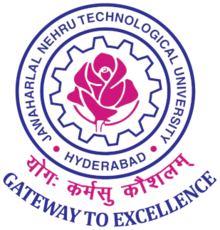
**WIRELESS BIOMETRIC LOCK USING ARDUINO AND MIT APP INVENTOR**

**P. SAHITYA (19JJ1A0435)**

**S. SAI TEJASWI (19JJ1A0447)**

**P. AKASH (19JJ1A0436)**



**Department of Electronics and Communication Engineering**

**Jawaharlal Nehru Technological University Hyderabad**

**University College of Engineering Jagtial**

Nachupally (Kondagattu), Jagtial Dist. - 505501, Telangana

**2023**

**WIRELESS BIOMETRIC LOCK USING ARDUINO AND MIT APP INVENTOR**

MINI PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY

IN

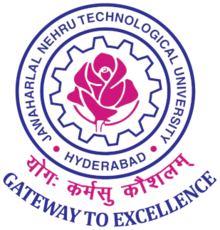
ELECTRONICS AND COMMUNICATION ENGINEERING

BY

P. SAHITYA (19JJ1A0435)

S. SAI TEJASWI (19JJ1A0447)

P. AKASH (19JJ1A0436)



**Department of Electronics and Communication Engineering**

**Jawaharlal Nehru Technological University Hyderabad**

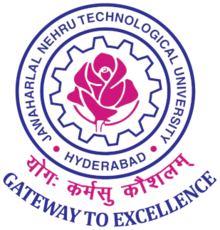
**University College of Engineering Jagtial**

Nachupally (Kondagattu), Jagtial Dist. - 505501, Telangana

**2023**

**Jawaharlal Nehru Technological University Hyderabad University College of Engineering**

Nachupally (Kondagattu), Jagtial Dist. - 505 501, Telangana



**Department of Electronics and Communication Engineering**

**CERTIFICATE**

Date: /10/2023

*This is to certify that the project work entitled* ***WIRELESS BIOMETRIC LOCK USING ARDUINO AND MIT APP INVENTOR*** *is a Bonafide work carried out by* ***P. SAHITYA, S. SAI TEJASWI*** *and* ***P. AKASH*** *bearing Roll Nos.* ***19JJ1A0435****,* ***19JJ1A0447*** *and* ***19JJ1A0436*** *in partial fulfillment of the requirements for the degree of* ***BACHELOR OF TECHNOLOGY*** *in* ***ELECTRONICS & COMMUNICATION ENGINEERING*** *by the Jawaharlal Nehru Technological University Hyderabad during the academic year 2022-23.*

*The results embodied in this report have not been submitted to any other University or Institution for the award of any degree or diploma.*

*-------------------------- ---------------------------*

|  |  |
| --- | --- |
| **Dr. Dhiraj Sunehra** | **Dr. Dhiraj Sunehra** |
| **Professor** | **Professor** |
| **Project Guide** | **Head of the Department** |

**Acknowledgements**

The success in this project would not have been possible without timely help and guidance by many people. We wish to express our sincere gratitude to all those who have helped and guided us for the completion of the project.

It is our pleasure to thank to our Project Guide and Project Coordinator, **Dr. Dhiraj Sunehra, Professor & Head, Department of ECE,** for the guidance and suggestions throughout the project, without which we would not have been able to complete this project successfully.

We express our sincere gratitude to **Prof. D. Ramesh, Principal and Prof. T. Venugopal, Vice-Principal, JNTUH UCEJ,** for their encouragement and providing facilities to accomplish our project successfully.

We would like to thank our classmates, all faculty members and non-teaching staff of ECE Department, JNTUH UCEJ for their direct and indirect help during the project work.

Finally, we wish to thank our family members and our friends for their interest and assistance that has enabled to complete the project work successfully.

**ABSTRACT**

Security and privacy are the important aspects to be considered when talking about the things that are vulnerable. Thus, it is obvious that the place where we live has to be secured as it contains valuable assets and important documents in order to protect them from theft. IoT has aided humans in many tasks and wireless biometric lock using IoT is one such way to help people overcome the above stated problem. It allows a homeowner to enter their home or provide access to others without requiring a traditional key. Instead, the user uses their smartphone to verify fingerprint and unlock the door.

This project aims at implementing a wireless biometric lock equipped with modules like NodeMCU, Bluetooth module and IoT implementation is done through the Blynk IoT application. HC-05 Bluetooth Module is a low cost, easy to-work and little assessed module used for distant correspondence in the Bluetooth range and is interfaced to the NodeMCU. MIT app inventor makes it possible to create applications for IoT and use the same for interacting with objects all around us.

The result of this project successfully helps people to lead a safe and secure life and the responsibility of people to take extra care for securing homes is eliminated. This model overcomes the limitations of previous models and is both effective and efficient. There would be a large-scale implementation of this model in future due to its advantages.

**CONTENTS**

**Certificate i**

**Acknowledgements ii Abstract iii Contents iv List of Figures vii**

|  |  |  |
| --- | --- | --- |
| **Chapter 1** | **Introduction** | 1 |
| 1.1 Introduction | 1 |
| 1.2 Aim of the Project | 1 |
| 1.3 Methodology | 1 |
| 1.4 Significance of the work | 2 |
| 1.5 Organization of the thesis | 2 |
| **Chapter 2** | **Overview of Arduino UNO** | 3 |
| 2.1 Introduction | 3 |
| 2.2 Types of Arduino | 3 |
| 2.3 Arduino UNO | 5 |
| 2.3.1 Features of Arduino UNO | 5 |
| 2.3.2 Description of Arduino UNO | 6 |
| 2.3.3 Pin Description of Arduino UNO | 7 |
| 2.4 Conclusions | 8 |
| **Chapter 3** | **Hardware Description** | 9 |
| 3.1 Introduction | 9 |
| 3.2 Block Diagram | 9 |
| 3.3 NodeMCU | 9 |
| 3.3.1 Features | 10 |
| 3.3.2 Description | 10 |
| 3.3.3 Pin Configuration | 11 |
| 3.4 HC-05 Bluetooth Module | 13 |
| 3.4.1 Features | 13 |
| 3.4.2 Description | 14 |
| 3.4.3 Pin Configuration | 14 |
| 3.4.4 Interfacing HC-05 with NodeMCU | 15 |
| 3.5 Power Supply | 16 |
| 3.5.1 Step down Transformer | 16 |
| 3.5.2 Bridge Rectifier | 17 |
| 3.5.3 Voltage Regulators LM7805 and LM7812 | 18 |
| 3.6 Relay | 19 |
| 3.6.1 Features | 19 |
| 3.6.2 Description | 19 |
| 3.6.3 Pin Configuration | 19 |
| 3.7 ULN2003 | 20 |
| 3.7.1 Features | 20 |
| 3.7.2 Description | 21 |
| 3.7.3 Pin Configuration | 21 |
| 3.7.4 Logic Diagram | 22 |
| 3.8 Solenoid Lock | 22 |
| 3.8.1 Features | 22 |
| 3.8.2 Description | 23 |
| 3.9 LED | 23 |
| 3.10 Conclusions | 24 |
| **Chapter 4** | **Software Tools** | 25 |
| 4.1 Introduction | 25 |
| 4.2 Arduino IDE | 25 |
| 4.3 Embedded C | 26 |
| 4.4 MIT App Inventor | 27 |
| 4.5 Blynk IoT App | 30 |
| 4.6 Proteus Software | 31 |
| 4.7 Conclusions | 34 |
| **Chapter 5** | **Results and Discussion** | 35 |
| 5.1 Introduction | 35 |
| 5.2 Schematic Diagram | 35 |
| 5.3 Flowchart | 36 |
| 5.4 Experimental Setup | 37 |
| 5.5 Results | 38 |
| 5.6 Conclusions | 39 |
| **Chapter 6** | **Conclusions** | 40 |
|  | **References** | 41 |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Description** | **Page No.** |
| 2.1 | Arduino Uno Board Description | 5 |
| 2.2 | Arduino UNO Top View | 6 |
| 3.1 | Block diagram of the Project | 9 |
| 3.2 | NodeMCU | 9 |
| 3.3 | Pin Configuration of NodeMCU | 11 |
| 3.4 | HC-05 Bluetooth Module | 13 |
| 3.5 | Pin Configuration of HC-05 Bluetooth Module | 14 |
| 3.6 | Interfacing HC-05 Bluetooth module with NodeMCU | 15 |
| 3.7 | Step Down Transformer | 17 |
| 3.8 | Bridge Rectifier | 17 |
| 3.9 | LM7805 & LM7812 Voltage Regulators | 18 |
| 3.10 | Relay | 19 |
| 3.11 | Pin Configuration of Relay | 19 |
| 3.12 | ULN2003 Relay Driver IC | 20 |
| 3.13 | Pin Configuration of ULN 2003A Relay Driver IC | 21 |
| 3.14 | Logic Diagram of ULN2003 Relay Driver | 22 |
| 3.15 | Solenoid Lock | 22 |
| 3.16 | LED | 23 |
| 4.1 | Arduino IDE | 25 |
| 4.2 | Arduino IDE tool bar | 26 |
| 4.3 | MIT App Inventor logo | 27 |
| 4.4 | MIT App Inventor Design Editor | 28 |
| 4.5 | MIT App Inventor Block Editor | 29 |
| 4.6 | Blynk IoT App | 30 |
| 4.7 | Proteus Software Window | 32 |
| 4.8 | Various options in Proteus Software | 33 |
| 5.1 | Schematic diagram of the complete project | 35 |
| 5.2 | NodeMCU flowchart | 36 |
| 5.3 | MIT App flowchart | 37 |
| 5.4 | Experimental Setup | 38 |
| 5.5 | Lock closed | 38 |
| 5.6 | Lock opened | 39 |
| 5.7 | Notifications received by the user | 39 |

**Chapter 1**

**Introduction**

**1.1 Introduction**

The primary protection that we provide to our houses is, locking it with the help of a traditional lock. Whenever we leave our home for work or when we travel somewhere, we lock our house. Not everyone can appoint a security guard to look after their home. Traditional locks have some disadvantages which led to idea of developing a wireless biometric lock providing extra security and notifying every time the lock is opened. This lock makes use of electronic components as well as the internet for its smooth functioning. It allows a homeowner to monitor the people entering their house and to keep it secure.

