8 kHz, and 32 kHz). This is also open drain and requires an external pull-up resistor. It requires application of either Vcc or Vb at to operate SQW/OUT, with an allowable pull up voltage of 5.5V and can be left floating, if not used.
- **Pin 8:** Primary power supply. When voltage is applied within normal limits, the device is fully accessible, and data can be written and read. When a backup supply is connected to the device and VCC is below VTP, read and writes are inhibited. However, at low voltages, the timekeeping function still functions.

❖ RTC Aadvantages.

1. Low power consumption.
2. High accuracy.
3. Releasing system time from time calculation.
4. Two-wire interface (I2C)
5. We can find these RTCs in many applications like embedded systems and computer mother board.
6. Battery Backup: arrangement of battery backup which keeps the clock/calendar running even if there is power failure. An exceptionally little current is required for keeping the RTC animated.

Figure 21: CR1220 3V lithium metal coin cell battery.

2.7. The ESP32-CAM

It is a small size, low power consumption camera module based on ESP32 WIFI development board. It comes has an OV2640 camera and it also features a micro-SD card slot that can be useful to store images taken with the camera or to store files to serve to clients.

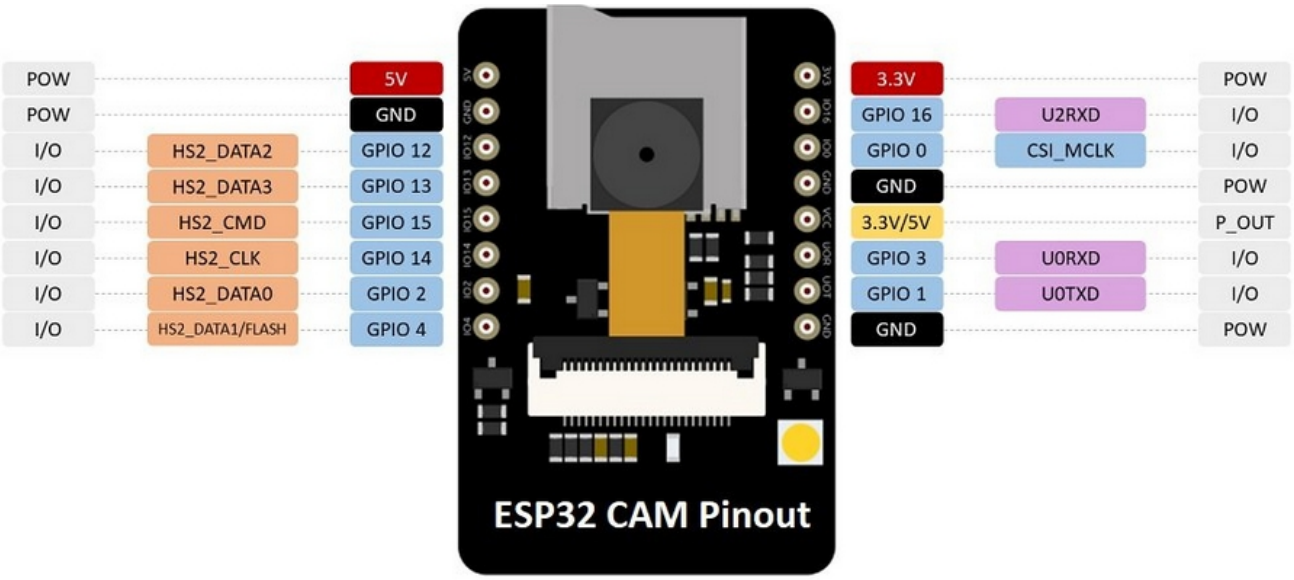


Figure 22: ESP32 CAM pin configuration.

ESP32-CAM has OV-2640 camera implemented, and different resolutions can be used, but the highest is 1600 × 1200. All available **r**esolutions are rectangular and not quadratic. But machine learning algorithms usually use low-resolution pictures, so it is recommended to set camera capture resolution to the lowest levels. using two lines I2C communication protocol to share information. One is used for the clock signal (**SCL**) and the other is used to send and receive data (**SDA**).

2.7.1. How ESP32-CAM work?

- The ESP32-CAM is in deep sleep mode.
- Press the RESET button to wake up the board.
- The camera takes a photo.
- The photo is saved in the microSD card with the name: picture X.jpg, where X corresponds to the picture number.
- The picture number will be saved in the ESP32 flash memory so that it is not erased during RESET, and we can keep track of the number of photos taken.

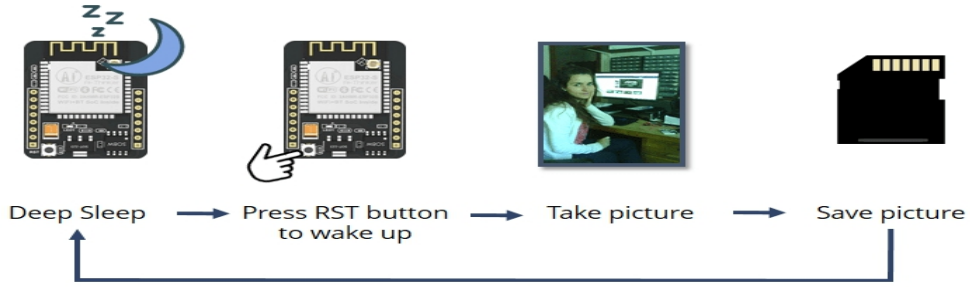


Figure 23: ESP32 CAM capturing image.

2.7.2. Connections

Esp32 CAM connected to ESP8266 micro controller by serial communication protocol called UART which detailed later, it also has been provided with the built-in camera module and internal flash. And connected with fan, water pumps, nutrient pumps which will be used when the esp8266 send a specific signal through the UART communication protocol Tx pin to the esp32 UART communication protocol Rx pin for taking a specific action with the connected components or to capture image of plant's state then upload it to the server through its Wi-Fi module.

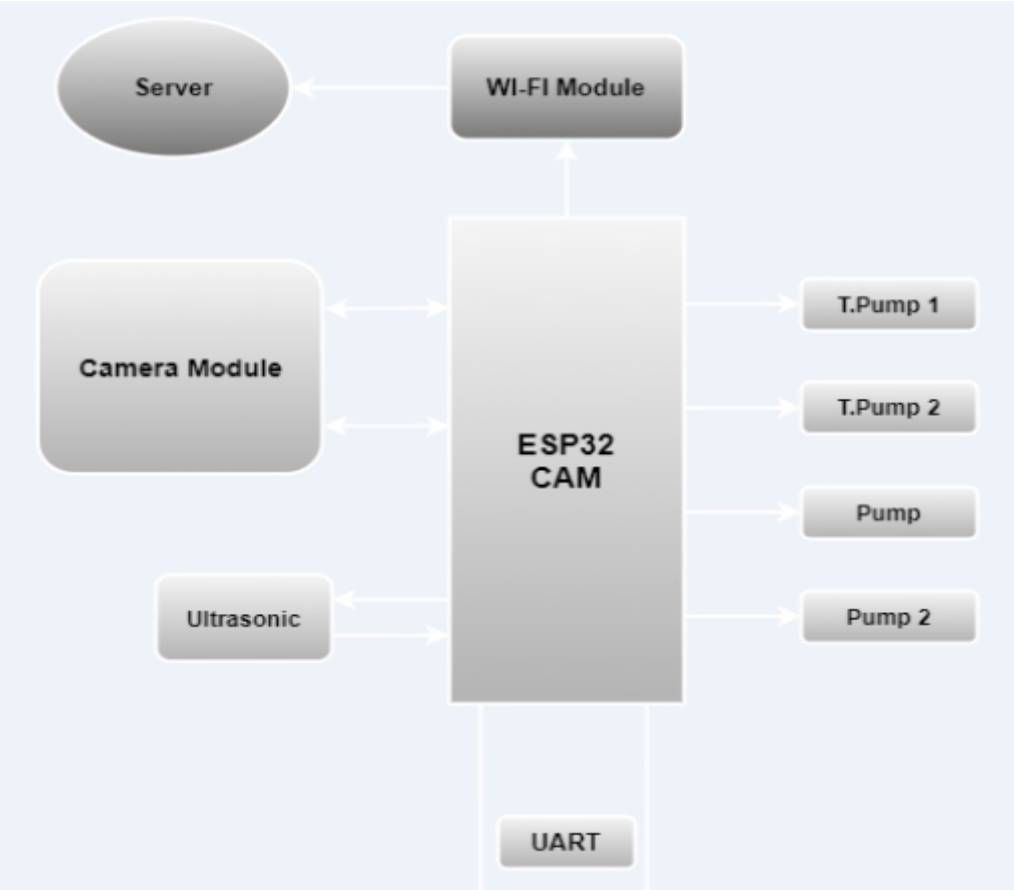


Figure 24: ESP32 connections diagram.

2.8. Serial Communication

In telecommunication and data transmission, serial communication is the process of sending data one bit at a time, sequentially, over a communication channel or computer bus. This is in contrast to parallel communication, where several bits are sent as a whole, on a link with several parallel channels.

Figure 25: Data Transfer in serial communication.

2.8.1. Serial Communication Modes

Serial data can be transferred in two modes:

1- Synchronous transmission.

In this communication mode there is a clock line which makes the synchronization. And another line is used for data transmission between the two devices, the data is sampled depending upon clock pulses, and since the clock sources are very reliable, so there is much less error in synchronous as compared to asynchronous.

Synchronous transmission is Preferred for short-distance transmission. Not used for long-distance transmission because sending the clock signal with the data signal is not an appropriate solution for the additional costs and clock skew.

Synchronous communication is usually more efficient because only data bits are transmitted between sender and receiver (no overhead). But can be more costly if extra wiring and circuits are required to share a clock signal between the senders and receiver

Figure 26: Synchronous transmission.

2- Asynchronous transmission.

In this communication mode the data bits are not "synchronized" with a clock line. There is no clock line at all. But The sender and receiver must agree use same data rate in advance and special bits (start bit, data bits, parity bit, stop bit) are added to each word which are used to synchronize the sending and receiving devices. When the transmitter pauses because it does not have data to transmit (idle state), it keeps a sequence of stop bits (logic high) in its output.

Asynchronous transmission Preferred for long distance communications because of Low cost because there is no wires for clock signals.

Figure 27: Asynchronous transmission.

2.8.2. Serial Communication Protocols

A variety of communication protocols have been developed based on serial communication in the past few decades. Some of them are:

- **I2C Protocol:** The term I2C stands for Inter Integrated Circuit, and it is a serial protocol which is used to connect low speed devices such as EEPROMs, micro controllers, A/D converters, etc. PIC micro controller support two wire Interface or I2C communication between two devices which can work as both Master and Slave device.
- **SPI Protocol:** The term SPI stands for Serial Peripheral Interface. This protocol is used to send data between PIC micro controller and other peripherals such as SD cards, sensors, and shift registers. PIC micro controller support three wire SPI communications between two devices on a common clock source. The data rate of SPI protocol is more than that of the USART
- **Fire Wire**–Developed by Apple, they are high-speed buses capable of audio/video transmission. The bus contains several wires depending upon the port, which can be either a 4-pin one, or a 6-pin one, or an 8-pin one.

2.8.3. The I2C Protocol

The I2C, or Inter-Integrated Circuit, is a communication protocol that we used with the LCD 20x4, and RTC module which connected to the ESP8266 for controlling sensors, pumps, Growing light, and capturing the plants image using ESP32 Camera module. So, we can say that it is often used in embedded systems as a way to transfer data between a master (or multiple masters) and a single slave (or multiple slaves) device. It is a bidirectional two-wire serial bus that uses serial clock (SCL) and serial data (SDA) wires to send and manage data between devices connected to the bus.I2C is a suitable protocol for connecting short-distanced, low-speed peripherals on printed circuit boards (PCBs). Common I2C applications can include reading memory, reading hardware sensors, and accessing DACs and ADCs.

The i2C protocol is used to connect devices like micro controllers, EEPROMs, I/O interfaces, and other peripheral devices in an embedded system. A micro controller is often used as the master device, and other peripheral devices are used as slave devices. Because all communication takes place on only two wires, all devices must have a unique address to identify it on the bus. By using the unique address, the master device can signal it's read/write command to exchange communication between the two over the bus.

Because I2C operates using a serial clock (SCL), it considered to be synchronous, which allows the output of bits to be synchronized to the sampling of bits by a clock signal shared between the master and the slave.

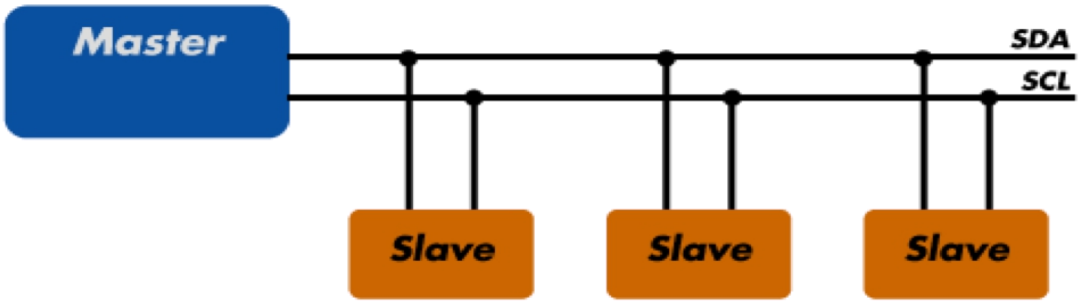


Figure 28: Sample I2C Implementation.

2.8.4. Universal Asynchronous Receiver/Transmitter (UART)

It is communication protocol plays a big role in organizing communication between devices. It is designed in different ways based on system requirements, and these protocols have a specific rule agreed upon between devices to achieve successful communication. Embedded systems, micro controllers, and computers mostly use UART as a form of device-to-device hardware communication protocol. Among the available communication protocols, UART uses only two wires for its transmitting and receiving ends.

By definition, the (UART) is a computer hardware device for asynchronous serial communication in which the data format and transmission speeds are configurable. The electric signaling levels and methods are handled by a driver circuit external to the UART. A UART is usually an individual (or part of an) integrated circuit (IC) used for serial communications over a computer or peripheral device serial port. One or more UART peripherals are commonly integrated in micro controller chips.

In UART communication, two UARTs communicate directly with each other. The transmitting UART converts parallel data from a controlling device like a CPU into serial form, transmits it in serial to the receiving UART, which then converts the serial data back into parallel data for the receiving device. Only two wires are needed to transmit data between two UART s. Data flows from the Tx pin of the transmitting UART to the Rx pin of the receiving UART.

