AUTOMATIC PASSENGER COUNT

**ABSTRACT**

The main objective of this Project is to identify the less crowded coaches for people who want to travel in metros or trains because this can help passengers make informed decisions , leading to a more even distribution of passengers across coaches and Real-time passenger occupancy data lets companies arrange additional vehicles to meet passenger needs making sure that no passenger is left behind even in bad weather conditions and improve passenger comfort and safety. Passenger counting systems are accurate, and they reduce expenses all around for any company that’s in the transportation industry. V-Count sensors collect and provide valuable data on transport performance, allowing cost reduction based on route and peak hour performance.

Embedded counting algorithms are proposed to count passenger arrivals and departures in real time. Experiments in a real scenario expose that the proposed scheme achieved a high percentage of success in passenger count.

In this project, we present a low cost WiFi-based system (Node MCU) for occupancy estimation in trains and bus. In particular, we propose a novel algorithm for occupancy estimation that reduces the problem of overestimation encountered in WiFi-based occupancy estimation approaches i.e, V-count sensors. We validate the accuracy and efficiency of the proposed algorithm and the WiFi-based occupancy estimation solution via a real-world experiment

### TABLE OF CONTENTS

[Approval sheet ii](#_Toc41878)

[Candidates declaration iii](#_Toc41879)

[Certificate iv](#_Toc41880)

[Acknowledgement v](#_Toc41881)

[Declaration vi](#_Toc41882)

[Abstract vii](#_Toc41883)

#### Table of Contents

1. **Chapter 1 : INTRODUCTION .......................................................................................................... 1-2** 
   1. Introduction ..................................................................................................................................................................1
   2. Objective of the work................................................................................................................................................... 1
   3. Scope of the project ...................................................................................................................................................... 2

1. **Chapter 2: LITERATURE REVIEW .................................................................................................... 3-8** 
   1. Introduction .......................................................................................................................................................................3
      1. Existing APC Systems ............................................................................................................................................. 3
         1. Video-based Systems...................................................................................................................................... 3
         2. Infrared Radiation (IR) Sensor-based Systems ............................................................................... 3
         3. Weight-based Systems .................................................................................................................................. 3
         4. Radio Frequency Identification(RFID) Systems ......................................................................... 4

* + 1. Recent Advancements In Sensors Technology In APC Systems ....................................................... 4
       1. Machine Learning ............................................................................................................................................. 4
       2. IoT Integration .............................................................................................................................................. 4

* + - 1. Sensor Fusion ................................................................................................................................................ 4

* 1. Proposal ............................................................................................................................................................................. 5
  2. Description ...................................................................................................................................................................... 5
     1. Block Diagram ......................................................................................................................................................... 6
     2. Data Flow Diagram ............................................................................................................................................... 6
     3. Circuit Diagram ...................................................................................................................................................... 7
  3. Problem Definition and Approach ....................................................................................................................... 8

1. **Chapter 3: IMPLEMENTATION AND METHODOLOGY ................................................................... 9-16** 
   1. Project Design ..................................................................................................................................................................9
   2. Components used ...................................................................................................................................................... 10
      1. Node MCU ESP8266 .......................................................................................................................................... 10
      2. IR Sensors ............................................................................................................................................................... 11
      3. 16\*2 LCD module ............................................................................................................................................... 12
      4. I2C Interface .......................................................................................................................................................... 14
      5. Breadboard ............................................................................................................................................................ 15
   3. Software Implementation ..................................................................................................................................... 16

1. **Chapter 4: RESULT AND DISCUSSION......................................................................................... 27-29** 
   1. Results ..............................................................................................................................................................................27
   2. Discussion .........................................................................................................................................................................29
   3. Applications ..................................................................................................................................................................... 29
   4. Advantages ....................................................................................................................................................................... 29
   5. Limitations ....................................................................................................................................................................... 29

4.4 Improvements ...............................................................................................................................................................29

1. **Chapter 5: SUMMARY AND CONCLUSION ...................................................................................... 30** 
   1. Summary ...........................................................................................................................................................................30
   2. Conclusion ........................................................................................................................................................................30

5.2 Future Scope ....................................................................................................................................................................30

1. **Chapter 6: REFERENCES.................................................................................................................. 31**

**TABLE OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Serial**  **Number** | **Name** | **Source** | **Page Number** |
| 1 | Fig 2.1: Block Diagram | - | 6 |
| 2 | Fig 2.2: Data Flow  Diagram | - | 6 |
| 3 | Fig 2.3: Circuit Diagram | - | 7 |
| 4 | Fig 2.4: Implementation | - | 8 |
| 5 | Fig 3.1: NodeMCU  ESP8266 | Components101 | 10 |
| 6 | Fig 3.2: IR Sensor | Dsmonline | 11 |
| 7 | Fig 3.3: 16x2 LCD | Electronicscomp | 13 |
| 8 | Fig 3.4: I2C Interface | Encrypted-tbn0.gstatic | 14 |
| 9 | Fig 3.5: Breadboard | Encrypted-tbn0.gstatic | 15 |
| 10 | Fig 4.1: LCD Display  Output | - | 27 |
| 11 | Fig 4.2: Webserver | - | 28 |
| 12 | Fig 4.3: Serial Monitor  Server Connection | - | 28 |

**The following Abbreviations have been used throughout our project and it should be understood the same for the whole document.**

|  |  |
| --- | --- |
| ABBRIVIATION | FULL FORM |
| APC | Automatic Passenger Count |
| LED | Light Emitting Diode |
| IR | Infrared Radiation |
| LCD | Liquid Crystal Display |
| MCU | Micro Controller Unit |
| LIDAR | Light Detection and Ranging |

### CHAPTER 1

INTRODUTION

#### 1.1 INTRODUCTION

In today's fast-paced world, efficient and comfortable public transportation is essential.

One significant challenge faced by commuters is finding a less crowded coach within a train.

Rail travel remains a popular mode of transportation, especially in densely populated regions. However, peak hours often lead to overcrowded coaches, causing discomfort and inconvenience for passengers. To address this issue, we propose an Automatic Passenger Count System designed to accurately estimate the number of passengers in each coach of a train. We propose an innovative solution utilizing IR sensors and NodeMCU to automatically count passengers and identify the least occupied coach.