**1.2 Aim of the Project**

The aim of this project is to implement a Wireless Biometric Lock using Arduino board and IoT which makes it possible to unlock the door without using a traditional key and notifies the owner every time the lock is opened.

The objectives of the project are as follows:

1. To understand the features and architecture of NodeMCU.
2. To interface the various hardware modules such as HC-05 Bluetooth Module to NodeMCU.
3. To develop the necessary code using Embedded C language in Arduino IDE software.
4. To develop an application using MIT App Inventor through which lock is opened.
5. To update the data to Blynk IoT App.

**1.3 Methodology**

The project is aimed at the implementation of a Wireless Biometric Lock using Arduino board. This lock is helpful in protecting houses and maintaining security. A person can monitor who is entering their house from any part of the world. The architecture of the ESP8266 NodeMCU is understood including the ports and pins. The written code is uploaded to the NodeMCU board. The HC-05 Bluetooth Module is interfaced with the NodeMCU. The system is tested for satisfactory results.

**1.4 Significance of the work**

In this project we attempt to develop a device which can be used to monitor the people entering a house and to allow access by using fingerprints. It has the following advantages:

* Efficient
* Economical
* Easy to operate
* Less power consumption
* Use of traditional lock is eliminated. A smartphone is used to wirelessly verify the person trying to enter the house and mechanically unlock the door.
* Notifications are sent through Blynk application and email as well.

**1.5 Organization of the thesis**

This thesis is divided into six chapters including introduction and conclusions. The block diagram, features, pin diagram and other functional units of Arduino UNO are explained in Chapter 2. The description of various hardware components and the software used in the project is explained in chapter 3 and chapter 4. The schematic diagram, flowchart, experimental setup and results are discussed in chapter 5.

**Chapter 2**

**Overview of Arduino UNO**

**2.1**  **Introduction**

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs like light on a sensor, a finger on a button or a Twitter message and turn it into an output activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so, you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE) based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for loT applications, wearable, 30 printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

**2.2 Types of Arduinos**

Different Arduino boards available in the market, with wide specifications. Each board uses different kinds of microcontroller with different operating voltages, CPU speed, RAM size and different number of analog and digital inputs. Some of the Arduino boards are given below:

**2.2.1 Arduino Uno**

Arduino Uno is the most preferred board among all the Arduino boards. The processor used is ATmega328P. Its operating and input voltage are 5V and 7 to 12 V respectively, 16MHz clock frequency, 6 analog and 14 digital IO, 1 kB EEPROM, 32 kB Flash, a regular USB and 1 UART.

## **2.2.2 Arduino Due**

First ARM based microcontroller Arduino board is the Arduino Due. It is a 32-bit microcontroller-based Cortex-M3 Central Processing Unit. It is a very powerful microcontroller with 54 digital I/O, 12 analog inputs, 2 analog outputs and 4 UART serial interfaces. If we give more than 5V power supply to the Arduino, it will damage. It is 32-bit ARM core, it has 512KB of flash memory, 96KB SRAM in which two banks are there and they are 64KB and 32KB.

**2.2.3 Arduino Mega 2560 Model**

The Arduino Mega 2560 model is an advanced version of the previous existing model Arduino Due. It has 256 KB of RAM memory and 54 input output pins. In that 16 pins are used as analog pins and 14 pins are used for Pulse Wave Modulation (PWM).

**2.2.4 Arduino Uno (R3)**

The Uno is a huge option for your initial Arduino. This Arduino board depends on an ATmega328P based microcontroller. It consists of 14-digital I/O pins, where 6-pins can be used as PWM output pins, 6-analog inputs, a reset button, a power jack, a USB connection, an In-Circuit Serial Programming header (ICSP) etc.

**2.2.5 Arduino Leonardo Board**

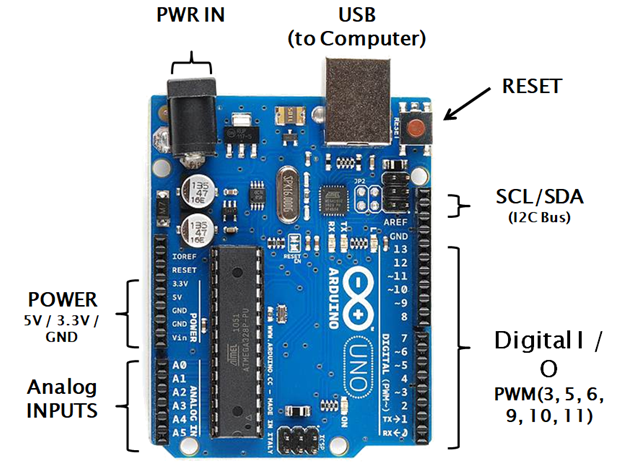
The first development board of an Arduino is the Leonardo board. This board uses one microcontroller along with the USB. That means, it can be very simple and cheap also. Because this board handles USB directly, program libraries are obtainable which let the Arduino board follow a keyboard of the computer, mouse, etc.

**2.3 Arduino UNO**

Arduino Uno, which is a small board but useful for many applications.

* Arduino Uno is a microcontroller board based on the ATmega328P.
* It consists of

1. 14 digital input/output pins (of which 6 can be used as PWM outputs)
2. 6 analog inputs
3. 16 MHz quartz crystal
4. USB connection
5. Power jack
6. ICSP header
7. Reset button



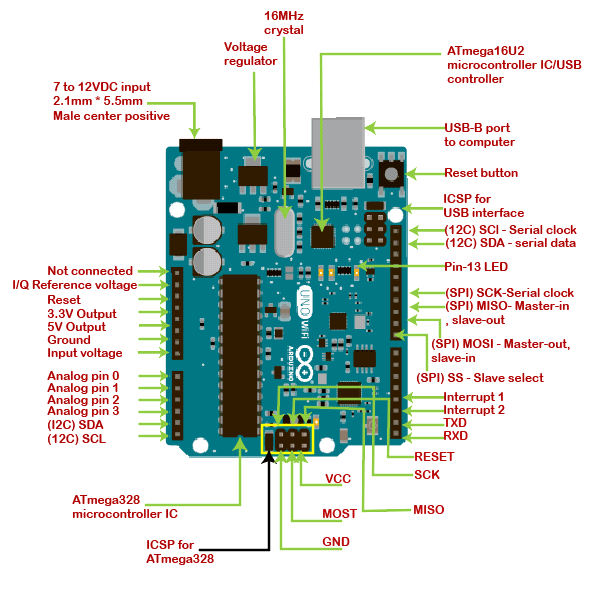
**Figure 2.1: Arduino Uno Board Description**

**2.3.1 Features of Arduino Uno**

Salient features of Arduino Uno board.

|  |  |
| --- | --- |
| * Microcontroller (MCU) | : ATmega328 |
| * Operating Voltage (DC) | : DC 5V |
| * Input Supply Voltage (External) | : 6 – 20 V DC |
| * Recommended Input Supply Voltage (DC) | : 7 – 12 V |
| * Number of Digital Input / Output (I/O) Pins | : 14 |
| * PWM (Pulse Width Modulation) outputs | : 6 (Pin 3, 5, 6, 9,10 & 11) |
| * Input Pins (Analog) | : 6 (A0-A5) |
| * DC Current (Max) | : 40 mA (per I/O Pin) |
| * DC Current for 3V Pin | : 50 mA |
| * Flash Memory | : 32 KB (ATmega328) |
| * Clock Speed (Ceramic Resonator) | : 16 MHz |
| * SRAM (Memory) | : 2 KB |
| * EEPROM (Memory) | : 1 KB |

**2.3.2 Description of Arduino UNO**



**Fig 2.2: Arduino UNO Top View**

**2.3.3 Pin Description of Arduino UNO**

The connections to the Arduino can be made through the pins. The black plastic ‘headers’ are used to plug and connect a wire into the board. The functions vary from pin to pin.

1. **GND:** GND means ‘Ground’. There are sufficient GND pins to connect more modules to the Arduino board.
2. **5V & 3.3V:** The power supply to the modules is the 5V and 3.3V pins. 5V pin supplies power at 5 volts and the 3.3V pin supplies power at 3.3 volts. In general, most of the components work at the specified voltages given above.
3. **Analog Pins:** A0 to A5 pins are analog. The input to these pins is analog in nature. These pins receive the data from an analog sensor. The smoke sensor and temperature sensor give analog values. There is an ADC to convert it into a digital value that will be processed by the controller.
4. **Digital Pins:** 0 to 13 are digital. These pins take the digital input from the sensor and give digital output to the load.
5. **PWM**: The tilde (~) is shown adjacent to some of the digital pins on the Arduino board which can be used as Pulse-Width Modulation pins & also as digital pins.
6. **AREF (Analog Reference):** It acts as an external reference voltage.

### **Reset Button:** When Reset is pressed, that pin is grounded and restarts the code loaded to Arduino.

### **Power LED Indicator:** When a power source is connected, the LED turns ON. If the LED doesn’t turn on, the connections need to be checked. If the connections are good then the board might have been damaged.

### **Tx Rx LEDs:** The Tx (transmitter) and Rx (receiver) LEDs will glow whenever there is data transfer between Arduino and PC. It also glows for other serial communication data transfer.

1. **Voltage Regulator:** The voltage regulator controls the fluctuations in the voltage that is supplied to the Arduino board.