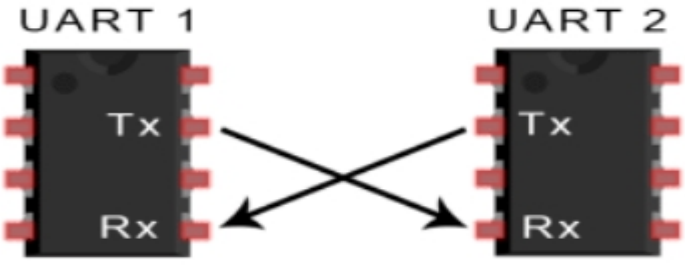
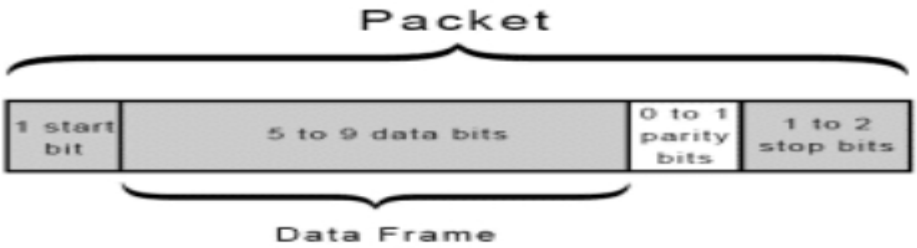


Figure 29: Data flows from the Tx pin of the transmitting UART to the Rx pin of the receiving UART.

UARTs transmit data asynchronously, which means there is no clock signal to synchronize the output of bits from the transmitting UART to the sampling of bits by the receiving UART. Instead of a clock signal, the transmitting UART adds start and stop bits to the data packet being transferred. These bits define the beginning and end of the data packet, so the receiving UART knows when to start reading the bits. When the receiving UART detects a start bit, it starts to read the incoming bits at a specific frequency known as the baud rate.

Baud rate is a measure of the speed of data transfer, expressed in bits per second (bps). Both UARTs must operate at about the same baud rate. The baud rate between the transmitting and receiving UARTs can only differ by about 10% before the timing of bits gets too far off. Both UARTs must also be configured to transmit and receive the same data packet structure.

for UART and most serial communications, the baud rate needs to be set the same on both the transmitting and receiving device. the set baud rate will serve as the maximum number of bits per second to be transferred.



- Start bit: To declare the start of transmission, always low „0".
- Data bits: 5,6,7, 8 or 9 bits of useful data bits.
- Parity bit: To check for transmission errors.
- Stop bit: one or two stop bits to declare end of frame, always high „1".

❖ Features

- Full Duplex Operation and Only uses two wires (Independent Serial Receive and Transmit Registers)
- No clock signal is necessary
- High Resolution Baud Rate Generator
- Has a parity bit to allow for error checking
- Supports Serial Frames with 5, 6, 7, 8, or 9 Data bits and 1 or 2 Stop Bits.
- Well documented and widely used method
- The structure of the data packet can be changed as long as both sides are set up for it.

- What are the Similarities and Differences Between I2C and UART?

Both I2C and UART are similar in that their purpose is to transfer data serial data at l low data rates between devices, and both use a two-wire interface to achieve this. Both are also commonly used for short/medium distanced data transmission. I2C, however, uses a master/slave configuration that uses clock signals to help synchronize the data being read or transmitted by the devices. UART, on the other hand, is hardware that is responsible for implementing asynchronous serial data streams for point-to-point connection and includes no clock signal.

❖ Advantages and Disadvantages to Using I2C vs UART

While both I2C and UART offer similar objectives in data transmission, there are reasons why one might be best over the other. I2C is flexible and useful for connecting multiple devices. It allows users to integrate multiple master and slave devices – up to 128 devices on a single bus. I2C also has flow control and performs data validation, so the integrity of the data and how it was transferred is reliable. I2C is also generally faster than UART and can reach speed of up to 3.4 MHz Some of the disadvantages of I2C include its increasing circuit complexity with additional master/slave setups, and is only able to operate in half-duplex, meaning data can only be transmitted in one direction at a time.

UART is often used for its simplicity in hardware implementations. It doesn't require a clock signal and is ideal for systems that need to send data between devices without waiting for a master poll request and generating unwanted traffic on the bus. Some drawbacks are that UART doesn't offer multiple master/slave support, which can limit how many devices are used on the bus. Additionally, each UART baud rate should be in 10% of each other or else data can be corrupted.

Chapter 3 | Deep Learning.

“Plant image classification System”

3.1. Plant disease classification problem

We want to make an automated system designed to help identify plant diseases by the plant appearance and visual symptoms which could be of help to amateurs and hobbyists who might want to grow their own garden or food and trained professionals as a verification system in disease diagnostics. The embedded system will send a signal regularly that triggers a camera to take a picture of the plant then pass it to our algorithm to perform classification of the image, based on the result of the classification a notification will be sent back to the user with the appropriate information, also users can take pictures manually at any time and pass it to the model for diagnostics there are different techniques to implement the previous task we will talk about them and which technique did we choose and why we choose it.

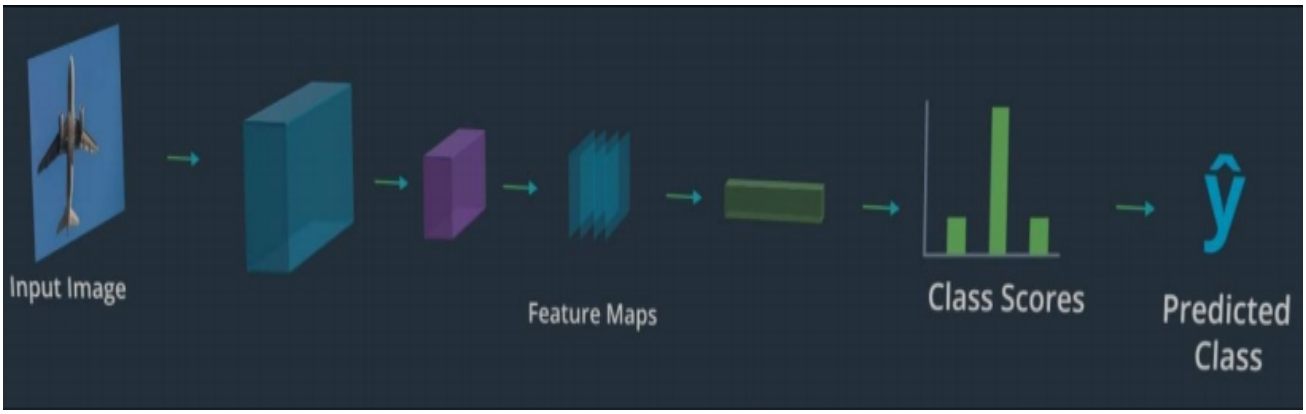


Figure 30: CNN for Image Classification

3.2. Deep learning vs traditional computer vision

3.2.1. The traditional approach

The traditional approach is to use well-established CV techniques such as feature descriptors (SIFT, SURF, BRIEF, etc.) for object detection. Before the emergence of DL, a step called feature extraction was carried out for tasks such as image classification. Features that are interesting, descriptive or informative patches in images. Several CV algorithms, such as edge detection, corner detection or threshold segmentation may be involved in this step. As many features as practicable are extracted from images and these features form a definition (known as a bag-of-words) of each object class.

At the deployment stage, these definitions are searched for in other images. If a significant number of features from one bag-of-words are in another image, the image is classified as containing that specific object (i.e., chair, horse, etc.). The difficulty with this traditional approach is that it is necessary to choose which features are important in each given image. As the number of classes to classify increases, feature extraction becomes more and more cumbersome. It is up to the CV engineer's judgement and a long trial and error process to decide which features best describe different classes of objects. Moreover, each feature definition requires dealing with a plethora of parameters, all of which must be fine-tuned by the CV engineer.

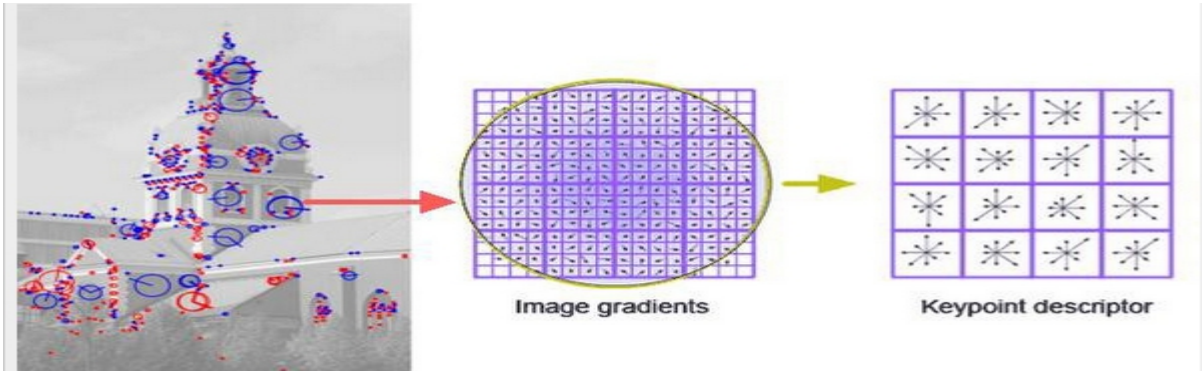


Figure 31: Traditional way of feature extraction

3.2.2. Deep learning approach

DL is a subset of machine learning. DL is based largely on Artificial Neural Networks (ANNs), a computing paradigm inspired by the functioning of the human brain. Like the human brain, it is composed of many computing cells or 'neurons' that each perform a simple operation and interact with each other to make a decision. Deep Learning is all about learning or 'credit assignment' across many layers of a neural network accurately, efficiently and without supervision.

To gain a fundamental understanding of Deep learning we need to consider the difference between descriptive analysis and predictive analysis. Descriptive analysis involves defining a comprehensible mathematical model which describes the phenomenon that we wish to observe. This entails collecting data about a process, forming hypotheses on patterns in the data and validating these hypotheses through comparing the outcome of descriptive models we form with the real outcome. Producing such models is precarious however because there is always a risk of unmodelled variables that scientists and engineers neglect to include due to ignorance or failure to understand some complex, hidden or non-intuitive phenomena. Predictive analysis involves the discovery of rules that underlie a phenomenon and form a predictive model which minimize the error between the actual and the predicted outcome considering all possible interfering factors.

Deep learning rejects the traditional programming paradigm where problem analysis is replaced by a training framework where the system is fed a large number of training patterns (sets of inputs for which the desired outputs are known) which it learns and uses to compute new patterns.

Deep learning enables CV engineers to achieve greater accuracy in tasks such as image classification, semantic segmentation, object detection and Simultaneous object detection and Simultaneous Localization and Mapping (SLAM). Since neural networks used in DL are trained rather than programmed, applications using this approach often require less expert analysis and fine-tuning and exploit the tremendous amount of video data available in today's systems. DL also provides superior flexibility because CNN models and frameworks can be re-trained using a custom dataset for any use case, contrary to CV algorithms, which tend to be more domain specific. Deep learning introduced the concept of end-to-end learning where the machine is just given a dataset of images which have been annotated with what classes of object are present in each image. Thereby a DL model is trained on the given data, where neural networks discover the underlying patterns in classes of images and automatically works out the most descriptive and salient features with respect to each specific class of object for each object. It has been well-established that DNNs perform far better than traditional algorithms, albeit with trade-offs with respect to computing requirements.

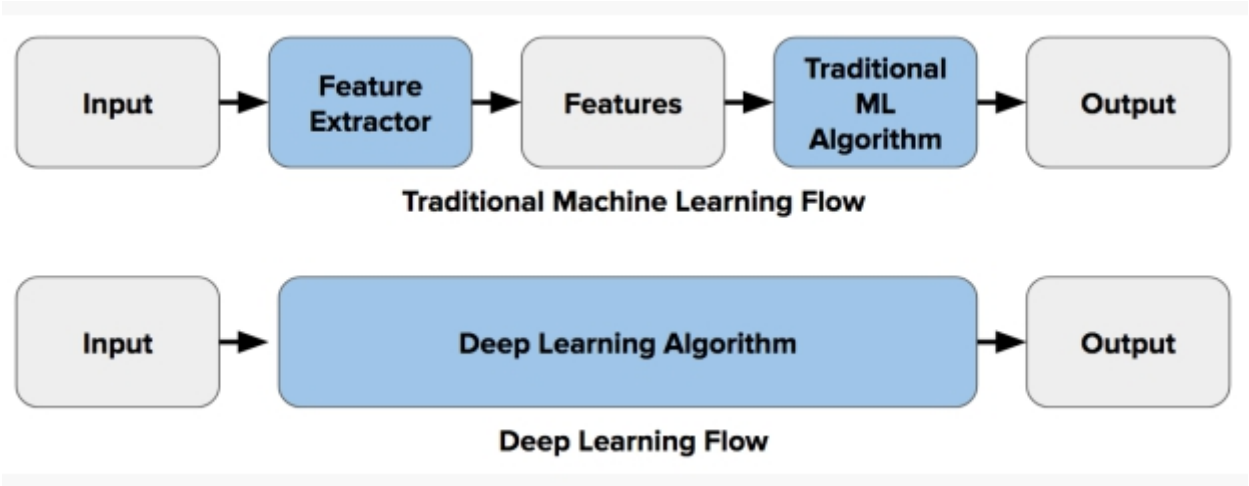


Figure 32: comparison between the traditional ML and Deep learning flow.

3.2.3. Neural networks

Deep neural network models were originally inspired by neurobiology. On a high level, a biological neuron receives multiple signals through the synapses contacting its dendrites and sends a single stream of action potentials out through its axon. The complexity of multiple inputs is reduced by categorizing its input patterns. Inspired by this intuition, artificial neural network models are composed of *units* that combine multiple inputs and produce a single output.

In Deep learning there are 3 types of neural network which are artificial neural network, Convolutional neural network, and Recurrent neural network, the first two are used in image classification tasks but we found that convolutional neural network suits better the image classification task because it preserves the spacial features while having far less parameters (weights) than artificial neural networks which make C-NN a faster deep learning therefore we will use C-NN to build our model and we will talk about it later in details.

