SMART SYSTEM FOR IRRIGATION

Members:	• Ahmed Abdelaal Ahmed Abohadeed. sec(1) • Abdallah Sabry Ahmed. sec(4) • Esraa Adel Shehata Ali. sec(7) • Shrouk Salama Mahmoud Mohamed. sec(7) • Abdullah Ashraf Abdel Fattah Amara. sec(4) • Ahmed Hassan Mohamed. sec(2) • Ahmed Emad Al-Dasouki Al-Jabri. sec(1)
Department:	AI

Topics

1. Preface

- This project is about creating a smart irrigation system that will help farmers to efficiently and effectively manage their irrigation needs. The system will be designed to be user-friendly and cost-effective, while also providing accurate and timely information about the irrigation needs of the crops. The goal of this project is to develop a system that will help farmers to save time, money, and resources while improving the quality of their crops

2.Introduction

- This system uses sensors to monitor soil moisture and weather conditions to determine when and how much water to give your plants. It can be programmed to water your plants at specific times, or it can be set to water them automatically based on the conditions. With this system, you can save time, money, and water, while ensuring your plants get the best care possible.

3.Glossary

Function	Resource
Coding Languages	Node MCU - ' C '

Development Platforms	Arduino - IDE Node-MCU Platform
Data Transfer	Github
Computer Boards	Node MCU
Sensors	Moisture Sensor Rain drop sensor Temperature and Humidity Sensor Relay Modules Motion sensor (PIR)
Other Hardware	Solenoid Valves 9v Batteries 1M Extra Copper Cabling Irrigation Piping/Equipment Electrical Tape Breadboard
Other Software	Draw.io Edraw

4.Overview

A-Problem:

Most of the farmers are facing a major problem with water. Irrigation plays an important role in agriculture. To overcome this water deficiency in country we have to save it as we can. Unnecessary irrigation should be avoided.

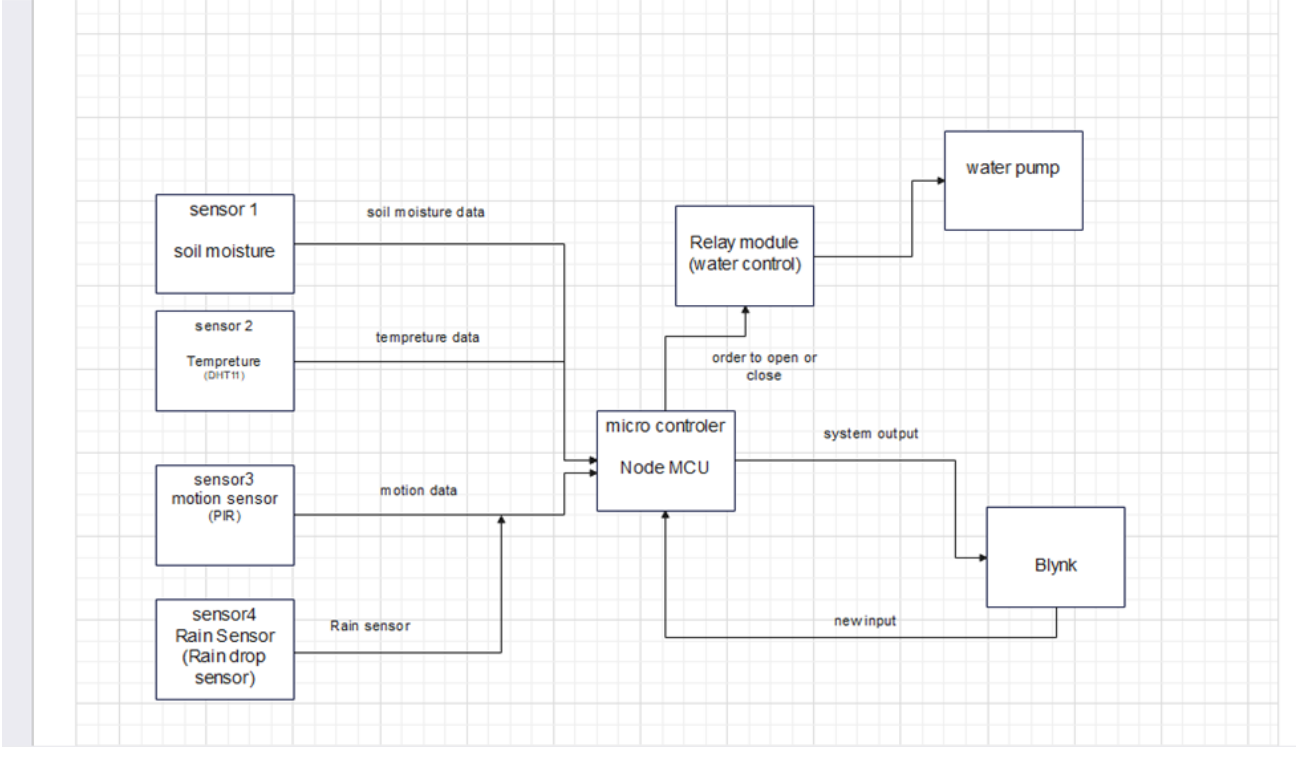
B-solution:

For this smart irrigation helps a lot. By arranging some useful sensors like temperature, humidity, soil moisture sensors will find the percent of moisture in soil. Then if the moisture percent is low, then motor will gets ON atomically and OFF when it maintains a sufficient percent. and it can Helps save water, money and manpower, Improving the quality of agricultural crops.

5. System User Requirement

The users of this system will require a simple yet effective and fully functioning automatic system. The mobile application should be simple and easy to use allowing users to navigate all of the sensor data with ease. Users will expect a system that once installed will actually work and water their plants when needed. This is the main requirement for users as the project will be a failure if it does not water the plants when they need to be watered.

6. System Architecture



7.Requirements

- **When broken down into its most basic elements, the main requirements of this**
- **project are as follows:**
  - All sensors must be connected to the Node MCU.
  - Data must be sent from Node MCU Platform to the Blynk Application.
  - Application has to display the received data.