### **Main IC:** The main IC on the Arduino is from the ATMEL Company, usually AT mega series.

**2.4 Conclusions**

In this chapter the history of Arduino and salient features of different types of Arduino boards are explained. The architecture of Arduino Uno board is discussed along with its pin configuration.

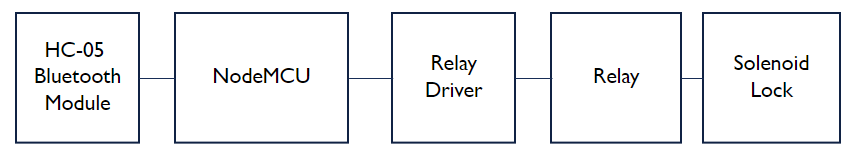
**Chapter 3**

**Hardware Description**

**3.1 Introduction**

NodeMCU is the heart of this project which can be considered as an Arduino with an in-built Wi-Fi module. HC-05 Bluetooth Module is interfaced with the NodeMCU. The Bluetooth module receives data from an application designed using MIT App Inventor which is used to open the solenoid lock using relay through the relay driver IC. Whenever the lock is opened, the data is uploaded to the Blynk App and notifications are sent to the owner through the same app and also through mail.

**3.2 Block Diagram**

**Fig 3.1: Block Diagram of the Project**

**3.3 NodeMCU**



**Fig 3.2: NodeMCU**

**3.3.1 Features**

* Open Source & Arduino like hardware.
* Micro USB port & Reset/Flash buttons.
* Interactive, Programmable & Low cost.
* ESP8266 with inbuilt Wi-Fi.
* USB to UART converter and Status LED.
* Voltage: 3.3V
* Current consumption: 10uA~170mA
* Flash memory attachable: 16MB max (512K normal)
* Integrated TCP/IP protocol stack.
* Processor: Tensilica L106 32-bit.
* Processor speed: 80~160MHz
* RAM: 32K+ 80K
* Maximum concurrent TCP connections: 5
* GPIOS: 17 (multiplexed with other functions). Analog to Digital: 1 input with 1024 step resolution.

**3.3.2 Description**

NodeMCU is an open-source firmware and development kit that plays a vital role in designing an IoT product using a few script lines. Multiple GPIO pins on the board allow us to connect the board with other peripherals and are capable of generating PWM, 12C, SPI, and UART serial communications.

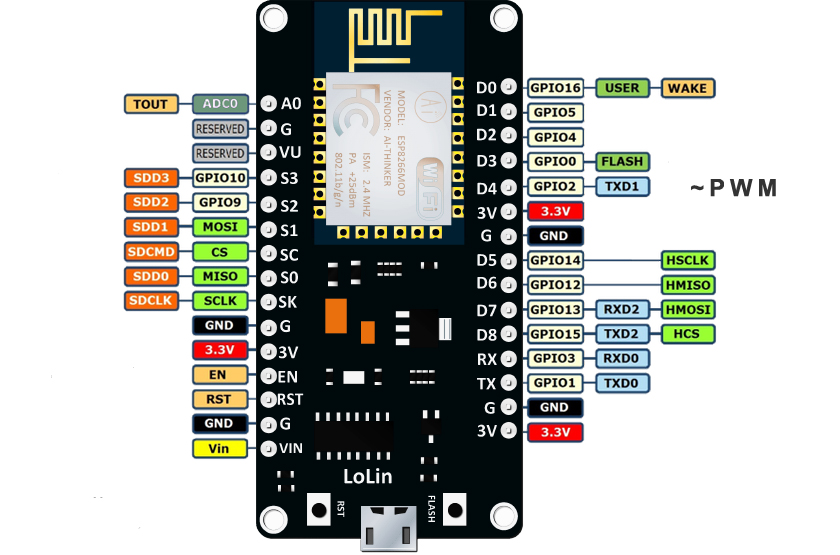
The interface of the module is mainly divided into two parts:

* Firmware – Runs on the ESP8266 Wi-Fi SoC.
* Hardware – Based on the ESP-12 Module

Open-source firmware allows us to edit, modify and rebuild the existing module and keep changing the entire interface until you succeed in optimizing the module as per your requirements.

* USB to UART converter is added on the module that helps in converting USB data to UART data which mainly understands the language of serial communication.
* Instead of the regular USB port, Micro USB port is included in the module that connects it with the computer for dual purposes: programming and powering up the board.
* The board incorporates status LED that blinks and turns off immediately, giving you the current status of the module if it is running properly when connected with the computer.
* The ability of module to establish a flawless Wi-Fi connection between two channels makes it an ideal choice for incorporating it with other embedded devices like Raspberry Pi.

**3.3.3 Pin Configuration**



**Fig 3.3: Pin Configuration of NodeMCU**

The Pin configuration of NodeMCU is as follows:

**1)** **Power Pins:** There are four power pins namely - Vin pin and three 3.3V pins.

* The Vin pin can be used to directly supply ESP8266 and its components if you have a controlled 5V voltage source.
* The 3.3V pins are the output of the voltage board controller. These pins can be used to supply power to external parts.

**2) GND (Ground):** It is the ground pin of the ESP8266 NodeMCU development board.

**3) I2C Pins:** These are used to integrate all types of I2C sensors and parameters in the project. Both I2C Master and I2C Slave are supported. The clock frequency is 100 kHz at maximum speed. It should be noted that the frequency of the I2C clock should be greater than the frequency of the slowest clock of the slave device.

**4) GPIO Pins:** The ESP8266 NodeMCU has 17 GPIO pins that can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light, and Button respectively. Each GPIO digitally-enabled can be adjusted to internal drag or drop or set to high intensity. When set as input, it can also be set to Edge-trigger or level-trigger to produce CPU interference.

**5) ADC channel:** NodeMCU is embedded 10-bit with SAR ADC accuracy. These two functions can be performed using the ADC viz. VDD 3P3 pin power supply and Tout pin power input. However, they cannot be used simultaneously.

**6) UART:** The ESP8266 NodeMCU has 2 UART domains, namely UART0 and UART1, which offer different connections (RS232 and RS485), and can communicate up to 4.5 Mbps. UART0 pins (TXD0, RXD0, RST0 & CTS0) can be used for communication. However, the UART1 (TXD1 pin) includes a data transfer signal only, so it is used to print the log.

**7) SPI Pins:** ESP8266 incorporates two SPIs (SPI and HSPI) into slave and master modes. These SPIs also support the following SPI features:

* 4-time modes for SPI format transfer
* Up to 80 MHz with split clocks of 80 MHz
* Up to 64-Byte FIFO

**8) SDIO Pins:** ESP8266 incorporates the Secure Digital Input / Output Interface (SDIO) which is used to connect directly to SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

**9) PWM Pins:** The board has 4 Pulse Width Modulation (PWM) channels. PWM output can be programmed and used to drive digital engines and LEDs. The frequency range of PWM ranges from 1000 μs to 10000 μs, eg between 100 Hz and 1 kHz.

**10) Control Pins:** These are used to control ESP8266. These anchors include Chip Enable pin (EN), Reset pin (RST), and WAKE pin.

* **EN Pin (Enable):** The ESP8266 chip is enabled when the EN pin is pulled INSIDE. When pulled LOW the chip works at low power.
* **RST Pin (Reset):** The RST pin is used to reset the ESP8266 chip.
* **Wake Pin:** The use of Wake pin is used to wake up the chip from a deep sleep.

**3.4 HC-05 Bluetooth Module**



**Fig 3.4: HC-05 Bluetooth Module**

**3.4.1 Features**

* Easy to use and minimum external components
* Status LED
* PIO control
* Config Button
* Range: 10 meters
* 5V power operation
* Typical -80dBm sensitivity
* Up to +4dBm RF transmit power
* Low Power 1.8V Operation ,1.8 to 3.6V I/O
* UART interface with programmable baud rate

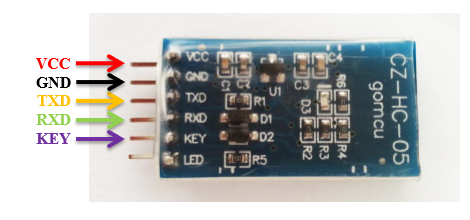
**3.4.2 Description**

HC-05 Bluetooth Module is an easy-to-use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. Its communication is via serial communication which makes an easy way to interface with controller or PC. HC-05 Bluetooth module provides switching mode between master and slave mode which means it able to use neither receiving nor transmitting data.

Major applications are as follows:

1. GPS Receiver
2. Industrial control
3. Mouse, keyboard, joystick
4. Microcontroller unit projects
5. Computer and peripheral devices

**3.4.3 Pin Configuration**



**Fig 3.5: Pin Configuration of HC-05 Bluetooth Module**

HC-05 Bluetooth Module has 6 pins. They are:

**1.**  **Key/EN:** It is used to bring Bluetooth module in AT commands mode. If Key/EN pin is set to high, then this module will work in command mode. Otherwise by default it is in data mode. The default baud rate of HC-05 in command mode is 38400bps and 9600 in data mode.

HC-05 module has two modes which are:

          1.  **Data mode:**Exchange of data between devices.

2.  **Command mode:**It uses AT commands which are used to change setting of HC-05. To send these commands to module serial (USART) port is used.