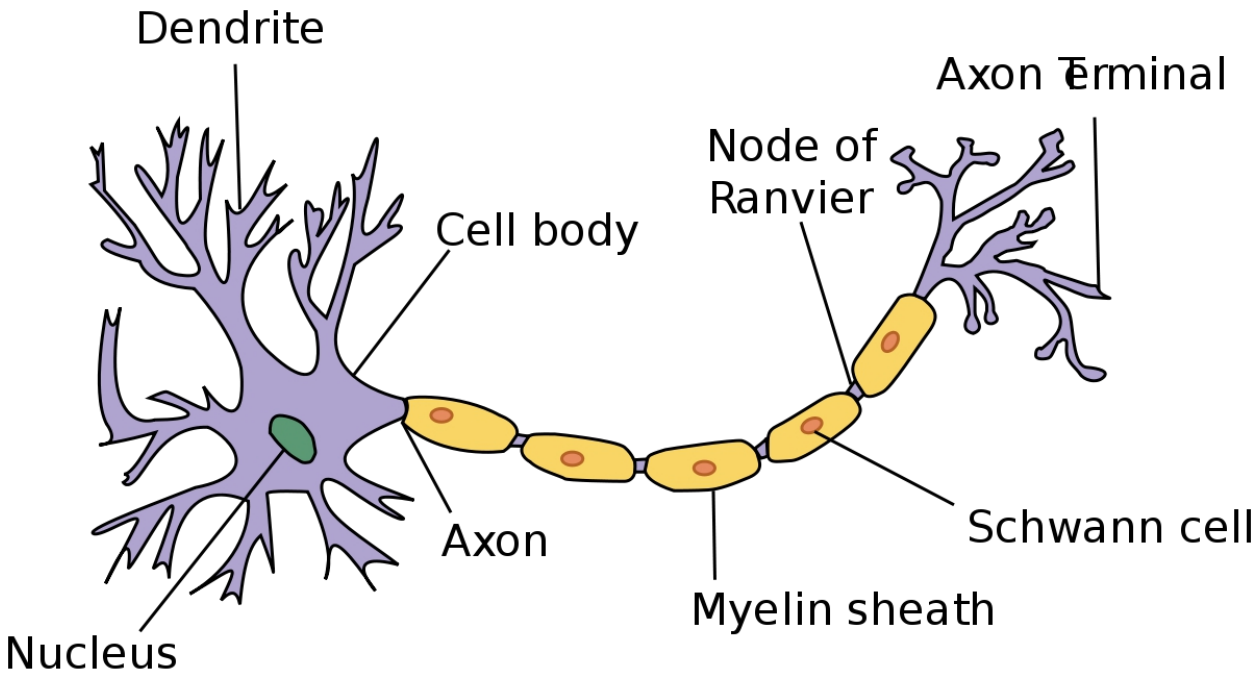


Figure 33: Biological Neuron.

3.3. Types

3.3.1. Artificial neural networks (ANNs)

They are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network. Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial intelligence, allowing us to classify and cluster data at a high velocity. Tasks in speech recognition or image recognition can take minutes versus hours when compared to the manual identification by human experts. One of the most well-known neural networks is Google's search algorithm.

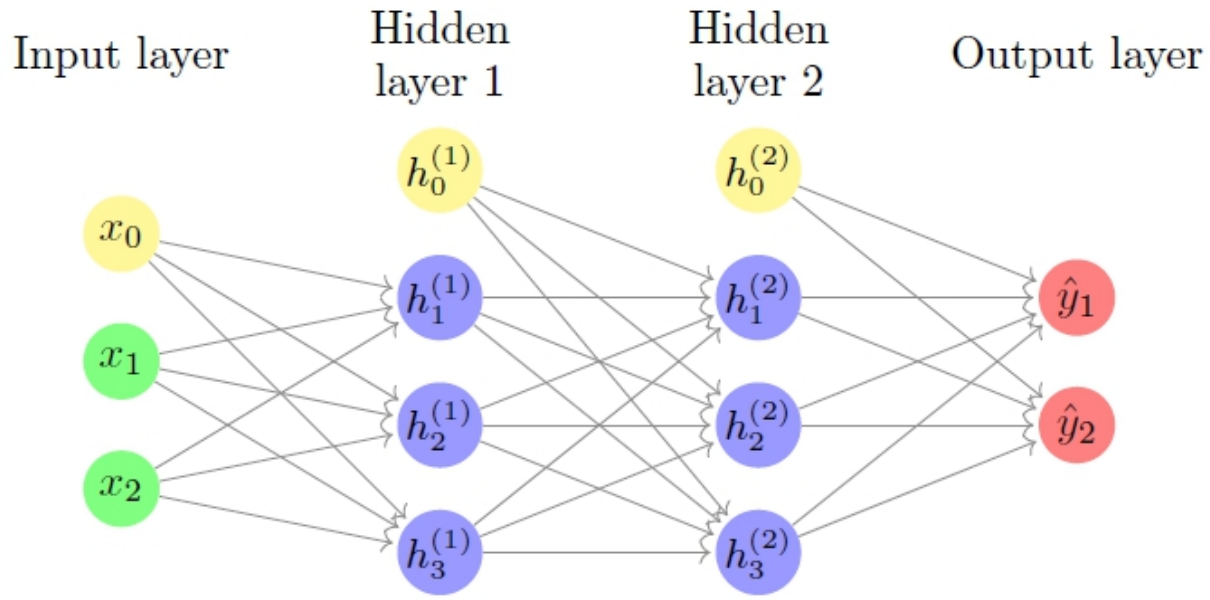


Figure 34: Artificial neural network (A-NN).

3.3.2. Convolutional neural network

It is a deep learning algorithm that is used with images, the architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlaps to cover the entire visual area. C-NN take the input image and pass it to each subsequent layer in which convolution process that is like the one used in digital signal processing but in 2-Dimensions is applied on the input image using layer's weights to extract features and learns the patterns that make a certain class.

3.4. Convolutional neural network (C-NN) Phases

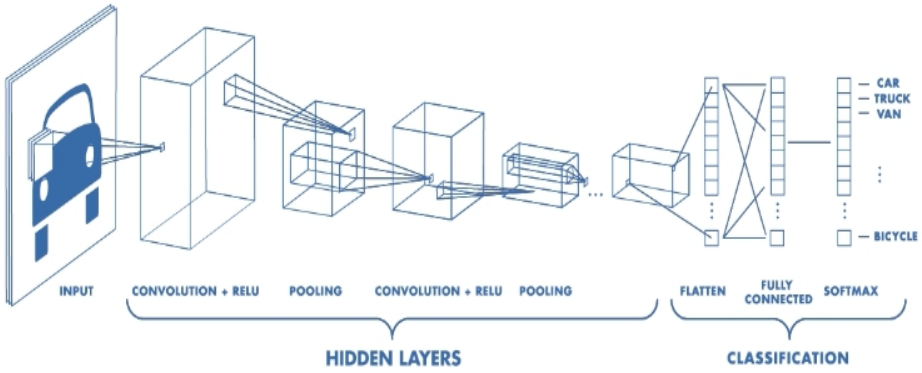


Figure 35: Convolutional neural network (C-NN)

To build our deep C-NN classifier there are 4 phases that we have to complete to have functional model that could be used in plants diagnostics these phases are:

- Dataset preparation and preprocessing
- C-NN model architecture
- Model training
- Model testing

3.4.1. Dataset preparation and preprocessing

In this phase we split the dataset into training data which is the sample of data used to fit the model on our classification task, validation dataset which is the sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyper parameters and testing data which is the sample of data used to provide the final evaluation on the model's fit on the training data, then process this data in a way that the model expects like making them of certain sizes and dimensions.

3.4.2. C-NN model architecture

This is the phase where we decide on the model's architecture, there are two approaches:

- Building a C-NN architecture from scratch as in implementing all the layers and initializing the weights to zero or randomly.
- Using a pre-trained architecture and its weights "Transfer learning".

3.4.3. Model training

Training data will be fed to the model and the model will extract features from it and tries to find the underlying relationships that makes the model classify an image to a certain class.

3.4.4. Model testing

Testing data will be used to measure the performance of the model and its accuracy in classifying the input images.

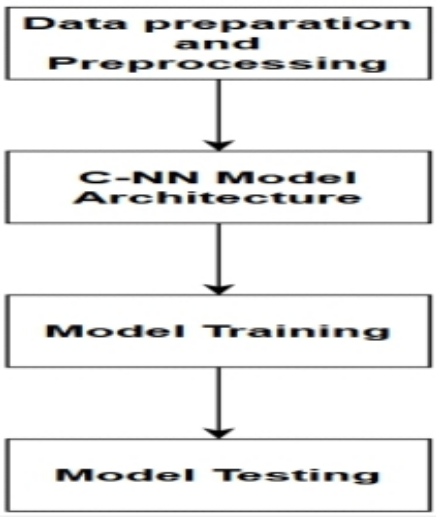


Figure 36: Phases of building Deep C-NN classifier

3.5. Steps

3.5.1. Dataset preparation and preprocessing

The backbone and the first step of any machine learning and deep learning algorithm is the data that is used to train and test the model, without the data the problem cannot be solved using said algorithms, In our plants classification problem we found a dataset that was uploaded on Kaggle called the plant village dataset that contain three types of vegetables: Tomato, pepper and potato each type is divided into folders that represents each class but for the sake of simplicity we will work with one type of plants which is tomato, tomato is divided into 10 classes which are healthy, bacterial spot, early blight, late blight, leaf mold, Septoria leaf spot, spider mite, target spot, curl virus and mosaic virus.

The plant village dataset is not split into train/validation/test but in order to evaluate the performance of the model accurately we need to test it against unseen data otherwise we can't be sure it didn't over fit to the data and ended up memorizing the images, therefore we wrote a python script to automatically splits the dataset for us into 3 splits: train, validation and test. The validation dataset also called the development dataset to not confuse it with the test dataset is used to evaluate the model's performance but during the training phase on the training data and can tune the model's parameters, but the test dataset is used when the model is completely trained and cannot change the model's parameters, this script also resizes the images into the desired size to match the model's input layer requirement.

3.5.2. Model's architecture

We chose the deep learning approach to build our model's architecture over any other approach because as we clarified earlier it will save us a huge amount of effort used in other approaches to manually extract features while deep learning architectures can do it on its own without the programmer interference there are various other approaches like support vector machine (SVM) and nearest neighbor (K-NN) but all of them require manual feature extraction.

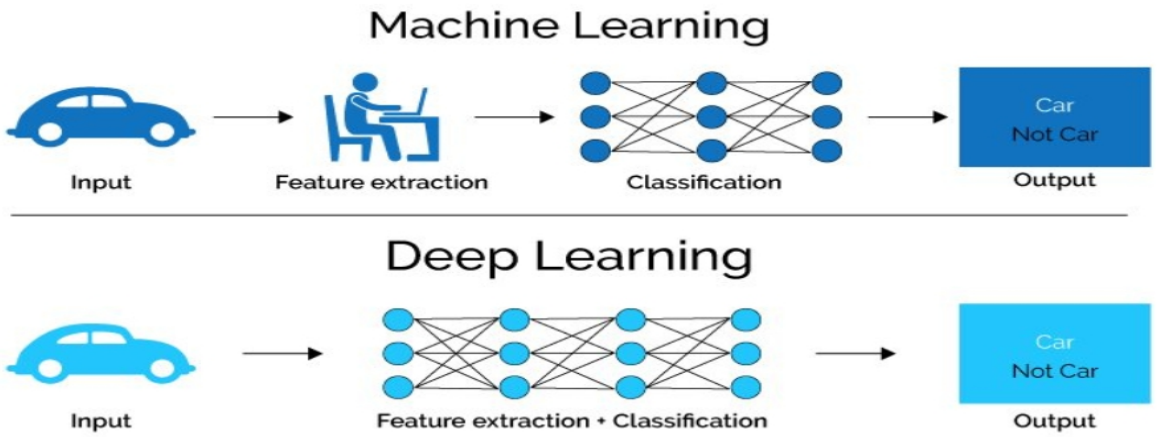


Figure 37: Advantage of deep learning over machine learning in classification task.

A-NN is a deep learning algorithm like C-NN and A-NN can be better than C-NN in some applications but in image classification C-NN will prevail, that is because every layer in A-NN is a fully connected layer which means that every neuron in the current layer receives input from every neuron in the previous layer, so when dealing with images that could lead to tremendous amount of parameters and huge increase in the complexity of the network making it harder to train and easier to over fit.

On the other hand, what makes C-NN so efficient is parameter sharing which means one filter has a constant number of parameters on multiple parts of the image leading to a huge decrease in the parameters, sparsity of connections also contributes in the efficiency of the C-NN because a filter applied on certain part of the image produces only 1 pixel as a result of the convolution process so each output depends on small number of inputs.

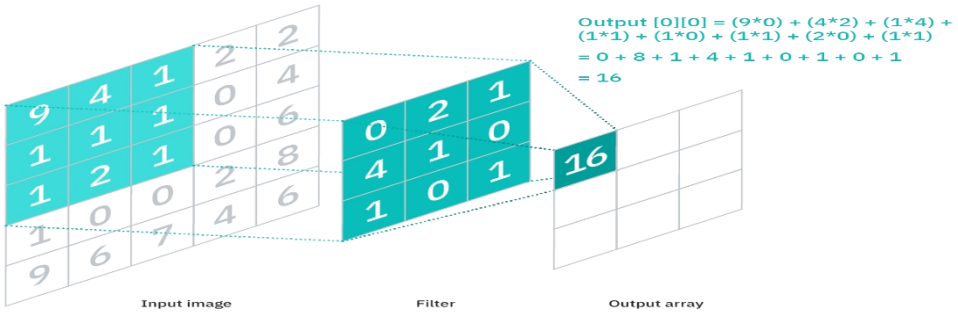


Figure 38: Output of the convolution process only depends on part of the input.

All the above reason prove that convolutional neural network is the more practical choice when dealing data that contains large number of features and produce large number of parameters such as image. Instead of building a C-NN architecture from scratch which will need huge amount of data that we don't have to train the network to an acceptable level of performance, we opted to make use of the transfer learning concept and use a pre-trained model with its custom weights that was produced during training on a similar problem, the architecture that we chose is the Mobile Net architecture from google which has much fewer parameters than vgg16 architecture for example and comparable performance to it, also it is a light weight architecture meaning that it can run on platforms with limited resources.

Table 8. MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

ImageNet Dataset

Figure 39: Comparison of popular models.

The architecture consists of 13 layers plus prediction layer, each layer consists of multiple operations which are convolution with zero padding, depth-wise separable convolution, batch normalization, normal convolution, and relu (rectified linear unit) activation function.

The most important operation out of the above and the responsible for the decreased computational power needs is the depth-wise separable convolution process where the normal convolution process is separated into two processes the depth-wise convolution process and the pointwise convolution process, in the depth-wise convolution process we convolve the input image with number of kernels equal to the input image depth resulting in a feature map that has the same depth as the input image, in the pointwise convolution method we convolve the feature map with a number of kernels of size (1X1Xfeature map depth) which will create a feature map with the desired depth.

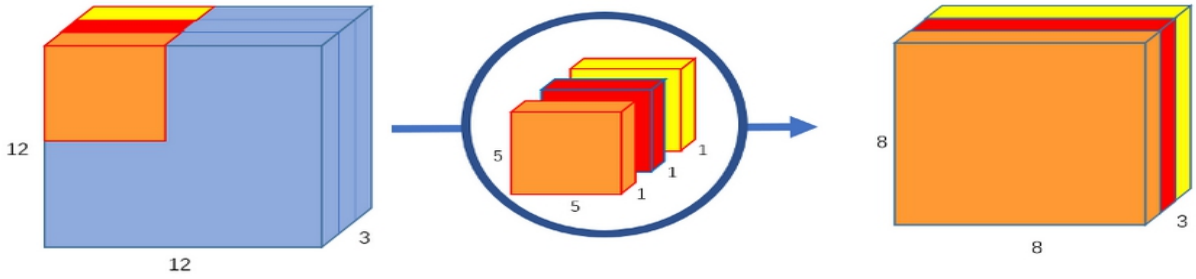


Figure 40: Depth-wise convolution process where a 12x12x3 convolved with three 5x5x1 filters to produce 8x8x3 feature map.

