

# 1. INTRODUCTION

The Internet of Things (IoT) is the network of physical objects, devices, vehicles, buildings and other items which are embedded with electronics, software, sensors, and network connectivity, which enables these objects to collect and exchange data. The Internet of Things allows objects to be sensed and controlled remotely across existing network infrastructure, creating opportunities for more-direct integration between the physical world and computer-based systems, and resulting in improved efficiency, accuracy and economic benefit. When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, smart homes, intelligent transportation and smart cities. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet infrastructure.

British entrepreneur Kevin Ashton first coined the term in 1999 while working at Auto-ID Labs. Typically, IoT is expected to offer advanced connectivity of devices, systems, and services that goes beyond machine-to-machine communications and covers a variety of protocols, domains, and applications. The interconnection of these embedded devices (including smart objects), is expected to usher in automation in nearly all fields, while also enabling advanced applications like a Smart Grid, and expanding to the areas such as smart cities.

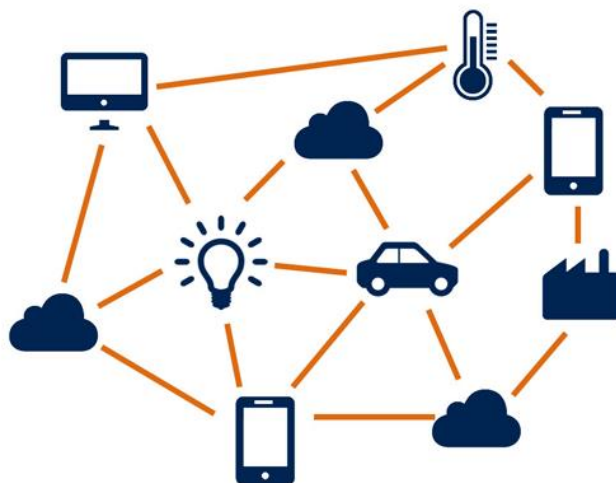
"Things," in the IoT sense, can refer to a wide variety of devices such as heart monitoring implants, biochip transponders on farm animals, electric clams in coastal waters, automobiles with built-in sensors, DNA analysis devices for environmental/food/pathogen monitoring or field operation devices that assist firefighters in search and rescue operations. These devices collect useful data with the help of various existing technologies and then autonomously flow the data between other devices. Current market examples include smart thermostat systems and washer/dryers that use Wi-Fi for remote monitoring.

The problem is, people have limited time, attention and accuracy -- all of which means they are not very good at capturing data about things in the real world. If we had computers that knew everything there was to know about things -- using data they gathered without any help from us - - we would be able to track and count everything and greatly reduce waste, loss and cost.

### a. Theoretical Background/History

The concept of a network of smart devices was discussed as early as 1982, with a modified Coke machine at Carnegie Mellon University becoming the first internet-connected appliance, able to report its inventory and whether newly loaded drinks were cold. Mark Weiser's seminal 1991 paper on ubiquitous computing, "The Computer of the 21st Century", as well as academic venues such as UbiComp and PerCom produced the contemporary vision of IoT. In 1994 Reza Raji described the concept in IEEE Spectrum as "moving small packets of data to a large set of nodes, so as to integrate and automate everything from home appliances to entire factories". Between 1993 and 1996 several companies proposed solutions like Microsoft's at Work or Novell's NEST. However, only in 1999 did the field start gathering momentum. Bill Joy envisioned Device to Device communication as part of his "Six Webs" framework, presented at the World Economic Forum at Davos in 1999.

The concept of the Internet of Things first became popular in 1999, through the Auto-ID Center at MIT and related market-analysis publications. Radio-frequency identification (RFID) was seen by Kevin Ashton (one of the founders of the original Auto-ID Center) as a prerequisite for the Internet of Things at that point. If all objects and people in daily life were equipped with identifiers, computers could manage and inventory them. Besides using RFID, the tagging of things may be achieved through such technologies as near field communication, barcodes, QR codes and digital watermarking.



*Fig. 1: Internet of Things*

## b. Objective and Scope of the Project

ADUNIK, the collaboration of Embedded electronic devices and an Android application, is a project that highlights the growing advancements in the Internet of Things (IoT) domain. Our project demonstrates the use of an Android Smartphone to manipulate the functions of Embedded electronic devices like Arduino Microcontrollers to control RGB Led Lights and Remote Controlled (RC) Car. This project enlightens the network of several devices and how they can be used in unison to produce better functionalities for the ease of human lifestyle.

The scope of the project is to enable Android smartphone owners to use their smartphone to control the electronic devices around them, thus eliminating the need of manually controlling the electronic devices, by integrating all the controls at one place and saving time.

This project is beneficial for people looking forward to Home Automation. All major appliances can be connected to the home automation system and can be controlled remotely. Depending on your unique requirements, new customized systems can be developed and installed. There is no limit to imagination and every feature one may need or think of can be developed now or in coming future.



*Fig. 2: Arduino and Android collaboration*

## **2. SYSTEM ANALYSIS**

### **a. Problem Definition**

The world in which we live is surrounded by electronic devices. Each device possessing its own unique quality or functionality. Some of these devices are what we use in everyday life, like mobile phones, TVs, computers etc.

But all these devices being independent, are not connected to each other in any way and each of them need to be manipulated or controlled separately.

Smartphone, a revolutionary innovation brings an entire online world to our fingertips. Weather forecast, E-mails, Media, Games, Social Networks, as far as personal assistance go, it can be anything we ask for. What if this smartphone could move and manipulate objects or devices for us, then we will have the perfect device.

### **b. Proposed System**

The solution to overcome the individuality of each device is to connect them together in a network. This will make sure that the abilities of one device is working in collaboration with the abilities of other devices to create something more efficient and productive.

The RGB led lights and the RC Car are connected with the arduino board. To interact with the LEDs and the RC Car, the user must first make a connection with the arduino board. The user opens the android application and turns on the Bluetooth. The user then selects the external Bluetooth device to pair with. The android application then makes a connection with the arduino board. The arduino board contains a program which receives the incoming value and processes it to generate appropriate results. The user can now change the color of the lights or turn them on and off as desired and can control the movement of the RC Car.

**c. Advantages**

1. Reduces time to go around and manually interact with the devices.
2. Less manual interaction.
3. This project offers solution that is helpful for home owners to ease their lifestyle.
4. It seems likely that automated system like this will become an integral part of many existing and future homes or offices.

### 3. REQUIREMENTS ANALYSIS

#### a. Functional Requirements

1. The Android application must be able to connect to the Bluetooth module.
2. The Android application must have appropriate buttons and sliders to control the behavior of RGB LEDs.
3. The Android application must have a switch/toggle button to toggle between 'Button Control' and 'Tilt Control' for the RC Car.
4. The Android application must be able to send data to the Arduino Board.
5. The Arduino Board must be able to read the value using the Bluetooth module.
6. The Arduino Board must be able to send data to the Dual Bridge Motor module.
7. The Dual Bridge Motor module must be able to control the speed and polarity of the DC Motors.

#### b. Non-functional Requirements

1. **Accessibility.** The user must be able to access the Android application and connect to the Arduino Board through Bluetooth.
2. **Efficiency.** The Bluetooth module must be efficient to receive.
3. **Fault tolerance.** The system must not be working on false values provided by the Application.
4. **Quality.** The quality of the components used in the system must be high as connectivity via Bluetooth plays major roles.
5. **Reliability.** The system must be reliable i.e. the output/result provided by the system must be correct so user can rely on the system.
6. **Response time.** The response time of the data provided by the Arduino to the Application must be minimal to achieve efficiency.

### **c. Hardware Requirements**

- a. Smartphone specifications:
  - i. Processor: 1GHz Dual Core or greater
  - ii. RAM: 1GB or greater
  - iii. OS: Android Ice-cream Sandwich (4.0.3) or higher
- b. Modules:
  - i. Arduino Uno R3
  - ii. RGB LEDs
  - iii. 74HC595 Shift Registers
  - iv. HC-05 Bluetooth Module
  - v. L298N Dual H-Bridge Motor Driver Module
  - vi. DC Motor
  - vii. Servo Motor
  - viii. 9V and 12V Batteries
  - ix. Resistors (270 ohms)

### **d. Software Requirements**

- a. Arduino IDE 1.6.7
- b. Android Studio 141.2456560

## 4. FEASIBILITY STUDY

As the name implies, feasibility study is an analysis of the viability of an idea. It ensures that a project is legally and technically feasible and economically justifiable.

Three different areas of Project Feasibility:

1. **Operational Feasibility**, under which we conduct a study to analyze and determine whether your business need can be fulfilled by using a proposed solution. It also measures how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters such as reliability, maintainability, supportability, usability, disposability, sustainability, affordability and others. This project fulfills the requirements and further enhancements can lead to a better Home Automation system.
2. **Economic Feasibility** studies enable organizations to assess the viability, cost and benefits of projects before financial resources are allocated. They also provide independent project assessment and enhance project credibility. It also helps to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification and identification of all the benefits expected. This assessment typically involves a cost/ benefits analysis of the project. Once this system is ready there is no need of additional hardware or software hence it is financially and economically stable and would give the best results.
3. **Technical Feasibility** assessment is focused on the present technical resource available in the organization. It studies if the technical resources including the technical team are capable of converting the ideas into working system. It also evaluates the hardware and the software requirement of the proposed system. In this project the Arduino Uno, Shift Register, Bluetooth Module, RGB LEDs, Dual Bridge Motor, DC Motors and Servo Motor are available in the market and the project is, therefore, technical feasible to implement.