Currently, passengers often rely on visual estimation to choose a less crowded coach, which can be inaccurate and time-consuming. This can lead to overcrowding, discomfort, and reduced overall passenger satisfaction.

Our system aims to provide real-time information about passenger occupancy in each coach, empowering passengers to make informed decisions. By strategically placing IR sensors within the coaches and connecting them to NodeMCU devices, we can accurately count the number of passengers entering and exiting each coach. The collected data will be transmitted wirelessly to a central server, where it will be processed and displayed on digital screens at train stations or through a mobile application.

#### 1.2 OBJECTIVE OF THE WORK

The main objective is to develop a cost efficient system which can help new passengers to automatically count passengers and identify the least occupied coach.

#### 1.3 SCOPE OF THE PROJECT

This project aims to develop and implement an automated system to accurately count passengers entering and exiting train coaches in real-time.

Utilize IR sensors to detect passenger movement. Employ NodeMCU devices to process sensor data and transmit it wirelessly. Establish a central server to collect, process, and store passenger count data from all coaches.

Analyze the collected data to identify trends and patterns in passenger behavior. Display real-time coach occupancy information on digital screens at train stations. Develop a mobile application to provide passengers with real-time updates. Integrate the system with existing train management systems for comprehensive data analysis.

Design and implement a robust system architecture for reliable data collection and transmission. Ensure the system is easy to install and maintain in various train environments.

Develop a comprehensive maintenance plan to address hardware and software issues. Implement measures to protect passenger privacy and data security. Anonymize collected data to prevent personal identification.

### CHAPTER 2

LITERATUREREVIEW

#### 2.1 INTRODUCTION

The increasing demand for efficient and reliable public transportation systems has led to the development of innovative technologies to improve passenger experience and operational efficiency. One such technology is Automatic Passenger Counting (APC) systems. These systems use various sensors and data processing techniques to accurately count passengers entering and exiting vehicles, providing valuable insights for transportation authorities and operators.

**2.1.1 Existing APC Systems:**

Several technologies have been explored for passenger counting, each with its own advantages and limitations:

**2.1.1.1 Video-based Systems:**

Advantages: High accuracy, potential for additional analytics (e.g., passenger behavior, demographics).

Disadvantages: High computational cost, privacy concerns, and sensitivity to lighting conditions.

**2.1.1.2 Infrared (IR) Sensor-based Systems:**

Advantages : Reliable, cost-effective , and less susceptible to environmental factors. Disadvantages: Potential for inaccuracies in crowded conditions or with specific passenger behaviors (e.g., carrying large bags).

**2.1.1.3 Weight-based Systems:**

Advantages: High accuracy, especially for stationary vehicles.

Disadvantages: High installation cost, sensitivity to road conditions and vehicle load distribution.

**2.1.1.4 Radio Frequency Identification (RFID) Systems:**

Advantages: High accuracy, potential for additional data collection (e.g., passenger demographics, travel patterns).

Disadvantages: High initial cost, requirement for passenger cooperation (e.g., carrying RFID cards).

**2.1.2 Recent Advancements In Sensor Technology In APC Systems:**

**2.1.2.1 Machine Learning:**

Advanced algorithms can enhance the accuracy of video-based systems by improving object detection and tracking.

**2.1.2.2 IoT Integration:**

IoT enables real-time data transmission and remote monitoring of APC systems.

**2.1.2.3 Sensor Fusion:**

Combining multiple sensor technologies (e.g., IR and video) can improve accuracy and robustness.

APC systems have the potential to revolutionize public transportation by providing accurate and timely passenger count data. By addressing the challenges and leveraging emerging technologies, we can develop more reliable and efficient APC systems that benefit both passengers and transportation operators. Future research should focus on improving sensor accuracy, developing robust data processing algorithms, and ensuring data privacy and security.

By exploring these research directions, we can further enhance the capabilities of APC systems and contribute to the development of smarter and more sustainable transportation solutions.

#### 2.2 PROPOSAL

There have been many past works on Automatic Passenger Count(APC) using NodeMCU. Here are some examples:

Bi-directional Visitor Counter : To allow only upto the maximum number of people into the particular room. They also use the 2 IR Sensors and the LCD 16\*2.

ESP8266 Object Counting : It is conveyor which is used to count the number of objects entered and it uses a dc motor along with the IR Sensor and the LCD 16\*2.

ESP8266 &MQTT Visitor Counter: To implement the counter they use MQ Telemetry Transport(MQTT) and it is a IoT based project.

In summary, there have been many past works. But the LCD turned off if both the IR Sensors detect at a time and the theme of the projects are also different. For the problem, we wrote an code to rectify it and the code worked perfectly.

#### 2.3 DESCRIPTION

Automatic Passenger Count (APC) is a bi-directional counter to display the count of the people in coaches separately when it comes to a train and count of total number of people in one bus. The project implemented using two IR Sensors and NodeMCU Module.

The IR Sensors which are connected two the NodeMCU detects the passenger and send signals to the NodeMCU, which processes the data i.e, whether there is an increment or decrement or nothing.The NodeMCU then send the message to the LCD 16\*2 to display the count and with the IP address we can watch the count in the webpage.

##### 2.3.1 BLOCK DIAGRAM

WiFi Module

NodeMCU

8266)