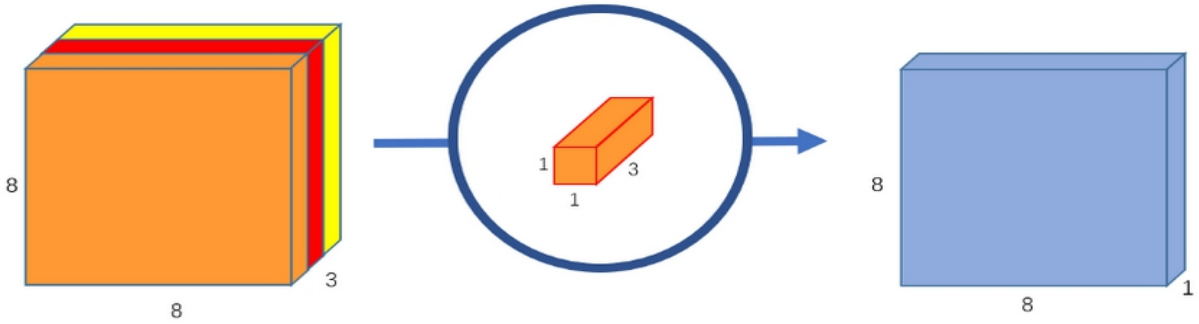


Figure 41: Point-wise convolution process where 8x8x3 feature map convolved with 1x1x3 filter produces 1-d feature map of same size 8x8.

The last layer consists of a flattening operation to turn feature map into 1-d vector then passed to a SoftMax function to predict the class of the input image out of 1000 classes but in our classification problem we do not have 1000 classes to predict from so we customize the architecture to our problem needs by removing the prediction layer and put our custom layer that suits our needs and we did that by adding a prediction layer with 10 nodes and a SoftMax activation function.

3.5.3. Model training

Now the model is nearly ready, we froze the model's parameters except for the last 23 layers, there is no reason why we ended up using that number, but it was a matter of trial and error this number of unfrozen layers resulted in the best training accuracy. We begin the training process by compiling the model to use Adam optimization algorithm which is an extension of combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems, we chose a learning rate of 0.0001 again this value was chosen based on trial with many other values and this value resulted in the best accuracy, the model's loss function is categorical cross entropy since we have more than 2 classes, lastly the model's performance evaluation metric will be accuracy.

After we compiled the model with the desired optimization algorithm, loss function and metric it will be ready to train so we call the fit function to start the training process for 11 cycles (epochs), we tried other number of cycles, but the model seems to diverge after 11 cycles (accuracy decreases), after the model finished training, we found the validation accuracy to be 96% which is a good accuracy.

3.5.4. Model testing

In this phase we test the model to see how well it perform on the test data, since the accuracy metric do not fully reflect how well the model is doing we need another evaluation metric which is confusion matrix, so we create confusion matrix with the prediction results after plotting the confusion matrix we were satisfied with the result so we saved the model in the h5 format using the save function which saves the architecture, the weights, the training configurations(loss, optimizer) and the state of the optimizer, allowing to resume training exactly where we left off.

3.6. Deep Learning Model testing Steps

1. choose a random image from the test dataset and perform the required preprocessing on it.

```
imgPath, tClass = ChooseTestImage(testPath)
```

```
# Display the random image before preprocessing
from IPython.display import Image
print(tClass)
Image(filename=imgPath, width=224,height=224)
```

```
T_Spider_mites_Two_spotted
```

```
processedImage = prepare_image(imgPath)
```

2. Pass the preprocessed test image to the deep learning model and call its predict function on that image, this function will return an array with the probabilities of all the classes

```
prediction = TDCMv1.predict(processedImage) #Use the loaded model to predict the state of the tomato plant
```

```
prediction
```

```
array([[6.8516181e-07, 2.4595158e-04, 3.9868793e-04, 3.2310512e-05,
        1.0138028e-04, 3.3729866e-06, 3.0444555e-05, 9.9768376e-01,
        1.4961467e-03, 7.2421335e-06]], dtype=float32)
```

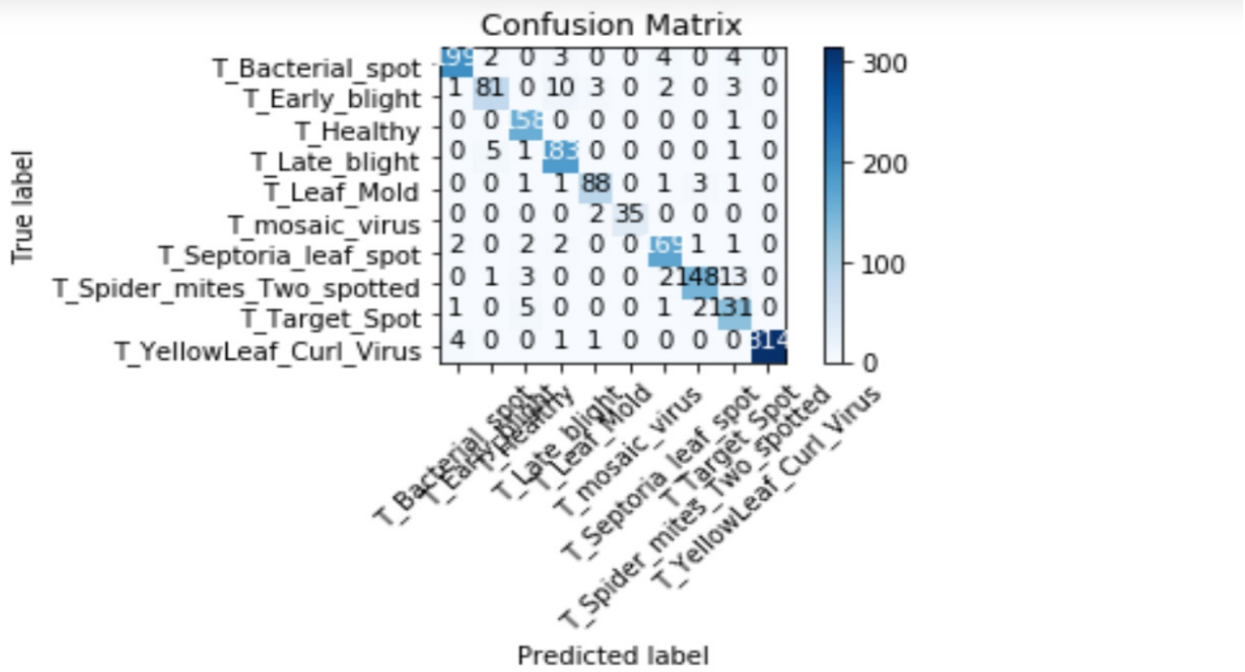
3. choose from that array the class with the highest probability then compare it with the true class of the image if it is the same class then the model has made a successful evaluation otherwise the model has failed in classifying the image.

```
predictedClass = classesDict[int(prediction.argmax(axis=1))]
```

```
print(f"True Class : {tClass}")
print(f"Predicted Class : {predictedClass}")
```

```
True Class : T_Spider_mites_Two_spotted
Predicted Class : T_Spider_mites_Two_spotted
```

❖ Model testing result on all test data (confusion matrix)



3.6 Deep learning model’ s visualization

We talked earlier about how deep learning algorithms introduced a concept of end-to-end learning in which we only provide the deep learning model with dataset of images that are labelled with the class that they belong and the neural network does the feature extraction and the prediction task without much interference from us, this can be a good thing especially if a person is not an expert in image processing and don’t know how to manually extract important features from an image that belong to a particular class he/she will still be able to build a functional deep learning model use it to infer the class of the image as long as he/she have enough and relevant data to train it with.

So basically what end-to-end learning means is that deep convolutional neural networks are a black box as you don't know how do they infer the class of the image and even if we studied the layers of the network we are not able to see a clear structure of the mathematical model being used to reach the prediction which is not a good thing because we humans love to know how stuff works and we want to learn from the deep learning model how it views and tries to solve the given task how t for safety reasons because in all fields using deep learning not know how a deep learning model view and try to solve its task could be problematic because we will not be able to control or modify it.

Figure 42: Neural networks as a black box

We couldn't settle with not knowing how our deep learning model solve our classification task so the solution we came to buy for that problem was to visualize the feature map (output) of each layer of the deep learning model in this way we can see how the image was altered at each layer and get a sense of what the kernels do.

3.6.1. Deep learning model visualization steps

1. we will choose a random picture from our dataset with known class.

T_Target_Spot

Figure 43: Test image that will be fed to the model visualizer.

2. In this step we will load our saved model, but we can't use it right away because the model's architecture as it is only outputting the result of the prediction and gives us no insight about how the deep C-NN infers the class.

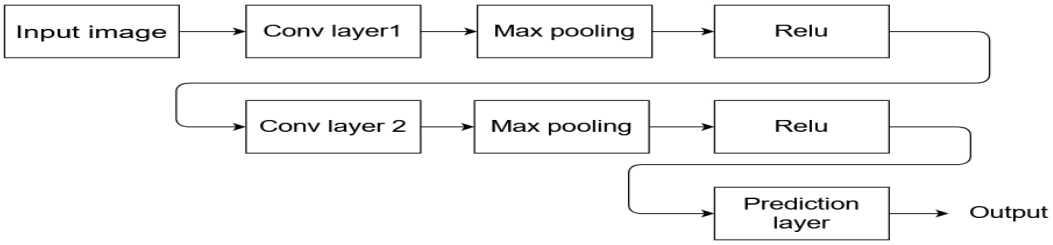


Figure 44: A simple C-NN model.

3. In this step we will modify the architecture of the model to output the result of each layer so that we can watch the resulted feature map and try to figure out what the kernels (filters) of this layer did.

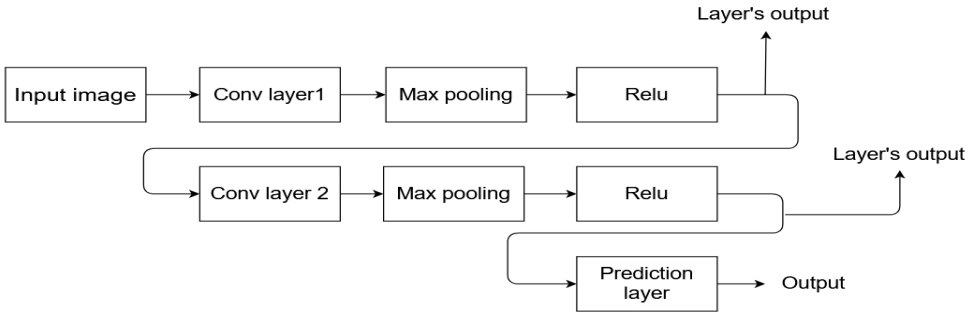


Figure 45: A simple C-NN model modified for visualization.

3.7. Results

After applying the test image to the modified deep learning model we noticed that in the first few layers bunch of the output feature maps are black which means the kernels responsible for these feature maps tries to detect something that is not in this image , some other kernels applies a form of thresholding on the image to the separate the object (leaf) from the background and finally some filter apply some form of high-pass filter to sharpen the image and detect edges.

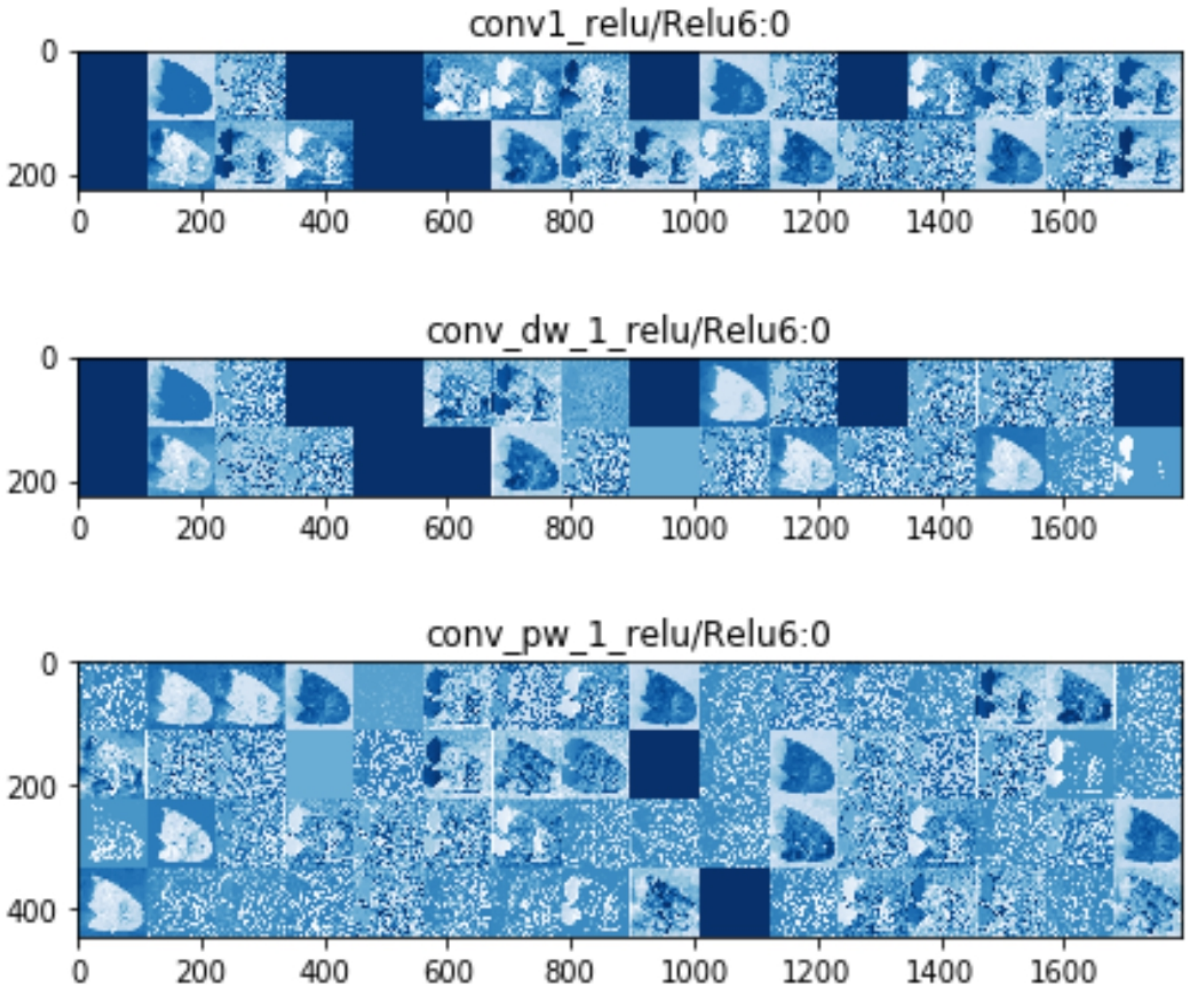


Figure 46: First few layers outputs visualized

So, this method helped us better understand what kernels of the first few layers does but unfortunately the deep the layer becomes the smaller the feature map gets which make it meaningless for us and we can't understand what the kernel of these layers does. This method is of great help with the first few layers, but its limitations is that it becomes useless when we try to visualize deeper layers' feature maps.