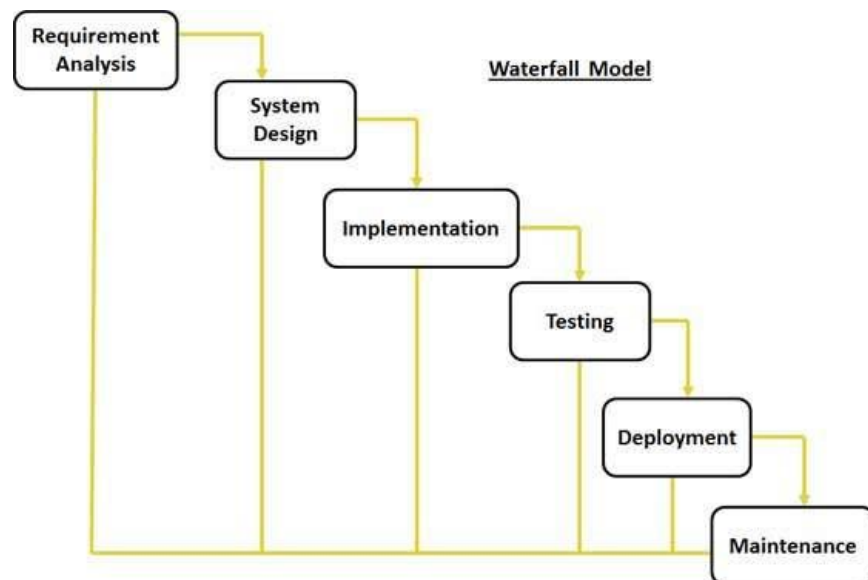
## 5. PROCESS MODEL INVOLVED

The waterfall model is a sequential design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, production/implementation and maintenance.

The waterfall development model originates in the manufacturing and construction industries: highly structured physical environments in which after-the-fact changes are prohibitively costly, if not impossible. Since no formal software development methodologies existed at the time, this hardware-oriented model was simply adapted for software development.

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. Waterfall model is the earliest SDLC approach that was used for software development.

The waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a linear-sequential life cycle model. This means that any phase in the development process begins only if the previous phase is complete. In waterfall model phases do not overlap.



*Fig. 3: Waterfall Model*

**The sequential phases in Waterfall model are:**

1. Requirement Gathering and Analysis
2. System Design
3. Implementation
4. Integration and Testing
5. Deployment of system
6. Maintenance

**In this project the waterfall model is used because:**

- a. All the requirements were clear and fixed.
- b. Requirements are not ambiguous.
- c. The design and architecture was developed.
- d. Ample resources with required expertise are available freely.
- e. As there is no scope for backtracking between the phases.
- f. The project is short.
- g. It is good to use this model when the technology is well understood.
- h. And objective of our system is the automation or computerization of an already existing manual working system.

## **6. TECHNOLOGY USED**

### **1. Arduino IDE**

Arduino programs may be written in any programming language with a compiler that produces binary machine code. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio.

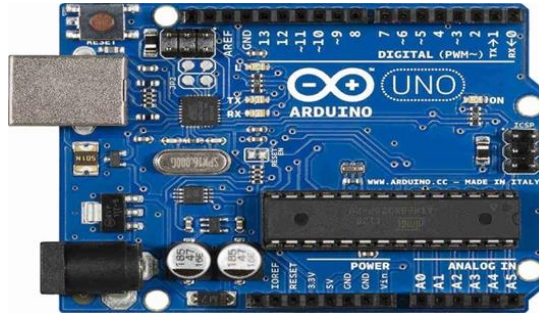
The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in Java. It originated from the IDE for the Processing programming language project and the Wiring project. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and provides simple one-click mechanism for compiling and loading programs to an Arduino board. A program written with the IDE for Arduino is called a "sketch".

The Arduino IDE supports the C and C++ programming languages using special rules of code organization. The Arduino IDE supplies a software library called "Wiring" from the Wiring project, which provides many common input and output procedures. A typical Arduino C/C++ sketch consists of two functions that are compiled and linked with a program stub `main()` into an executable cyclic executive program:

- `Setup ()`: a function that runs once at the start of a program and that can initialize settings.
- `Loop ()`: a function called repeatedly until the board powers off.

After compilation and linking with the GNU tool chain, also included with the IDE distribution, the Arduino IDE employs the program `avrdude` to convert the executable code into a text file in hexadecimal coding that is loaded into the Arduino board by a loader program in the board's firmware.

## 2. Arduino Uno R3



*Fig. 4: Arduino Uno R3*

The Uno is a microcontroller board based on the [ATmega328P](#). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started... You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the [Arduino index of boards](#).

Each of the 14 digital pins on the Uno can be used as an input or output, using `pinMode()`, `digitalWrite()`, and `digitalRead()` functions. They operate at 5 volts.

Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

- a. Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- b. External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the `attachInterrupt()` function for details.
- c. PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the `analogWrite()` function.
- d. SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- e. LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- f. TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though it is possible to change the upper end of their range using the AREF pin and the `analogReference()` function.

There are a couple of other pins on the board

- g. AREF. Reference voltage for the analog inputs. Used with `analogReference()`.

Reset: Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

### 3. HC-05 Bluetooth Module



*Fig. 5: HC-05 Bluetooth Module*

These low-cost Bluetooth Sub-modules work well with Arduino and other Microcomputers.

HC-05 is a more capable module that can be set to be either Master or Slave.

The HC-05 Bluetooth Module has 6 pins- Vcc, GND, TX, RX, Key, and LED. It comes pre-programmed as a slave, so there is no need to connect the Key pin, unless you need it change it to Master Mode.

The major difference between Master and Slave modes is that, in Slave mode the Bluetooth module cannot initiate a connection; it can however accept incoming connections. After the connection is established the Bluetooth module can transmit and receive data regardless of the mode it is running in. If you are using a phone to connect to the Bluetooth module, you can simply use it in the Slave mode. The default data transmission rate is 9600kbps.

The range for Bluetooth communication is usually 30m or less. The module has a factory set pin of “1234” which is used while pairing the module to a phone.

The module has two modes of operation, Command Mode where we can send AT commands to it and Data Mode where it transmits and receives data to another Bluetooth module.

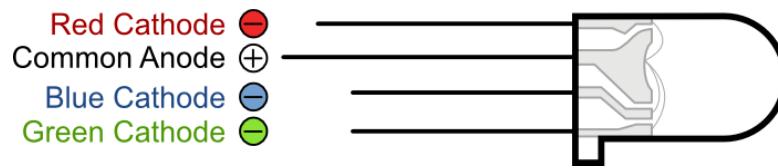
The default mode is DATA Mode, and this is the default configuration, that may work fine for many applications:

- a. Baud Rate: 9600 bps, Data : 8 bits, Stop Bits: 1 bit, Parity : None, Handshake: None
- b. Passkey: 1234
- c. Device Name: HC-05

HC05 Pin Out:

- a. KEY: If brought HIGH before power is applied, forces AT Command Setup Mode.  
LED blinks slowly (2 seconds)
- b. VCC: +5 Power
- c. GND: System / Arduino Ground
- d. TXD: Transmit Serial Data from HC-05 to Arduino Serial Receive. NOTE: 3.3V  
HIGH level: OK for Arduino
- e. RXD: Receive Serial Data from Arduino Serial Transmit
- f. STATE: Tells if connected or not

#### 4. RGB LEDs (Common Anode)



*Fig. 6: RGB LED (Common Anode)*

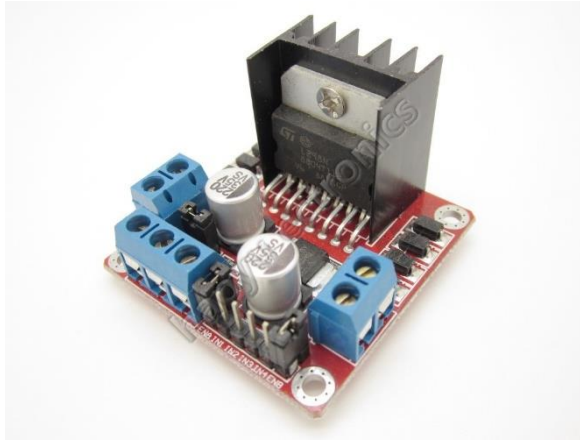
#### 5. 74HC595 Shift Register



*Fig. 7: 74HC595*

The 74HC595 is an 8-bit serial-in/serial or parallel-out shift register with a storage register and 3-state outputs. Both the shift and storage register have separate clocks. The device features a serial input (DS) and a serial output (Q7S) to enable cascading and an asynchronous reset MR input. A LOW on MR will reset the shift register. Data is shifted on the LOW-to-HIGH transitions of the SHCP input. The data in the shift register is transferred to the storage register on a LOW-to-HIGH transition of the STCP input. If both clocks are connected together, the shift register will always be one clock pulse ahead of the storage register. Data in the storage register appears at the output whenever the output enable input (OE) is LOW. A HIGH on OE causes the outputs to assume a high-impedance OFF-state. Operation of the OE input does not affect the state of the registers. Inputs include clamp diodes. This enables the use of current limiting resistors to interface inputs to voltages in excess of VCC.

## 6. L298N Dual H-Bridge Motor Driver



*Fig. 8: L298N Dual H-Bridge Motor Driver*

This module is based on the very popular L298 Dual H-Bridge Motor Driver Integrated Circuit. The circuit will allow you to easily and independently control two motors of up to 2A each in both directions.

It is ideal for robotic applications and well suited for connection to a microcontroller requiring just a couple of control lines per motor. It can also be interfaced with simple manual switches, TTL logic gates, relays, etc.

Features:

1. Motor supply: 7 to 24 VDC
2. Control Logic: Standard TTL Logic Level
3. Output Power: Up to 2 A each
4. Enable and Direction Control Pins
5. Heatsink for IC
6. Power-On LED indicator
7. 4 Direction LED indicators



## 7. METHODOLOGY ADOPTED

**Structured programming** is a programming paradigm aimed at improving the clarity, quality, and development time of a computer program by making extensive use of subroutines, block structures, for and while loops—in contrast to using simple tests and jumps such as the goto statement which could lead to "spaghetti code" which is difficult both to follow and to maintain.

**Object-oriented programming (OOP)** is a programming paradigm based on the concept of "objects", which may contain data, in the form of fields, often known as attributes; and code, in the form of procedures, often known as methods. A distinguishing feature of objects is that an object's procedures can access and often modify the data fields of the object with which they are associated (objects have a notion of "this" or "self"). In OO programming, computer programs are designed by making them out of objects that interact with one another. There is significant diversity in object-oriented programming, but most popular languages are class-based, meaning that objects are instances of classes, which typically also determines their type.