**2.**  **VCC:**Connect 5 V or 3.3 V to this Pin.

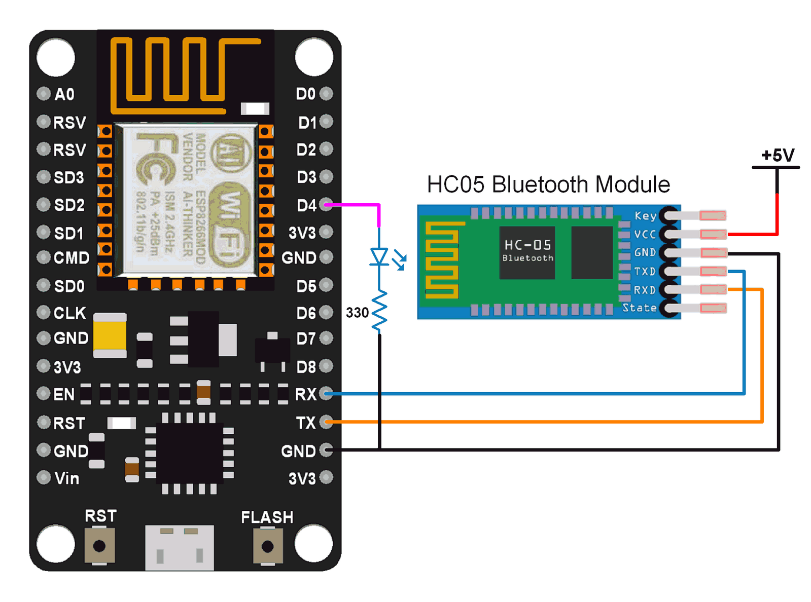
**3.**  **GND:**Ground Pin of module.

**4.**  **TXD:**Transmit Serial data (wirelessly received data by Bluetooth module transmitted out serially on TXD pin)

**5.** **RXD:** Receive data serially (received data will be transmitted wirelessly by Bluetooth module).

**6.**  **State:**It tells whether module is connected or not.

**3.4.4 Interfacing HC-05 with NodeMCU**



**Fig 3.6: Interfacing HC-05 Bluetooth module with NodeMCU**

The connections for interfacing of HC-05 Bluetooth Module with NodeMCU are simple.

1) Connect the Tx pin on the NodeMCU to the RxD pin of the HC-05 Bluetooth Module.

2) Connect the Rx Pin on the NodeMCU to the TxD pin of the HC-05 Bluetooth Module

3) Connect the GND pin on the NodeMCU to the GND pin of the HC-05 Bluetooth Module.

**3.5 Power Supply**

The power requirements of various components involved in this project are met with the help of the following three components.

1. Step Down Transformer
2. Bridge Rectifier
3. Voltage Regulators LM7805 and LM7812

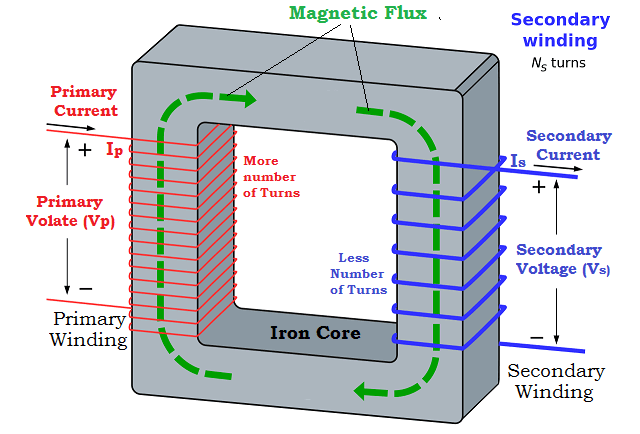
**3.5.1 Step down Transformer**

A transformer is a type of static electrical equipment that transforms electrical energy (from primary side windings) to magnetic energy (in transformer magnetic core) and again to the electrical energy (on the secondary transformer side). A transformer works on the principle of mutual induction.

### Mutual Induction:

It means that a current gets induced in a coil when it comes in proximity of a current-carrying coil having varying magnetic flux. This induced current is directly proportional to the rate of change in current.

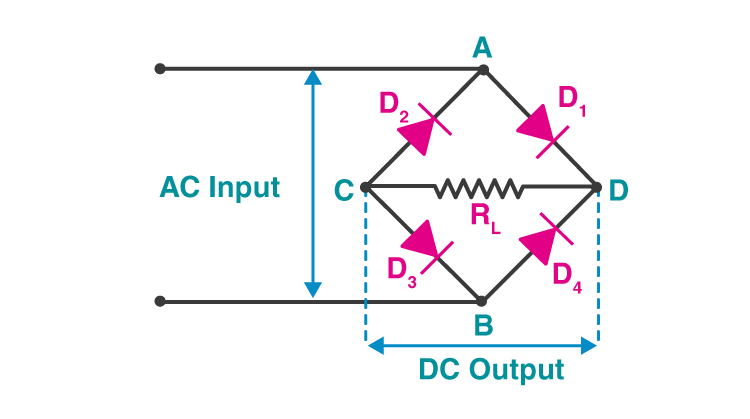
A step-down transformer is a type of transformer that converts the high voltage (HV) and low current from the primary side of the transformer to the low voltage (LV) and high current value on the secondary side of the transformer. A step-down transformer has a larger number of turns in the primary winding and a smaller number for secondary winding is called a step-down transformer. It has a wide variety of applications in electrical systems and transmission lines.



**Fig 3.7: Step Down Transformer**

**3.5.2 Bridge Rectifier**

The bridge rectifier circuit is made of four diodes D1, D2, D3, D4, and a load resistor RL. The four diodes are connected in a closed-loop configuration to efficiently convert the alternating current (AC) into Direct Current (DC). The main advantage of this configuration is the absence of the expensive center-tapped transformer. Therefore, the size and cost are reduced. ridge rectifier is a type of full-wave rectifier that uses four or more diodes in a bridge circuit configuration to efficiently convert alternating (AC) current to a direct (DC) current.



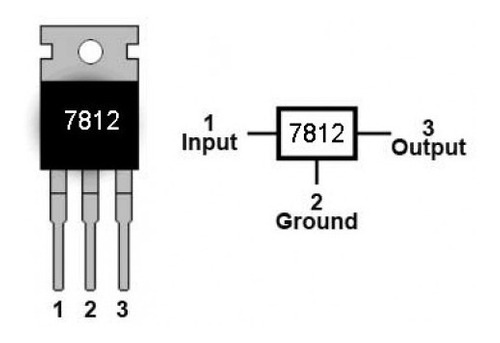
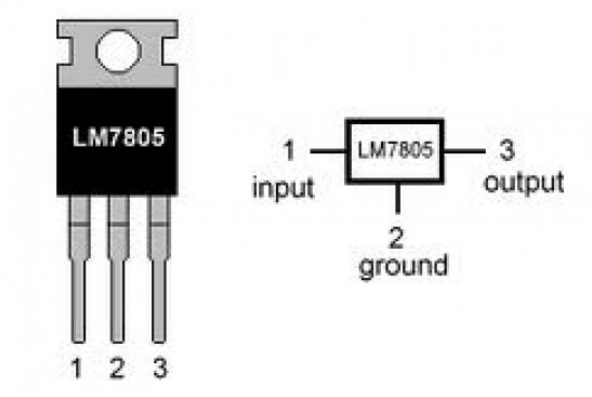
**Fig 3.8: Bridge Rectifier**

**3.5.3 Voltage Regulators LM7805 and LM7812**

The MC78XX/LM78XX/MC78XXA are a series of three terminal positive voltage regulators. This series of fixed-voltage integrated-circuit voltage regulators are designed for a wide range of applications such as on-card regulation for elimination of noise and distribution problems associated with single-point regulation. Each of these regulators can deliver up to 1.5 A of output current. The internal current-limiting and thermal-shutdown features of these regulators essentially make them immune to overload. In addition to use as fixed-voltage regulators, these devices can be used with external components to obtain adjustable output voltages and currents, and also can be used as the power-pass element in precision regulators.

Features of these voltage regulators:

* Output Current up to 1A
* Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
* Thermal Overload Protection
* Short Circuit Protection
* Output Transistor Safe Operating Area Protection

**Fig 3.9: LM7812 & LM7805 Voltage Regulators**

**3.6 Relay**



**Fig 3.10: Relay**

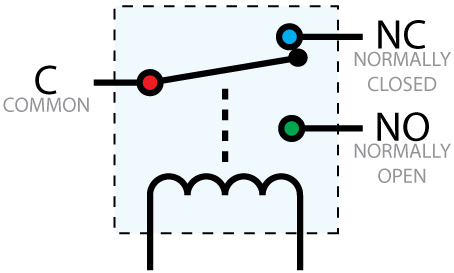
**3.6.1 Features**

* 12V DC SPDT Relay
* Rated up to 7A @240VAC
* Fully Sealed

**3.6.2 Description**

The Single Pole Double Throw SPDT relay is quite useful in certain applications because of its internal configuration. It has one common terminal and 2 contacts in 2 different configurations: one can be Normally Closed and the other one is opened or it can beNormally Openand the other one closed. So basically, you can see the SPDT relay as a way of switching between 2 circuits: when there is no voltage applied to the coil one circuit “receives” current, the other one doesn’t and when the coil gets energized the opposite is happening.

**3.6.3 Pin Configuration**



**Fig 3.11: Pin Configuration of Relay**

**1.** **Coil End 1**- Used to trigger (On/Off) the Relay, normally one end is connected to 12V and the other end to ground.

**2. Coil End 2** - Used to trigger (On/Off) the Relay, normally one end is connected to 12V and the other end to ground.

**3.** **Common (COM)** - Common is connected to one End of the Load that is to be controlled.

**4.** **Normally Close (NC)** - The other end of the load is either connected to NO or NC. If connected to NC the load remains connected before trigger.

**5. Normally Open (NO)** - The other end of the load is either connected to NO or NC. If connected to NO the load remains disconnected before trigger.