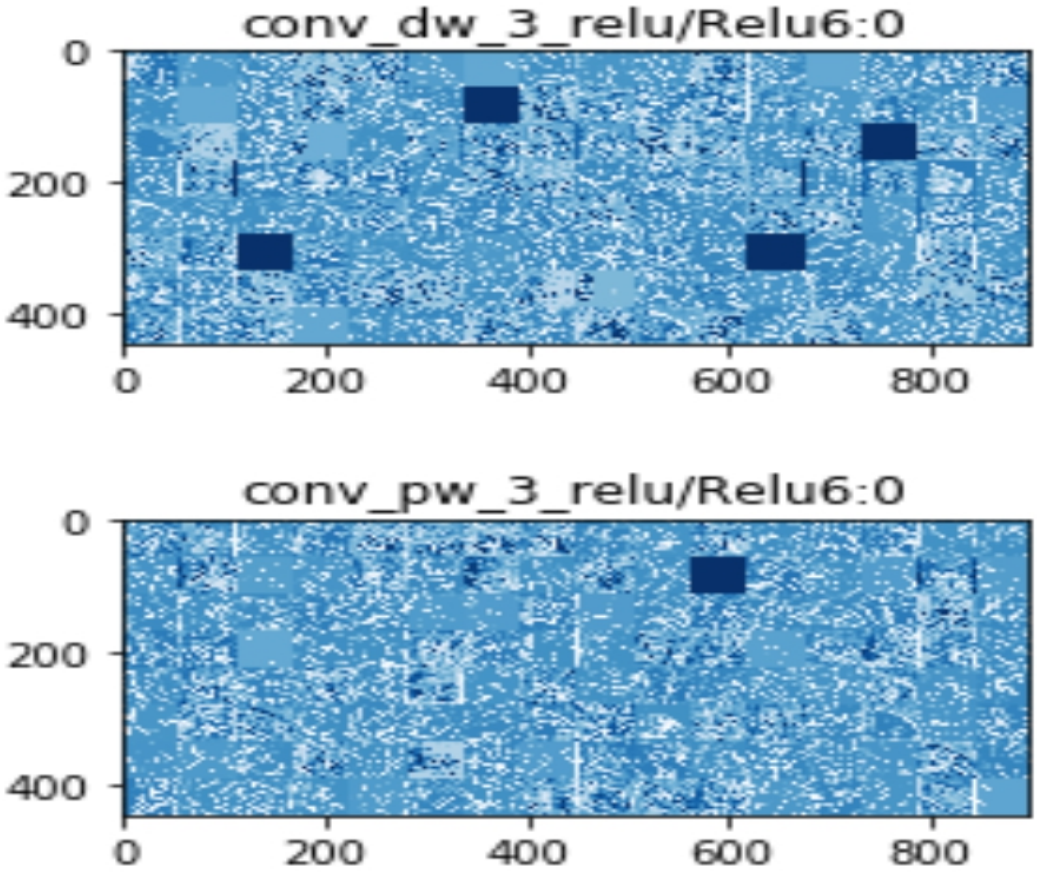


Figure 47: Deep layers visualized

4. Chapter| Django Web Application

4.1. Django web application

Django is a free and open-source web application framework written in Python. A framework is nothing more than a collection of modules that make development easier. They are grouped together and allow you to create applications or websites from an existing source, instead of from scratch.

With the uses of the Django framework, we can develop and deploy web applications within hours as it takes care of much of the hassle of web development. Django is very fast, fully loaded such as it takes care of user authentication, content administration, security as Django takes it very seriously and helps to avoid SQL injection, cross-site scripting etc. and scalable as applications can be scalable to meet high demands and used to build any type of applications that's why we call it as a versatile framework. We can build different applications from content management to social networking websites using the Django framework. It offers lots of resources and good documentation, which helps new learners to learn and experienced people for reference.

This web application is designed to help identify some of the more common plant diseases and provides friendly solutions. This web application is not just about detecting disease it also gives directions about how to avoid it and the best practice guide. It uses the deep learning model for image detection and classification of diseases. This feature can be a standalone application that targets everyone and helps them to grow their plants healthy.

4.2. Construction of the application

It is built using the Django framework which is a high-level Python Web framework that encourages rapid development and clean, pragmatic design.

It also supports the MVC pattern that is based on three components: Model, View, Control. The Model-View-Template (MVT) is slightly different from MVC. The main difference between the two patterns is that Django itself takes care of the Controller part (Software Code that controls the interactions between the Model and View), leaving us with the template. The template is an HTML file mixed with Django Template Language (DTL).

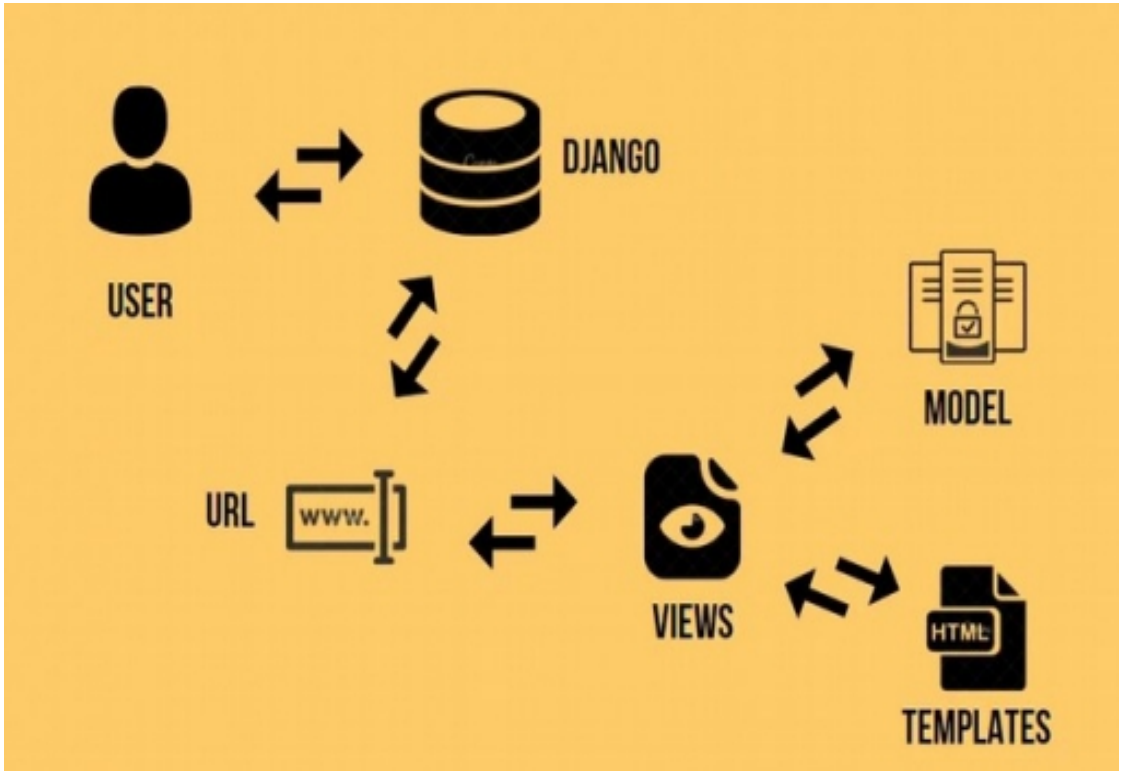


Figure 48 : View to render the form page.

4.3. Django Benefits

With the uses of the Django framework, we can develop and deploy web applications within hours as it takes care of much of the hassle of web development. Django is very fast, fully loaded such as it takes care of user authentication, content administration, security as Django takes it very seriously and helps to avoid SQL injection, cross-site scripting etc. and scalable as applications can be scalable to meet high demands and used to build any type of applications that's why we call it as a versatile framework. We can build different applications from content management to social networking websites using the Django framework. It offers lots of resources and good documentation, which helps new learners to learn and experienced people for reference.

1. **Fast development**

   Django was designed to help developers take applications from concept to completion as quickly as possible. It takes care of much of the hassle of Web development. It is also free and open source.

2. **Reassuringly secure**

   Django takes security seriously and helps developers avoid many common security mistakes.

3. **Exceedingly scalable**

   Some of the busiest sites on the Web leverage Django's ability to quickly and flexibly scale.

4. **Object-Relational Mapping (ORM) Support**

   Django provides a bridge between the data model and the database engine, and supports a large set of database systems including MySQL, Oracle, Postgres, etc.Django also supports NoSQL database through Django-nonrel fork. For now, the only NoSQL databases supported are MongoDB and google app engine.

5. Multilingual Support

Django supports multilingual websites through its built-in internationalization system. So you can develop your website, which would support multiple languages.

6. Framework Support

Django has built-in support for Ajax, RSS, Caching and various other frameworks.

7. Administration GUI

Django provides a nice ready-to-use user interface for administrative activities.

8. Development Environment

Django comes with a lightweight web server to facilitate end-to end application development and testing.

The used database is MySQL database which has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. It is the de-facto standard database system for websites with HUGE volumes of both data and end-users (like Face-book, Twitter, and Wikipedia). The great thing about MySQL is that it can be scaled down to support embedded database applications. A big advantage is that it has the feature of cross-platform support.

Django expects the database to support Unicode (UTF-8 encoding) and delegates to it the task of enforcing transactions and referential integrity. It is important to be aware of the fact that the two latter ones aren't actually enforced by MySQL when using the MyISAM storage engine.

MySQL has a couple of drivers that implement the Python Database API which are: mysql client and MySQL Connector/Python. The one that is used in this application is mysql client which is thread-safe and provides connection pooling. In addition to a DB API driver, Django needs an adapter to access the database drivers from its ORM.

Django provides an adapter for mysql client. Django abstracts the needed SQL by modelling data then migrate it to be applied to the database. The model formsets validate unique fields in a case-sensitive manner. Thus when using a case-insensitive collation, a formset with unique field values that differ only by case will pass validation, but upon calling save(), an Integrity Error will be raised. There are known issues in even the latest versions of MySQL that can cause the case of a table name to be altered when certain SQL statements are executed under certain conditions. It is recommended that you use lower case table names, if possible, to avoid any problems that might arise from this behaviour. Django uses lower case table names when it auto-generates table names from models, so this is mainly a consideration if you are overriding the table name via the db_table parameter. In this application, there exist two models. The Image model is to store the name and the directory of the image. By default, Django stores file locally, using the MEDIA_ROOT and MEDIA_URL settings.

The examples below assume that you're using these defaults. The file is saved as part of saving the model in the database, so the actual file name used on the disk cannot be relied on until after the model has been saved. The other model is the Recommendations model that is used to store the diseases and the recommendations to deal with them. It is used to compare and match the output of the model to fetch the needed record.

The Deep Learning model is based on MobileNets which is an efficient convolutional neural network for vision applications. MobileNets are based on a streamlined architecture that uses depth-wise separable convolutions to build lightweight deep neural networks. It introduces two simple global hyper-parameters that efficiently trade-off between latency and accuracy. These hyper-parameters allow the model builder to choose the right sized model for their application based on the constraints of the problem. It presents extensive experiments on resource and accuracy trade off and shows strong performance compared to other popular models on ImageNet classification. We then demonstrate the effectiveness of MobileNets across a wide range of applications and use cases including object detection, fine-grain classification, face attributes, and large-scale geo-localization.

The main advantage of the Mobile Network is the low computational cost and the small storage size which allows to embed it into the server.

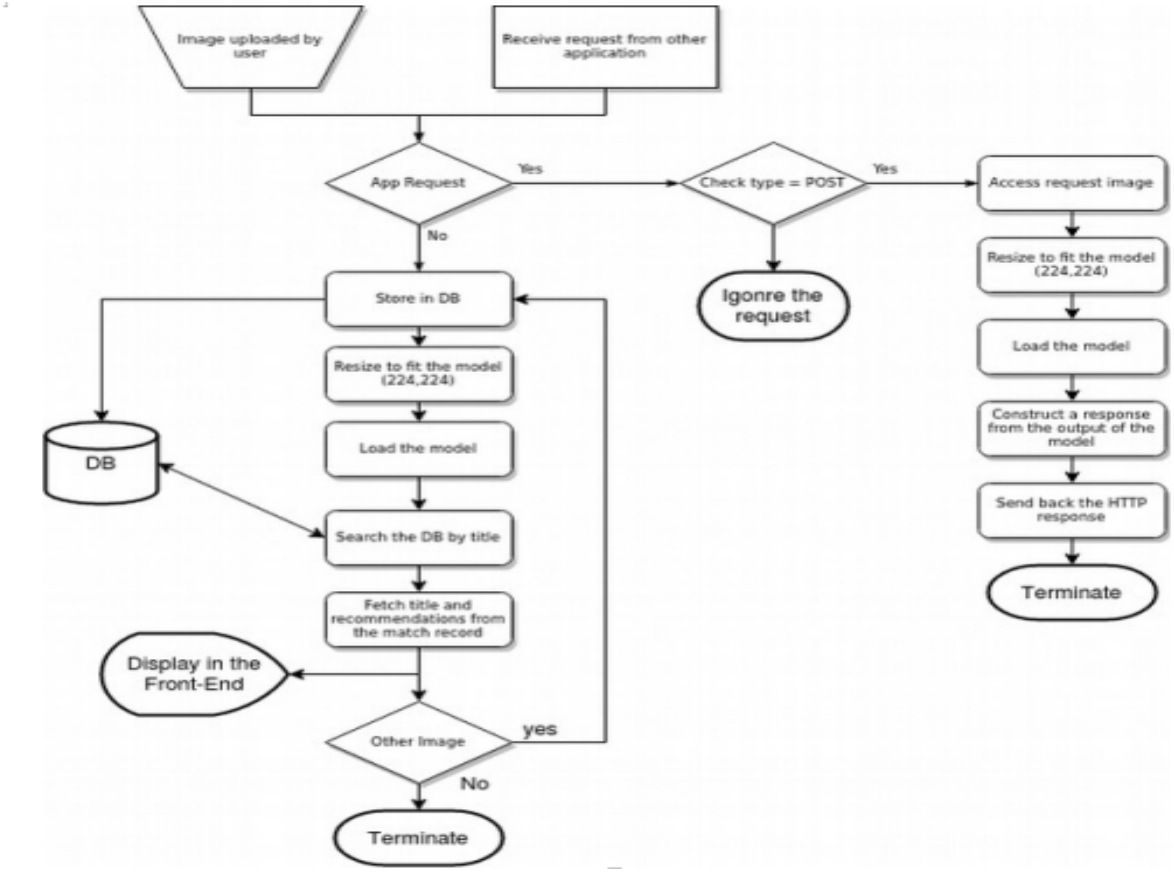


Figure 49:data transfere algorithm

4.4. The connection between the front-end and the back-end.

The connection is made through the Django helper function "render". This function Combines a given template with a given context dictionary and returns an HttpResponse object with that rendered text. One of its arguments is the context which is a dictionary of values to add to the template context. By default, this is an empty dictionary. If a value in the dictionary is callable, the view will call it just before rendering the template. for example on the first page, the form is used to upload the image then this image is inserted in the media folder however, the path to it is stored in the database.

```
@method_decorator(csrf_exempt, name='dispatch')
class firstpage(CreateView):
    model = Imaage
    fields = ['imagefile']
    template_name = "project/firstPage.html"
    success_url = reverse_lazy('project:state')
```

Figure 50: view to render form page

By extending the CreateView class that is built in Django. That class is A view that displays a form for creating an object, redisplaying the form with validation errors (if there are any), and saving the object. It takes the model of the database as a parameter and its fields to be updated from the form and the template name that is the front-end HTML file that contains the form itself and the success URL which is the address to be routed to after the form is saved successfully.