*This project supports the concept that a system be constructed using the structured methodology in the early stages of its development, and using object-oriented methodology at later, more detailed, levels. During the complete system development process, formal methods are applied in a manner that demonstrates their practicability.*

Performing both structured and object-oriented analysis provides a better design. One major advantage of following this approach is that regardless of which design (OO or structured) you end up implementing, you can come up with a much better solution by looking at the problem from both a structured approach and an object oriented approach.

For example, by doing structured analysis and design, you can find ways make an object oriented design more efficient – without sacrificing flexibility or reuse. It's also typically that you can develop a more flexible/reusable-structured design by seeing how the system would be implemented using an object oriented approach.

**Conclusion:** Systems engineering has changed significantly over the last 10 years. In the past we typically designed and developed large systems from scratch. Those days are gone. Today systems have to be built faster, cheaper, and they have to be more flexible than the systems in the past. To build today's systems, we have to rely on COTS and reuse. This new focus requires us to use new methods and techniques.

Methodologies like object oriented analysis and design can help with some of the problems, but also bring new challenges. We have to expand our “toolbox” to include a wide variety of techniques to design these systems. The process described provides a complete set of system engineering techniques that can be used to address the challenges of designing systems today.

In this project object oriented approach is used for android application development whereas structure oriented approach is being used for arduino and other programming.

## 8. CIRCUIT DIAGRAM

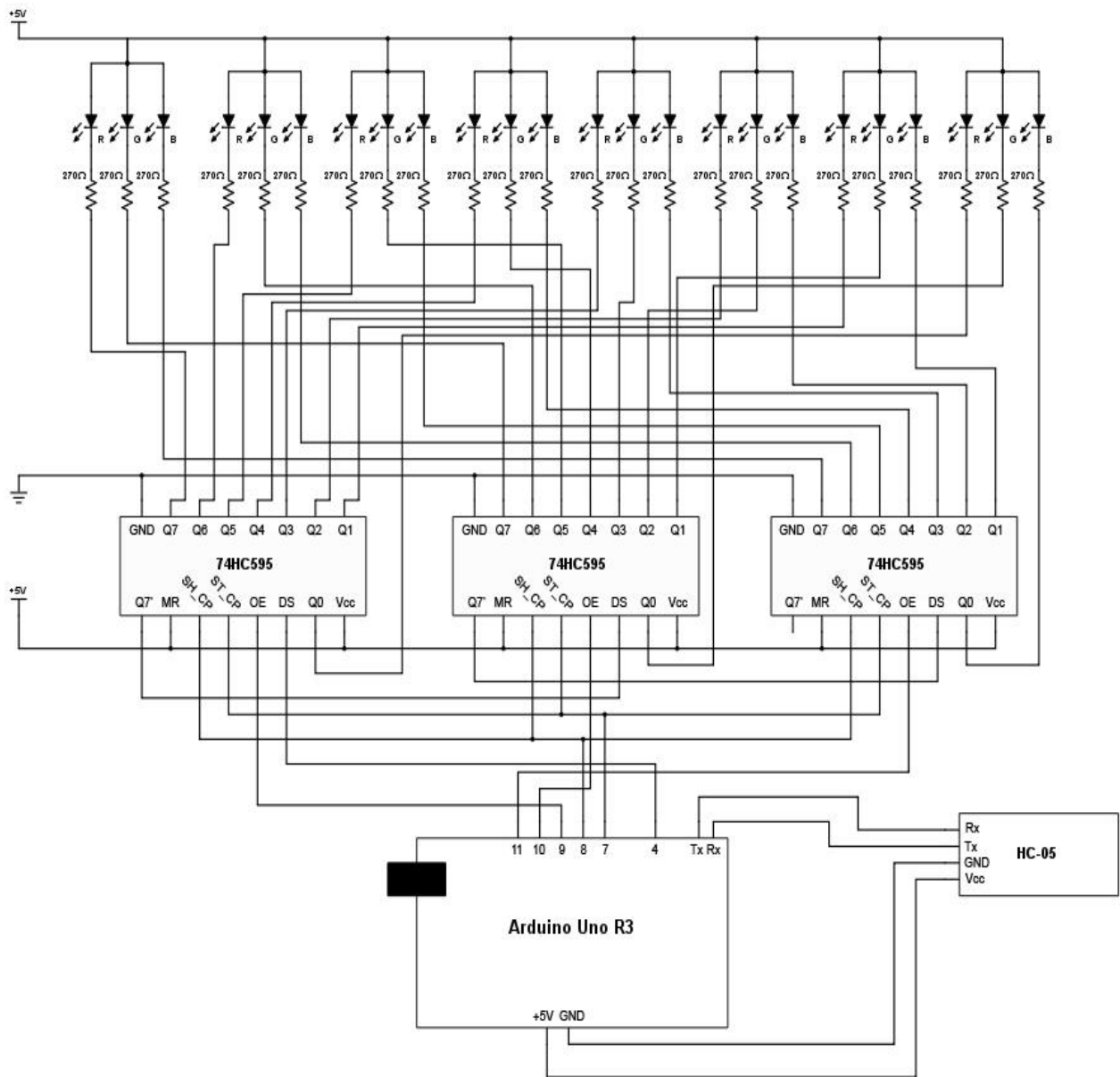
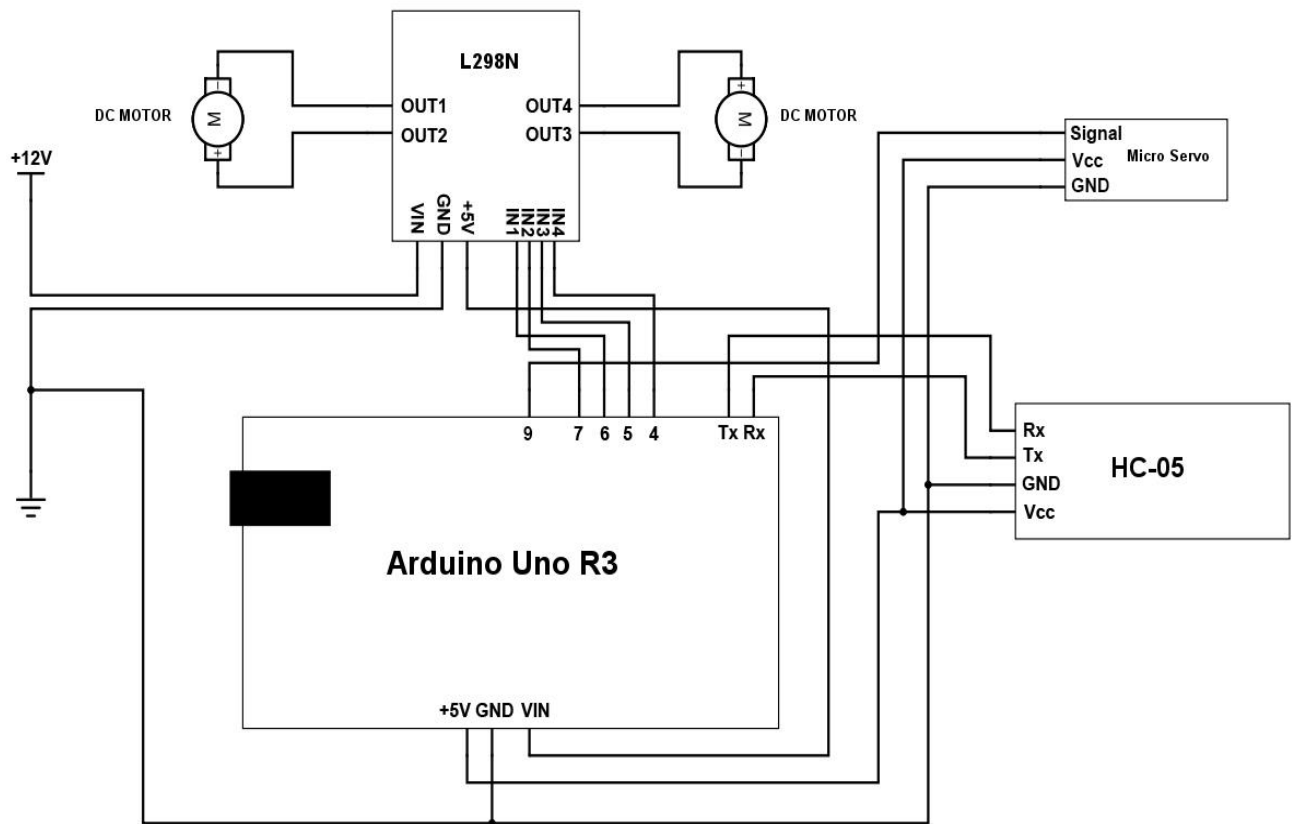


Fig. 9: Circuit Diagram (LED)



*Fig. 10: Circuit Diagram (RC Car)*

## 9. SYSTEM PLANNING

### Gantt Chart

	i	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1			Project Management	119 days	Wed 10/14/15	Mon 3/28/16	
2			Requirement Gathering	22 days	Wed 10/14/15	Thu 11/12/15	
3			Analysis	14 days	Wed 10/14/15	Sat 10/31/15	
4			Feasibility Study	9 days	Mon 11/2/15	Thu 11/12/15	3
5			Analysis Completed	3 days	Fri 11/13/15	Tue 11/17/15	4
6			Design	21 days	Fri 11/13/15	Fri 12/11/15	2
7			Software Design	12 days	Fri 11/13/15	Mon 11/30/15	5
8			Use Case Diagram	3 days	Wed 11/18/15	Fri 11/20/15	
9			Sequence Diagram	4 days	Wed 11/18/15	Sat 11/21/15	8
10			Class Diagram	5 days	Mon 11/23/15	Fri 11/27/15	9
11			Activity Diagram	3 days	Mon 11/30/15	Wed 12/2/15	10
12			Interface Design	7.5 days	Thu 12/3/15	Mon 12/14/15	11
13			Design Complete	4 days	Mon 12/14/15	Fri 12/18/15	12
14				46 days	Mon 12/14/15	Mon 2/15/16	6
15			Development System Module	25 days	Fri 12/18/15	Fri 1/22/16	
16			Integrate System Module	27 days	Mon 1/18/16	Tue 2/23/16	15
17			Implementation Complete	5 days	Wed 2/24/16	Tue 3/1/16	16
18			Testing	28 days	Mon 2/15/16	Wed 3/23/16	14
19			System Testing	7 days	Wed 3/2/16	Thu 3/10/16	
20			Black Box Testing	9 days	Fri 3/11/16	Wed 3/23/16	19
21			White Box Testing	11 days	Thu 3/24/16	Thu 4/7/16	20
22			Cost Estimation	2 days	Fri 4/8/16	Mon 4/11/16	21
23			Testing Complete	3 days	Tue 4/12/16	Thu 4/14/16	22
24			Deployment	7 days	Wed 3/23/16	Thu 3/31/16	18
25			Document	8 days	Fri 4/15/16	Tue 4/26/16	23