ESP

(

IR Sensor 2

Power Supply

LCD

16\*2

Display

Webserver

IR Sensor 1

**Fig 2.1** Block Diagram

##### 2.3.2 DATA FLOW DIAGRAM

Software

Text Count

IR Sensors

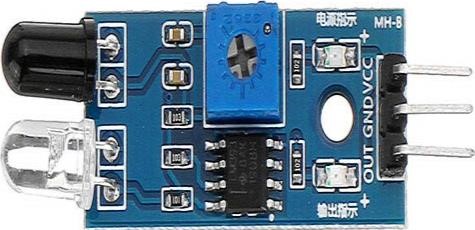
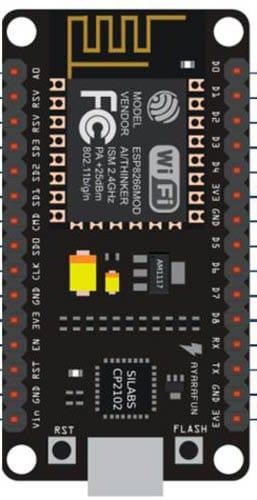
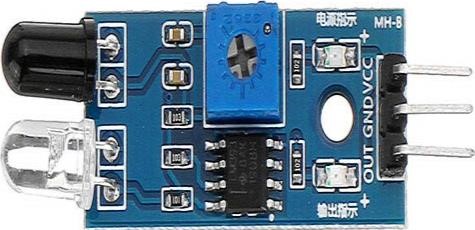
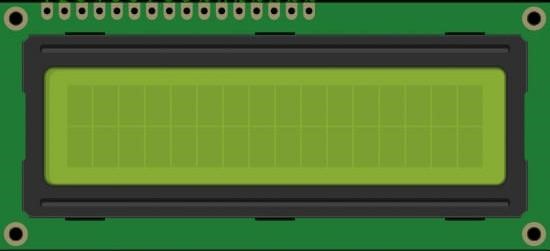
NodeMCU

**Fig 2.2** Data Flow Diagram

##### 2.3.3 CIRCUIT DIAGRAM

**Fig 2.3**

Circuit Diagram



#### 2.4 PROBLEM DEFINITION AND APPROACH

To address the difficulties by new passengers to identify the less crowded coaches. The primary goal is to develop an automated system that accurately counts passengers entering and exiting a vehicle (e.g., bus, train) using IR sensors and a NodeMCU board. This system aims to replace manual passenger counting, which is often prone to errors and inefficiencies. IR Sensors: Position IR sensors at strategic locations, such as entrances and exits, to detect the presence of passengers.

NodeMCU: This microcontroller will process the sensor data and communicate with other devices (e.g., a server) to transmit the passenger count.

Power Supply: Ensure a reliable power source for the sensors and NodeMCU, considering the vehicle's power system or using batteries.

Wiring: Connect the IR sensors to the NodeMCU's digital input pins.

Server-Side Processing: Set up a server to receive the passenger count data from the NodeMCUs. Store the data in a database for analysis and reporting.

By following these steps and considering the additional factors, you can develop a robust and efficient automatic passenger counting system using IR sensors and NodeMCU.

Initializing Hardware

components

Input Sensor

Passenger

Sensor recognition

Text count

Wireless communication

### CHAPTER 3

IMPLEMENTATION AND

METHODOLOGY

#### 3.1 PROJECT DESIGN

Automatic Passenger Count (APC) is a bi-directional counter to display the count of the people in coaches separately when it comes to a train and count of total number of people in one bus. We place two IR Sensors one after another consecutively. For entry or for exit we have to use both the IR sensors that’s why it is called a bi-directional counter.

The basic work of the project starts from IR sensors detecting the count of the passengers. This system utilizes two IR (Infrared) Sensors strategically place to detect passenger entry and exit. The NodeMCU (ESP8266) board, a versatile microcontroller, processes the sensor signals and updates a passenger count.

We connect the NodeMCU to the Breadboard for our convenience and the two IR Sensors to the NodeMCU and then the LCD 16\*2 via an interface called I2C interface to the NodeMCU.

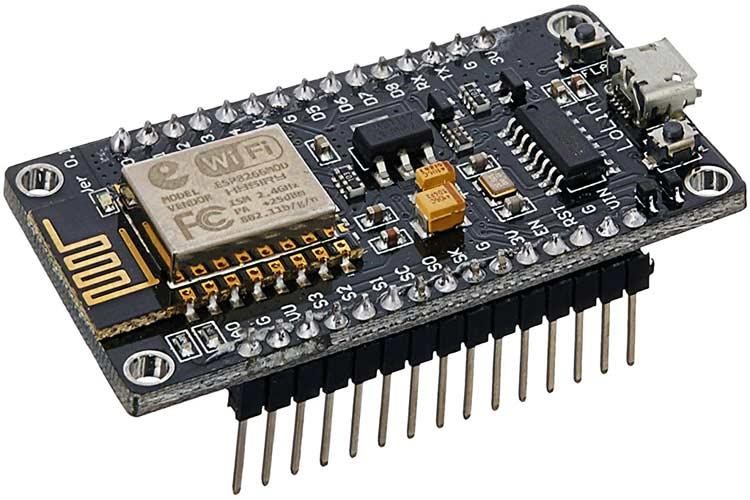
Firstly, when the IR Sensor 1 detects the passenger the LED of the IR Sensor 1turned on after some delay only the IR Sensor 2 has to be turned on so we place it at some distance from the IR Sensor 1. After the detection of IR Sensor 2 the count of the Passengers is incremented. In the same way if IR Sensor 2 detects the passenger first and then the IR Sensor 1 then the count of the Passenger is decremented and it is showcased on the LCD display.

For the webserver we use the HTML Code in the Arduino IDE by writing for the WiFi Server Where an IP Address is displayed in the serial monitor if we connect the system with a WiFi service. We can paste thatIP address in the Chrome or Google to get the count of the passengers

#### 3.2 COMPONENTS USED

Automatic Passenger Counting (APC) systems are innovative solutions designed to monitor and manage passenger flow in public transportation systems. One effective implementation involves using a NodeMCU microcontroller, which facilitates Wi-Fi connectivity for real-time data transmission. The system employs infrared (IR) sensors to accurately detect when passengers enter or exit a vehicle by interrupting an IR beam. This data is then processed and displayed on an LCD module, providing immediate feedback on passenger counts. The entire setup is typically assembled on a breadboard, allowing for easy prototyping and adjustments. This combination of components enables efficient monitoring of passenger numbers, ultimately aiding in better resource management and service optimization.