- C code must activate the valve Relay Moduol.
- Water is required to flow through the system and water the soil samples

7.a.Functional Requirements

- **Moisture Sensor Data Collection:** The sensor is collecting analog data about the moisture content of the soil they are placed into. This data will be represented on the android application where it will be labelled and display the info for the user to see.

- **Temperature and Humidity Sensor Data Collection:**This sensor is being added to the system in order to collect data about the enclosed environment surrounding the soil sample (moisture_sensor_C). This specific data is being collected to see if there is a difference in how the moisture content of the soil changes whilst being set at unique temperature and humidity to the other soil samples.

- **Rain Drop Sensor :**this sensor was added to collect a Data about Rains and Air Condition.

- **Mobile Application Notifications:** Node MCU must send a push notification to the mobile device of the user once the values being collected from the moisture sensors are at a specific value. The mobile app will alert the user to this event.

- **Mobile Application Display :** The mobile application that will be accompanying the irrigation system will act as a display hub for all the information gathered by the sensors throughout the build. The function of the app will be a display for users and will also include the ability to manually open and close the solenoid valves.Due to the nature of the project there is a very minimal need for the application to have any functionality. The purpose of this system is to remove the need for usersto look after their plants, therefore creating a system that implements too much functionality would be pointless and the system is meant to be installed once and it should run on its own without any interaction.

7.b. Non - Functional Requirements

- **Specifies any other particular non-functional attributes required by the system.**

Examples are provided below.

- **Performance/Response time requirement :** The project should update the application with the relevant information quickly but there is no need for extremely fast data transfer. Node-MCU data transfers seem nearly instantaneous to the user so this should be sufficient.

- **Availability requirement :** The data gathered from the Node - MCU sensors will be available to the project and the application as long as the 'C' code is running.

- **Recover requirement:** All of the code will be saved using Github. Here the project files can be stored without the fear of losing them through hardware or software malfunction.

- **Robustness requirement :** The project is certainly robust but has got some smaller parts that could be easily broken or damaged. The over functionality of the system should continue throw some wear and tear.

- **Reliability requirement :**Thankfully due to the robustness of the hardware and the simple nature of the software, the project is unlikely to fail. There may be some cable damage over time but that would be a simple fix.

Maintainability requirement : This project should be very simple to maintain due to the cheap sensors and other easily replaceable parts. There are some customised wiring fixes that would take more time to maintain but still these components are very cheap to replace.

- **Portability requirement :** Most elements of this project are very portable and can be quickly and easily moved around from one place to another. Once the components have been wired up and set in place it will be less portable, but the system can be moved with relative ease.

- **Extendibility requirement :**This project uses many different sensors and has the potential the be repurposed or expanded upon. The system could be added to and more sensors connected for larger area coverage if needs be. The project would work perfectly if scaled up for bigger projects.

- **Reusability requirement :** The code of this project would be very simple to reuse for other projects or even when creating another automatic irrigation system. Some of the hardware elements of this project have been fitted purposefully for this exact project but there are sensors that could be reused.

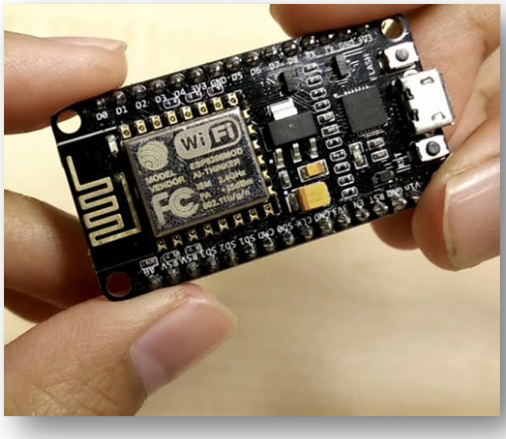
8.Component Details

- Micro controller (Node MCU) :
  - is a programable CPU or small computer used to control system input(sensors)and output.
  - we use it because it have a Wi-Fi module embedded in board that can reduce the cost .

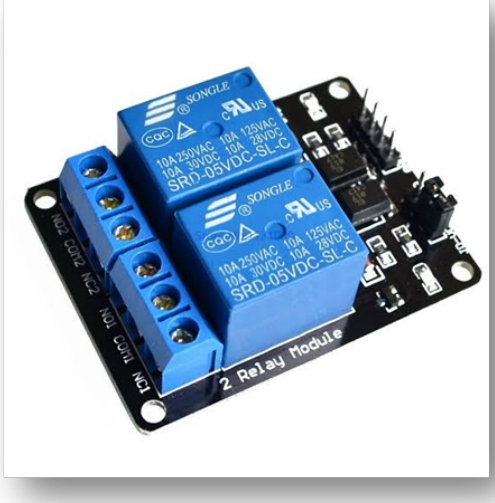
- **sensors:**
  - **soil monster sensor:**
  - The soil sensor is used to measure or estimate the amount of water in the soil. After estimating the amount of water in the soil, we can determine if the soil needs water or not. Then we measure the amount
  - **Temperature sensor (DHT11 or DHT22):**
    - DHT sensor is a basic sensor that is used to measure the temperature and humidity in the surrounding air. It's known for its low power consumption.
  - **Rain sensor (Rain drop sensor):**
    - The rain sensor is used for detecting if there's rain or not, which's a crucial thing for the system work upon. Knowing that there's rain determines the whole process, that's why it's the first step in our design to check if there's rain or not. If rain exists, the system stops irrigation till the rain is done. So, I the previous points you now get that how the sensors system is operating and in the next figure you can find out our hardware circuit for the sensor's system.
  - **Motion sensor (PIR):**
    - PIR sensors allow you to sense motion, almost always used to detect whether a human has moved in or out of the sensors range. They are small, inexpensive, low-power, easy to use and don't wear out.
- **Blynk App:**
  - It is responsible for all the communications between the smartphone and hardware, it can make a notification if a changes has been accurate in a system , it can be run in a cloud or in Local network and control in all system remotely.
- **Arduino IDE:**
  - is a program that use in compile and run the system code .
- **C Language:**
  - is a language that we used to code this system because it very fast and it is high level language and it is a suitable language the dealing with Hardware.

