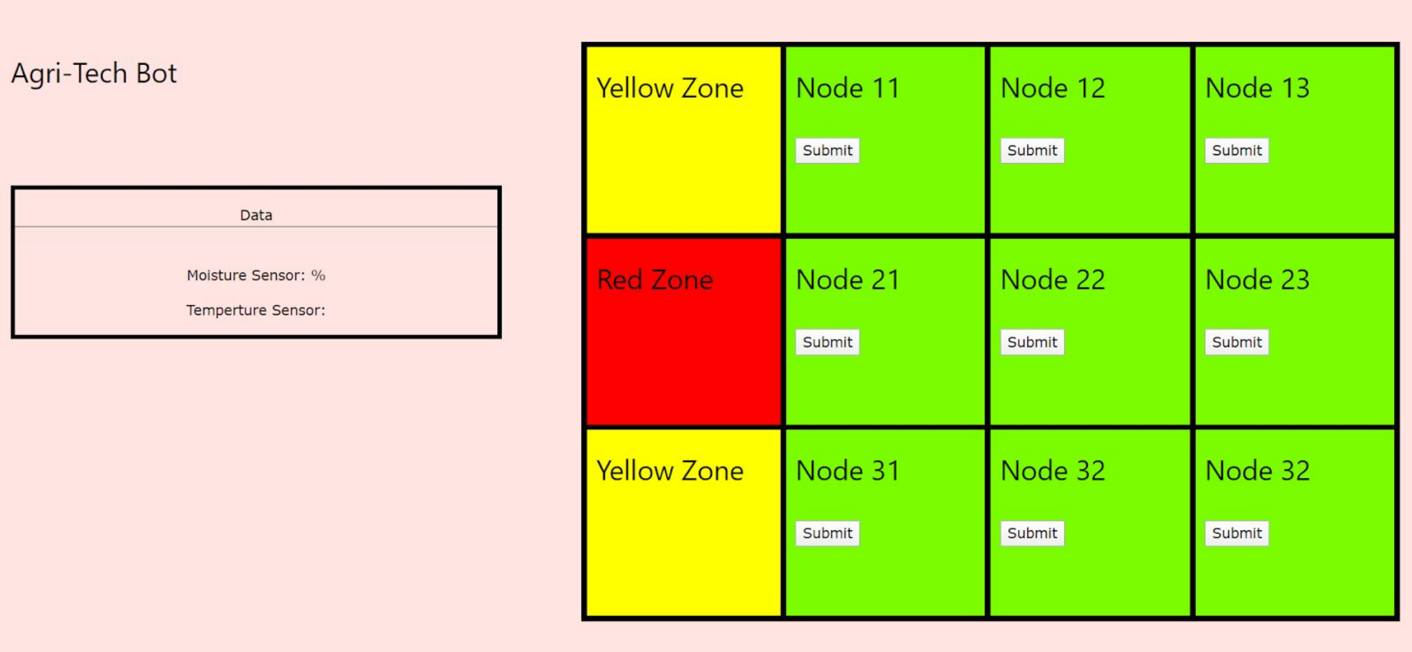
Working of the IoT.

The reading from the different types of sensors (moisture, humidity, temperature etc) will be taken by a microcontroller and transferred wirelessly to a database. The readings will be stored in the database using file handling. The files with the reading will be sorted according to the nodes present on the field. A **node** is a region/area on the actual cultivated land whose details can be obtained in order to analyse the condition of the soil and the crops.

All the sensors are connected to the **Arduino AT Mega** microcontroller, which will be responsible for registering the input data from the sensors. Since Arduino AT-Mega does not have an in-built WiFi or Bluetooth system, so an **ESP-8266 NODE-MCU** can be used to transfer the data wirelessly to the database. The transfer of data from Arduino to the esp8266 can take place through interrupt pins, then the esp8266 will create a local server, another python script will run in the background on the same local server as created by the esp8266 which will, in turn, read the data from the esp8266 and store the received data in the database using file handling. The micro-header name **ESP_MICRO.h** is a very important header file needed for the esp8266 which is essential for the data transfer as described.

After the data is stored in the database in a sorted manner, we will need to render the files to the web page which will display the data to the user. The web page is thus an essential entity here. The web page contains a responsive replica of the land under cultivation. A basic snapshot of the web page looks like the picture as shown below.



