**CHAPTER 1 INTRODUCTION OF PROJECT**

The project aims at designing a LED based scrolling message display controlled from an android mobile phone. The proposed system makes use of Google’s technology to communicate using an android phone with the LED display board. This project is to develop an embedded system, which is used for instant information display using LED by using android Gmail module. A standard LED display board consists of led lights arranged in 8 rows x 8 columns. These display units are capable of displaying messages of multiple kinds including alphabetic, alphanumeric, numbers etc., in static or scrolling formats.

The main objective of the project using Gmail module we can send message to any distant location and to develop a wireless notice board that display notice in the form of text. Consume less power and easy to operate also notification can be delivered in within second. The voice calling feature can be added with the proposed system as a further enhancement for using the system.

**1.1 MOTIVATION**

LED display system aims at revolutionizing the use of traditional wooden, pin-paper notice boards in educational institutions like schools and colleges. Notice boards are required in almost every commercial building as well as schools and all other educational institutions. The use of traditional notice boards is becoming a problem, as they usually go unnoticed resulting in missed notices. Also, the traditional notice display requires typing, editing, printing and getting authorization done manually which is a really tedious job to do. So, these notice boards clearly need a technology driven upgrade, catering to the modern digitalization needs. This is an IOT based project that focuses on the use of speech to voice input method to display Notices on an LED matrix based Notice board as output. We have projected our ideas to implement Speech to text conversion using Google Assistant and display the resultant output. This saves time and eliminates chances of forged authorization with the help of a secure Google account.

**1.2 BACKGROUND**

Educational institutions still use attendance registers for managing student’s attendance. The teacher has to call out names during the lecture time. Increased number of students causes wastage of reasonable amount of time from the valuable lecture time.

Provides a systematic and effective solution for administrator/lecturer to manage the students’ attendance record.

**1.3 NEED OF THE PROJECT**

LED display system is aimed at the colleges and universities for displaying day-to-day information continuously or at regular intervals during the working hours.

The LED display is easy to expand and it allows the user to add more displays at any time and at any location dependent or requirement.

**CHAPTER 2 LITERATURE SURVEY**

1. **IJSR-CSEIT, 17 May 2017: - Smart rolling LED Display using Arduino and Bluetooth**

By Diptanuprasad Chakraborty, Shubham Yadav, Sona Rathore, Sunil Kumar, Ruchita Agarwal, Pallavi Chandrakar.

This system introducing the overall design of “Smart rolling LED Display using Arduino and Bluetooth” with low cost and user can access multiple applications. If anyone wants to display the message, they can send message through using android Bluetooth by using this project. This project deals with advanced wireless Arduino development board. The main objective of this project is to design a wireless board that displays messages sent from android phone user using Bluetooth. The main controlling device of the whole system is Arduino. Matrix LED Display module, Bluetooth module are interfaced to Arduino and android development tool with APK application. The introducing concept of wireless technology using Bluetooth in the field of communication we can make our communication more efficient and faster, with higher efficiency. We can display the messages with less errors and maintenance. This model can be used very efficiently used in schools, chain restaurants in colleges wherein students and staffs can be informed simultaneously in time. It will used at public transport places like railways, bus station, and airport and also at roadside for traffic control and in emergency situations. It is cost efficient system and user friendly.

1. **IRJET, 3 March 2019- A paper on IOT based digital notice board using Arduino AT Mega 328**

by Pooja Pawar, SuvarnaLangade, MohiniBandgar.

LED display system is aimed at the colleges and universities for displaying day-to-day information continuously or at regular intervals during the working hours. The LED display system mainly consists of a receiver and a display board which can be programmed from an Arduino.

It receives the message through serial port and display the desired information after necessary code conversion. The LED display is easy to expand and it allows the user to add more displays at any time and at any location dependent or requirement. As the technology is changes every day the, display board system is changing from normal display to digital LED display. It saving time and resources and making the information available instantly to the intended person. The system is simple low cost and easy to use that interact with intended user instantly. The system can be used in various applications like banking, schools, restaurants, offices, hospitals, score boards for sports etc

1. **IJEDR, 2014: - Scrolling LED Display using wireless transmission**

By Anuradha Mujumdar, Vaishali Niranjane, Deepika Sange

The main objective of the project is display of important public messages to the masses. without much manual efforts Important notices can be displayed within a short span of time and these notices can be upgraded within minutes. Thus, we can achieve flexibility in notifications. The project provides us with easy and user-friendly interface to display any message on the LED matrix. There is no need to manually write the message like in conventional display boards thus it is more popular. Moreover, the LED display is catchier and can be seen from a distance compared to the conventional display boards.