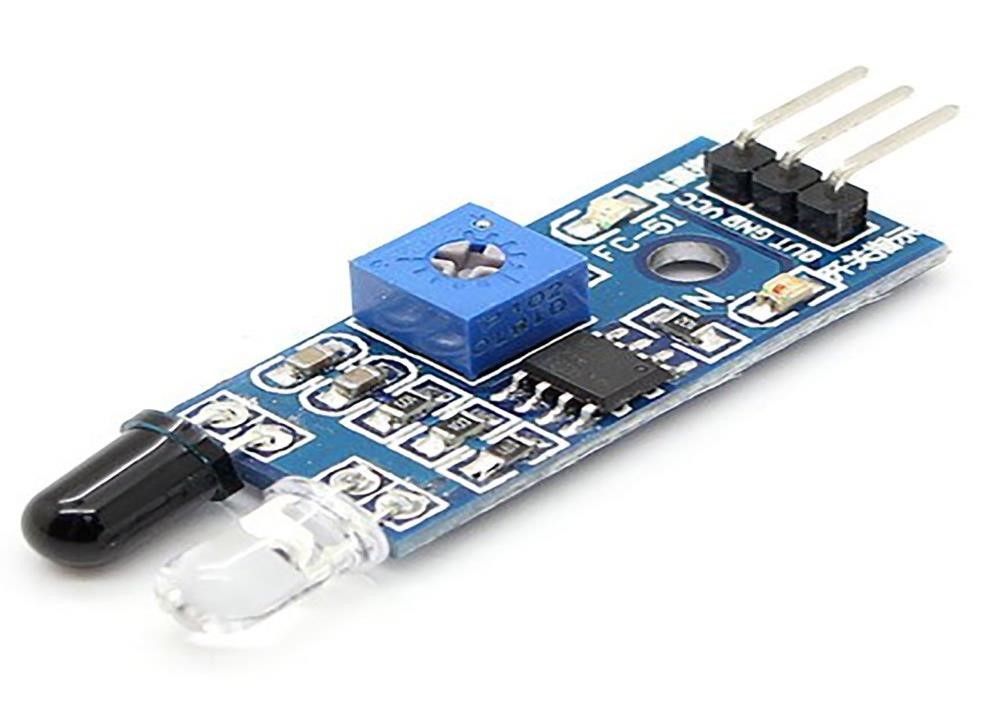
##### 3.2.1 NODEMCU ESP8266



###### **Fig 3.1** NodeMCU ESP8266

NodeMCU ESP8266 is a highly versatile open-source development platform built around the ESP8266 Wi-Fi chip, making it a popular choice for Internet of Things (IoT) applications. This platform operates at 3.3V and typically includes 4MB of Flash memory and 128KB of RAM, providing ample resources for various projects. The microcontroller features the Tensilica Xtensa 32-bit LX106 RISC processor, with an adjustable clock frequency ranging from 80MHz to 160MHz, allowing for efficient processing. One of its standout features is its integrated Wi-Fi capability, which facilitates seamless connectivity and supports multiple communication protocols, including UART, SPI, and I2C. NodeMCU is powered via a Micro USB port or an external power supply and offers 16 GPIO pins, one analog pin (A0), and dedicated pins for SPI and UART, enabling versatile input/output options. The platform iseasily programmable using the Arduino IDE or LUA scripting, making it accessible for both beginners and experienced developers. NodeMCU ESP8266 finds extensive applications in home automation, allowing users to remotely control lights, fans, and appliances through smartphone apps or web interfaces. It is also employed in smart agriculture, enabling real-time monitoring of soil moisture, temperature, and humidity for optimized irrigation. Environmental monitoring projects utilize NodeMCU to track air quality and other parameters, transmitting data to cloud platforms for analysis. Moreover, it plays a crucial role in IoT-based solutions for smart cities, contributing to traffic monitoring, waste management, and public safety initiatives. Additionally, the platform is favored for DIY projects, such as weather stations, smart mirrors, and automated pet feeders, thanks to its ease of use and robust community support. Overall, NodeMCU ESP8266 serves as an ideal foundation for developing a wide range of connected devices and applications across diverse domains, solidifying its position as a staple in the IoT ecosystem

##### 3.2.2 IR SENSORS



###### **Fig 3.2** IR Sensor

An infrared (IR) sensor is an electronic device that detects infrared radiation, which is emitted by objects based on their temperature. These sensors are widely used in various applications due to their ability to detect heat and motion without physical contact. There are two primary types of IR sensors: active and passive. Active IR sensors emit infrared light and measure the reflection from nearby objects, while passive IR sensors detect the infrared radiation emitted by objects, such as human bodies or warm surfaces. Common applications of IR sensors include motion detection for security systems, where they can trigger alarms or lights when movement is detected, and in remote controls for televisions and other devices, where they enable wireless communication. Additionally, IR sensors are utilized in temperature sensing applications, such as non-contact thermometers, and in robotics for obstacle detection and navigation. Their low cost, compact size, and ease of integration make IR sensors a popular choice in consumer electronics, automotive systems, and industrial automation, contributing to their widespread use in modern technology.

IR sensors operate based on the principle of detecting infrared radiation, which falls outside the visible spectrum and is primarily associated with heat emitted by objects. They are composed of various components, including a photodiode or phototransistor that responds to incoming infrared light, a lens to focus the IR signals, and often an amplifier to enhance the sensor's output. The sensitivity and range of IR sensors can vary depending on their design and intended application; for instance, passive infrared (PIR) sensors are specifically designed to detect motion by sensing changes in infrared radiation levels caused by moving warm bodies, making them ideal for security lighting and automatic door systems. In contrast, active IR sensors, such as those used in proximity detection, can measure distances by emitting IR light and calculating the time it takes for the light to return after reflecting off an object. Furthermore, IR sensors are also used in various non-contact applications, including industrial automation for object detection, in automotive systems for parking assistance, and in healthcare for monitoring body temperature without physical contact. Their versatility, reliability, and ability to function in low-light conditions make IR sensors essential components in a wide array of technological solutions.

##### 3.2.3 16\*2 LCD MODULE

The 16x2 LCD module is a widely used display component in electronics, particularly in microcontroller projects. This module features a character display that can show 16 characters per line and has two lines, making it ideal for displaying text information such as sensor readings, status messages, or user prompts. The display uses the HD44780 controller, which provides a simple interface for controlling the screen and is compatible with various microcontrollers, including Arduino, Raspberry Pi, and others.