Agri-Tech Bot

Data

Moisture Sensor: %

Temperture Sensor:

Yellow Zone	Node 11 Submit	Node 12 Submit	Node 13 Submit
Red Zone	Node 21 Submit	Node 22 Submit	Node 23 Submit
Yellow Zone	Node 31 Submit	Node 32 Submit	Node 32 Submit

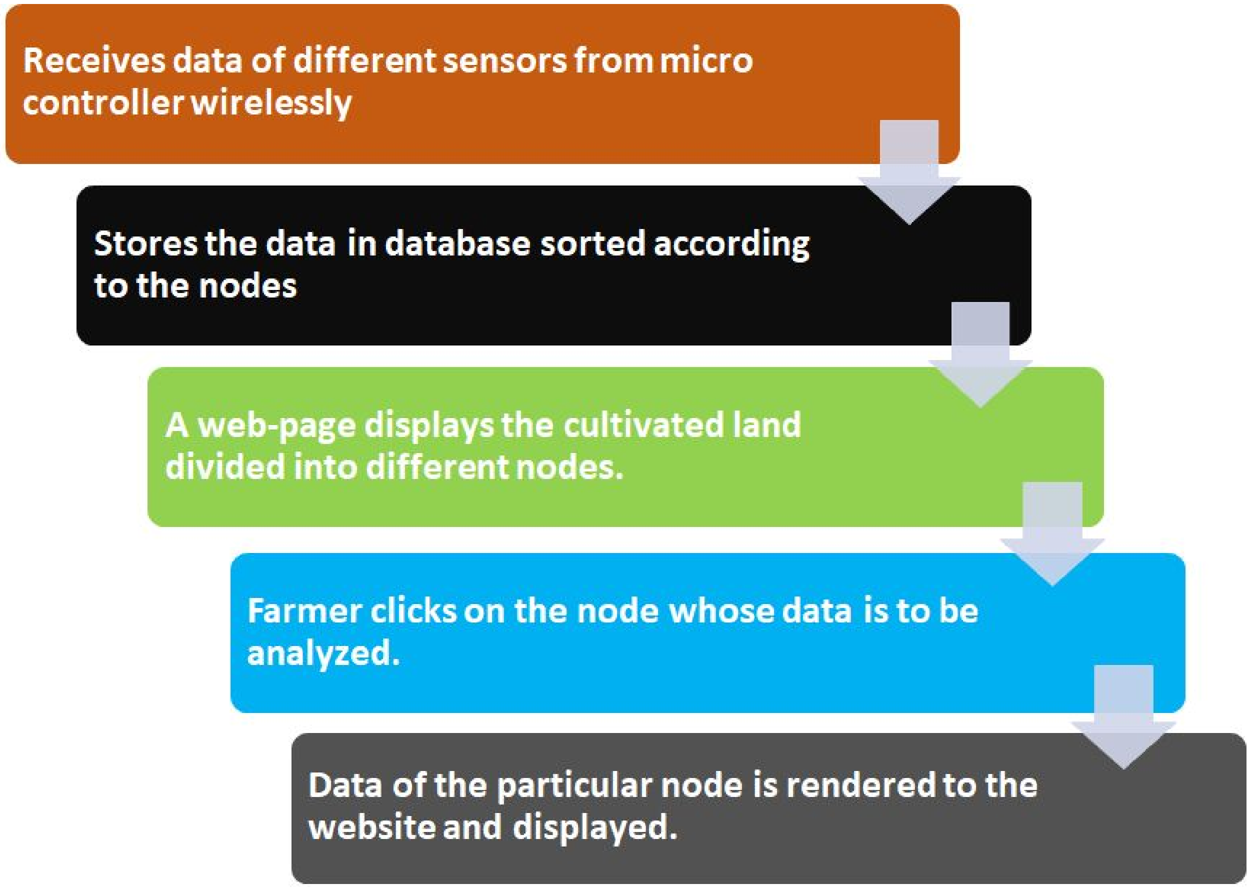
The Web page Interface

The table on the right-hand side is the replica of the cultivated land and in our case is the arena for the problem statement. The table is **responsive** and will respond to the click of the user. Each step is divided into different nodes. It is the region where the Agritech bot will take the readings from the sensors. The user has to click on the particular nodes whose conditions/readings he wants to analyze. The details of the sensor reading will be displayed on the box present on the right-hand side of the web page. The data from the database will be rendered to the website using flask. A note/message can be delivered regarding what can be expected of the readings.

Using Flask

Flask is a micro web framework written in python. Flask has been used in this project to render the web pages to a local host which can be accessed by the farmer. The sensor data stored in a local system cannot be directly read in the front end by any web pages and hence the necessity to read the data and render it to localhost. The flask reads the data of the particular node from the database and renders the data to the web page hosted on a local server.

A simplified flow chart of the process:-



Receives data of different sensors from micro controller wirelessly

Stores the data in database sorted according to the nodes

A web-page displays the cultivated land divided into different nodes.

Farmer clicks on the node whose data is to be analyzed.

Data of the particular node is rendered to the website and displayed.

One of the problems faced by the farmers in terrace farming is the lack of data from the land but this feature of the Agritech bot is the solution to the problem faced. The Agritech robot is envisioned to make the farmers smart, robust and decrease human dependency.