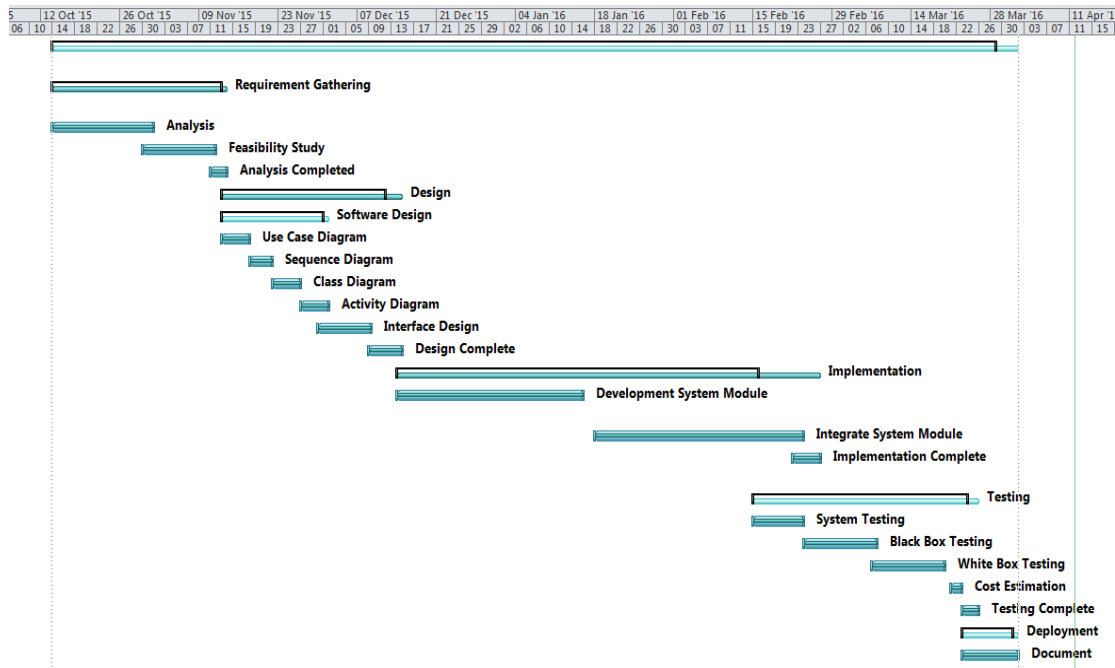


Fig. 11: Gantt Chart

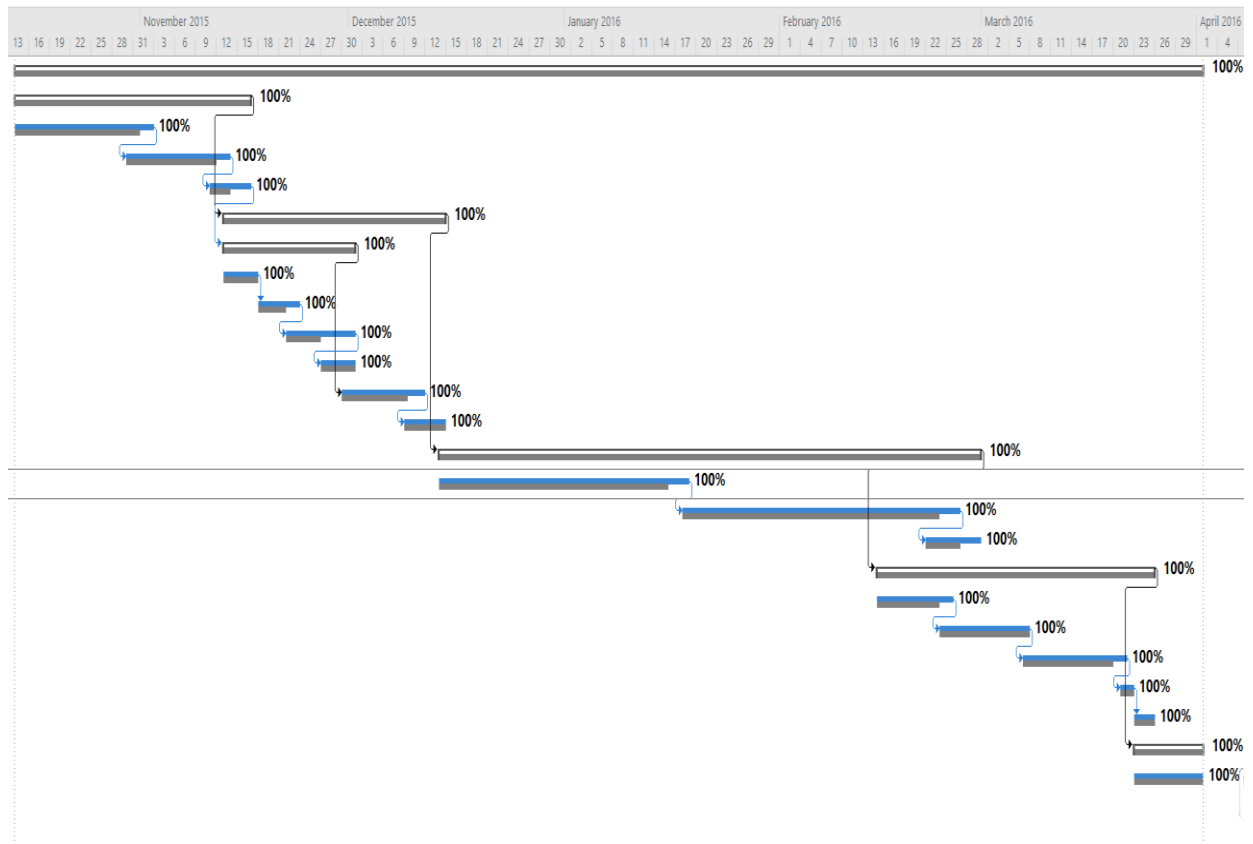


Fig. 12: Tracking Gantt Chart

## 10. DESIGN

For object-oriented approach:

### a. Use Case Diagram

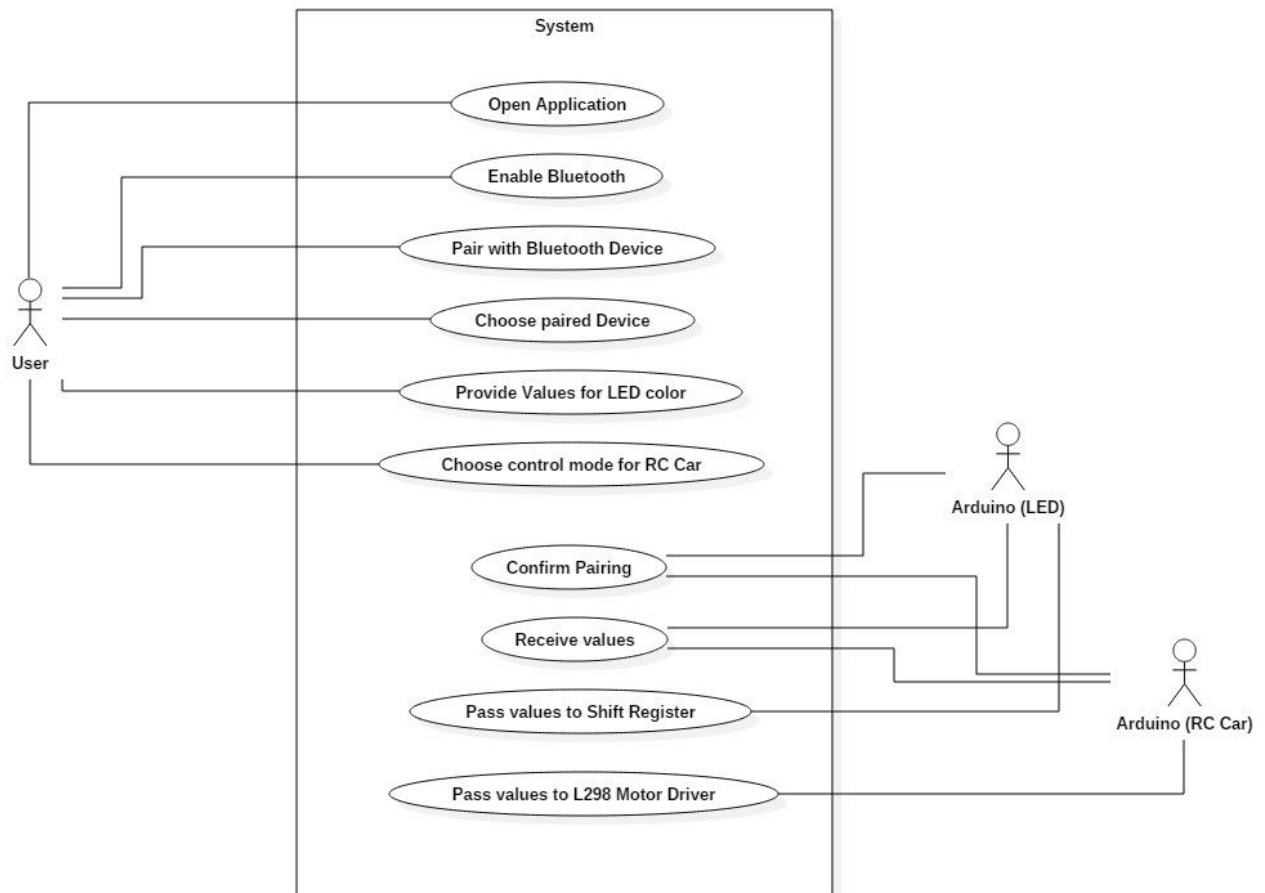


Fig. 13: Use Case Diagram

## b. Activity Diagram

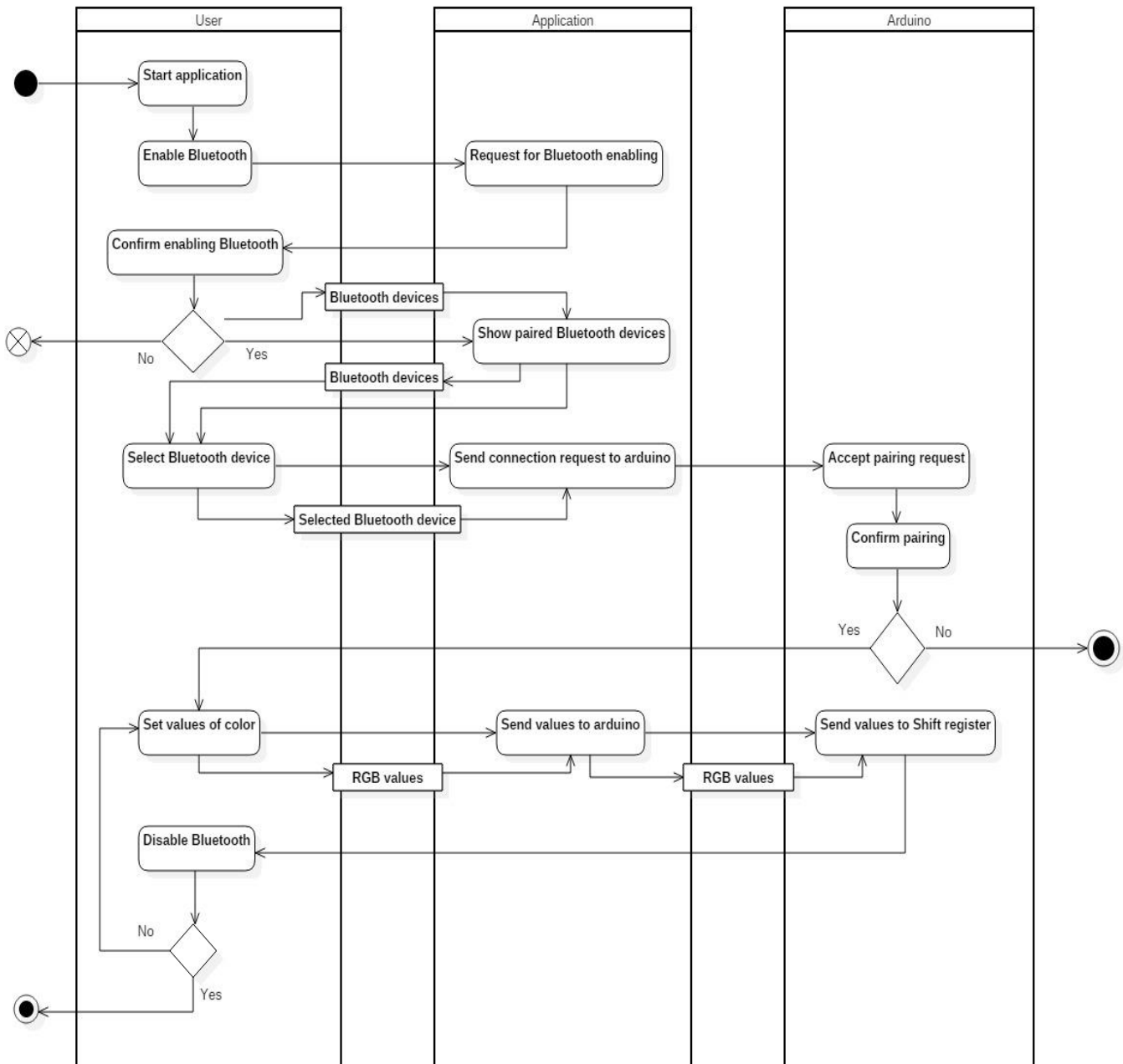


Fig. 14: Activity Diagram (LED)



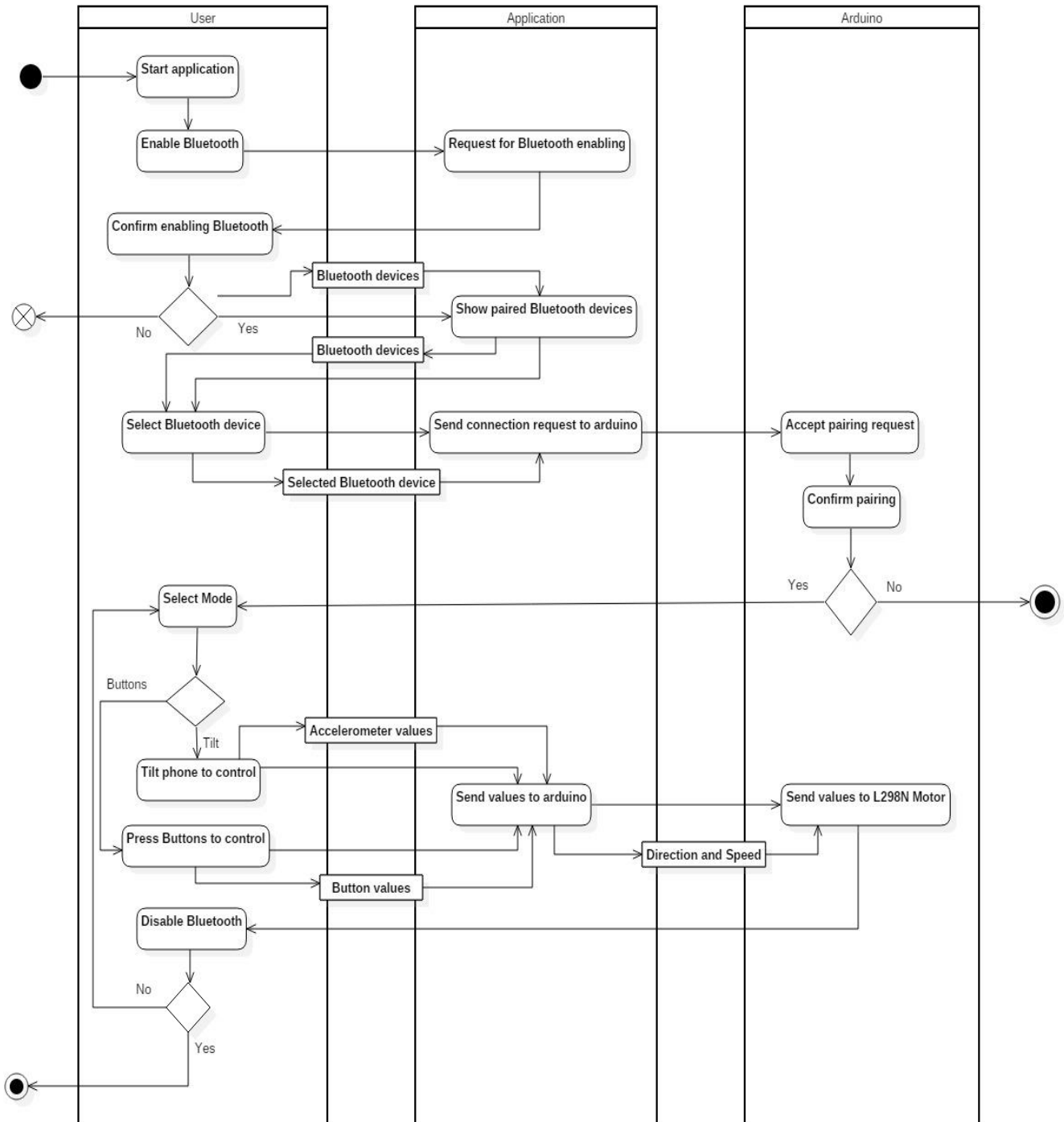


Fig. 15: Activity Diagram (RC Car)