9.components picture

- **Micro controller (Node MCU)**



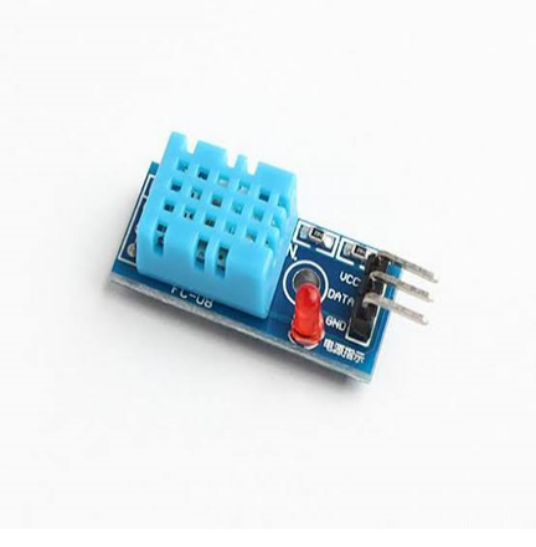
- **Relay Modul:**

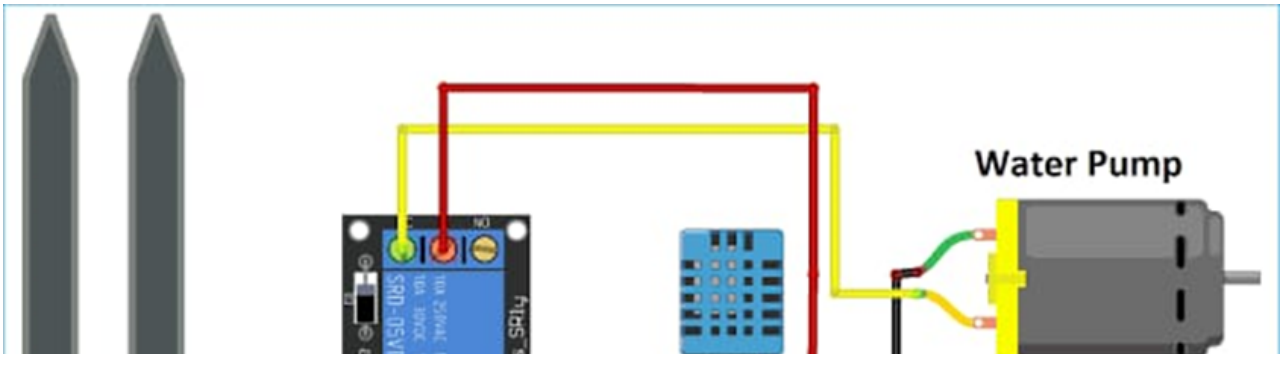


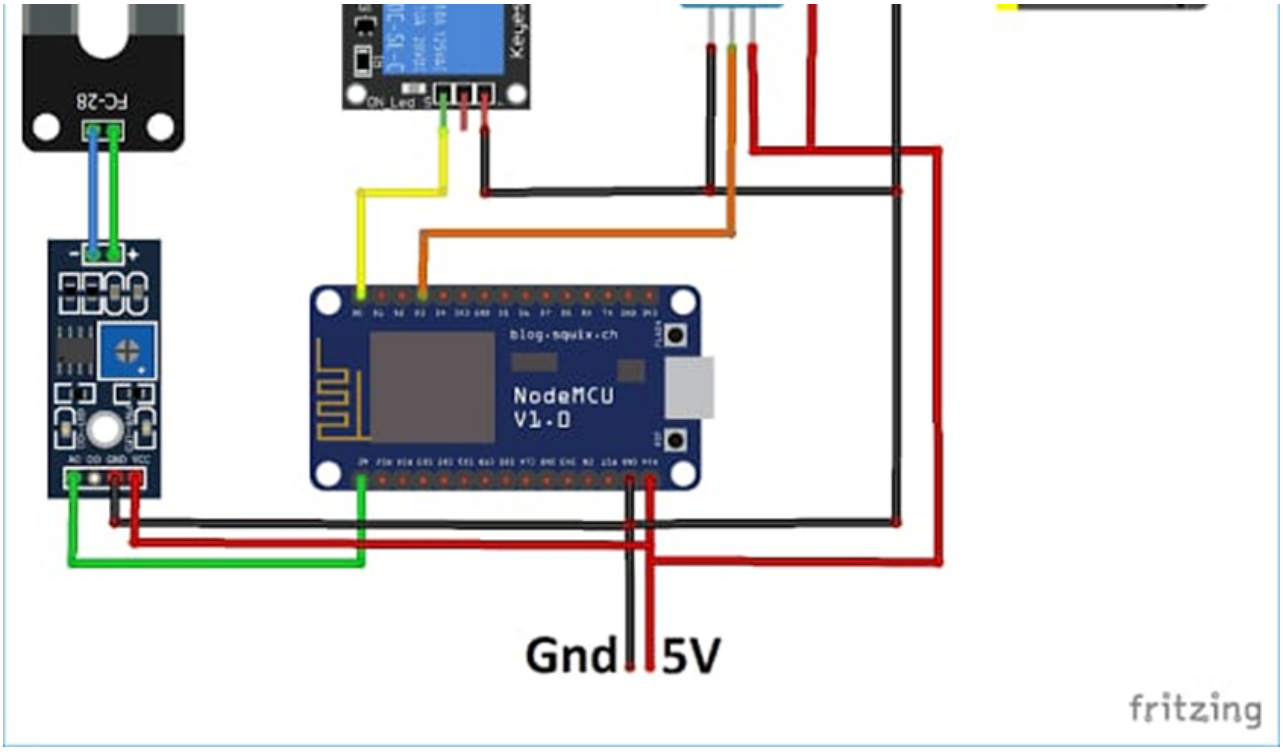
- **Soil moisture:**

- **Rain Drop sensor**

- **DHT11**

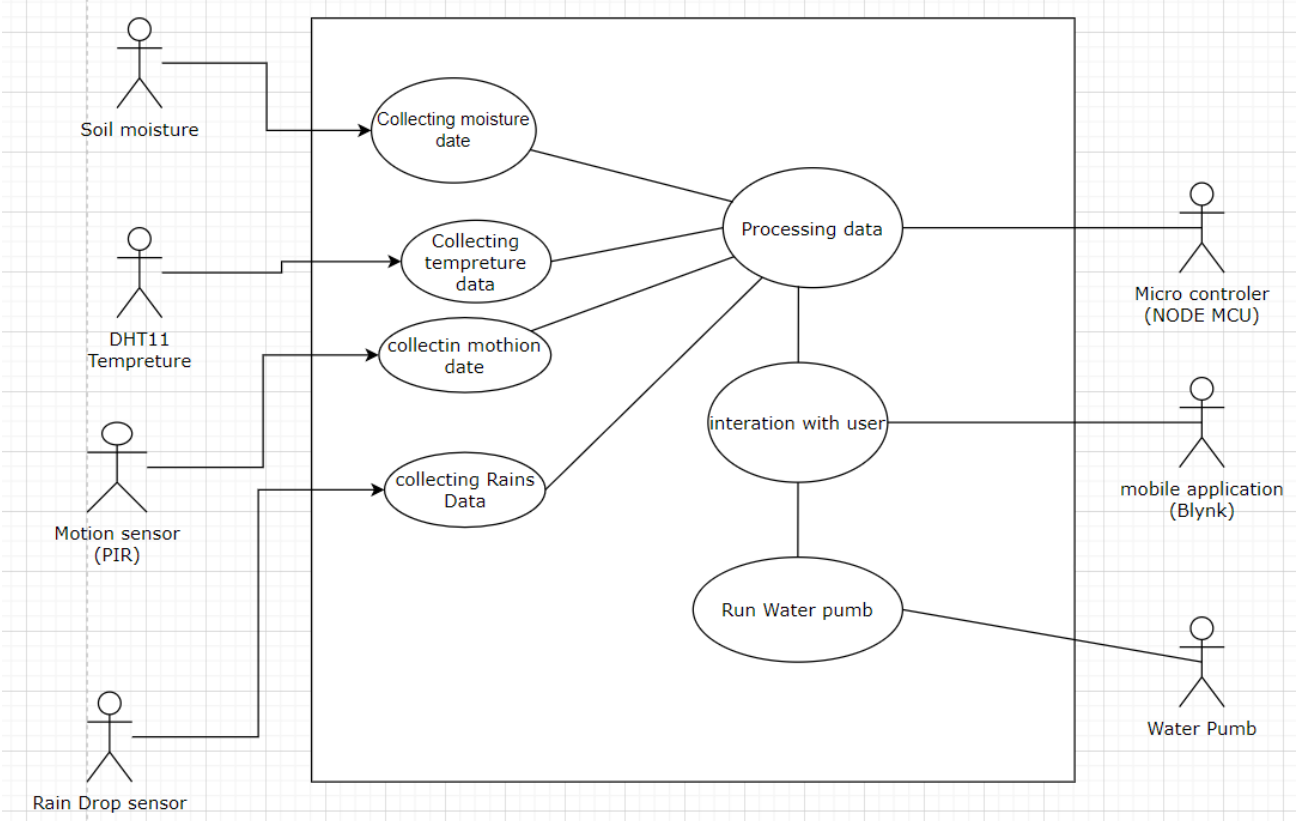






10. System Models

10.1.USE CASE



- **1.1.Collecting moisture data**

Use Case ID:	1
Use Case Name:	Collecting moisture date
Actors:	Soil moisture , Micro controller(node MCU)
Preconditions:	The system sits at stand by waiting for the moisture sensors to gauge a specific value from the soil.
Postconditions:	when a moisture sensor reads a certain value for the soil data and sends it to the node mcu And is Collecting percentage of moisture in a soil successfully.
Normal Flow:	1.connect soil moisture with node mcu correctly 2.start soil moisture sensor read Moisture percentage in the soil 3- send to mcu correct data about moisture soil orsend to mcu Moisture percentage in the soil is low
Alternative Flows:	In step 1: connect soil moisture with node mcu incorrectly. In step 2: Moisture percentage in the soil is not accurate or Moisture percentage in the soil is high

- **1.2 Collecting temperature data :**

Use Case ID	2
Use Case Name:	Collecting temperature date
Actors:	temperature sensor (DHT11), Micro controller(node MCU)
Preconditions:	The system sits at stand by waiting for the temperature sensor to gauge a specific value from the environment

Postconditions:	when a temperature sensor reads a certain value for the environment data and sends it to the node mcu And is Collecting percentage of temperature successfully
Normal Flow:	1-connect temperature sensor with node mcu correctly 2- start temperature sensor read temperature in the air 3-send to mcu temperature correct data or send to mcu temperature percentage in the air is high
Alternative Flows:	In step 1: connect temperature sensor with node mcu incorrectly In step 3: send to mcu temperature that a correct data or temperature percentage in the air is low

- **1.3 collecting Motion data:**

Use Case ID:	3
Use Case Name:	Collecting motion date
Actors:	Motion sensor(PIR) , Micro controller(node MCU)
Preconditions:	The system sits at stand by waiting for the motion to gauge a specific value from the environment
Postconditions:	The motion sensor detect a motion around the system
Normal	1-connect component of the system correctly 2- Writing the

Flow:	" c" code successfully 3-connect the system with power supply 4.sensor detect no motion
Alternative Flows:	In step 1:connect component of the system incorrectly In step 2: The "c" code cannot be run because it have an errors and bugs. in step 3: the power supply is damaged. in step 4:sensor detect motion around the system