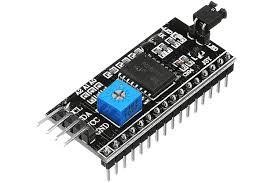
###### **Fig 3.3** 16x2 LCD

The 16x2 LCD typically operates in either 4-bit or 8-bit mode, allowing users to choose the most suitable communication method for their project. In 4-bit mode, only four data pins are used for communication, while in 8-bit mode, all eight data pins are utilized, which can simplify programming but requires more connections. The module consists of a liquid crystal display (LCD) panel, a backlight (usually LED), and a set of pins for interfacing with a microcontroller.

To connect the 16x2 LCD to a microcontroller, you generally need to connect the following pins: VSS (ground), VDD (power supply), VO (contrast adjustment), RS (register select), RW (read/write), E (enable), and the data pins (D0-D7). The contrast of the display can be adjusted using a potentiometer connected to the VO pin.

Applications of the 16x2 LCD module are vast and include displaying data in embedded systems, user interfaces for various devices, and educational projects for learning about interfacing and programming. Its simplicity, low cost, and ease of use make it a favorite among hobbyists and professionals alike for creating interactive and informative displays in electronic projects.

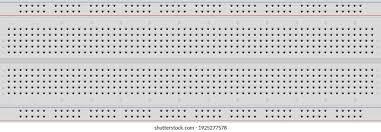
##### 3.2.4 I2C INTERFACE



###### **Fig 3.1** I2C Interface

I2C (Inter-Integrated Circuit), also known as TWI (Two Wire Interface), is a popular communication protocol used for connecting low-speed peripherals to microcontrollers and processors. Developed by Philips (now NXP Semiconductors), I2C allows multiple slave devices to communicate with a single master device using only two wires: a serial data line (SDA) and a serial clock line (SCL). This simplicity in wiring makes I2C an attractive choice for embedded systems and applications where space and complexity need to be minimized. Each device on the I2C bus has a unique address, allowing the master to communicate with multiple slaves without conflicts. The protocol supports different data rates, typically up to 100 kbit/s in standard mode and up to 400 kbit/s in fast mode, with even higher speeds available in fast-mode plus and high-speed mode. I2C also incorporates features like acknowledgment signals and the ability to transmit data in both directions, making it versatile for various applications, including sensors, displays, and memory devices. Its ease of use, combined with the ability to daisy-chain multiple devices on the same bus, has made I2C a widely adopted protocol in consumer electronics, automotive systems, and industrial automation.

##### 3.2.5 BREADBOARD



###### **Fig 3.5** Breadboard

A breadboard is an essential tool for prototyping electronic circuits without the need for soldering. It provides a convenient and reusable platform for building and testing circuits, making it particularly popular among hobbyists, students, and engineers. Breadboards come in various sizes, but they typically consist of a rectangular plastic board with a grid of holes arranged in rows and columns. These holes are used to insert electronic components such as resistors, capacitors, integrated circuits (ICs), and wires.

The internal structure of a breadboard is designed to facilitate easy connections between components. It usually contains metal clips or strips that run horizontally and vertically beneath the surface, allowing for electrical connections to be made. The central area of the breadboard is often used for placing components, while the outer rows are typically reserved for power distribution. Most breadboards have a pair of long horizontal power rails running along the top and bottom, marked with red (for positive voltage) and blue or black (for ground), enabling easy access to power and ground connections.

One of the key advantages of using a breadboard is its flexibility. Users can easily insert and remove components, making it simple to modify the circuit as needed without the permanence of soldering. This feature is particularly useful during the design and testing phases, where adjustments and troubleshooting are common. Additionally, breadboards support a variety of connection types, including jumper wires, which can be used to connect different components and create complex circuits.

#### 3.3 SOFTWARE IMPLEMENTAION

**SOURCE CODE:**

// Load Wi-Fi library

#include <ESP8266WiFi.h>

#include <Wire.h>

#include <LiquidCrystal\_I2C.h> // Replace with your network credentials const char\* ssid = "Induv.\_\_."; const char\* password = "Indu@2005";

// Set web server port number to 80

WiFiServer server(80);

// Variable to store the HTTP request

String header;

LiquidCrystal\_I2C lcd(0x27, 16, 2); const int in= 14 ; const int out=12; int s1 =0; int s2 =0; int count=0, pos=0;

// Current time unsigned long currentTime = millis();

// Previous time unsigned long previousTime = 0;

// Define timeout time in milliseconds (example: 2000ms = 2s) const long timeoutTime = 2000;

void setup() { Serial.begin(115200);

lcd.init(); // initialize the lcd lcd.backlight(); pinMode(in,INPUT); pinMode(out,INPUT);

// Connect to Wi-Fi network with SSID and password

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password); while (WiFi.status() != WL\_CONNECTED) { delay(500); Serial.print(".");

}

// Print local IP address and start web server

Serial.println("");

Serial.println("WiFi connected.");

Serial.println("IP address: "); Serial.println(WiFi.localIP()); server.begin();

}

void loop(){

// put your main code here, to run repeatedly:

lcd.setCursor(0,0); lcd.pr

int("COUNT: ");

if((digitalRead(in))==0){ delay(20);

/\*

Arrangement or placing of sensors: while entering the room from outside, sensor1 will be encounterd first and sensor2 will be next.

pos will tell the position of a person, entering/leaving the room

If pos=0, default value; No person is entering/leaving the room/hall

If pos=1, person is entering the room and crossed sensor1 (in)

If pos=2, person has entered the room after crossing both the sensors

If pos=3, person is going out of the room and crossed the sensor2 (out)

If pos=4, person has gone out of the room after crossing both the sensors

\*/ if(pos==0) pos=1;

else if(pos==3)

pos=4;

delay(2000); if((digitalRead(in))==0){

pos =0;

}

}

if(pos==4 && count!=0){

count--;

//person has left the room, decrement the count

lcd.setCursor(2,1); lcd.print(" "); lcd.setCursor(2, 1);

lcd.print(count); pos=0;

}

if((digitalRead(out))==0){

delay(20) ;

if(pos==1)

pos=2;

else if(pos==0)

pos=3;

delay(2000); if((digitalRead(out))==0){

pos =0;

}

}

if(pos==2){ count++; //person has entered the room, increment the count

lcd.setCursor(2,1);

lcd.print(" "); lcd.setCursor(2, 1);

lcd.print(count); pos=0; }

else if(pos==4 && count!=0){ count--; //person has left the room, decrement the count

lcd.setCursor(2,1); lcd.print(" "); lcd.setCursor(2, 1);

lcd.print(count); pos=0;

}

delay(50);

WiFiClient client = server.available(); // Listen for incoming clients

if (client) { // If a new client connects,

Serial.println("New Client."); // print a message out in the serial port

String currentLine = ""; // make a String to hold incoming data from the client currentTime = millis(); previousTime = currentTime; while (client.connected() && currentTime - previousTime <= timeoutTime) { // loop while the client's connected currentTime = millis(); if (client.available()) { // if there's bytes to read from the client, char c = client.read(); // read a byte, then

Serial.write(c); // print it out the serial monitor header += c; if (c == '\n') { // if the byte is a newline character

// if the current line is blank, you got two newline characters in a row.