### c. Class Diagram

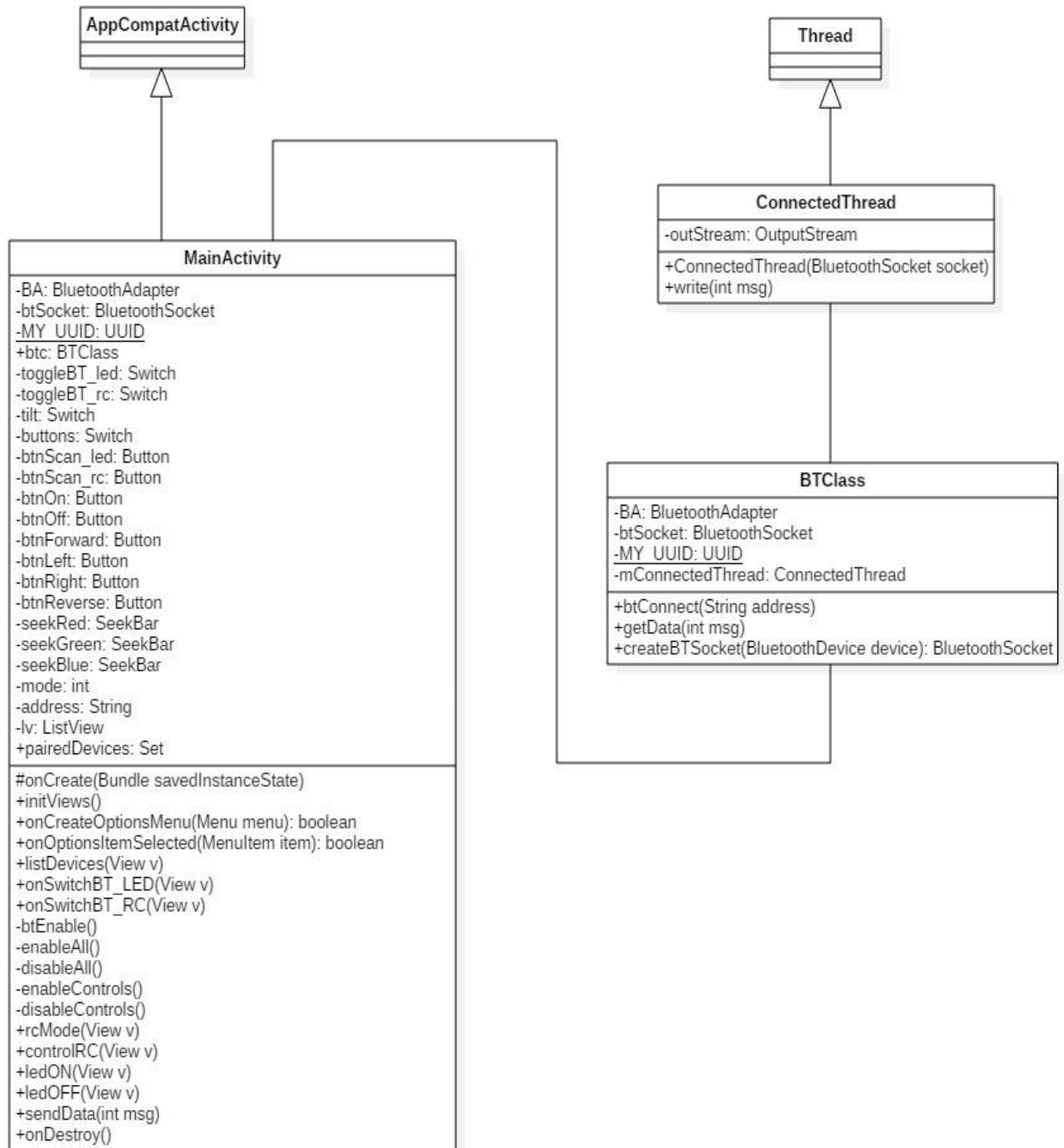


Fig. 16: Class Diagram

#### d. Sequence Diagram

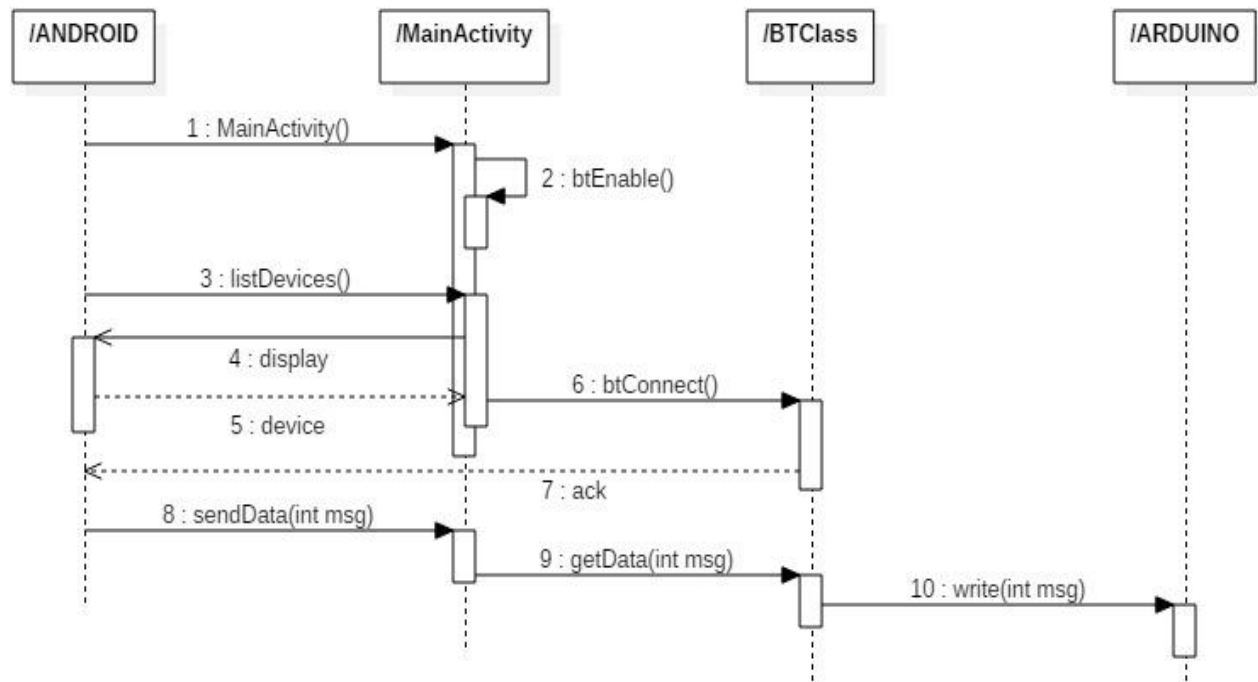


Fig. 17: Sequence Diagram

## **For structured oriented approach**

### **a. Flowchart**

Flowcharts are also called process flowcharts or process flow diagram. A flowchart is a picture of the separate steps of process in sequential order. Elements that might be included are sequences of actions, materials or services entering or leaving process, decisions that must be made, people who become involved, time involved in each step or process measurement.

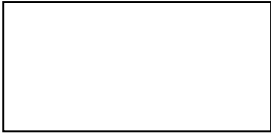

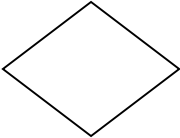
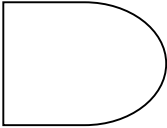

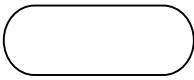
Common alternative names include: flowchart, process flowchart, functional flowchart, process map, process chart, functional process chart, business process model, process model, process flow diagram, work flow diagram, business flow diagram. The terms "flowchart" and "flow chart" are used interchangeably.

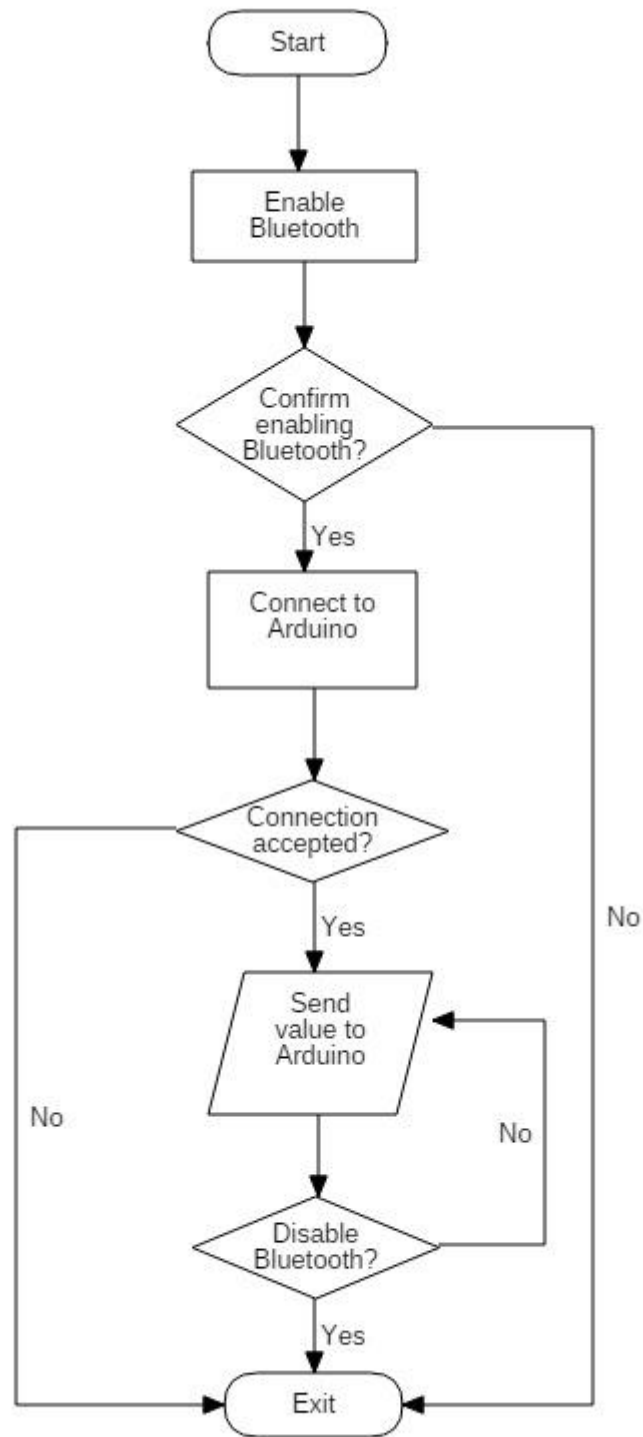
The underlying graph structure of a flow chart is a flow graph, which abstracts away node types, their contents and other ancillary information.

When to use a flowchart:

- a) To develop an understanding of how a process is done.
- b) To study a process for improvement
- c) To communicate to others how a process is done
- d) To document a process
- e) When planning a project

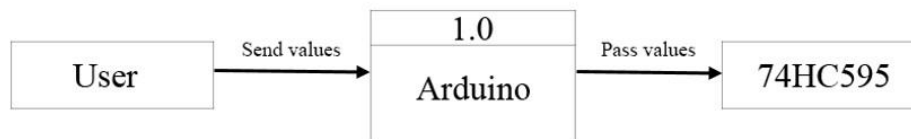
SYMBOLS USED:

	Steps are written in this box
	Direction of flow from one step to another
	Decision based on equation
	Delay or wait
	Input or output
	Start or end