- **1.4 Collecting Rains data :**

Use Case ID:	4
Use Case Name:	Collecting moisture date
Actors:	Rain Drop sensor , Micro controller(node MCU)
Preconditions:	The system has been run ,the soil moisture sensor in suitable environment
Postconditions:	Collecting Rains data successfully
Normal Flow:	1-connect component of the system correctly 2- Writing the " c" code successfully 3-connect the system with power supply
Alternative Flows:	In step 1:connectcomponent of the system incorrectly

	In step 2:The"c" code cannot be run because it have an errors and bugs

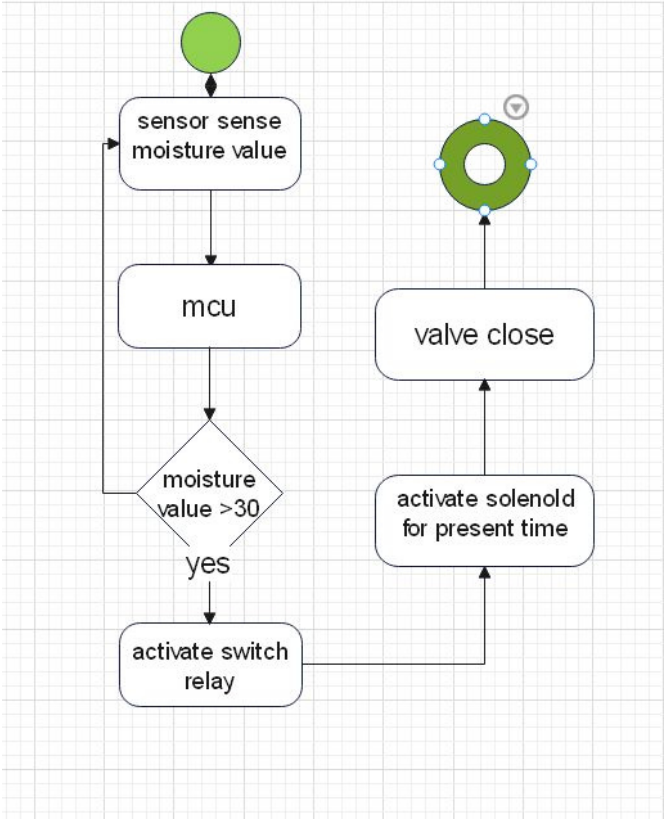
- **1.5 processing data:**

Use Case ID:	5
Use Case Name:	Processing date
Actors:	Motion sensor(PIR) , soil moisture, motion sensor, temperature sensor and Micro controller(node MCU)
Preconditions:	• The system has been run ,the soil moisture sensor in suitable environment • The sensors collecting date successfully
Postconditions:	The data has been send to the micro controller And send to the next process to be send to the user
Normal Flow:	1-connect component of the system correctly 2- Writing the " c" code successfully 3-connect the system with power supply 4-the date has been collecting successfully from all sensors 5-the processes that
Alternative Flows:	In step 1: connect component of the system incorrectly In step 2:The"c" code cannot be run because it have an errors and bugs