// that's the end of the client HTTP request, so send a response: if (currentLine.length() == 0) {

// HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK) // and a content-type so the client knows what's coming, then a blank line: client.println("HTTP/1.1 200 OK"); client.println("Content-type:text/html"); client.println("Connection: close"); client.println();

// Display the HTML web page client.println("<!DOCTYPE html><html>"); client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">"); client.println("<link rel=\"icon\" href=\"data:,\">");

client.println("<body><h1 style =\"color :Tomato;\",\" border:2px solid Violet;\">ESP8266 Web Server</h1>"); client.println("<h2 style =\"color :DodgerBlue;\">Count:</h2>"); client.println(count);

client.println("</body></html>");

// The HTTP response ends with another blank line client.println();

// Break out of the while loop break;

} else { // if you got a newline, then clear currentLine currentLine = "";

}

} else if (c != '\r') { // if you got anything else but a carriage return character, currentLine += c; // add it to the end of the currentLine

}

}

}

// Clear the header variable header = "";

// Close the connection client.stop();

Serial.println("Client disconnected.");

Serial.println("");

}

}

##### HOW THE CODE WORKS

The first thing you need to do is to include the ESP8266WiFi library.

// Load Wi-Fi library

#include <ESP8266WiFi.h>

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

As mentioned previously, you need to insert your ssid and password in the following lines inside the double quotes const char\* ssid = ""; const char\* password = "";

Then, you set your web server to port 80.

// Set web server port number to 80

WiFiServer server(80);

The following line creates a variable to store the header of the HTTP request: String header;

You also need to assign a GPIO to each of your outputs. Here we are using GPIO 14 and GPIO 12. You can use any other suitable GPIOs.

// Assign ir sensors to GPIO pins const int in= 14 ; const int out=12; int s1 =0; int s2 =0; int count=0, pos=0;

Now, let’s go into the setup(). The setup() function only runs once when your ESP first boots.

First, we start a serial communication at a baud rate of 115200 for debugging purposes Serial.begin(115200);

The following lines begin the Wi-Fi connection with WiFi.begin(ssid, password), wait for a successful connection and prints the ESP IP address in the Serial Monitor.

// Connect to Wi-Fi network with SSID and password

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password); while (WiFi.status() != WL\_CONNECTED) { delay(500);

Serial.print(".");

}

// Print local IP address and start web server

Serial.println("");

Serial.println("WiFi connected.");

Serial.println("IP address: "); Serial.println(WiFi.localIP()); server.begin();

let keep the setCursor to 0,0 to display the count

lcd.setCursor(0,0); lcd.print("COUNT: "); if((digitalRead(in))==0){ delay(20);

Arrangement or placing of sensors: while entering the room from outside, sensor1 will be encounterd first and sensor2 will be next.pos will tell the position of a person, entering/leaving the room

If pos=0, default value; No person is entering/leaving the room/hall

If pos=1, person is entering the room and crossed sensor1 (in)

If pos=2, person has entered the room after crossing both the sensors

If pos=3, person is going out of the room and crossed the sensor2 (out)

If pos=4, person has gone out of the room after crossing both the sensors

if(pos==0) pos=1;

else if(pos==3)

pos=4;

delay(2000); if((digitalRead(in))==0){

pos =0;

}

}

if(pos==4 && count!=0){ count--;

When a request is received from a client, we’ll save the incoming data. The while loop that follows will be running as long as the client stays connected. We don’t recommend changing the following part of the code unless you know exactly what you are doing.

if (client) { // If a new client connects,

Serial.println("New Client."); // print a message out in the serial port

String currentLine = ""; // make a String to hold incoming data from the client while (client.connected()) { // loop while the client's connected if (client.available()) { // if there's bytes to read from the client, char c = client.read(); // read a byte, then Serial.write(c); // print it out the serial monitor header += c;

if (c == '\n') { // if the byte is a newline character

// if the current line is blank, you got two newline characters in a row.

// that's the end of the client HTTP request, so send a response: if (currentLine.length() == 0) {

// HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK) // and a content-type so the client knows what's coming, then a blank line:

client.println("HTTP/1.1 200 OK");

client.println("Content-type:text/html"); client.println("Connection: close");

By using the above instruction we can write the code for the remaining cases.

Displaying the HTML Web Page

The next thing you need to do, is generate the web page. The ESP8266 will be sending a response to your browser with some HTML text to display the web page.

The web page is sent to the client using the client.println() function. You should enter what you want to send to the client as an argument.

The first text you should always send is the following line, that indicates that we’re sending

HTML.