For the front-end I used the Django template language is Django's own template system. Until Django 1.8 it was the only built-in option available. It's a good template library even though it's fairly opinionated and sports a few idiosyncrasies. If you don't have a pressing reason to choose another backend, you should use the DTL, especially if you're writing a pluggable application and you intend to distribute templates. Django's contrib apps include templates, like Django.contrib.admin, use the DTL.

A Django template is a text document or a Python string marked-up using the Django template language. Some constructs are recognized and interpreted by the template engine. The main ones are variables and tags.
A template is rendered with a context. Rendering replaces variables with their values, which are looked up in the context, and executes tags. Everything else is output as-is. The syntax of the Django template language involves four constructs. The first one is the variables. A variable outputs a value from the context, which is a dict-like object mapping keys to values.
Variables are surrounded by {{ and }}. the second is the Tags. Tags provide arbitrary logic in the rendering process.

This definition is deliberately vague. For example, a tag can output content, serve as a control structure e.g. an “if” statement or a “for” loop, grab content from a database, or even enable access to other template tags.
Tags are surrounded by {% and %}. the third is the Filters which transform the values of variables and tag arguments. the fourth is the comments.

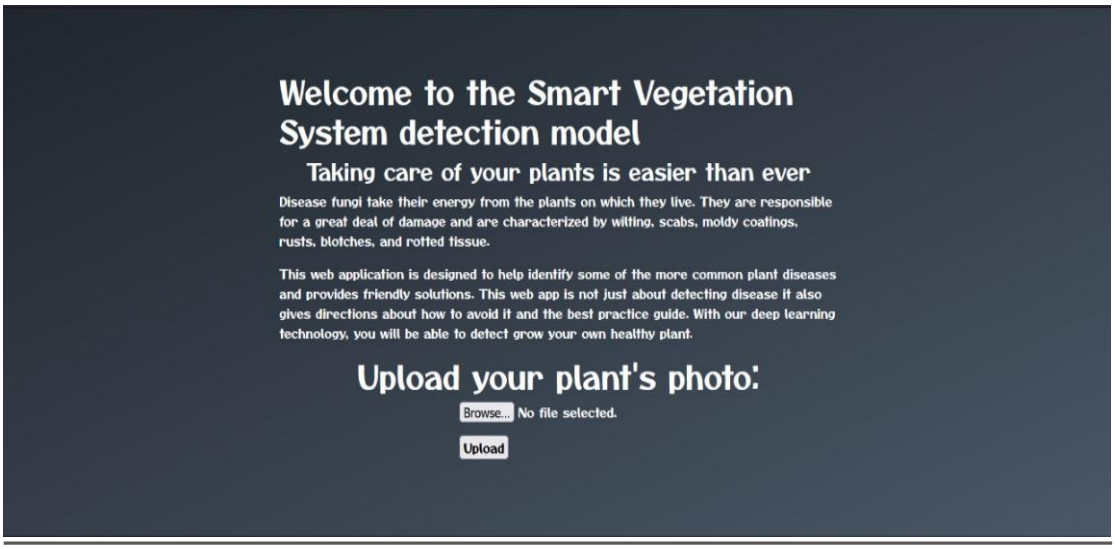


Figure 51: Form page

The second page is considered as the processing that loads the model and fetches the image from the database then gives it to the model as input then predicts the disease. After prediction, it fetches the recommendations from the database by id.

```
def state(request):
    lastimage= Imaage.objects.order_by('-id')[0]

    testimg = '/home/YoussefFekry/django-projects/mysite/media/'+str(lastimage.imagefile)
    img = image.load_img(testimg, target_size=(img_height, img_width))
    x = image.img_to_array(img)
    x = x.reshape(1, img_height, img_width, 3)
    x = tf.keras.applications.mobilenet.preprocess_input(x)
    predi = model.predict(x)
    label = int(predi.argmax(axis=1))
    label = label + 1

    recomend = Rec.objects.get(pk=label)

    db = Imaage(id = lastimage.id , name = recomend.disease , imagefile = lastimage.imagefile)
    db.save()

    ctx = {'recomend' : recomend,'myimage' : lastimage}
    return render(request, "project/recPage.html", ctx)
```

Figure 52: View to render the state page

All the data are then collected into a dictionary then rendered in the front-end. In the recPage.html I used inline styling. for the CSS I used the bootstrap and the font awesome icons.

Figure 53: Status page

The ESP-CAM captures images and POST them to the server periodically. For the user to access it we need to make a history page that shows all the images and their states.

Figure 54: History page

To be able to show the image with its prediction by using the model again, we need to model them in a many-to-one relationship. This means that for every one image, there is only one prediction but not vice versa. This modeling is made by giving each image a name attribute that stores its prediction.

In order to construct the page, I Inherited the Django ListView class which is used for a page representing a list of objects. It takes the model and the template name as the arguments and returns a list of that objects that are then used to iterate it using the python loops in our case I used the for a loop. I also override the default "get_context_data" function to add the count of the images to the dictionary before rendering the template.

```
class history(generic.ListView):
    model = Imaage
    template_name = 'project/img_list.html'

    def get_context_data(self , **kwargs):
        context = super().get_context_data(**kwargs)
        context['counter'] = Imaage.objects.all().count()
        return context
```

Figure 55: View to render the history page

5. Chapter| Web Application

5.1. Description

This is a web application that aims to provide assistive information through visual representations (graphs) and verbal messages (notification) to help in decision making and monitoring of the plant condition, it is connected to an embedded system providing readings like temperature, and uploads images that get processed by a deep learning algorithm on a seperate server to know the state of the plant.

5.2. Features

5.2.1. Home Page

The home page contains the latest readings from each sensor, providing a quick preview of the status of the plant, color change depending on the sensor values.

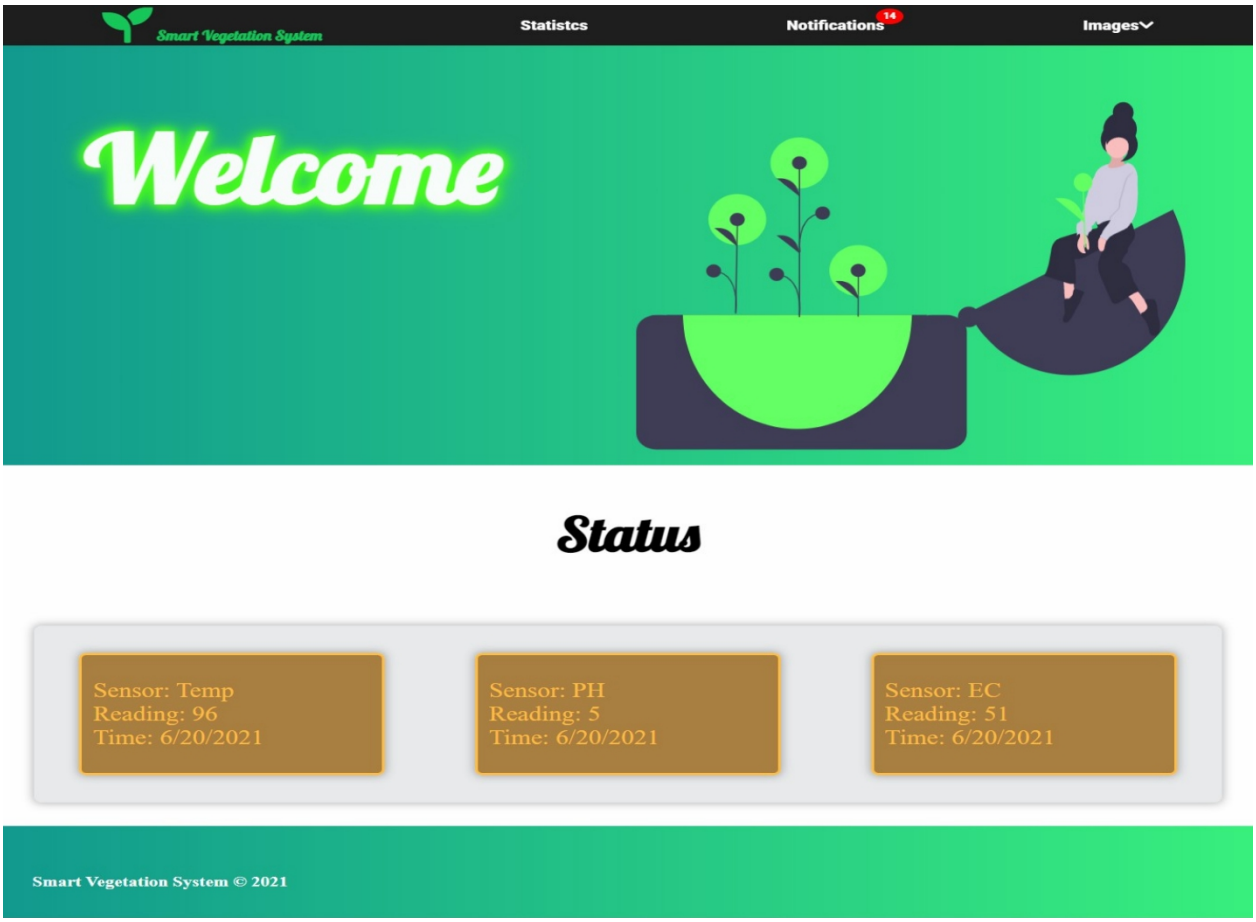


Figure 56: Home Page.

5.2.2. Stats Page

You can now navigate to statistics page and get data from 3 tables (temperature, PH, electric conductivity) You can do the following:

1. Get previous/next pages in each graph (you can choose a graph from the 3 icons at the bottom of the graph, temperature graph is chosen by default), when no pages are available the chevron will disappear.
2. You can specify a filter on the minimum value and the max value , the beginning date and the last date as well as the limit of points of each graph, on submitting you will get the result displayed on the corresponding graph , if the filtered result is greater than the limit you specified you can simply use the next/previous buttons to navigate through the result.
3. If no filter is specified, the default value will be used.
4. You can use the reset button to reset the form data to the default.

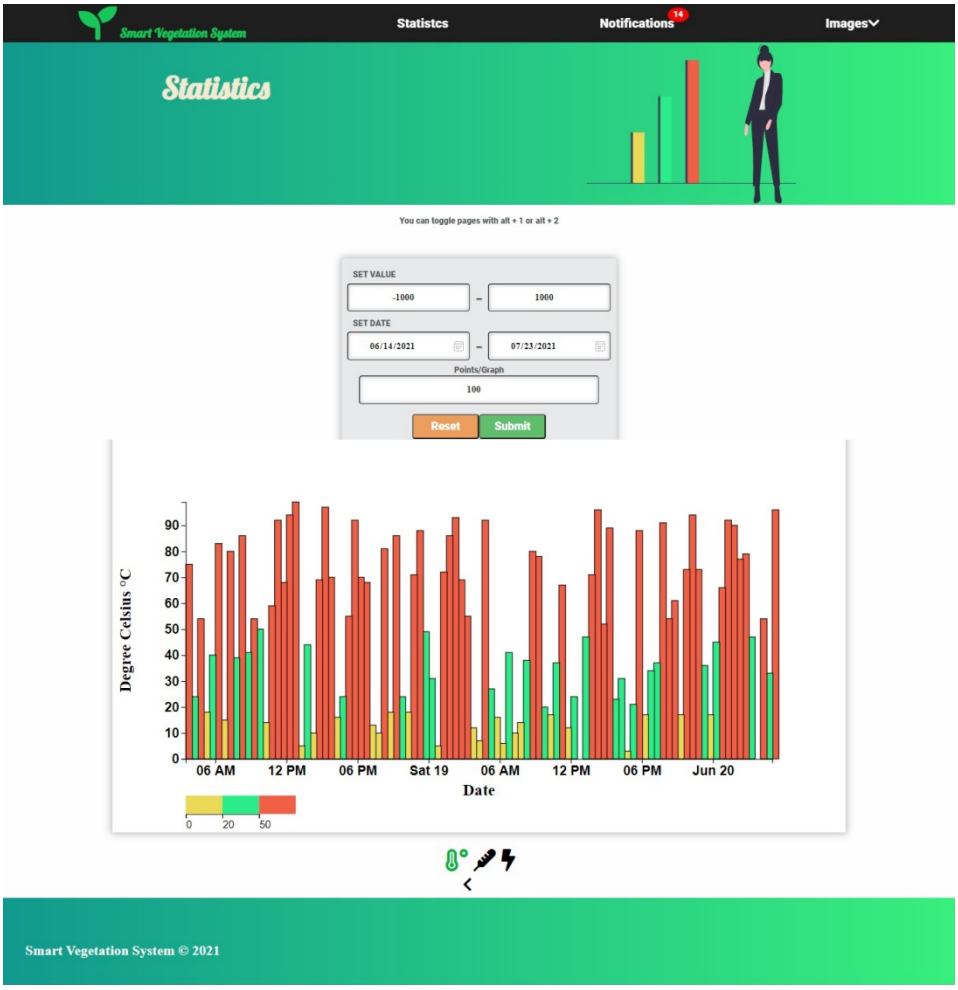


Figure 57: Stats Page.

5.2.3. Notification page

Any reading from the MC is checked if they are within desired levels, otherwise, they will be stored as notification for the user. You can do the following:

1. Check if new notifications are present, notifications include useful information like the measured value, its deviation from norm, its date, if it's below or above average and which table it comes from, if none are present, a notification message will be displayed indicating no notifications are present.
2. You can mark notifications as read by clicking any of the notification's edges, the notification color will change accordingly, any read notifications will not be shown again.
3. You can dismiss any notification which will delete the notification and mark it as read; so it won't show again.
4. When no notification is present; a default message will inform that everything is OK.