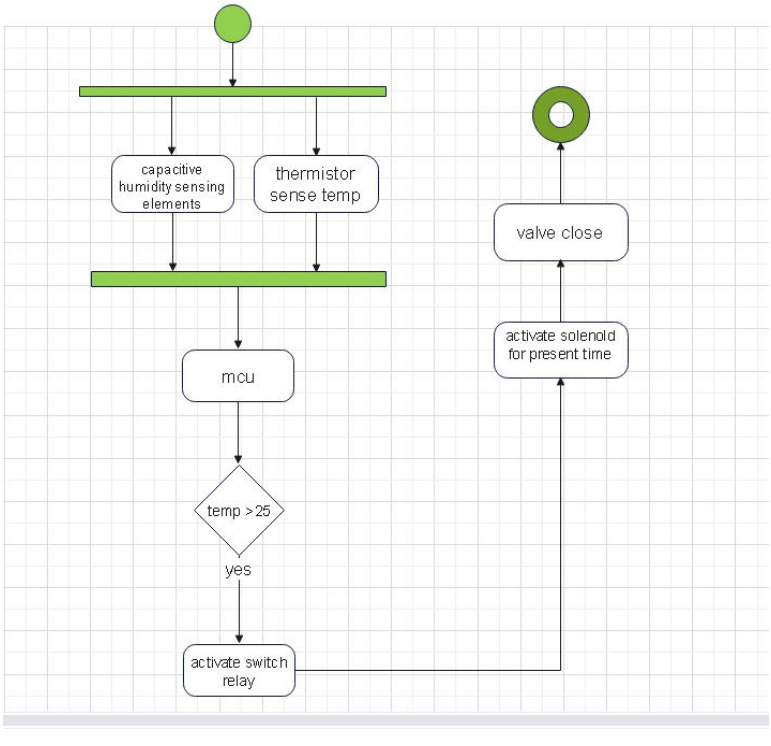
-

10.2.Activity Diagrams

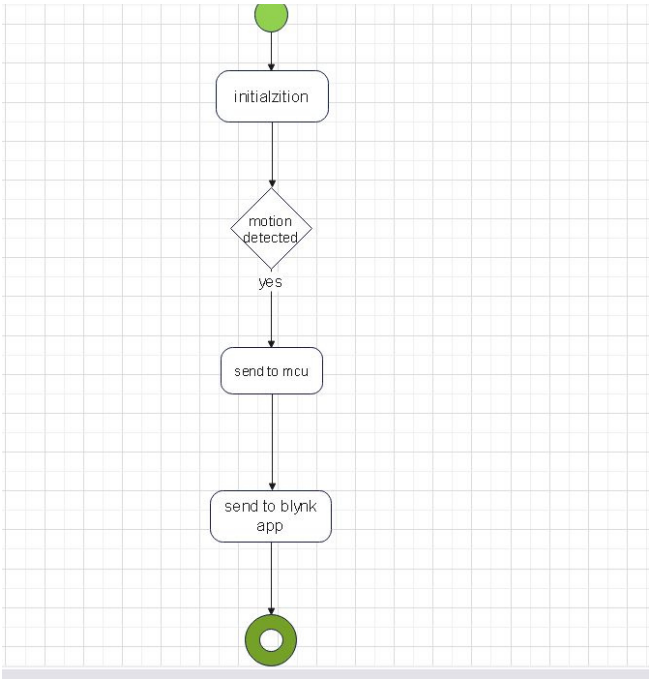
- **Collecting moisture Data:**



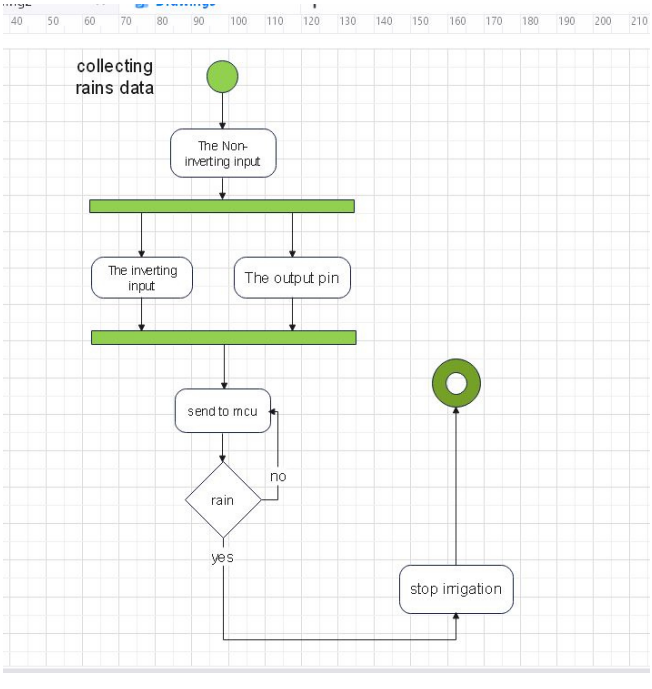
- **COLLECTING TEMPRTEURE DATA:**



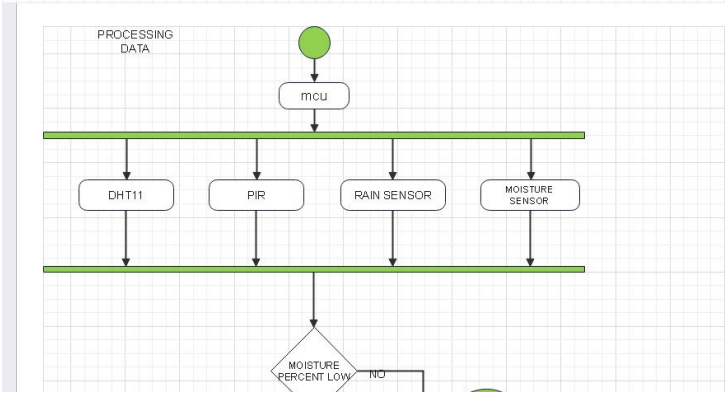
- **COLLECTING MOTION DATA:**

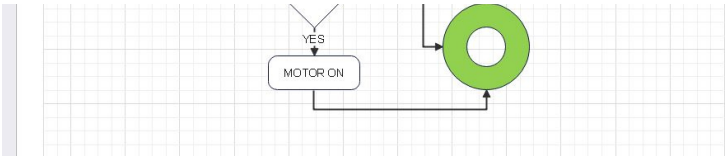


