The LNM Institute of Information Technology, Jaipur-302031

Detailed Project Report
DPIoT 2023
Group 2

LNMIIT

Collaborative Bot System

> Why do we need it?

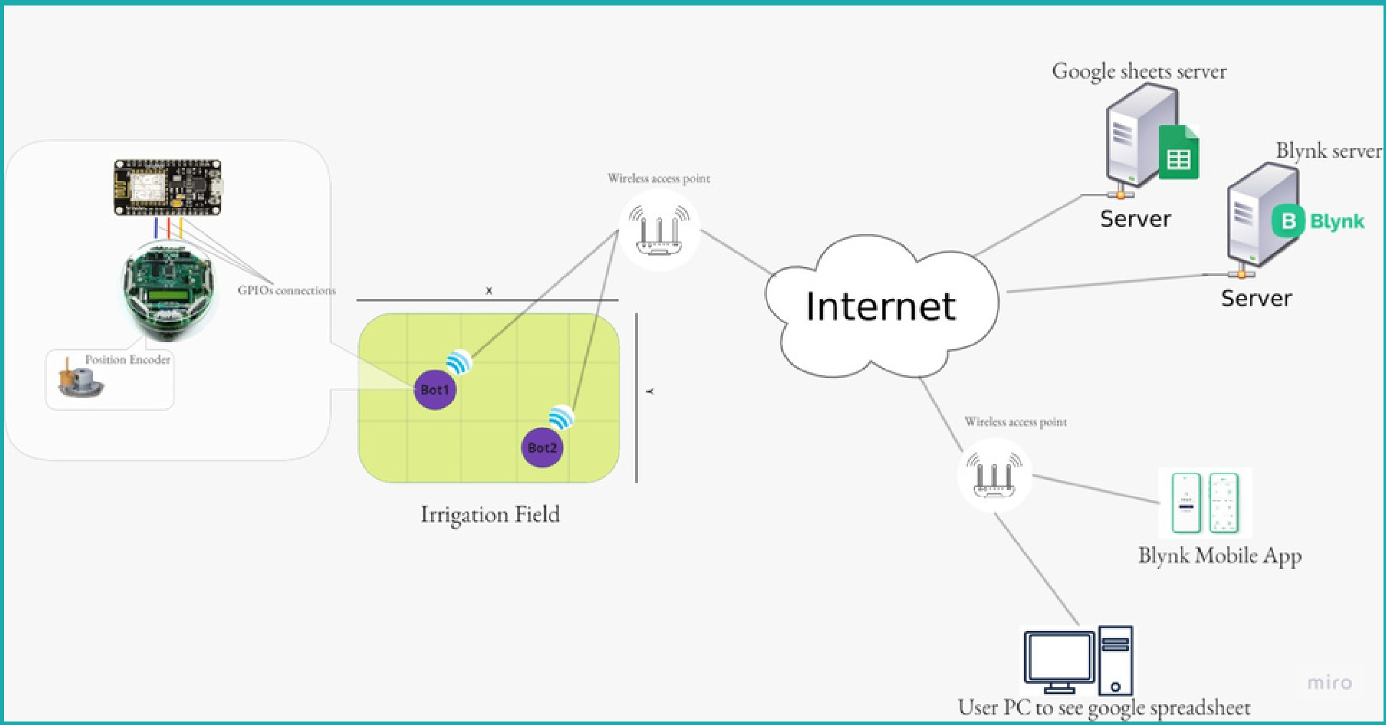
- Limited resources
- Maximum Optimization
- Least Wastage
- Less Human Intervention
- Maximum Growth

>What are we trying to do?

In terms of irrigation/addition of fertilizers, given the user's settings for a rectangular area (i.e the length and breadth), two firebird V robots will cover the complete area without overlaps or collisions, irrigating/fertilizing the entire area while also keeping track of the bots' positions.This information of position is then continuously sent to the cloud (here we are using google sheets to store the data).Thus this allows the user to also see the position of the bot at any point of time.

LNMIIT

>Scenario Diagram



>Logic/Working

- To cover a large area with two collaborating robots, we first partitioned the rectangular field into a matrix. This matrix may now be copied and used to direct each bot on their next step.

Thus, within our controller, we construct a matrix that represents the field, and the dimensions of this matrix are supplied by the user based on the physical field. In addition, the matrix constructed here contains two extra rows and columns to define the field's border. Each border element is assigned a number of 3, whereas the internal components are assigned a number of 0.