**3.7 ULN2003**



**Fig 3.12: ULN2003 Relay Driver IC**

**3.7.1 Features**

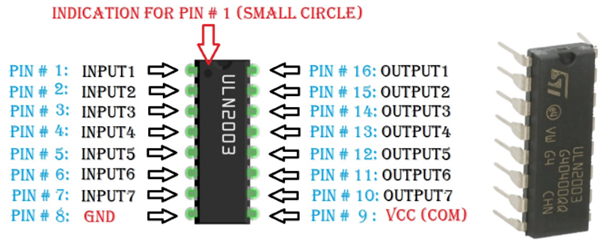
* Seven Darlington Pairs per package
* Output current 500mA per driver (600mApeak)
* Output voltage 50 V
* Integrated suppression diodes for inductive loads
* Outputs can be paralleled for higher current
* TTL/CMOS/PMOS/DTL Compatible inputs
* Inputs pinned opposite outputs to simplify layout
* Relay driver application

**3.7.2 Description**

The ULN2003 is a monolithic high voltage and high current Darlington transistor arrays. It consists of seven NPN Darlington pairs that features high-voltage outputs with common-cathode clamp diode for switching inductive loads. The Darlington pairs may be paralleled for higher current capability. Each channel rated at 500mA and can withstand peak currents of 600mA. Suppression diodes are included for inductive load driving and the inputs are pinned opposite the outputs to simplify board layout.

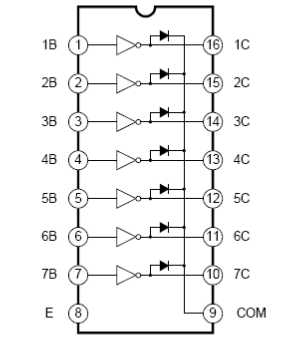
Applications include relay drivers, hammer drivers, lamp drivers, display drivers (LED gas discharge), line drivers, and logic buffers. The ULN2003 has a 2.7kW series base resistor for each Darlington pair for operation directly with TTL or 5V CMOS devices.

**3.7.3 Pin Configuration**

**Fig 3.13: Pin Configuration of ULN 2003A Relay Driver IC**

1. **Pins 1 to 7 (Input 1 to Input 7)**: Seven Input pins of Darlington pair, each pin is connected to the base of the transistor and can be triggered by using +5V
2. **Pin 8 (Ground):** Ground Reference Voltage 0V
3. **Pin 9 (COM):** Used as test pin or Voltage suppresser pin (optional to use)
4. **Pins 10 to 16 (Output 1 to Output 7):** Respective outputs of seven input pins. Each output pin will be connected to ground only when its respective input pin is high(+5V)

**3.7.4 Logic Diagram**



**Fig 3.14: Logic Diagram of ULN2003 Relay Driver**

**3.8 Solenoid Lock**



**Fig 3.15: Solenoid Lock**

**3.8.1 Features**

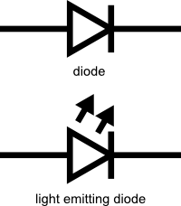
* Operating voltage: 12V DC (you can use 9-12 DC volts, but lower voltage results in weaker/slower operation)
* Designed for 1-10 seconds long activation time
* Wire length: 222.25mm / 8.75"
* Fully enclosed heavy-duty stainless-steel construction
* Draws 650mA at 12V, 500 mA at 9V when activated
* Max Dimensions: 41.85mm / 1.64" x 53.57mm / 2.1" x 27.59mm / 1.08"
* Dimensions: 23.57mm / 0.92" x 67.47mm / 2.65" x 27.59mm / 1.08"
* Weight: 147.71g

**3.8.2 Description**

It is an electromagnetic lock, designed for a basic cabinet, safe or door. When (9-12) V DC is applied, the slug pulls in so it does not stick out and the door can be opened. It does not use any power in this state. It is very easy to install for automatic door lock systems like electric door lock with the mounting board.

Solenoids are basically electromagnets: they are made of a big coil of copper wire with an armature (a slug of metal) in the middle. When the coil is energized, the slug is pulled into the center of the coil. This makes the solenoid able to pull from one end. This solenoid in particular is nice and strong, and has a slug with a slanted cut and a good mounting bracket. It's basically an electronic lock, designed for a basic cabinet or safe or door. Normally the lock is active so you can't open the door because the solenoid slug is in the way. When 9-12VDC is applied, the slug pulls in so it doesn't stick out anymore and the door can be opened.

**3.9 LED**



**Fig 3.16: LED**

A Light emitting diode (LED) is essentially a pn junction diode. When carriers are injected across a forward-biased junction, it emits incoherent light. Most of the commercial LEDs are realized using a highly doped n and a p Junction. LEDs are a particular type of diodes that convert electrical energy into light.

LEDs are like tiny light bulbs. LEDs require a lot less power to light up by comparison. They're also more energy efficient, so they don't tend to get hot like conventional light bulbs do. LEDs work on the principle of Electroluminescence.

Electroluminescence is the basis of light-emitting diodes (LEDs) in which a p-n junction diode can emit light when an appropriate voltage is applied, so that electrons recombine with electron holes in the device to release energy in the form of photons.

Applications:

* Devices, medical applications, clothing, toys
* Remote Controls (TVs, VCRs)
* Lighting
* Indicators and signs

**3.10 Conclusions**

In this chapter different hardware modules involved in the project are discussed and their interfacing is explained.

**Chapter 4**

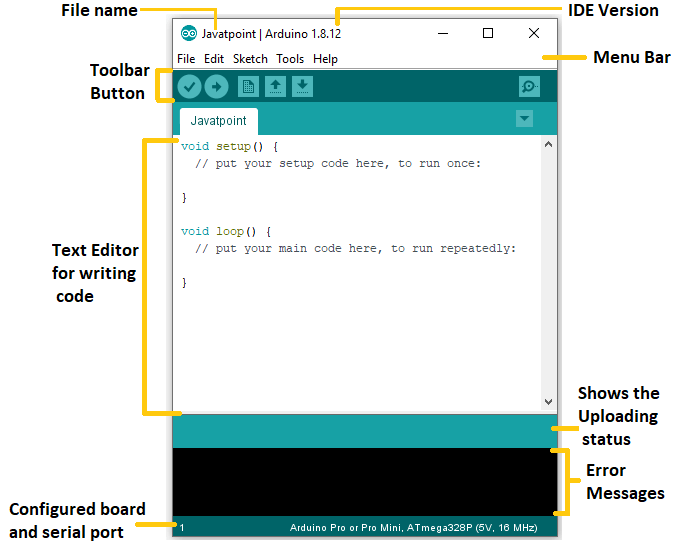
**Software Tools**

**4.1 Introduction**

The software tools which are used in this project are Arduino IDE, Embedded C, Blynk IoT Application and Proteus Software.

**4.2 Arduino IDE**

Arduino IDE (Integrated Development Environment) is used to write code to interface different modules with an Arduino board. An Arduino board connects to the computer via USB cable. Arduino IDE is user friendly where code can be written and various toolbars are available for easy programming. Arduino IDE interface is shown in Fig. 4.1.



**Fig 4.1: Arduino IDE**

Steps to follow for Arduino programming IDE:

**Step 1:** Choose a suitable Arduino board and connect it to PC using USB cable.  
**Step 2:** Arduino IDE Software should be downloaded and installed from the website.

**Step 3:** Provide power supply. Most Arduino boards have USB as a power source. Adaptor can also be a choice. The power LED named PWR glows on the power supply.

**Step 4:** Start Arduino IDE. Open the installed software by double click.

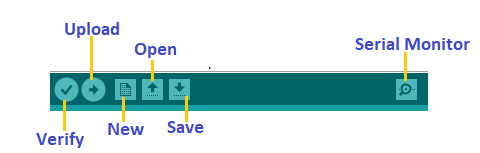
**Step 5:** Initiate a project. It can be done in two ways:

1. A new project can be designed.
2. An existing project can be reviewed.

**Step 6:** Type of Arduino board needs to be selected. Any discrepancy can be eliminated while uploading code if a compatible board is chosen. Board can be chosen from the toolbar.

**Step 7:** Configuration of Serial port. See the port to which Arduino is connected and select that port like COM3.

**Step 8:** The program can now be uploaded to the board. The code is finally dumped and the result can be verified. Done uploading is seen on screen if upload is successful.



**Fig 4.2: Arduino IDE tool bar**

## **4.3 Embedded C**

Embedded C is a set of language extensions for the [C programming language](https://en.wikipedia.org/wiki/C_(programming_language)) by the [C Standards](https://en.wikipedia.org/wiki/ISO/IEC_JTC_1/SC_22) Committee to address commonality issues that exist between C extensions for different [embedded systems](https://en.wikipedia.org/wiki/Embedded_system). In 2008, the C Standards Committee extended the C language to address these issues by providing a common standard for all implementations to adhere to. It includes a number of features not available in normal C, such as, fixed point arithmetic, named address spaces, and basic I/O hardware addressing.

**4.4 MIT App Inventor**



**Fig 4.3: MIT App Inventor Logo**

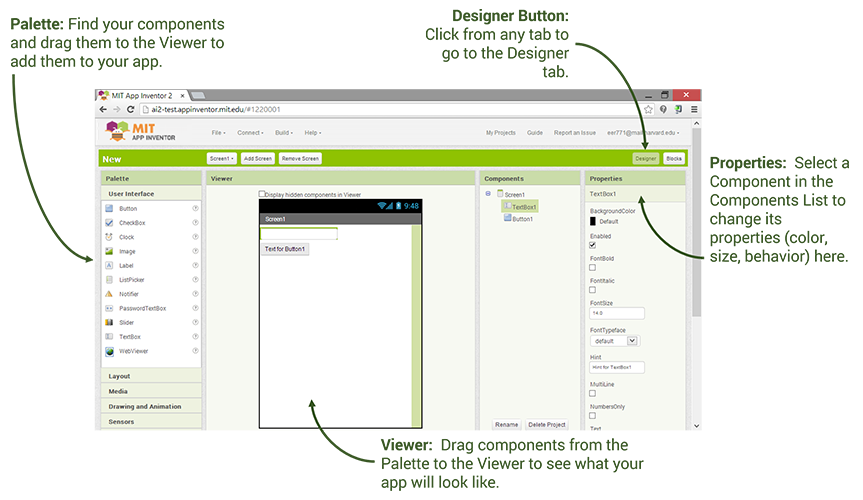
**Introduction**

MIT App Inventor is an online development platform that anyone can leverage to solve real-world problems. It provides a web-based “What you see is what you get” (WYSIWYG) editor for building mobile phone applications targeting the Android and iOS operating systems. It uses a block-based programming language built on Google Blockly and inspired by languages such as Star Logo TNG and Scratch, empowering anyone to build a mobile phone app to meet a need.