- **COLLECTING RAINS DATA:**

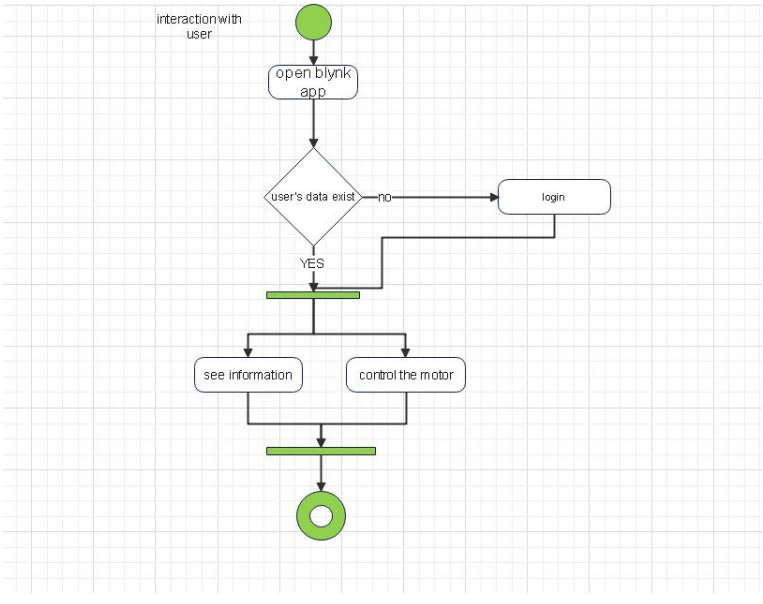


- **PROCESSING DATA:**

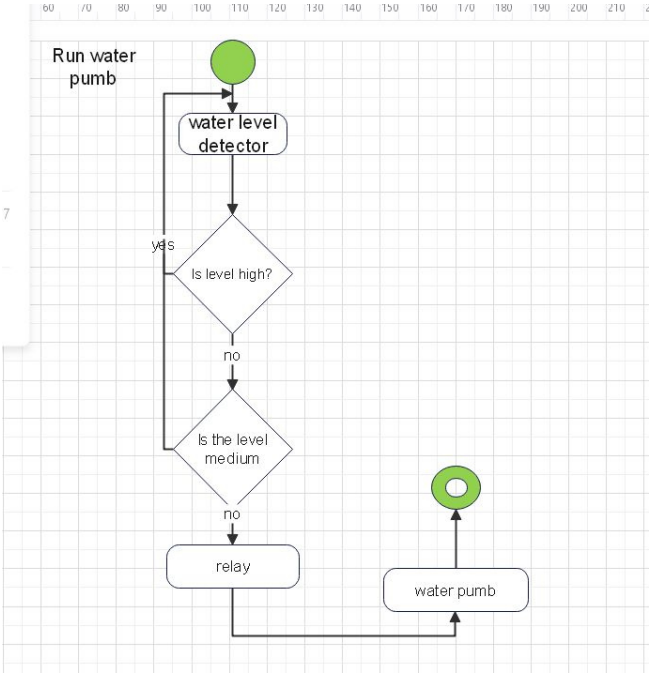




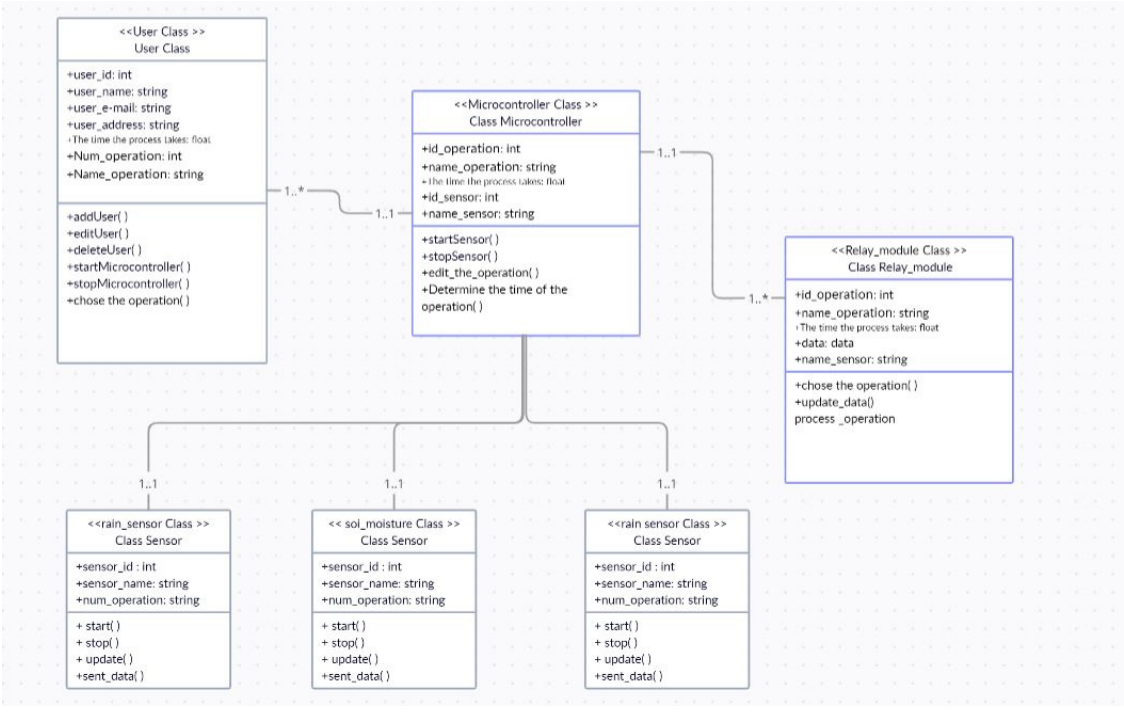
- **INTERACTION WITH USER:**



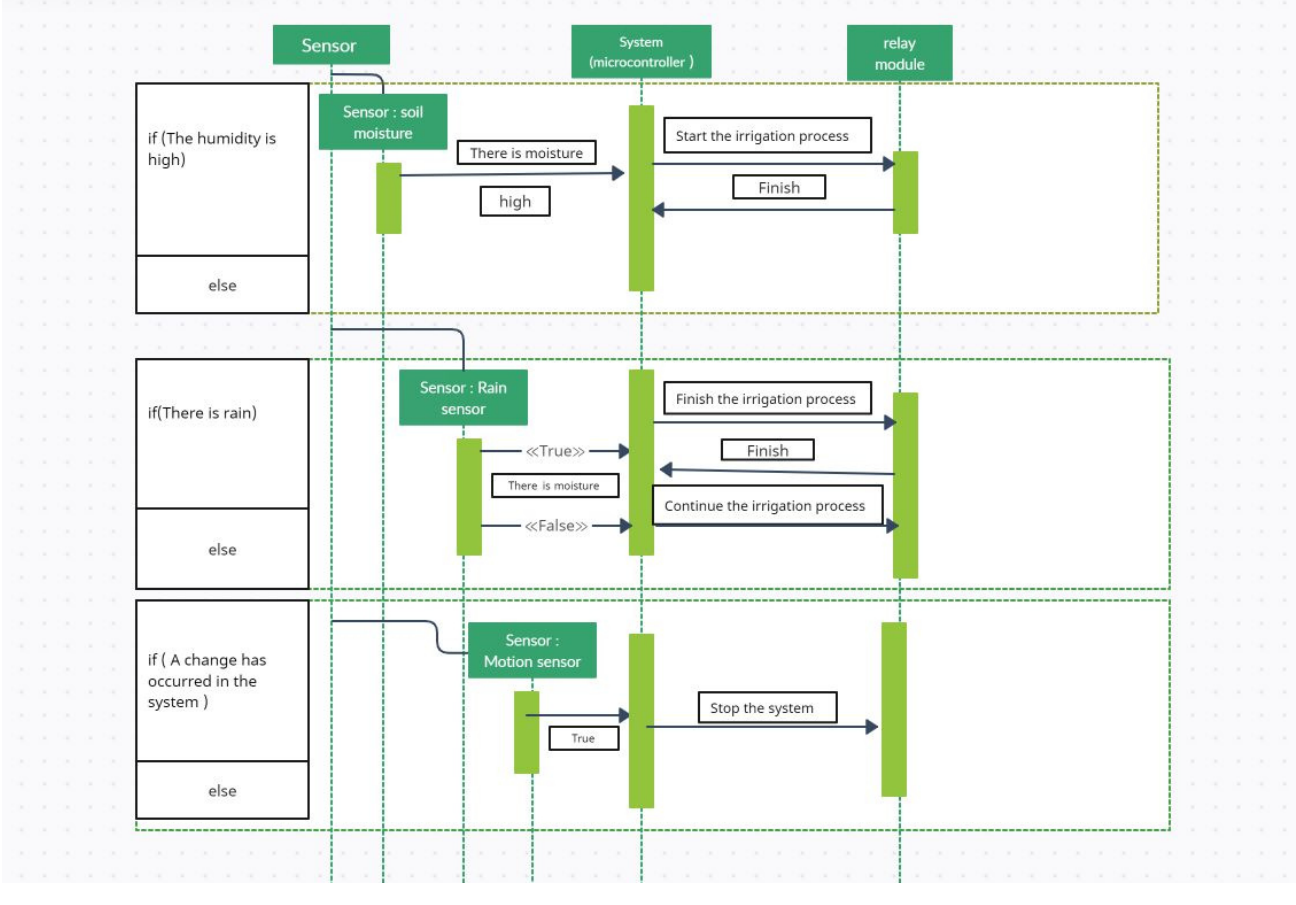
- **RUN WATER PUMB:**



10.3.CLASS DAIGRAM

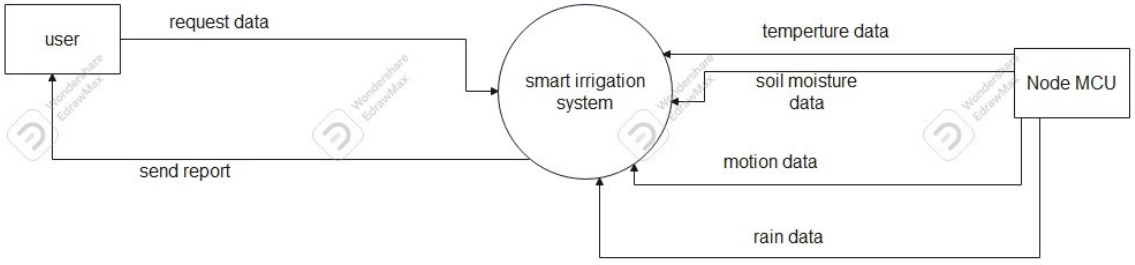


10.4.SEQUENCE DAIGRAM