- Bot1 Function

After this matrix is generated in nodeMCU of Bot1, the bot is instructed to move according to the algorithm. Now using the position encoders we can precisely move the bot for a particular distance thus we move it equal to the length of one matrix block so that it reaches to the next block of the matrix. Now as soon as it moves from one block to another the data entry in the previous bot is set to 1.This position is also sent to Bot2 so that it can keep track of the location of Bot1 as well inside its matrix thus preventing overlaps.

This coordinate information is also supplied to Google Sheets in order to maintain track of the Bot's present location.

- Bot2 Function

Bot2 operates similarly to Bot 1, in that it updates the matrix with 2 and the location covered by Bot1 with 1.

This coordinate information is also supplied to Google Sheets in order to maintain track of the Bot's present location.

- ## Movement Pattern

The bots' movement pattern is in the shape of a square wave here to simplify complexity for time being, but in reality, for quicker and more comprehensive communication between bots, we may utilize Random area coverage techniques (which are a little more difficult to comprehend and implement).We can also use additional bots instead of two.

3	3	3	3	3	3
3	0	0	0	0	3
3	0	0	0	0	3
3	0	0	0	0	3
3	3	3	3	3	3

3	3	3	3	3	3
3	1	0	0	0	3
3	1	0	0	0	3
3	0	0	0	0	3
3	3	3	3	3	3

3	3	3	3	3	3
3	1	0	2	2	3
3	1	0	0	2	3
3	1	0	0	2	3
3	3	3	3	3	3

Matrix Generation

>Description of each component

- FireBird V Robot 2

- Sensor : Position Encoder(Image)

Position encoders are used in Firebird V to track the movement of the robot's wheels and to determine the robot's position and orientation in space.

The position encoders are used to measure the rotation of the wheels and provide feedback to the robot's control system, allowing it to accurately determine how far the robot has traveled and in which direction. This information is crucial for tasks such as navigation and mapping, where the robot needs to know its position relative to a map or to a set of predefined waypoints.

Our use: they are used to move the bot by a distance equal to the length of one matrix block.

- Actuator : 2 DC geared Motors(Image)

In Firebird V, DC geared motors are used to drive the robot's wheels and provide the necessary torque and speed to navigate various terrain types. The gearbox allows the robot to move slowly and precisely, which is important for tasks such as navigation, mapping, and obstacle avoidance. The motors are also equipped with position encoders, which allow the robot's control system to accurately track the position and orientation of the robot.

- NodeMCU (Microcontroller : ESP-8266 32-bit)

This the main computing unit of the entire iot project where in all the algorithms to run the robots in a particular fashion with suitable GPIOs connected to FireBird V to communicate with it and also the wifi based connection with the outside network(that is with the servers) as well as within network(that is with other nodeMCU) for sharing the data is initiated and established.

Connection with the FireBird V:

NodeMCU GPIO	FireBird V GPIO
Input	Output
D2	PD2(Extension Board pin 38)
Output	Input
D6	PD0 (Extension Board pin 20)
D7	PD1(Extension Board pin 19)

D6	D7	Movement
1	1	Forward
1	0	left
0	1	right
0	0	stop

Servers

Google sheets server : Here the data current positions of bot1 and bot2 are being stored along with the timestamp .To implement this first 2 sheets each for Bot1 and Bot2 were created and an App script also a service by Google was used to Deploy a web app that upon calling could be used to enter records in the tables.