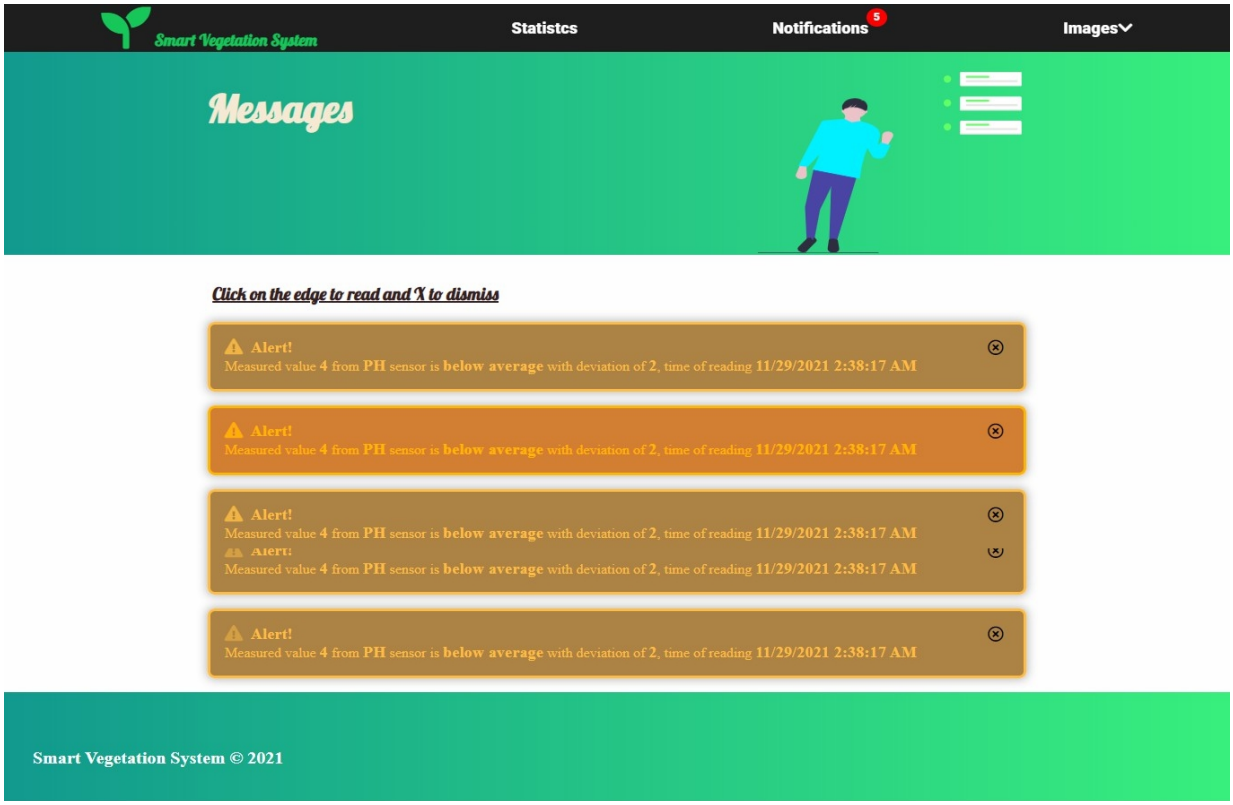


Figure 58: Notification page.

5.2.4. Upload Page

The user can upload an image and if it's of a valid format and size (PNG/JPEG max 5MB) then the image will be compressed to 224*224 pixels and converted to JPEG to be suitable for the python server to handle it, the image is then sent to the python server and processed, and the returned image object is then saved in the database for later view.

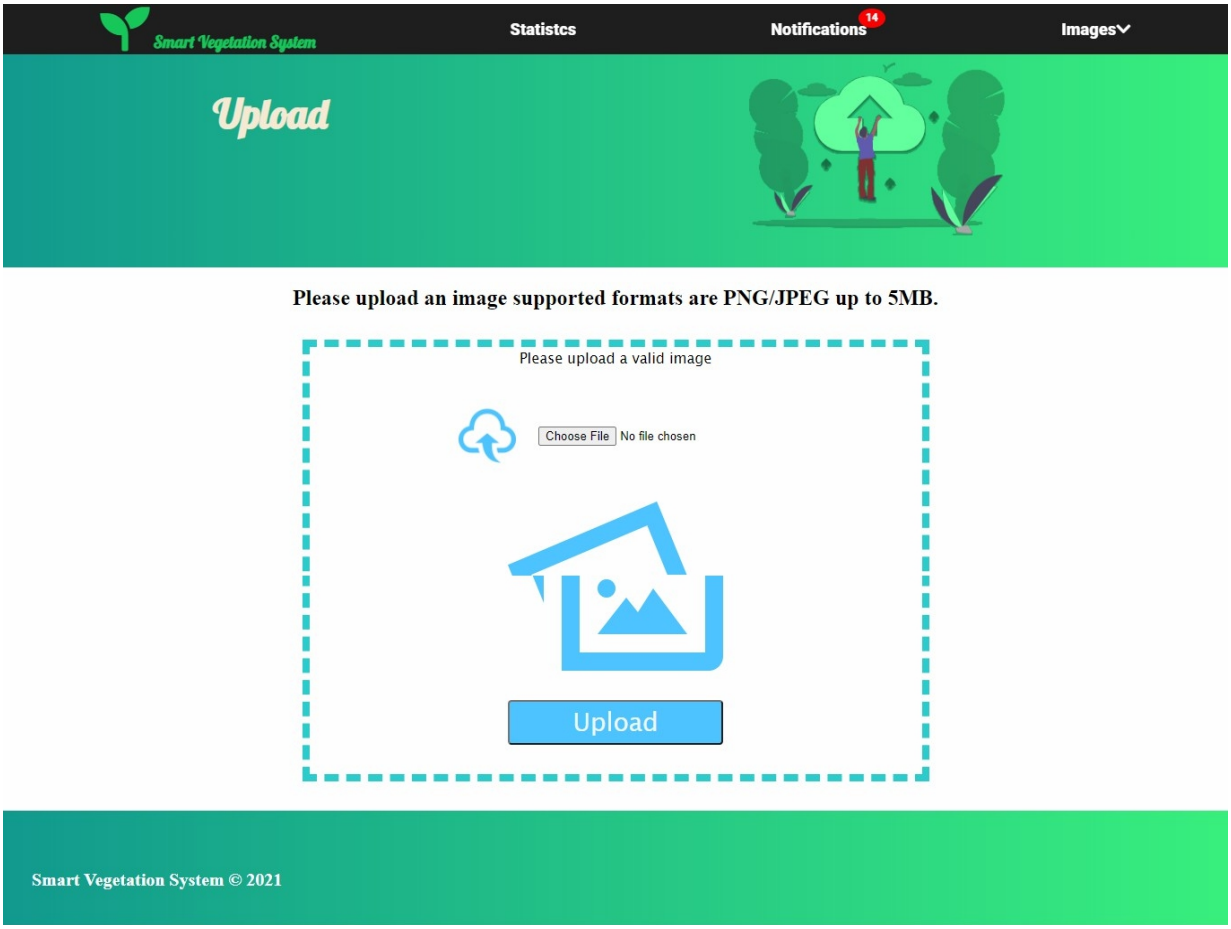


Figure 59 : Uploading page.

5.2.5. Result Page

Here the user is redirected to the python server and can view the latest processed images from the deep learning algorithm giving important information about the plant and if it's infected or not.

5.3. Tools

This is a web application using :

1. HTML to define the content of web pages (later used a templating engine (EJS).
2. CSS to specify the layout of web pages.
3. JavaScript to program the behavior of web pages.
4. D3 Library for graphics.
5. Node JS and Express for backend development (server initialization, routing and serving of web content).
6. MongoDB for database storage.
7. Mongoose for object modeling of the database (definition of schemas for each table in the database).
8. Replit/VS-Code as an IDE and the current server hosting.
9. Github for version control.
10. Many packages and modules

5.4. Diagrams

5.4.1. Use Case

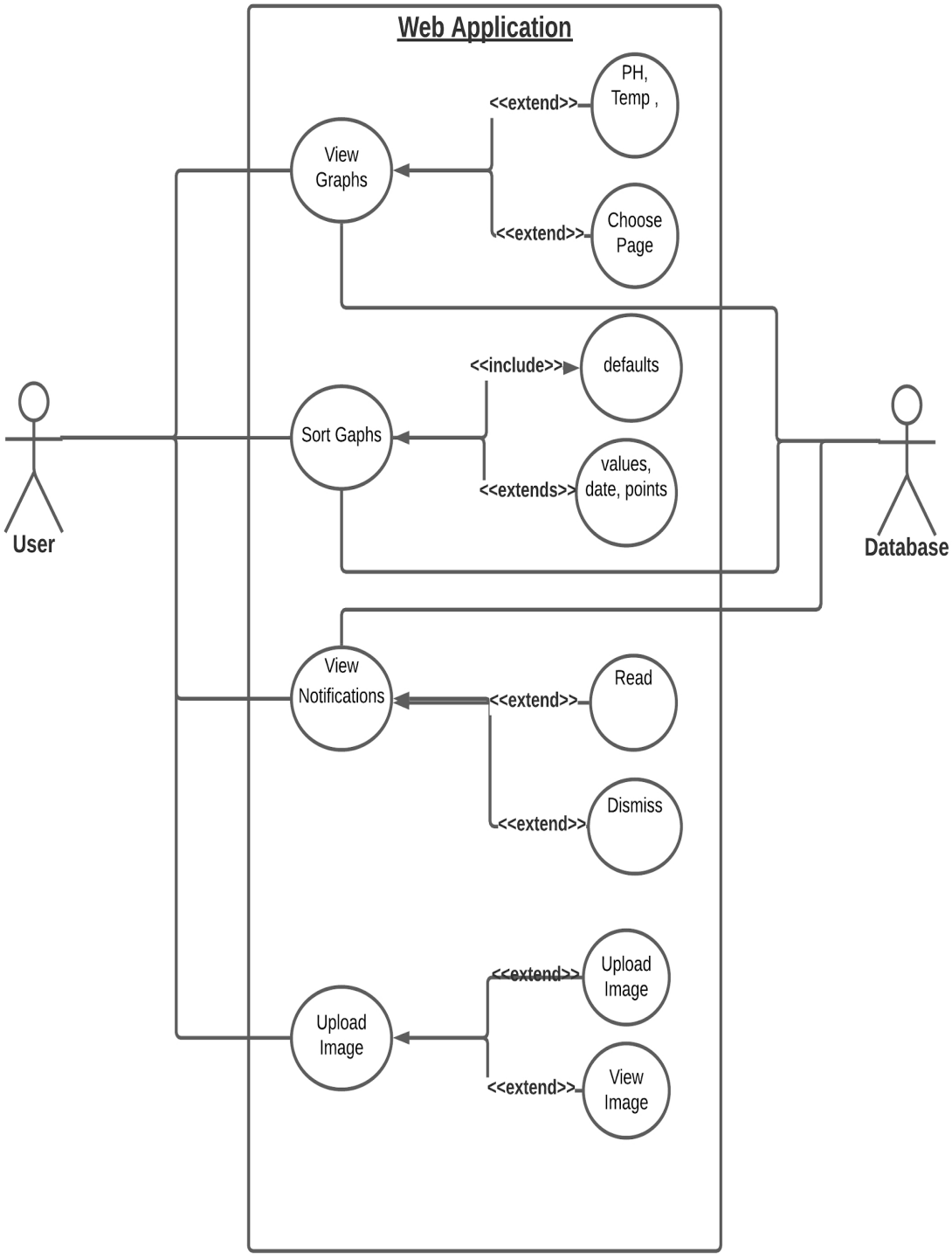


Figure 60: Use Case diagram.

5.4.2. Stats Page Flow Diagram

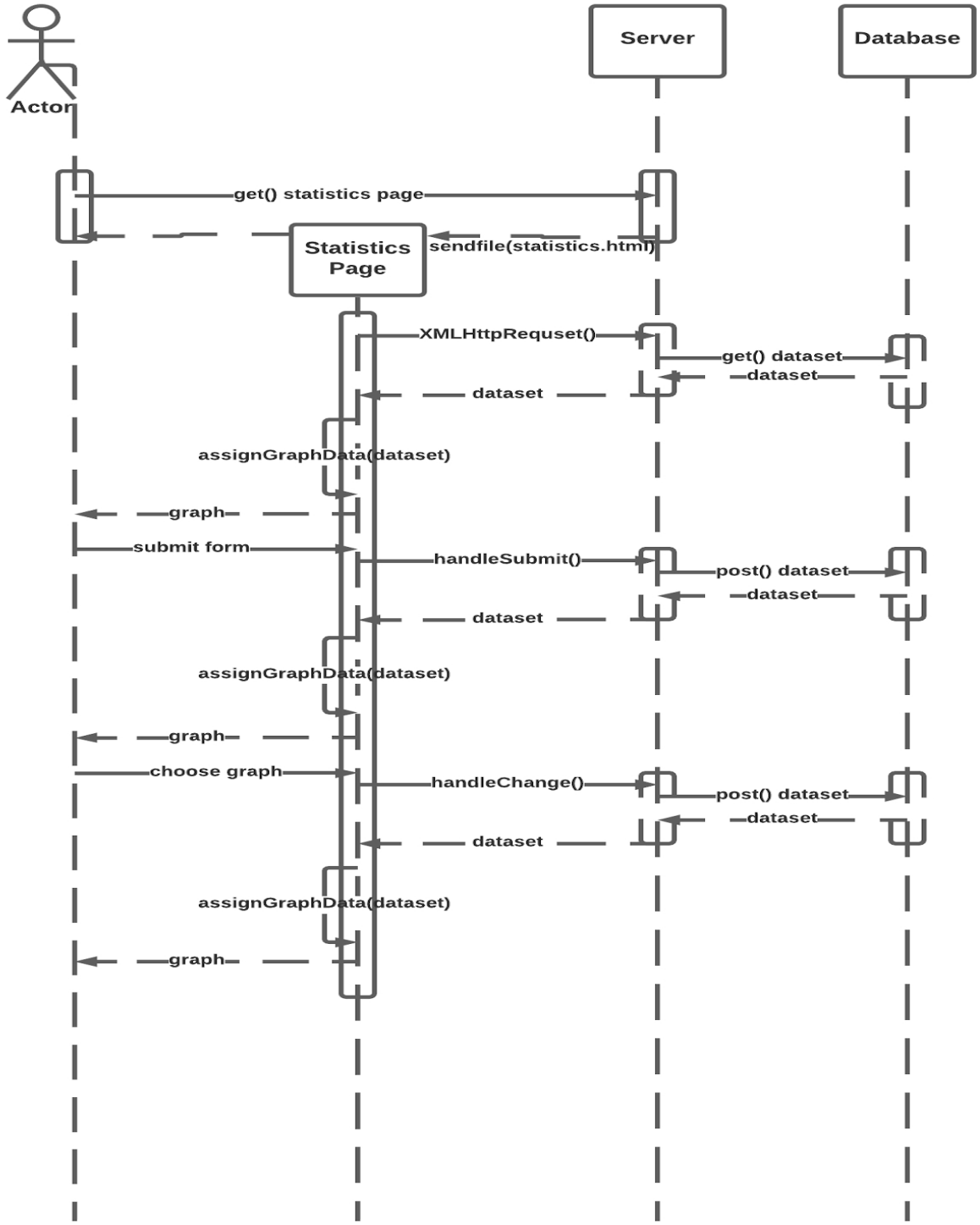


Figure 61: Stats Page Flow Diagram.