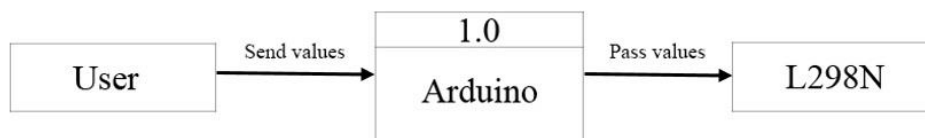


*Fig. 18: Flow Chart*

## b. Context Flow Diagram

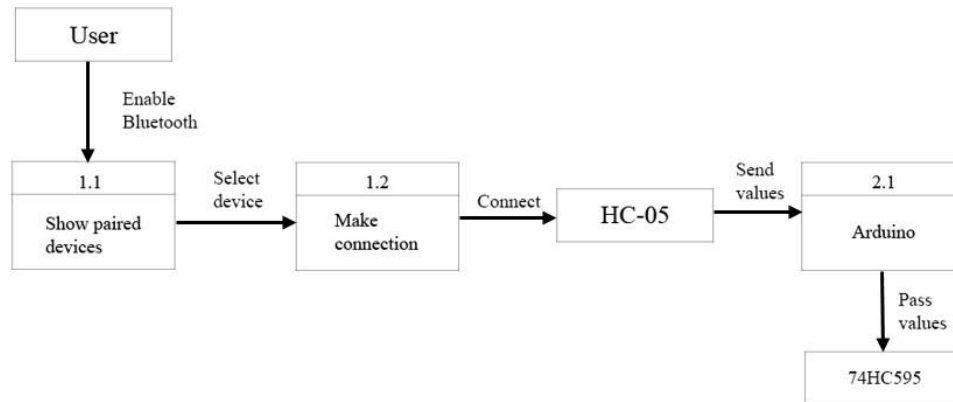


*Fig. 19: Context Flow Diagram (LED)*

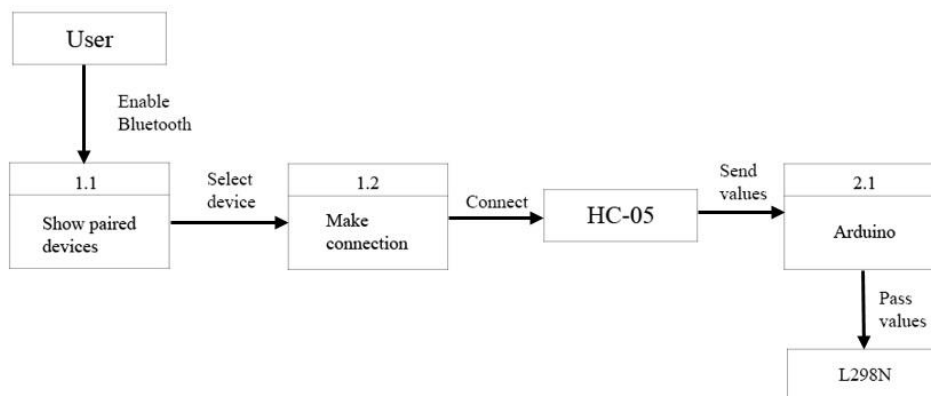


*Fig. 20: Context Flow Diagram (RC Car)*

### c. Data Flow Diagram



*Fig. 21: Data Flow Diagram (LED)*

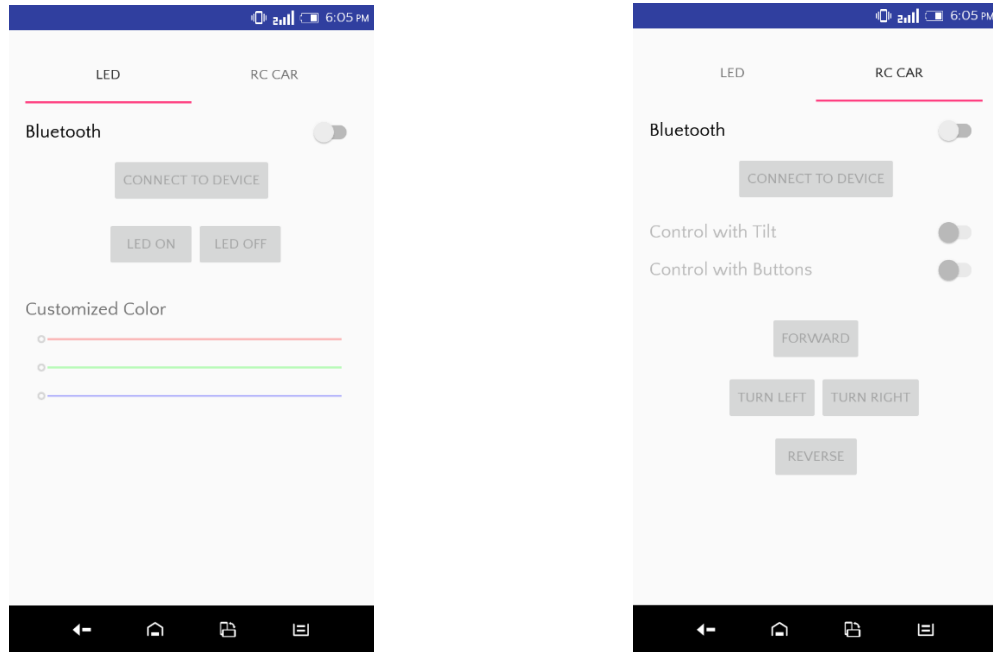


*Fig. 22: Data Flow Diagram (RC Car)*

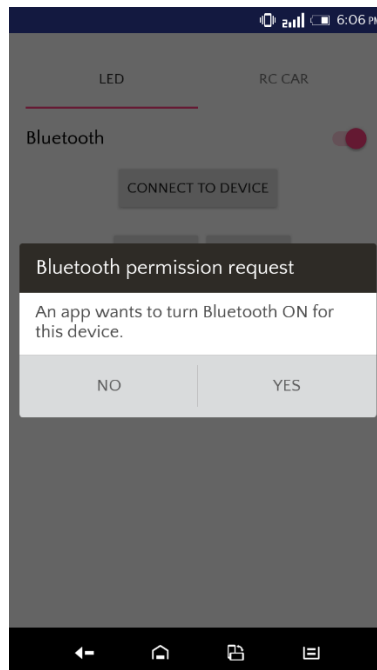


## 11. LAYOUT

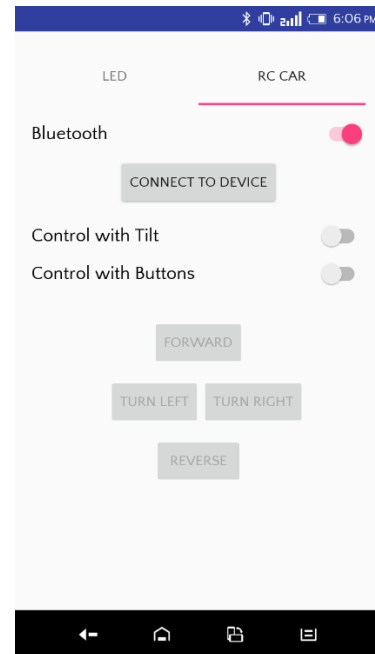
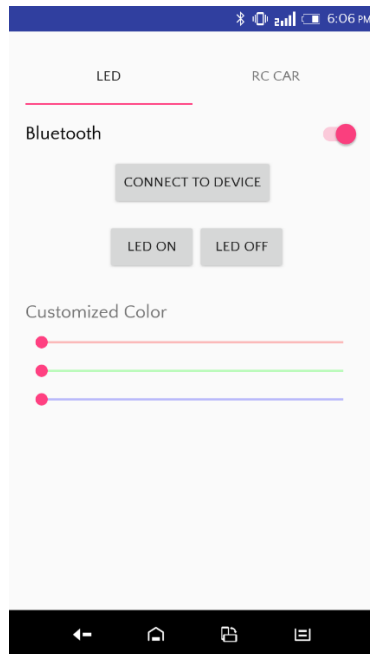
### a. When the application starts



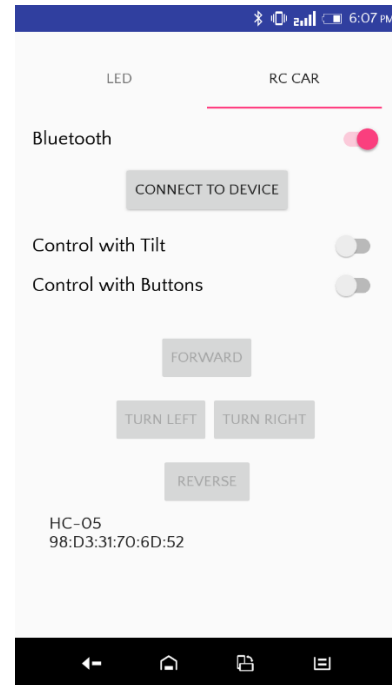
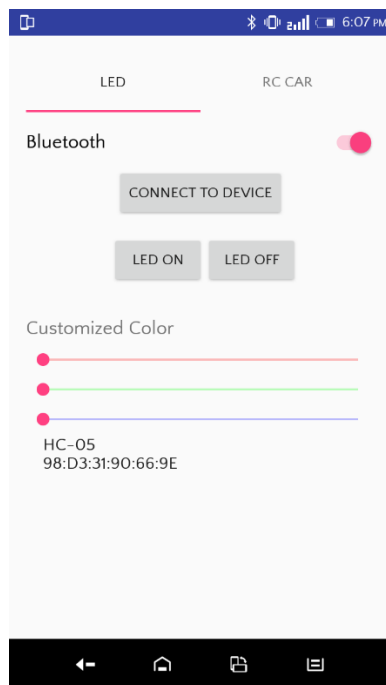
### e. Requesting permission to enable Bluetooth



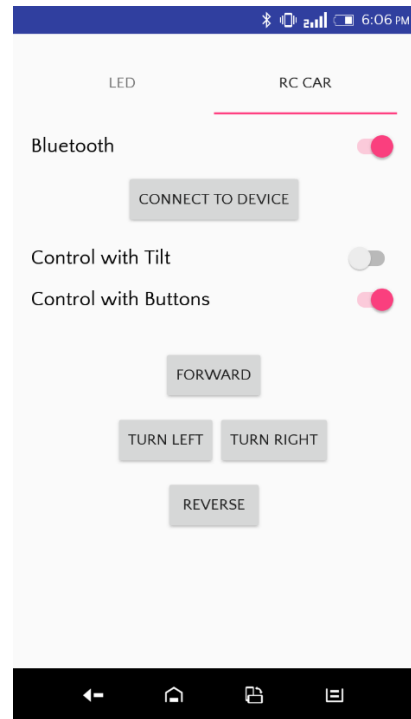
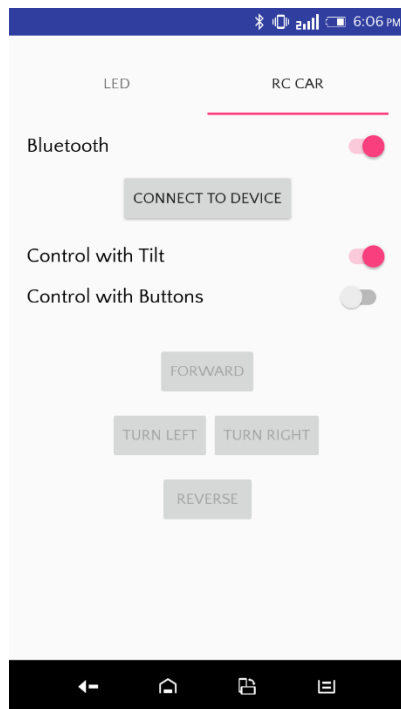
*f. When Bluetooth is enabled*



*g. Paired Device is shown when "Connect to Device" button is pressed*



*h. Two modes to control RC Car*



## **12.TESTING**

### **a. Methodology for Testing**

System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is the process of executing the program with the intent of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied. The ultimate aim is quality assurance. The various tests performed are unit testing, integration testing and user acceptance testing.