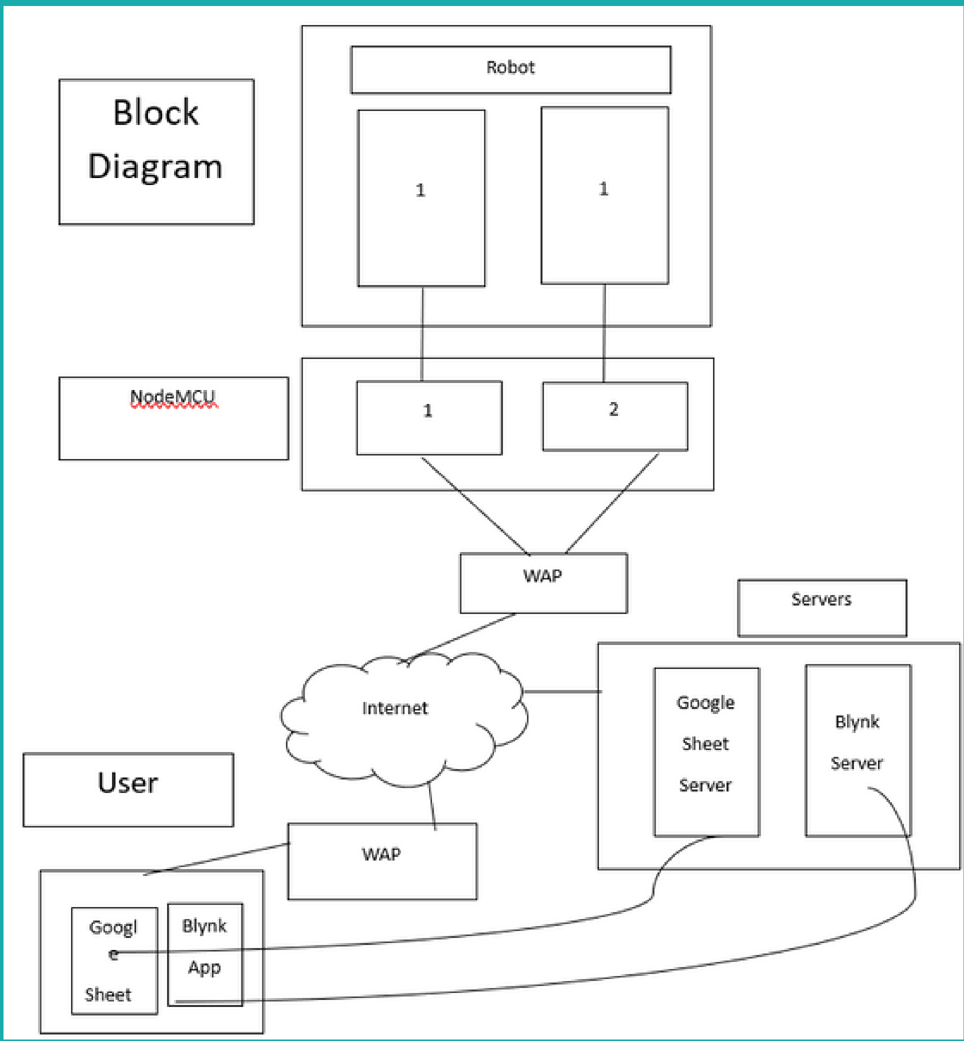
Blynk server : The app used by the user is requested from the blynk server which provides the user service to control any iot device which is connected to it through wifi.

User

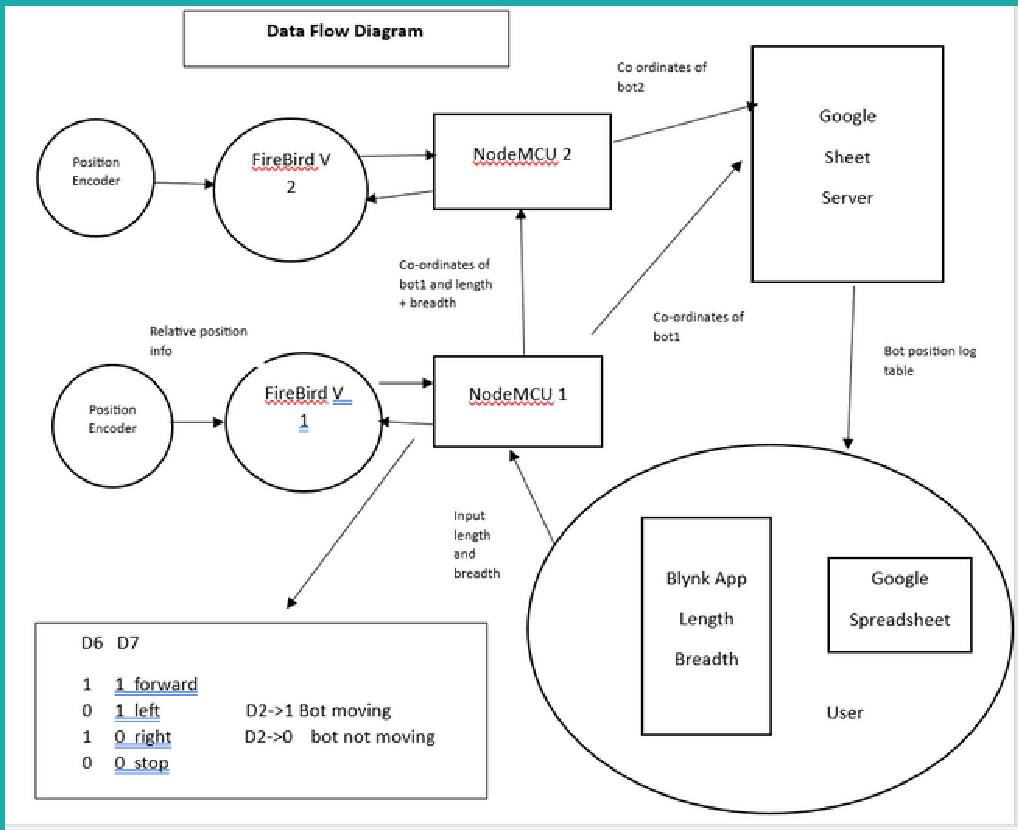
Blynk mobile App : users can customize the features in the app to align it to their needs and thus send and receive data with iot devices connected