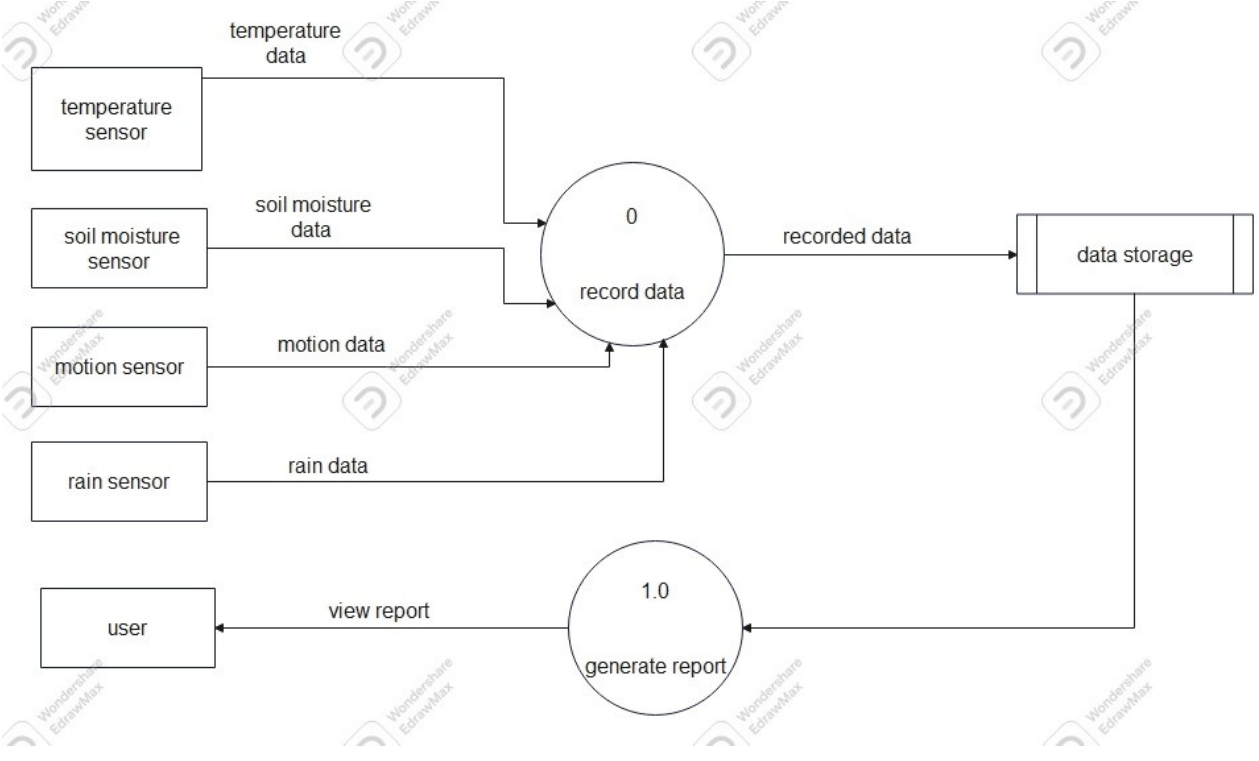


10.5.CONTEXT AND DFD

- **CONTEXT :**



- **DFD :**



11.SYSTEM Evolution

- **we can increase soil moisture sensor.**
- **Add water flow sensor to detect amount of water.**
- **we can use a microcontroller more efficient like Raspberry Pi.**
- **implementing a particular app for the system .**