<!DOCTYPE html><html>

Then, the following line makes the web page responsive in any web browser.

client.println("<head><meta name=\"viewport\" content=\"width=device-width, initialscale=1\">");

The next one is used to prevent requests related to the favicon – You don’t need to worry about this line.

client.println("<link rel=\"icon\" href=\"data:,\">");

Styling the Web Page

Next, we have some CSS to style the buttons and the web page appearance. We choose the Helvetica font, define the content to be displayed as a block and aligned at the center.

client.println("<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center;}");

We style our buttons with the some properties to define color, size, border, etc… client.println(".button { background-color: #195B6A; border: none; color: white; padding:

16px 40px;"); client.println("text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}");

Then, we define the style for a second button, with all the properties of the button we’ve defined earlier, but with a different color. This will be the style for the off button. client.println(".button2 {background-color: #77878A;}</style></head>");

Displaying the output in web client.println(count);

Finally, when the response ends, we clear the header variable, and stop the connection with the client with client.stop(). // Clear the header variable header = "";

// Close the connection client.stop();

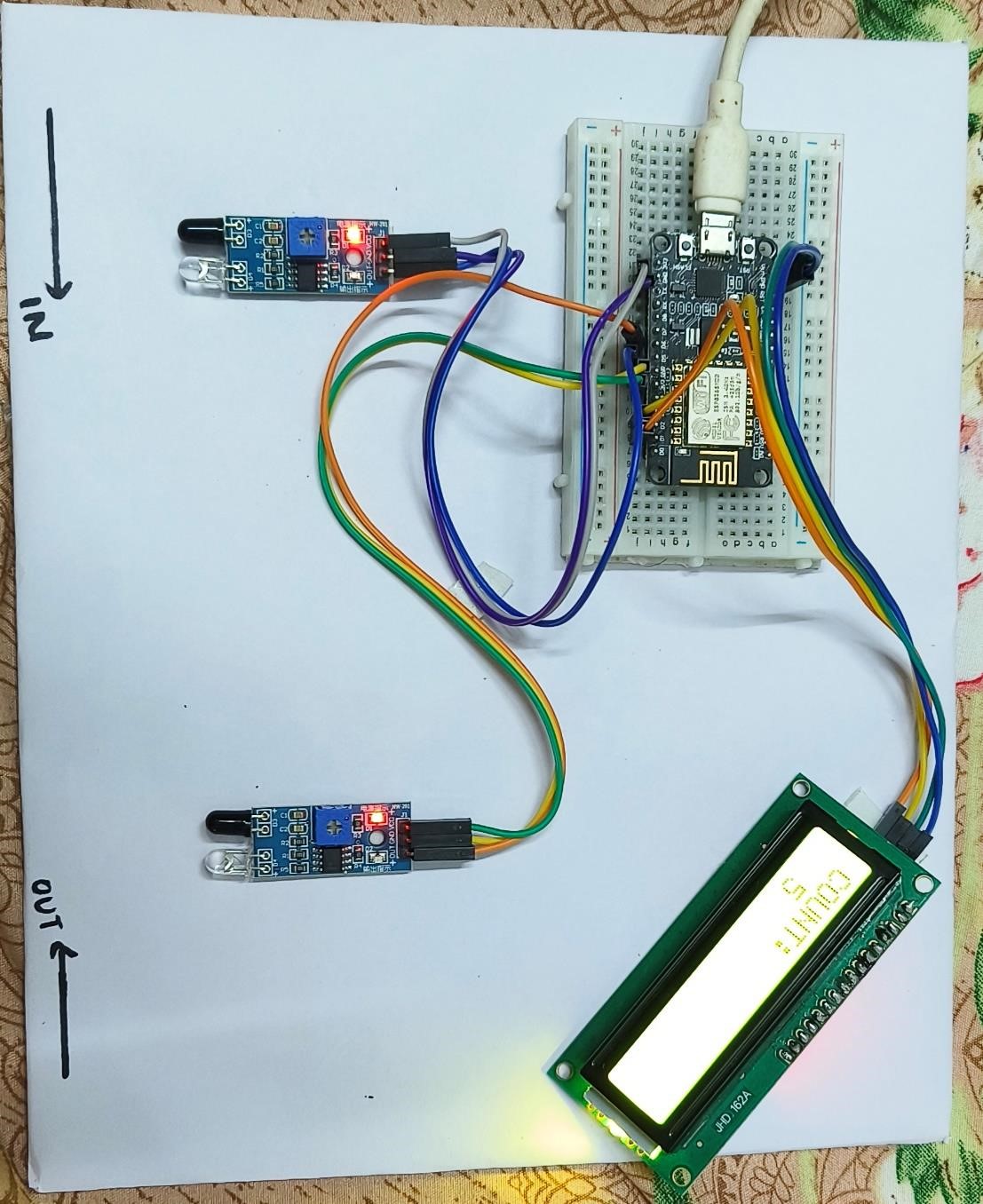
### CHAPTER 4

RESULTS AND DISCUSSION

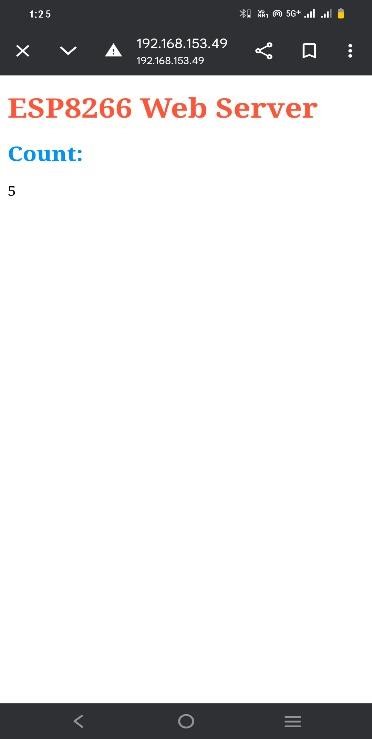
#### 4.1 RESULTS

The Automatic Passenger Count system successfully monitored and recorded the count of objects entering and exiting a defined area. The following key results were observed:

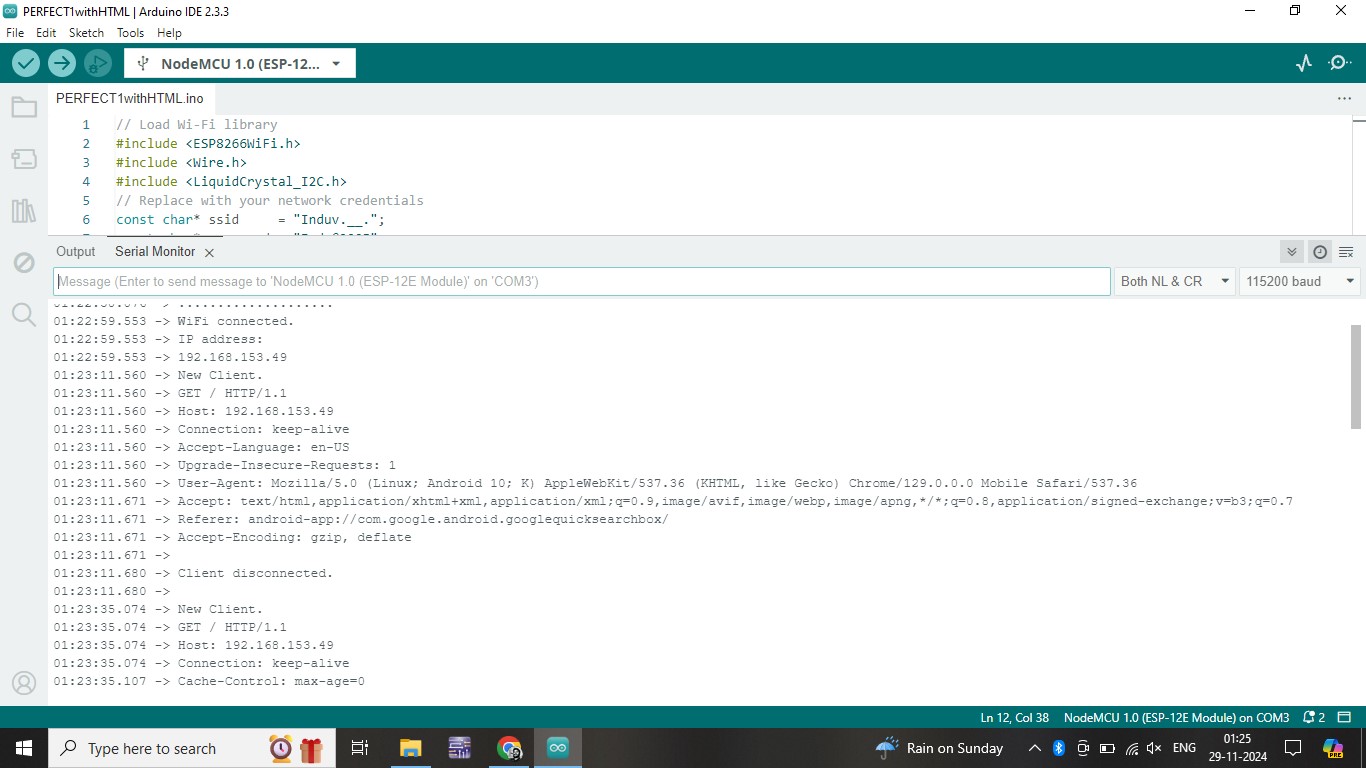
* **Counting Accuracy:** The system achieved an accuracy in detecting and counting objects, depending on optimal placement of the IR sensors.
* **Real-Time Display:** Counts were updated in real time on the LCD display, providing immediate feedback to users.
* **Website Integration:** Data synchronization with the website was seamless, ensuring remote monitoring capabilities.



##### Fig 4.1 LCD Display Output



##### Fig 4.2 Webserver



**Fig 4.3** Serial Monitor Server Connection

#### 4.2 DISCUSSIONS

* The project demonstrated the effective integration of hardware and software for realtime passenger count.
* This project of real time tracking and monitoring is used in various applications such as inventory management, automated counting system.

#### 4.3 ADVANTAGES

* Compact design and low-cost components like NodeMCU and IR sensors.
* High reliability in environments with consistent lighting conditions.

#### 4.3 LIMITATIONS

Network connectivity: When network connectivity is poor or unstable, the system may not be able to transmit data between sensors and monitoring systems in real time.

Detection: APC system may detect luggage also.

**4.4 IMPROVEMENTS**

 The software could be improved to make the system faster and more accurate.

### CHAPTER 5

SUMMARY AND CONCLUSION

#### 5.1 SUMMARY

The passenger counting system was designed to automatically count passengers moving through a predefined area. The project successfully demonstrated the use of NodeMCU, IR sensors, and an LCD display to achieve real-time monitoring. Additionally, the integration of a website allowed for enhanced accessibility and remote supervision.

The system effectively detects objects as they pass through the monitored area, increasing or decreasing the count accordingly. The integration of the NodeMCU allowed for smooth data processing and synchronization between the hardware and web interface, providing live updates for better accessibility.

The use of IR sensors provided an effective way to detect objects entering and exiting a designated space. Despite some challenges with environmental factors, such as varying light conditions, the system performed reliably when tested in stable environments. The LCD display offered a direct, visual representation of the count, while the website interface enhanced user interaction and monitoring.

#### 5.2 CONCLUSION

This project highlights the feasibility of developing low-cost automation systems for counting and monitoring applications. With accurate and reliable hardware integration, the system meets its intended objectives.

The project aimed to develop a low-cost, easy-to-implement solution for real-time object tracking and has the potential to be applied in various fields, including transportation, inventory management, and automated systems.

**5.3 FUTURE SCOPE**

To further enhance the system:

* For further development we can replace IR sensors with **LIDAR** for improvement range and accuracy.
* We can use image processing to distinguish between objects and people.

### CHAPTER 6

REFERENCES

1. **DonskyTech , ESP8266 Object Counting with Mobile Application**

<https://www.youtube.com/watch?v=aA_XuV8N_DA>

1. **Brandon Cannaday, How to Connect an LCD Display to ESP8266 NodeMCU**

<https://www.losant.com/blog/how-to-connect-lcd-esp8266-nodemcu>

1. [**Alsan Parajuli,**](https://iotprojectsideas.com/author/alsan/) **IoT Based Bidirectional Visitor Counter using ESP8266 & Blynk**

<https://iotprojectsideas.com/iot-based-bidirectional-visitor-counter-using-esp8266-blynk/>

#### [4] IoT Design Pro ,IoT Based Smart Currency Counter using NodeMCU and Arduino IDE

[https://iotdesignpro.com/projects/iot-based-smart-currency-counter-using-nodemcu-andarduino-ide](https://iotdesignpro.com/projects/iot-based-smart-currency-counter-using-nodemcu-and-arduino-ide)

©**All copyrights reserved to Department of Electronics and Communication**

**Engineering, Rajiv Gandhi University of Knowledge Technologies – Ongole Campus**

**© O.U.R**