Google Spreadsheet : users can access the positions_Log Tables of the bots on their desktop by using the url to open the google spreadsheet.

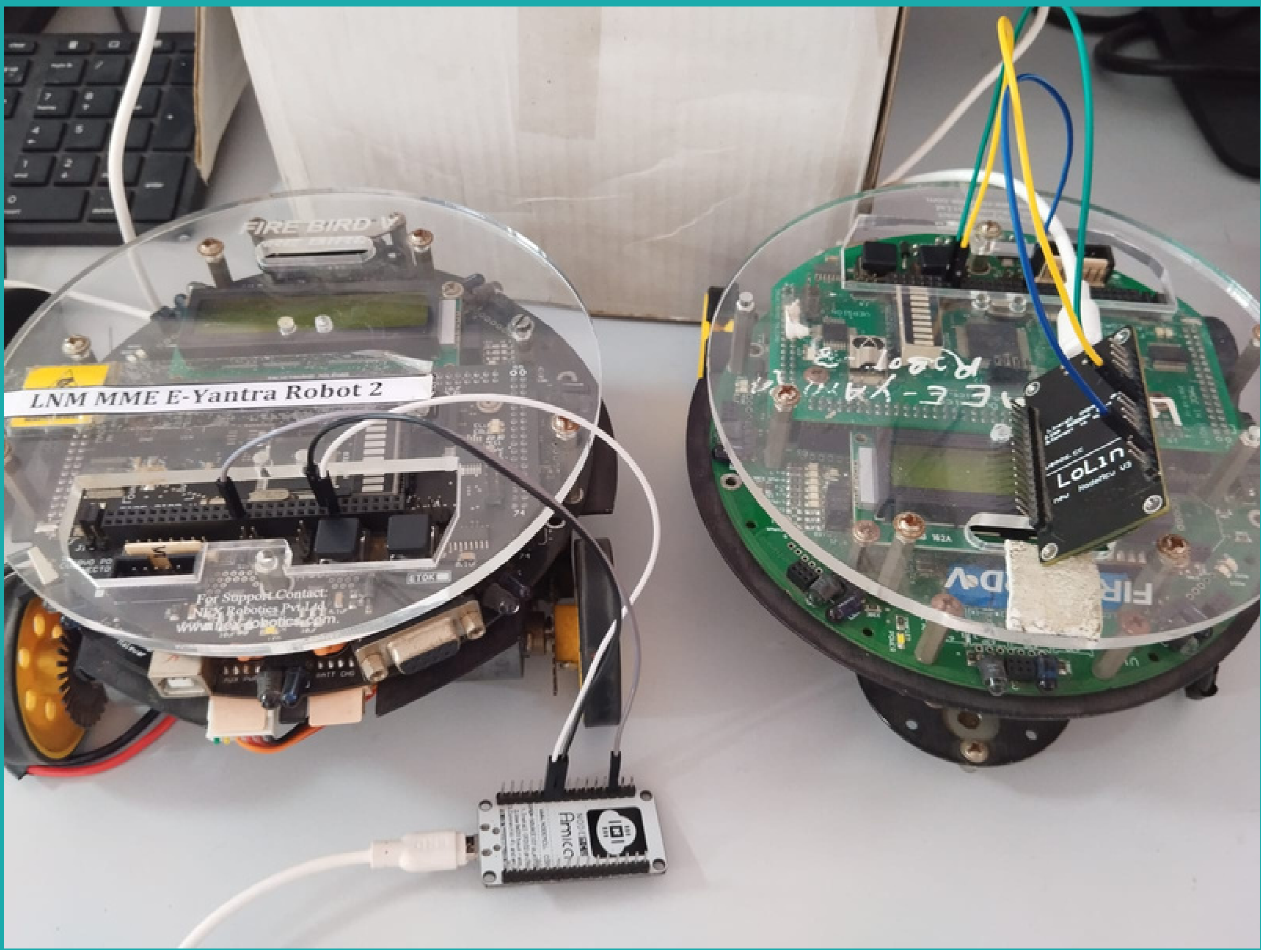
>Block Diagram