 MIT App Inventor is designed to democratize technology and is used as a tool for learning computational thinking in a variety of educational contexts, teaching people to build apps to solve problems in their communities. The MIT App Inventor user interface includes two main editors:

1. Design editor: The design editor, or designer, is a drag and drop interface to lay out the elements of the application’s user interface (UI).
2. Blocks editor: The blocks editor is an environment in which app inventors can visually lay out the logic of their apps using color-coded blocks that snap together like puzzle pieces to describe the program.

To aid in development and testing, App Inventor provides a mobile app called the App Inventor Companion that developers can use to test and adjust the behaviour of their apps in real time. In this way, anyone can quickly build a mobile app and immediately begin to iterate and test.

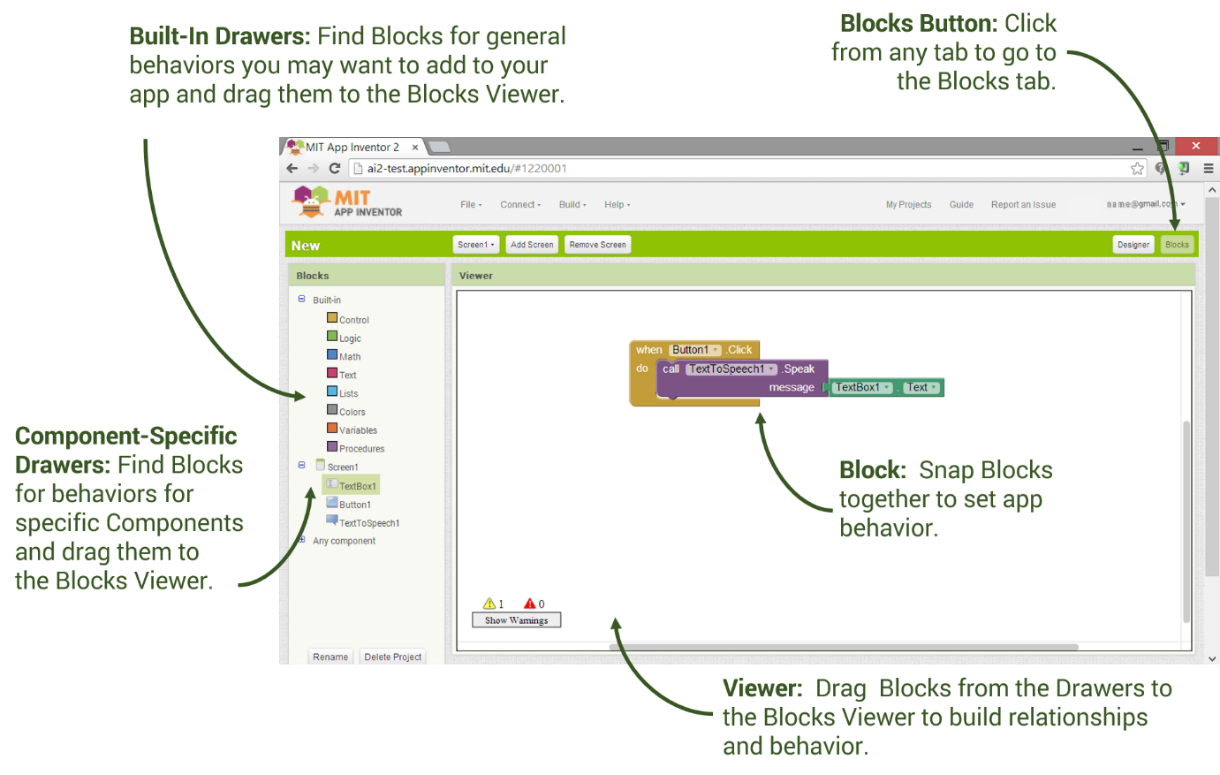


**Fig 4.4: MIT App Inventor Design Editor**

**Blocks as Logic**

In MIT App Inventor, users code application behaviour using a block-based programming language. There are two types of blocks in App Inventor:

1. Built-in blocks: The built-in blocks library provides the basic atoms and operations generally available in other programming languages, such as Booleans, strings, numbers, lists, mathematical operators, comparison operators, and control flow operators.
2. Component blocks: Developers use component blocks (properties, methods, and events) to respond to system and user events, interact with device hardware, and adjust the visual and behavioural aspects of components.



**Fig 4.5: MIT App Inventor Block Editor**

**What you See is What You Get**

The design editor for App Inventor allows developers to see how the app will appear on the device screen and adjust the form factor of the visualized device (e.g., phone or tablet). Adjustments to properties of the visual components, for example, background colour and size, are reflected in real time. Apps can also be run in a livedevelopment mode using the Companion, which we will be discussed in more detail below.

The App Inventor team recently added capability for creating map-based applications. The functionality allows app inventors to drag, drop, and edit markers, lines, polygons, rectangles, and circles in their maps, as well as integrate web-based data from geographic information systems (GIS) to build content-rich apps. This way, the user can move the content around easily to achieve great results without needing to provide most of the logic for this in code.

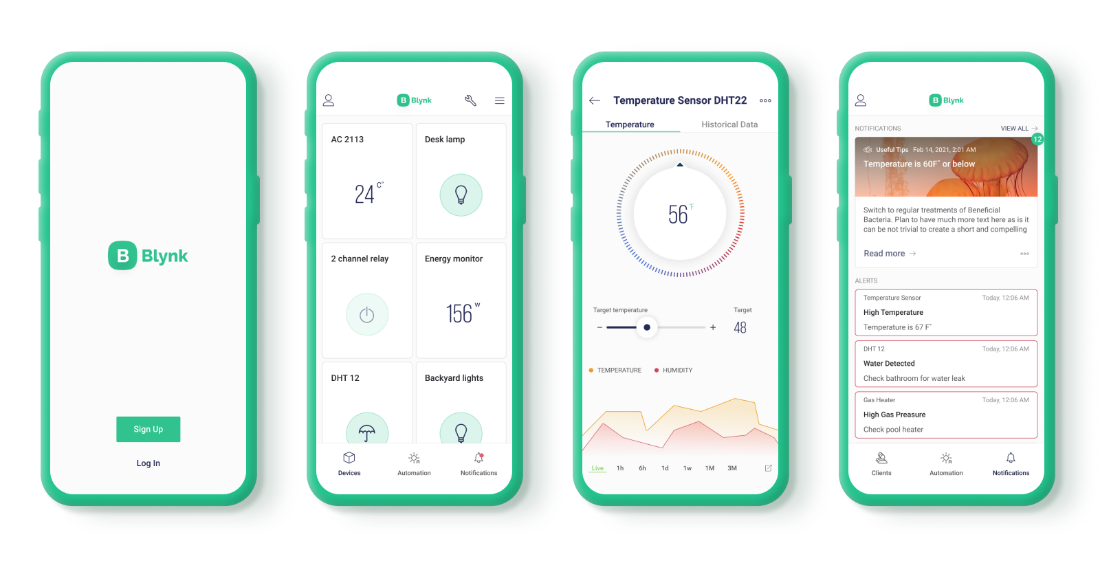
**Fast Iteration and Design Using the Companion**

A key feature of MIT App Inventor is its live development environment for mobile applications. App Inventor provides this by means of a companion app installed on the user’s mobile device. The App Inventor web interface sends code to the companion app, which interprets the code and displays the app in real time to the developer. This way, the user can change the app’s interface and behaviour in real time.

The traditional build cycle for an Android app involves writing code in a text editor or integrated development environment, and rebuilding the application for testing may often take minutes, whereas making a change in the live development environment typically takes effect in 1–2 s. Seeing changes reflected in the app quickly means that students can explore and even make mistakes while exploring, because the time cost of those mistakes is relatively small.

## **4.5 Blynk IoT App**

Blynk is an IOT platform for iOS or Android smartphones that is used to control Arduino, Raspberry Pi and NodeMCU via the Internet. This application is used to create a graphical interface or human machine interface (HMI) by compiling and providing the appropriate address on the available widgets.



**Fig 4.6: Blynk IoT App**

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, visualize it and do many other cool things.

There are three major components in the platform:

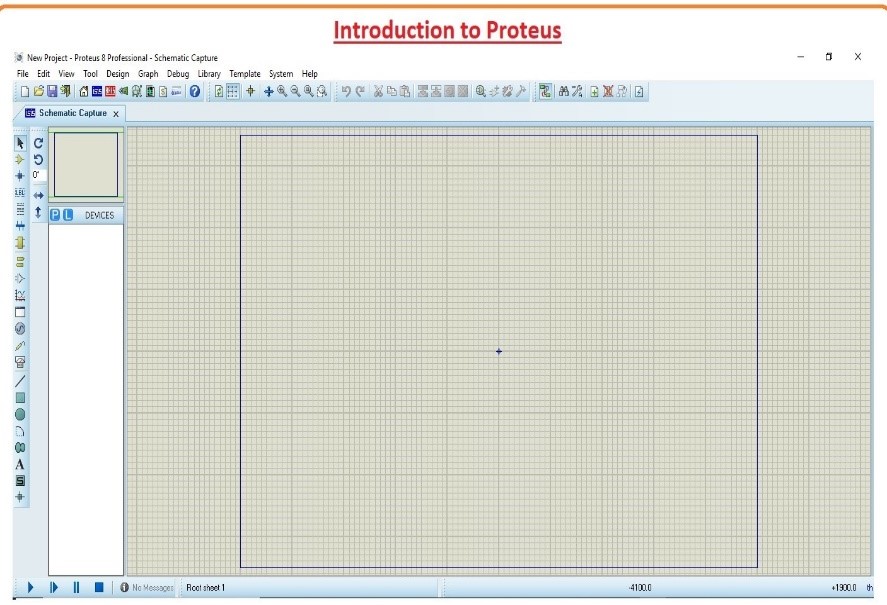
1. **Blynk App**: - It allows you to create amazing interfaces for your projects using various widget which are provided.
2. **Blynk Server**: - It is responsible for all the communications between the smartphone and hardware. You can use the Blynk Cloud or run your private Blynk server locally. It is open source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.
3. **Blynk Libraries**: - It enables communication, for all the popular hardware platforms, with the server and process all the incoming and outcoming commands.