**CHAPTER 3 SCOPE OF PROJECT**

* 1. **PROJECT OBJECTIVES**

The main objective of this project using Google Assistant is that we can send speech message to any distant location and thus develop a wireless notice board that display notice in the form of text. The objective is to consume less power and make it easy to operate, as also notifications can be delivered within seconds. This project will display important public messages to the masses without much manual efforts. Important notices can be displayed within a short span of time and these notices can be upgraded within minutes.

* 1. **PROJECT FEATURES**
* Saves time
* Easy to implement as voice input is required which will convert into text display.
* Good security assisted by Google services (Gmail)
* Less power consumption
  1. **PROJECT COST**

COCOMO Model

Boehm proposed COCOMO (Constructive Cost Estimation Model) in 1981.COCOMO is one of the most generally used software estimation models in the world. COCOMO predicts the efforts and schedule of a software product based on the size of the software.

The necessary steps in this model are:

1. Get an initial estimate of the development effort from evaluation of thousands of delivered lines of source code (KDLOC).
2. Determine a set of 15 multiplying factors from various attributes of the project.
3. Calculate the effort estimate by multiplying the initial estimate with all the multiplying factors i.e., multiply the values in step1 and step2.

The initial estimate (also called nominal estimate) is determined by an equation of the form used in the static single variable models, using KDLOC as the measure of the size.

Embedded: A development project is treated to be of an embedded type, if the software being developed is strongly coupled to complex hardware, or if the stringent regulations on the operational method exist. For Example: ATM, Air Traffic control.

For three product categories, Bohem provides a different set of expression to predict effort (in a unit of person month)and development time from the size of estimation in KLOC(Kilo Line of code) efforts estimation takes into account the productivity loss due to holidays, weekly off, coffee breaks, etc. Basic COCOMO Model: The basic COCOMO model provide an accurate size of the project parameters. The following expressions give the basic COCOMO estimation model:

**Effort=a1\*(KLOC) a2 PM**  
                **Tdev=b1\*(efforts)b2 Months**

Where KLOC is the estimated size of the software product indicate in Kilo Lines of Code, a1,a2,b1,b2 are constants for each group of software products,

Tdev is the estimated time to develop the software, expressed in months, Effort is the total effort required to develop the software product, expressed in person months (PMs).In our system,

A1=3.6

A2=1.20

B1=2.5

B2=0.32

Effort= a1\*(KLOC) a2 PM  
 =3.6\*(380)1.2  =4487.89 PM

Tdev=b1\*(efforts)b2 Months

=2.5 \*(4487.89)0.32

=36.86 Months

Cost:

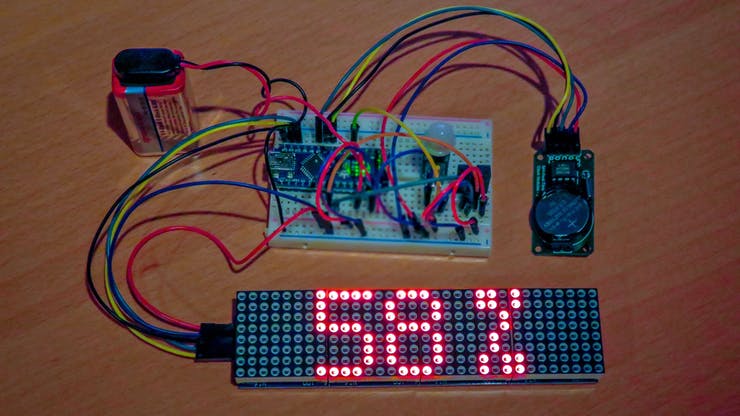
|  |  |
| --- | --- |
| Project Expense Category | Cost (In Rupees) |
| Man Power | |
| Order Quantity | 1.0 |
| Production Time(hrs.) | 200 |
| Labor Rate (Rs/Hr.) | 200 |
| Number of Workers | 4 |
| Total | 160,000 |
| Software cost | |
| Android Studio | 0 |
| Python | 0 |
| Tensar Flow | 0 |
| Open Weather API | 0 |
| Total | 0 |
| Total Cost | 160,000 |