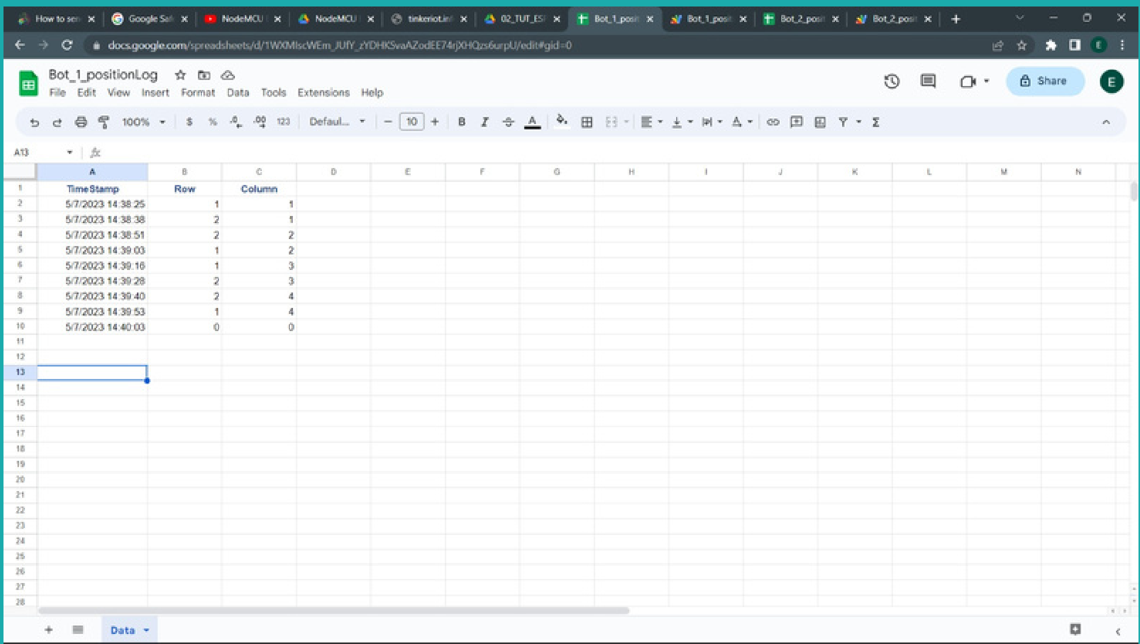
>Data Flow Chart



>Some Snippets of our Project



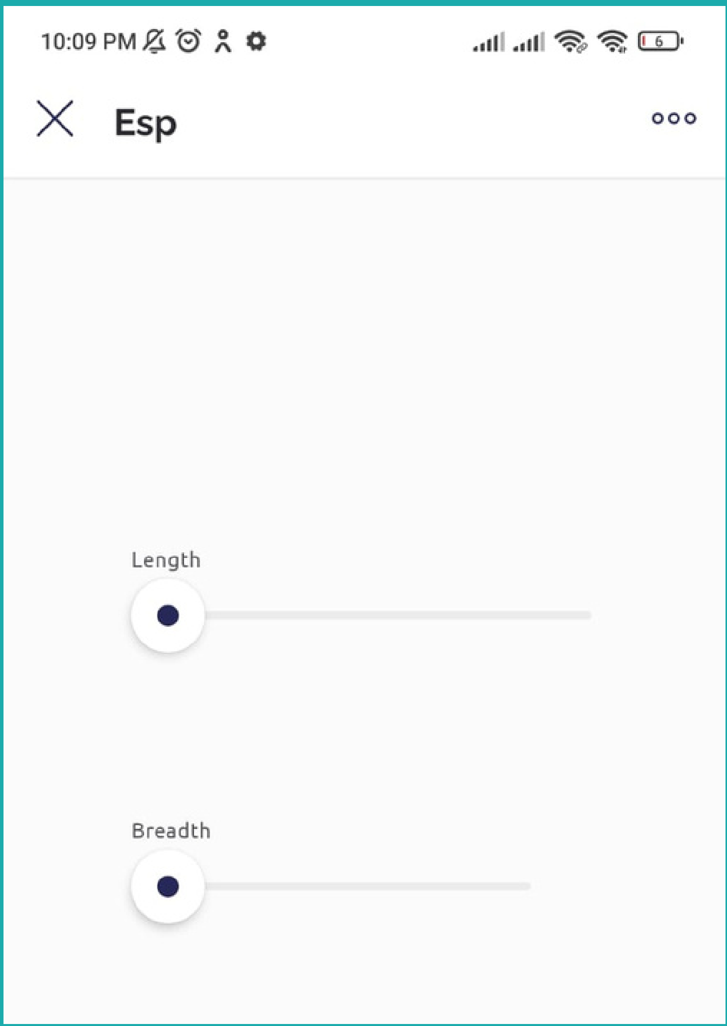
Robots



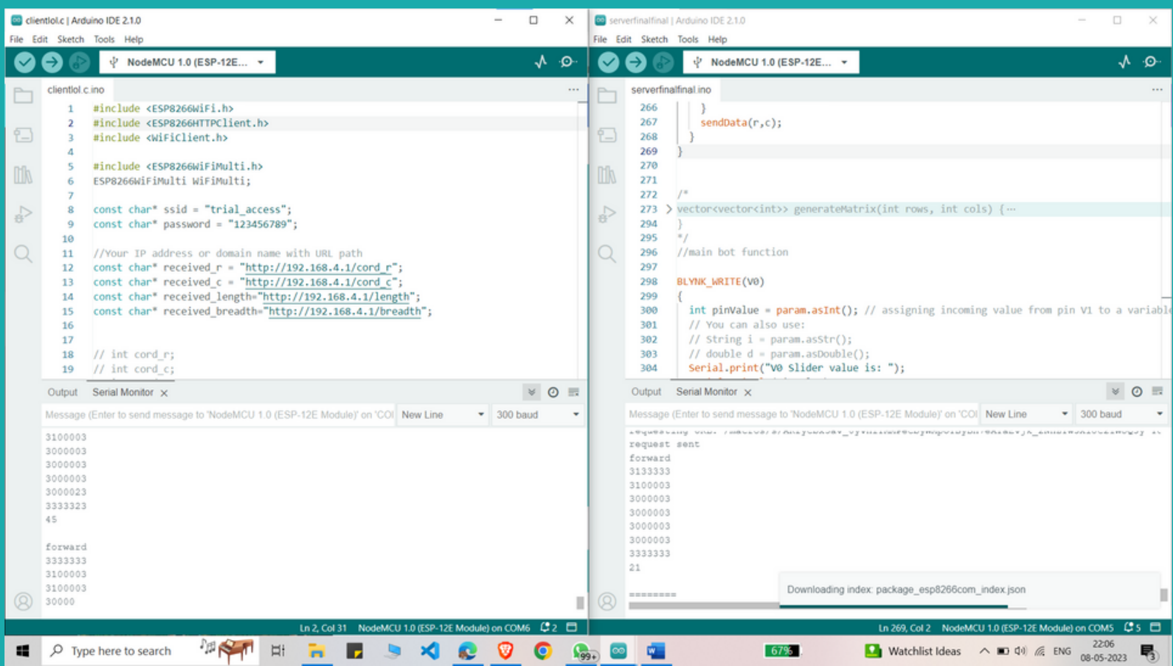
Bot_1_positionLog

TimeStamp	Row	Column
5/7/2023 14:38:25	1	1
5/7/2023 14:38:38	2	1
5/7/2023 14:38:51	2	2
5/7/2023 14:39:03	1	2
5/7/2023 14:39:16	1	3
5/7/2023 14:39:28	2	3
5/7/2023 14:39:40	2	4
5/7/2023 14:39:53	1	4
5/7/2023 14:40:03	0	0