All the components were first tested individually and then integrated and tested.

#### **Unit Testing**

The software units in a system are modules and routines that are assembled and integrated to perform a single function. Unit testing focuses first on modules, independently of one another to locate errors. This enables to detect errors in coding and logic that are contained within each module. The testing includes entering data and to make certain if the value to the type and size supported by java. The various controls are tested to ensure that each performs its actions as required.

#### **Integration Testing**

Data can be lost across any interface, one module can have adverse effect on another, sub functions when combined, may not produce the desired major functions. Integration testing is a systematic testing to discover errors associated within the interface. The objective is to take unit tested modules and build a program structure. All the modules are combined and tested as a whole. Here the server module and client module options are integrated and tested.

#### **User Acceptance Testing**

User acceptance testing is the key factor for the success of any system. The system under construction is tested for user acceptance by constantly keeping in touch with the system users at time of developing and making changes whenever required.

## b. Test Report

Test Case No.	Test Description	Test Data	Expected Output	Actual Output	Pass/Fail
1.	Ask for enabling Bluetooth when the Bluetooth switch is turned ON.	Turn ON the Bluetooth switch.	Ask for permission to enable Bluetooth.	Same as expected.	Pass
2.	List of paired devices should be displayed when clicked on “Connect to Device” button.	Click on “Connect to Device” button.	Display list of paired devices.	Same as expected.	Pass
3.	Connection should be established with HC-05 Bluetooth module when HC-05 is selected from the list.	Click on the HC-05 in the paired devices list.	Interval of 2 seconds between the blinking light on HC05 module.	Same as expected.	Pass
4.	LEDs should turn OFF when clicked on “Led OFF” button.	Click on “Led OFF” button.	LEDs turn off.	Same as expected.	Pass
5.	LEDs should turn ON when clicked on “Led ON” button.	Click on “Led ON” button.	LEDs turn on.	Same as expected.	Pass
6.	LEDs should be set to desired color using the three sliders.	Slide the knob on the sliders to desired position.	LEDs are set to desired color.	Same as expected.	Pass

7.	RC Car should move forward when “Forward” button is clicked.	Click on “Forward” button.	RC Car moves forward.	Same as expected.	Pass
8.	RC Car should move backward when “Reverse” button is clicked.	Click on “Reverse” button.	RC Car moves backward.	Same as expected.	Pass
9.	RC Car should turn left when “Turn Left” button is clicked.	Click on “Turn Left” button.	RC Car turns left.	Same as expected.	Pass
10.	RC Car should turn right when “Turn Right” button is clicked.	Click on “Turn Right” button.	RC Car turns right.	Same as expected.	Pass
11.	RC Car should move forward when phone is tilted forward.	Tilt the phone forwards.	RC Car moves forward.	Same as expected.	Pass
12.	RC Car should move backward when phone is tilted backward.	Tilt the phone backward.	RC Car moves backward.	Same as expected.	Pass
13.	RC Car should turn left when phone is tilted left.	Tilt the phone to the left.	RC Car turns left.	Same as expected.	Pass
14.	RC Car should turn right when phone is tilted right.	Tilt the phone to the right.	RC Car turns right.	Same as expected.	Pass
15.	Bluetooth connection should be terminated when application is closed.	Close the application.	HC-05 light blinks faster.	Same as expected.	Pass

## 13. COST ANALYSIS

### Methodology used:

### COCOMO Model:

The **Constructive Cost Model (COCOMO)** is an algorithmic software cost estimation model developed by Barry W. Boehm. The model uses a basic regression formula with parameters that are derived from historical project data and current as well as future project characteristics.

**Basic COCOMO** computes software development effort and cost as a function of program size. Program size is expressed in estimated thousands of source lines of code (SLOC, KLOC).

COCOMO applies to three classes of software projects:

Organic projects - "small" teams with "good" experience working with "less than rigid" requirements

Semi-detached projects - "medium" teams with mixed experience working with a mix of rigid and less than rigid requirements

Embedded projects - developed within a set of "tight" constraints. It is also combination of organic and semi-detached projects. (Hardware, software, operational ...)

The basic COCOMO equations take the form

**Effort Applied (E)** =  $a_b (\text{KLOC})^{b_b}$  [**man-months**]

**Development Time (D)** =  $c_b (\text{Effort Applied})^{d_b}$  [**months**]

**People required (P)** = Effort Applied / Development Time [**count**]

Where, **KLOC** is the estimated number of delivered lines (expressed in thousands) of code for project. The coefficients  $a_b$ ,  $b_b$ ,  $c_b$  and  $d_b$  are given in the following table:

Software project	$a_b$	$b_b$	$c_b$	$d_b$
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

### Cost of project

The total source lines of code are (Android application and Arduino) = 945.

Hence KLOC = 0.945

$$\begin{aligned}
 1. \text{ Effort Applied (E)} &= 3.6(\text{KLOC})^{1.20} \\
 &= 3.6 \times (0.945)^{1.20} \\
 &= \mathbf{3.3 \text{ [man-months]}}
 \end{aligned}$$

$\therefore$  Efforts Applied (E) = **03 Months 09 Days [man-months]**

The above calculated value is the number of months that would be required by one person to complete the project.



$$\begin{aligned}
2. \text{ Development Time (D)} &= 2.5(E)^{0.32} \\
&= 2.5 \times 3.3^{0.32} \\
&= \mathbf{3.6 \text{ [months]}} \\
&= \mathbf{3 \text{ Months 18 Days [months]}}
\end{aligned}$$

$$\therefore \text{ Development Time (D)} = \mathbf{4 \text{ Months [months]}}$$

The above calculated value is the number of months required to complete the project by a specific number of members in a team which is calculated below using the above two values.

$$3. \text{ People required (P)} = \mathbf{2 \text{ [count]}}$$

$$4. \text{ Cost of one programmer: Rs. 5,000}$$

We get the number of people required to complete the project as 2.

If the salary of one programmer is Rs. 5,000

$$5. \text{ Total cost of Programmers: } \mathbf{2 \times 5,000 = \text{Rs. 10,000}}$$

$$6. \text{ Total cost of Project: } \mathbf{10,000 \times 4[\text{months}] = \text{Rs. 40,000}}$$

**7. Cost of Components:**

- a. 2 x Arduino Uno R3: Rs. 850
- b. 1 x Arduino Cable: Rs. 40
- c. 2 x HC-05 Bluetooth Module: Rs. 580
- d. 3 x Breadboard: Rs. 195
- e. 90 x Jumper Wires: Rs. 234
- f. 8 x RGB LEDs: Rs. 68
- g. 3 x 74HC595 Shift Registers: Rs. 37
- h. 1 x L298N Dual H-Bridge Motor: Rs. 230
- i. 2 x DC Motors: Rs. 260
- j. 1 x Servo Motor: Rs. 150
- k. 1 x 12V Battery: Rs. 350

$$\mathbf{\text{Total cost of components: Rs. 2,994}}$$

Hence the Total cost of the project with respect to the Basic COCOMO Model is calculated to be as **Rs. 42,994**

## 14. SYSTEM MAINTENANCE

Maintenance involves the software industry captive, typing up system resources. It means restoring something to its original condition.

It follows conversion to extend that changes are necessary to maintain satisfactory operations relative to changes in the user's environment. Maintenance often includes minor enhancements or corrections to problems that surface in the system's operation.

It is also done on the based on fixing problems reported, changing the interface with other software or hardware enhancing the software. Any system developed should be secured and protected against possible hazards.

Security measures could be provided to prevent unauthorized access of database at various levels. And uninterrupted power supply should be so that power failure or voltage fluctuations will not erase the data in the files. Password protection and simple procedures to prevent unauthorized access are provided to the users.

This system should be maintained by:

- a. Time to time checking of all the hardware modules used.
- b. Enhancing the system by using Wi-Fi instead of Bluetooth for large scale implementation.

Finally, the last maintenance process, also an event which does not occur on a daily basis, is the retirement of a piece of software.

## **15. FUTURE ENHANCEMENTS**

The objective of the proposed system is to provide a cost effective solution involving minimum manual intervention that provides higher level of efficiency for Home Automation.

### **a. Limitations**

- a. If the Bluetooth module stops working the system will remain shut down.
- b. If any hardware stops working the system may fail.
- c. If the Bluetooth adapter of the phone is faulty the system will not work.
- d. If the phone is not available for some particular reasons, the system will remain shut down.

### **b. Future Enhancements**

- a. WiFi can be used for wireless communication to improve the network of connected devices.
- b. Proximity sensors can be used to automate the turning of lights on and off.

## **16. CONCLUSION & REFERENCES**

### **Conclusion**

To conclude, “ADUNIK” would be useful for home owners to easily control devices like LED lights and RC Cars remotely rather than interacting with them manually.

It is said that no system is perfect. Every system has some limitations and flaws.

Taking consideration, the project of “ADUNIK” has been carried out successfully.

This project will help the user to connect with the arduino device and control the devices around them.

There is surely scope for the enhancement of the project by replacing communication with the android via Bluetooth to Wi-Fi.

### **Websites used for reference:**

- a. <https://www.arduino.cc/>
- b. <https://www.instructables.com/>
- c. <https://www.youtube.com/>
- d. <http://www.github.com/>