The process that occurs when someone presses the Button in the Blynk application is that the data will move to Blynk Cloud, where data finds its way to the hardware, happens in a blink of an eye.

**4.6 Proteus Software**

**Introduction to Proteus**

Proteus is used to simulate, design and draw electronic circuits. It was invented by Labcenter Electronic. By using Proteus, one can make two-dimensional circuit designs as well. With the use of this engineering software, it is possible to construct and simulate different electrical and electronics circuits on personal computers or laptops. There are numerous benefits to simulate circuits on Proteus before making them practically. Designing of circuits on the Proteus takes less time than practical construction of the circuit. The possibility of error is less in software simulation such as loose connection that takes a lot of time to find out connection problems in a practical circuit.



**Fig 4.7: Proteus Software Window**

Circuit simulations provide the main feature that some components of circuits are not practical then you can construct your circuit on proteus. There is zero possibility of burning and damaging of any electronic component in proteus. The electronic tools that are very expensive can be easily used in Proteus such as an oscilloscope.

There are 2 main parts of Proteus.

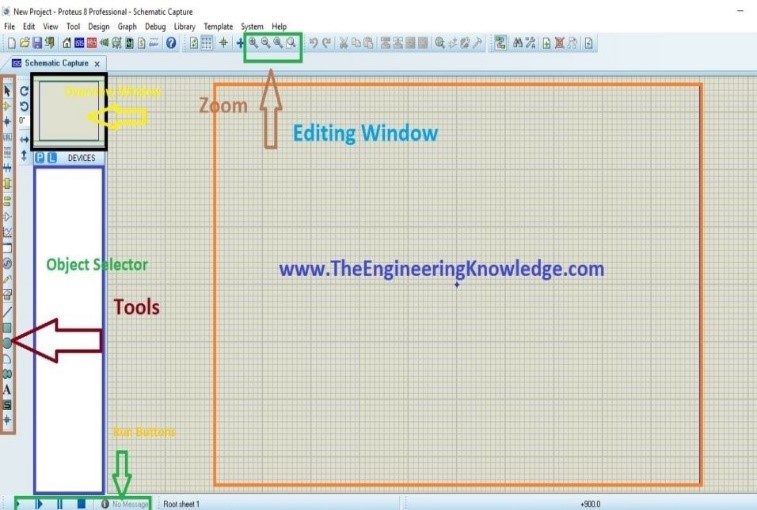
1. ISIS that used to design and simulate circuits.
2. ARES that used for designing of a printed circuit board.

It also provides features related to the three-dimensional view of design in PCB.

**Proteus Layout**

In the designing process, first run the program by clicking on the icon, and a new splash screen will appear. Next, a grid-like workspace will appear. There are buttons available that will help us in designing our PCB. There is a blue rectangle outline; our circuit will be designed within this rectangular space.

When you click on the Icon of installed Proteus in your computer then you will see this window from a new file option.



**Fig 4.8: Various options in Proteus Software**

**Editing Window:** This is a drawing portion of proteus where you simulate your engineering circuits and projects.

**Overview Window:** In the overview window, you see complete view of your complete design.

**Overview Selector:** This section has 2 buttons P and E.

1. P is used to select different components.
2. E button is for editing something. For example, you want to vary any value of components than you can use this edit button.

**Zoom Option:** By using this option, you can easily zoom in and zoom out your layout and can observe complete simulation very clearly.

**Tool Option:** By using this option there are 4 buttons run, stop, pause and stop. These buttons are like the remote control and on and off your circuit.

**How to make circuit in Proteus**

**Step 1: First of all, click on Proteus icon in your computer and click on a new file option.**

**Step 2: After that, you will see the drawing sheet as shown in the below figure. Save it according to your project.**

**Step 3: After this, go to the component option and select the elements for your projects.**

**Step 4: After clicking on components mode, you will see two buttons P and L, if you move to P button you will get “Pick from Libraries” above it.**

**Step 5: When you will click on the P button you will see a box. Type your required component for a circuit.**

**Step 6: When you select components for your project, you will see them in a box at the left.**

**After the selection of components, make the circuit layout of your project and connect all these components with the wires. For connection of one component to other, click on left of first one terminal of component and drag it to other components. To remove any component or remove its connection, just double click on respective component of wire. To change the values of any component such as resistance, capacitor, then click right on that component and select the desired value and click OK button.**

**Step 7: When you connect all components in the circuit like run button in the left bottom, see the practical working of your circuit.**

**Step 8: When you observe the simulation of your circuit, then click on stop button on the left bottom to stop the working of the circuit.**

**4.7 Conclusions**

In this chapter, software tools like Arduino IDE, Embedded C, MIT App Inventor, Blynk IoT App and Proteus Software are studied.

**Chapter 5**

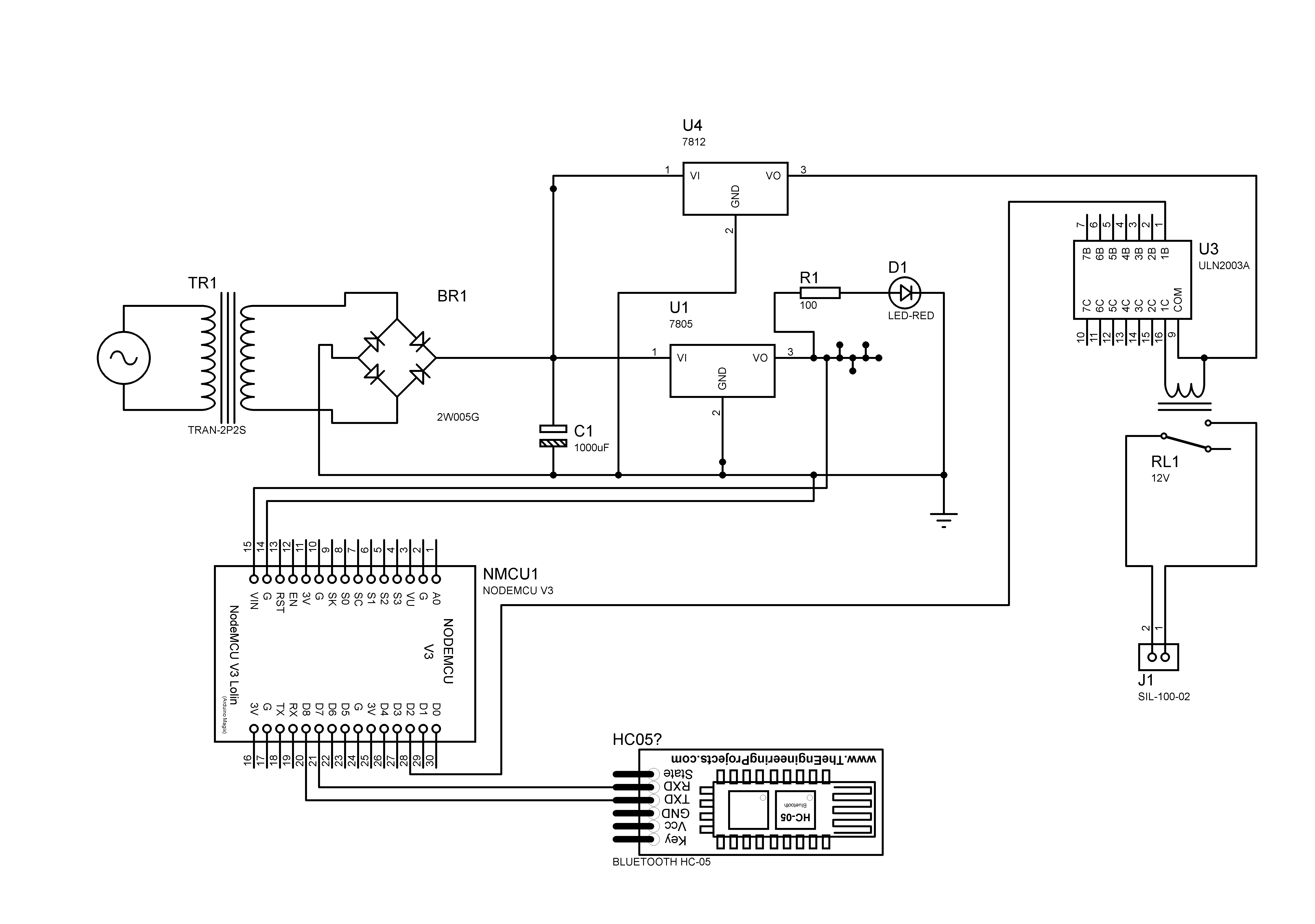
**Results and Discussion**

**5.1 Introduction**

In this chapter we discuss the project outputs and the sequential process of the project, i.e., flowchart of the project, and schematic diagram of setup (by Proteus software), explanation of experimental setup, and results of the project.

**5.2 Schematic diagram**

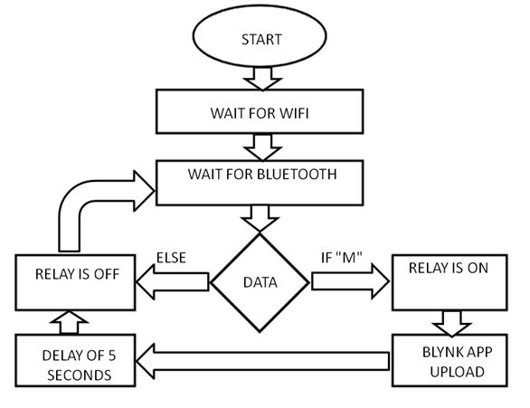
Figure 5.1 shows the schematic diagram of the project.