5.4.3. Notification Page Sequence Diagram

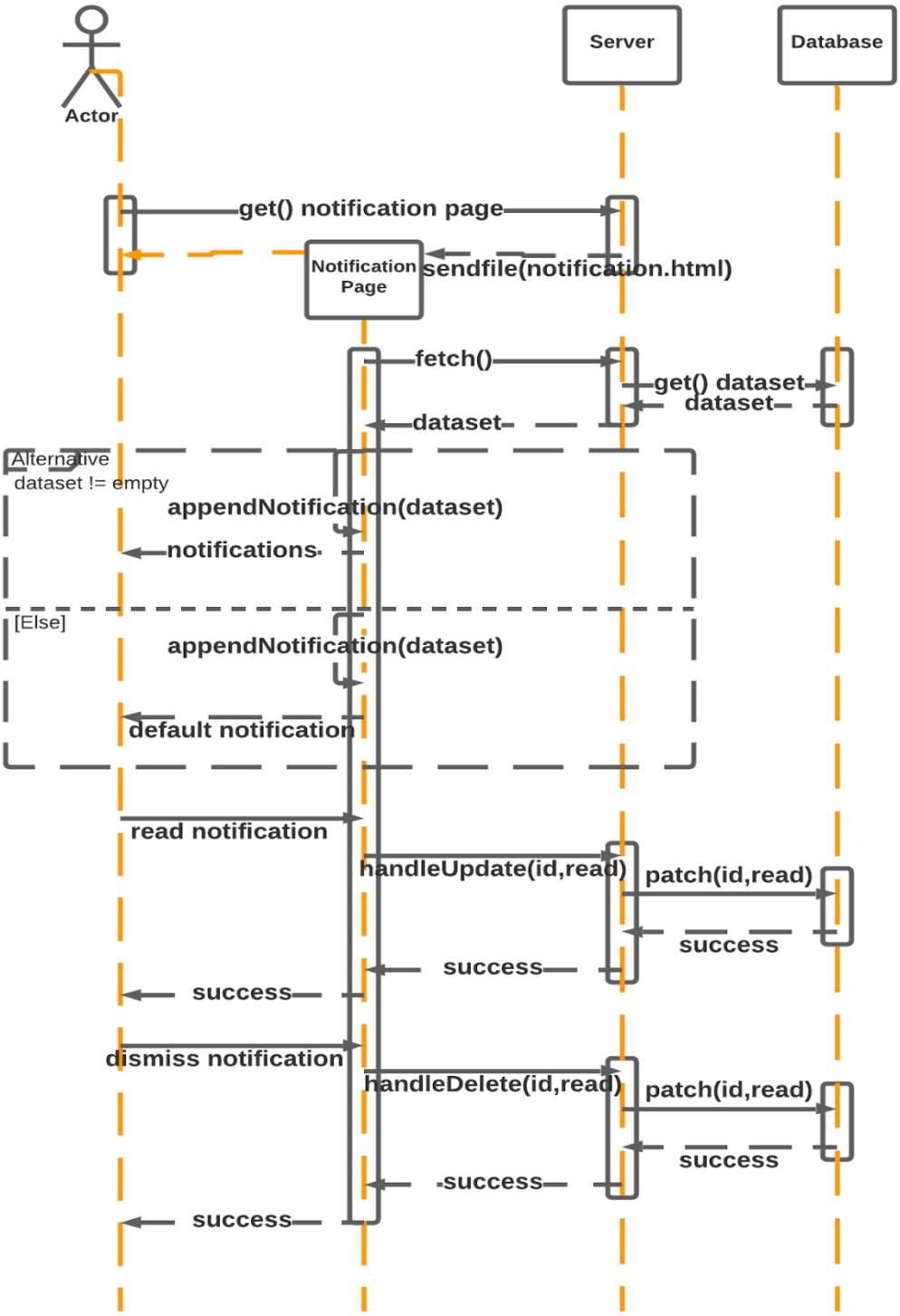


Figure 62: Notification Page Sequence Diagram.

5.4.4. Upload Page Sequence Diagram

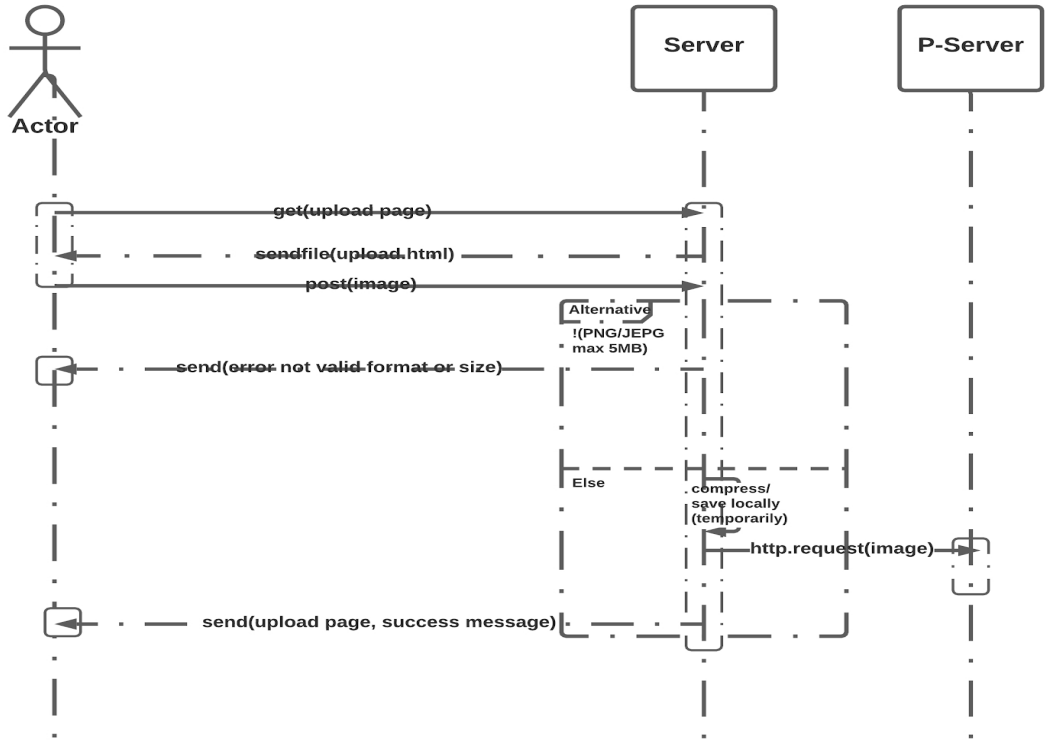


Figure 63: Upload Page Sequence Diagram.

5.4.6. Old Site vs New Site

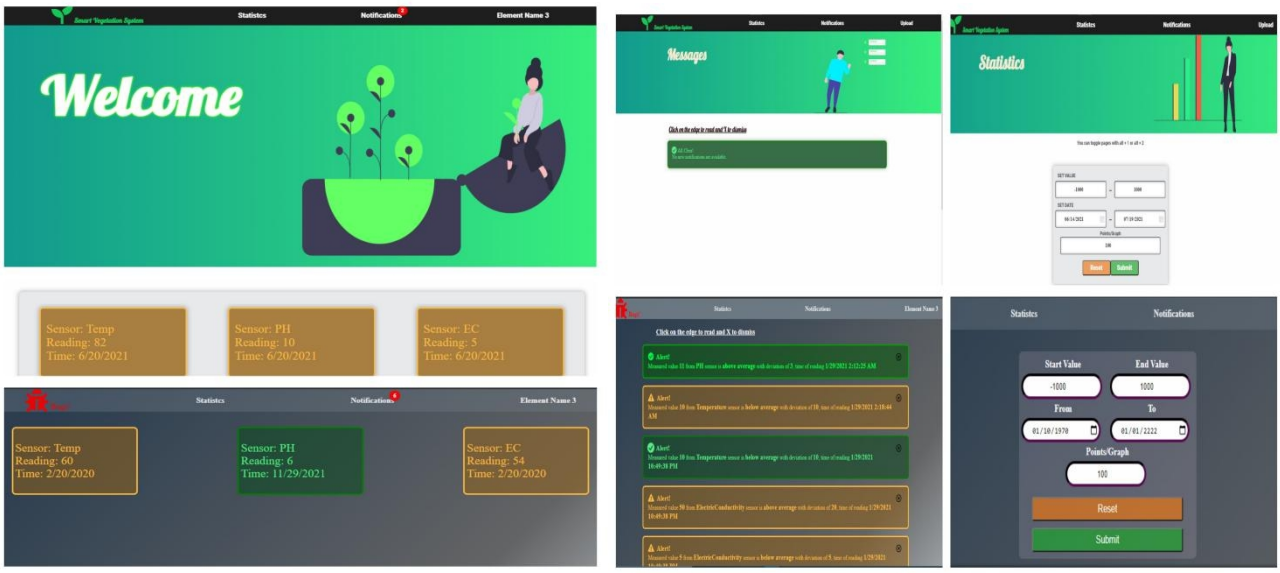


Figure 64: Comparison between the old site and the new site.

5.5. Web Application Steps

1. Boilerplate for Project
2. Statistics Page
3. Navigation Menu
4. Navigation Routing
5. Graph for each measured unit
6. Make dummy data
7. Store data in database
8. Plot graphs
9. Notification Page
10. Use Case Diagram
11. Sequence Diagram Notification Page
12. Sequence Diagram Statistics Page
13. Make a diagram of the project
14. Add notification counter
15. Current status for each table in home page
16. Receive data from embedded app
17. Page to upload images
18. Saving images locally/DB
19. Diagram for upload page
20. Diagram for view page

6. References

[1] DIY Hydroponic Gardens: How to Design and Build an Inexpensive System for Growing Plants in Water ,© 2018 Quarto Publishing Group USA Inc. Text © 2018 Tyler Baras. *https://b-ok.africa/book/3516760/9112ce*

[2] How-To Hydroponics, Fourth Edition Copyright © 1994-2003, ***Keith Roberto***

https://b-ok.africa/book/646828/366786

[3] *https://www.epicgardening.com/deep-water-culture-get-started/*

https://www.gardeningknowhow.com/special/containers/deep-water-culture-for-plants.htm

[5] *https://en.wikipedia.org/wiki/Hydroponics*

[6]*http://scienceinhydroponics.com/2009/02/faq-electrical-conductivity-ec-in-hydroponics.html*

[7] *https://www.freshwatersystems.com/blogs/blog/what-are-hydroponic-systems*

https://scienceinhydroponics.com/

[9] *https://wiki.dfrobot.com/PH_meter_SKU__SEN0161_*

[10] *https://en.wikipedia.org/wiki/Real-time_clock*

[11] *https://www.elprocus.com/rtc-ds1307/*

[12] *https://docs.platformio.org/en/latest/boards/espressif32/esp32cam.html*

[13] Internet of Projects with ESP32: Build exiting and powerful IoT projects using the all-new Expresif ESP32, ***Agus Kurniawan***

https://b-ok.africa/book/5222049/8ab286

[14] *https://robotzero.one/esp32-cam-arduino-ide/*

[15]*https://robu.in/product/ai-thinker-esp32-cam-development-board-wifibluetooth-with-ov2640-camera-module/*

[16] Internet of Things with ESP8266, by *Marco Schwartz*

https://b-ok.africa/book/2928339/5b3df6

[17] ESP8266 Internet of Things Cookbook, Copyright © 2017 Packet Publishing

https://b-ok.africa/book/3713244/f2e1b5

[18] NodeMCU ESP8266 Communication Methods and Protocols : Programming with Arduino IDE 2018 , by *Manoj R. Thakur* ***https://b-ok.africa/book/3598348/8cdc9d***

[19] ***https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf***

[19]***https://www.terraelectronica.ru/pdf/show?pdf_file=%2Fz%2FDatasheet%2F1%2F1420644897.pdf***

[20] ***https://cdn.shopify.com/s/files/1/0672/9409/files/Datasheet_DHT11.pdf?733***

[21] ***https://www.alldatasheet.com/datasheet-pdf/pdf/1132088/ETC2/DHT11.html***

[22] ***https://www.ti.com/lit/ds/symlink/pcf8574.pdf?ts=1626846604622***

[23] Handbook of Serial Communications Interfaces: A Comprehensive Compendium of Serial Digital Input/Output I/O) Standards 2015, by **Louis Frenzel**. ***https://b-ok.africa/book/2662704/10b6cd***

[24] ***https://learn.sparkfun.com/tutorials/serial-communication/all***

[25] ***https://en.wikipedia.org/wiki/Serial_communication***

[26] MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications (paper) ***https://arxiv.org/abs/1704.04861***

[27] Deep Learning vs. Traditional Computer Vision (paper)

https://arxiv.org/ftp/arxiv/papers/1910/1910.13796.pdf

[28Toward datascience (website)

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53?gi=7ca5b05e3665

Deep Learning. Ai Andrew ng (Course)

https://www.coursera.org/specializations/deep-learning?utm_source=gg&utm_medium=sem&utm_campaign=17-DeepLearning-ROW&utm_content=17-DeepLearning-ROW&campaignid=6465471773&adgroupid=77415260637&device=m&keyword=coursera%20deep%20learning%20ai&matchtype=p&network=g&devicemodel=&adpostion=&creativeid=506751438660&hide_mobile_promo&gclid=CjwKCAjwruSHBhAtEiwA_qCppsfuqrT4nZwTaHTYJrdfhQ2-mFBWRYHc6cNjUN_emCpKAT1xUb-5GxoCuQ4QAvD_BwE

[30] PlantVillage Dataset, Dataset of diseased plant leaf images and corresponding labels ..

https://www.kaggle.com/emmarex/plantdisease

[31] ***https://docs.djangoproject.com/en/3.2/ref/class-based-views/generic-display/***
[32] ***https://docs.djangoproject.com/en/3.2/topics/http/shortcuts/***
[33] ***https://docs.djangoproject.com/en/3.2/ref/databases/***
[34] ***https://docs.djangoproject.com/en/3.2/topics/templates/***
[35] Our Project Repository Link

https://github.com/Xeonart/Graduation-Project

[36] Web Repository Link

https://github.com/Piemaker/automated-vegetation-system

[37] QR Code For Site

[38] Link for Site

https://automated-vegetation-system.piemaker1.repl.co/