Google Spread Sheet



Blynk Console



Client Server Model for NodeMCU

LNMIIT

Google Appscript Code

Google AppScript Code

```
function doGet(e) {
  Logger.log( JSON.stringify(e) );
  var result = 'Ok';
  if (e.parameter == 'undefined') {
    result = 'No Parameters';
  }
  else {
    var sheet_id = '1WXMIscWEm_JUfY_zYDHKSvaAZodEE74rjXHQzs6urpU';  //
Spreadsheet ID
    var sheet = SpreadsheetApp.openById(sheet_id).getActiveSheet();
    var newRow = sheet.getLastRow() + 1;
    var rowData = [];
    var Curr_Date = new Date();
    rowData[0] = Curr_Date; // Date in column A
    for (var param in e.parameter) {
      Logger.log('In for loop, param=' + param);
      var value = stripQuotes(e.parameter[param]);
      Logger.log(param + ':' + e.parameter[param]);
      switch (param) {
        case 'Row':
          rowData[1] = value;
          result = 'Row Written on column B';
          break;
        case 'Column':
          rowData[2] = value;
```

Atmel Studio Code Snippet

ATMEL STUDIO CODE

```
#define F_CPU 14745600
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

volatile unsigned long int ShaftCountLeft = 0; //to keep track of left position encoder
volatile unsigned long int ShaftCountRight = 0; //to keep track of right position encoder
volatile unsigned int Degrees; //to accept angle in degrees for turning

//Function to configure ports to enable robot's motion
void motion_pin_config (void)
{
	DDRA = DDRA | 0x0F;
	PORTA = PORTA & 0xF0;
	DDRL = DDRL | 0x18;   //Setting PL3 and PL4 pins as output for PWM generation
	PORTL = PORTL | 0x18; //PL3 and PL4 pins are for velocity control using PWM.

}

//Function to configure INT4 (PORTE 4) pin as input for the left position encoder
void left_encoder_pin_config (void)
{
	DDRE  = DDRE & 0xEF;  //Set the direction of the PORTE 4 pin as input
	PORTE = PORTE | 0x10; //Enable internal pull-up for PORTE 4 pin
}

//Function to configure INT5 (PORTE 5) pin as input for the right position encoder
void right_encoder_pin_config (void)
{
	DDRE  = DDRE & 0xDF;  //Set the direction of the PORTE 4 pin as input
	PORTE = PORTE | 0x20; //Enable internal pull-up for PORTE 4 pin
```

>Conclusion and Recommendation

In conclusion, our IoT-based project demonstrated the use of two bots for efficient irrigation of a field, or any related purpose that requires physical coverage of a field in a random/non-random fashion by simulating a pseudo-random path. The bots were able to cover all parts of the field by communicating with each other and a cloud-based system used for initial and final executions. The project leveraged IoT technology to enable real-time monitoring and control of the bots' movements, ensuring that the irrigation process was completed in an optimal manner. Through the use of sensors and communication protocols, our system was able to deliver accurate and timely data to the cloud, enabling users to make informed decisions about their needs. Overall, this project showcases the potential of IoT technology to improving efficiency, reducing waste or even for security purposes if equipped with a camera covering a given field at all times.

Also, we recommend certain additions to the bot which could further make these bots more powerful and efficient which would be the use of path tracing and being totally random in its movement and still covering the field without colliding and overlapping

>References

- https://roboram.files.wordpress.com/2016/03/fire-bird-v-atmega2560-hardware-manual-2010-12-21.pdf
- https://github.com/akshar100/eyantra-firebird-resources/blob/master/Fire%20Bird%20V%20ATMEGA2560%20Robots%202010-12-29/Expirements/5_Position_Control_Interrups/Pos_Con_Interrupts/Pos_Con_Interrupts.c
- https://randomnerdtutorials.com/esp8266-nodemcu-client-server-wi-fi/
- https://www.youtube.com/watch?v=okNECYf2xlY&t=617s
- https://www.youtube.com/watch?v=AvSCAxbqvvE

>Team Members

Lakshya Singh
21UCS115

Elish Baraiya
21UCS077

Chirag Bhayana
21ucs050