**Fig 5.1: Schematic diagram of the project**

The HC-05 Bluetooth module is interfaced with NodeMCU. All the other components are connected with their respective pin connections. The circuit is connected as per the schematic diagram shown above.

**5.3 Flowchart**

Figure 5.2 shows the flowchart of NodeMCU.



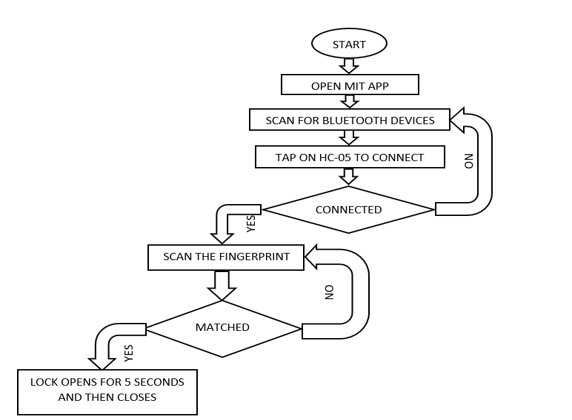
**Fig 5.2: NodeMCU flowchart**

1. The execution starts from NodeMCU.
2. Initially, NodeMCU waits for the Wi-fi connectivity. Once connected to the Wi-fi, NodeMCU waits for Bluetooth.
3. Further execution is carried out in MIT application.
4. As NodeMCU waits for Bluetooth, user opens the MIT Application and scans for Bluetooth devices.
5. If the user connects to HC-05 Bluetooth Module, the application allows the user to scan his/her fingerprint. Else, it will ask to connect to Bluetooth.
6. If the scanned fingerprint matches, Bluetooth sends ‘M’ to the NodeMCU. Else, it will ask to scan fingerprint again.

Now, the execution flow goes back to NodeMCU.

1. If the received data is ‘M’, relay is turned ON which opens the lock and the owner is alerted through notifications. After 5 seconds, relay is turned OFF and the lock is automatically closed.

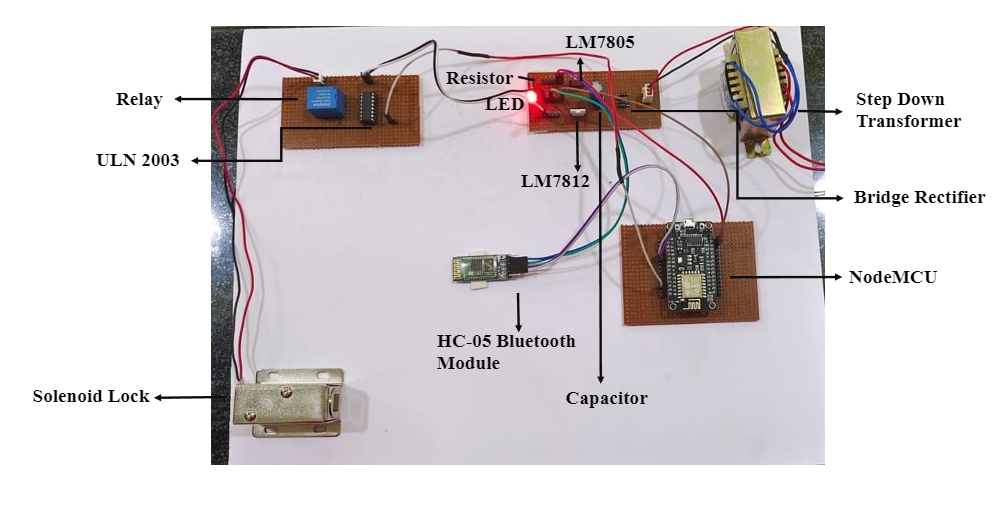
Figure 5.3 shows the flowchart of MIT Application.



**Fig 5.3: MIT App flowchart**

**5.4 Experimental Setup**

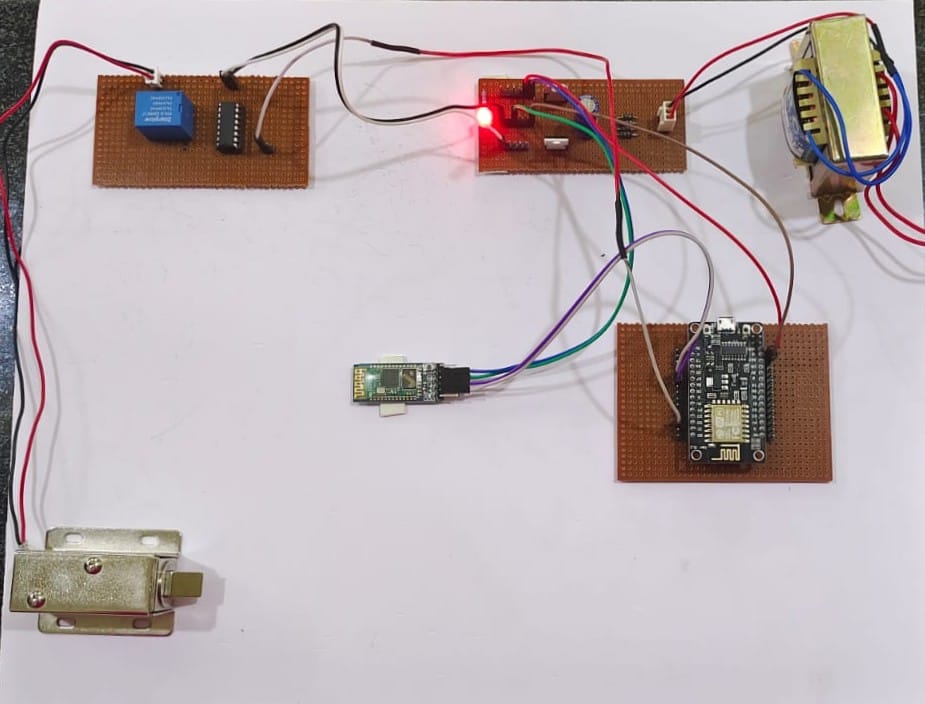
Figure 5.4 shows the experimental setup of the project.



**Fig 5.4: Experimental Setup**

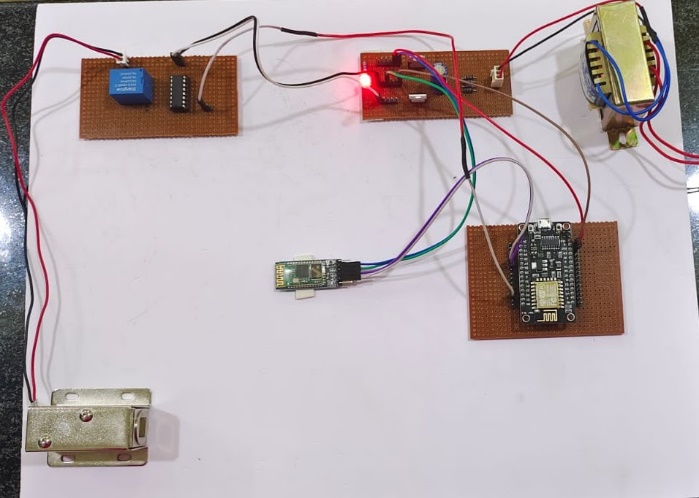
**5.5 Results**

The following figures show the results obtained during this project.



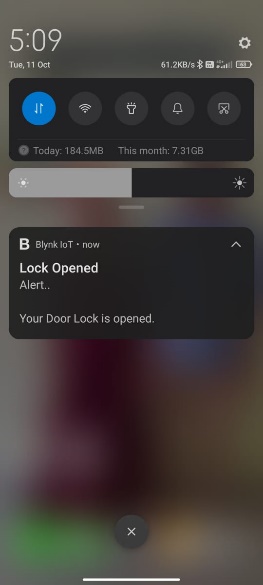
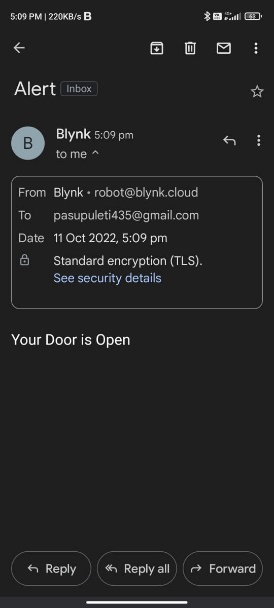
**Fig 5.5: Lock Closed**

In figure 5.5 it can be observed that the slug is sticking out from the solenoid lock indicating that the lock is closed.



**Fig 5.6: Lock Opened**

In figure 5.6 it can be observed that the slug is not sticking out from the solenoid lock indicating that the lock is opened.



**Fig 5.7 Notifications received by the owner**

Figure 5.7 shows the notifications received by the owner when the lock is opened.

**5.6 Conclusions**

In this chapter we have seen the schematic diagram, flowchart, experimental setup of the project and their results during various operations.

**Chapter 6**

**Conclusions**

The wireless biometric lock developed in this project is capable of securing a house by providing entry access only to authorized people, along with real-time notifications to the homeowner. A homeowner can monitor the number of times the lock is opened using the data uploaded to the Blynk IoT Application. Advancements in biometric identification management technology are moving so fast, in future, this system can be enhanced with the use of multiple functions like image recognizing process and password-based system. Also, eye retina recognition for password which helps in authenticating authorized persons for entrance. So, biometric technology makes an individual comfortable in real life. So, fingerprint-based door lock system needs some improvements for better functioning and efficiency.

**References**

1. https://www.arduino.cc/
2. https://www.electronicshub.org/arduino-introduction/
3. https://ieeexplore.ieee.org/document/9760583