Table .1: Cost Estimation Table

**CHAPTER 4 PROPOSED METHODOLOGY**

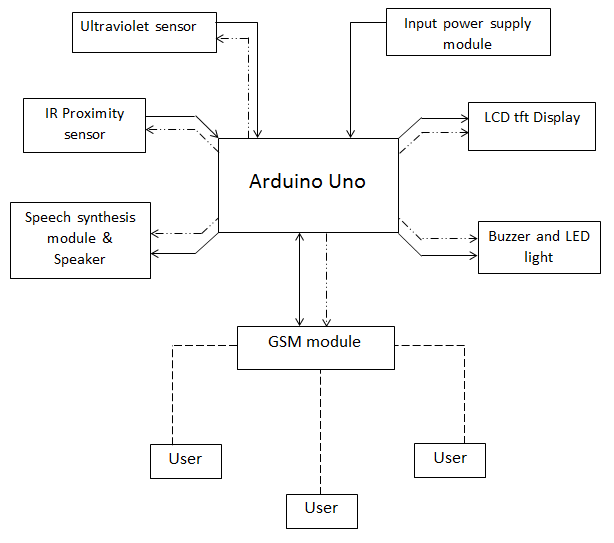
The implementation of the project includes the following steps:

* Discussion regarding objectives of the project according to the daily requirements.
* Visualisation of the practical applications of the smart activity scheduler in current scenario with the help of you tube videos and research papers.
* Modifications in the present routine pattern to avoid imbalance in achieving daily schedule.
* Discussion with the respected guide, research papers and reference books on feasibility on every proposed modification and market survey for resource availability.
* Market survey of the appropriate electronic equipment
* Proposing a prototype block diagram for the system.
* Designing of the interfacing of buzzer, sensor and voice synthesis module with the controller Arduino.
* Purchasing of such equipment which make the system energy efficient.
* Development of software platform with use of computer languages for accessing the real time data.
* Assembling the components together and wiring all the components according to the design.
* Testing the prototype.



**4.1 SYSTEM ARCHITECHTURE**

Presenting data is an important part of consuming it, since the data can be efficiently used if it is represented properly. Our system is based on the concept of displaying data in a manner to be understood by all and to maintain flow of data. Smart display system will gather data from users (in our case an administrator). This data is then processed by Arduino and feed to the display in a predefined format. The user can feed important schedules or upcoming events so the controller (Arduino) which further will notify the user accordingly. These event or schedules will be displayed on a display along with an alert. Our system also consists of speech synthesizer module which intake raw speech data processes it and passes it on to the Arduino. These commands then can be processed into some particular functions.



**Fig: System Architecture**

**4.2 DETAILED DESCRIPTION OF MODULES**

**SOFTWARE**

**Arduino IDE**

The [Arduino](https://en.wikipedia.org/wiki/Arduino) integrated development environment ([IDE](https://en.wikipedia.org/wiki/Integrated_development_environment)) is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) application (for [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS), [Linux](https://en.wikipedia.org/wiki/Linux)) that is written in the programming language [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

## The source code for the IDE is released under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), version 2. The Arduino IDE supports the languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) using special rules of code structuring. The Arduino IDE supplies a [software library](https://en.wikipedia.org/wiki/Software_library) from the [Wiring](https://en.wikipedia.org/wiki/Wiring_(development_platform)) project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable [cyclic executive](https://en.wikipedia.org/wiki/Cyclic_executive) program with the [GNU toolchain](https://en.wikipedia.org/wiki/GNU_toolchain), also included with the IDE distribution.

## [Arduino](http://arduino.cc) is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a [microcontroller](http://en.wikipedia.org/wiki/Microcontroller)) and a piece of [software](http://arduino.cc/en/Main/Software), or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

[](https://cdn.sparkfun.com/assets/9/1/e/4/8/515b4656ce395f8a38000000.png)

This is an Arduino Uno

The Uno is one of the more popular boards in the Arduino family and a great choice for beginners. We'll talk about what's on it and what it can do later in the tutorial.

Believe it or not, those 10 lines of code are all you need to blink the on-board LED on your Arduino. The code might not make perfect sense right now, but, after reading this tutorial and the many more Arduino tutorials waiting for you on our site, we'll get you up to speed in no time!

The Arduino IDE employs the program *argued* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

**HARDWARE COMPONENTS**

[](https://cdn.sparkfun.com/assets/9/1/e/4/8/515b4656ce395f8a38000000.png)

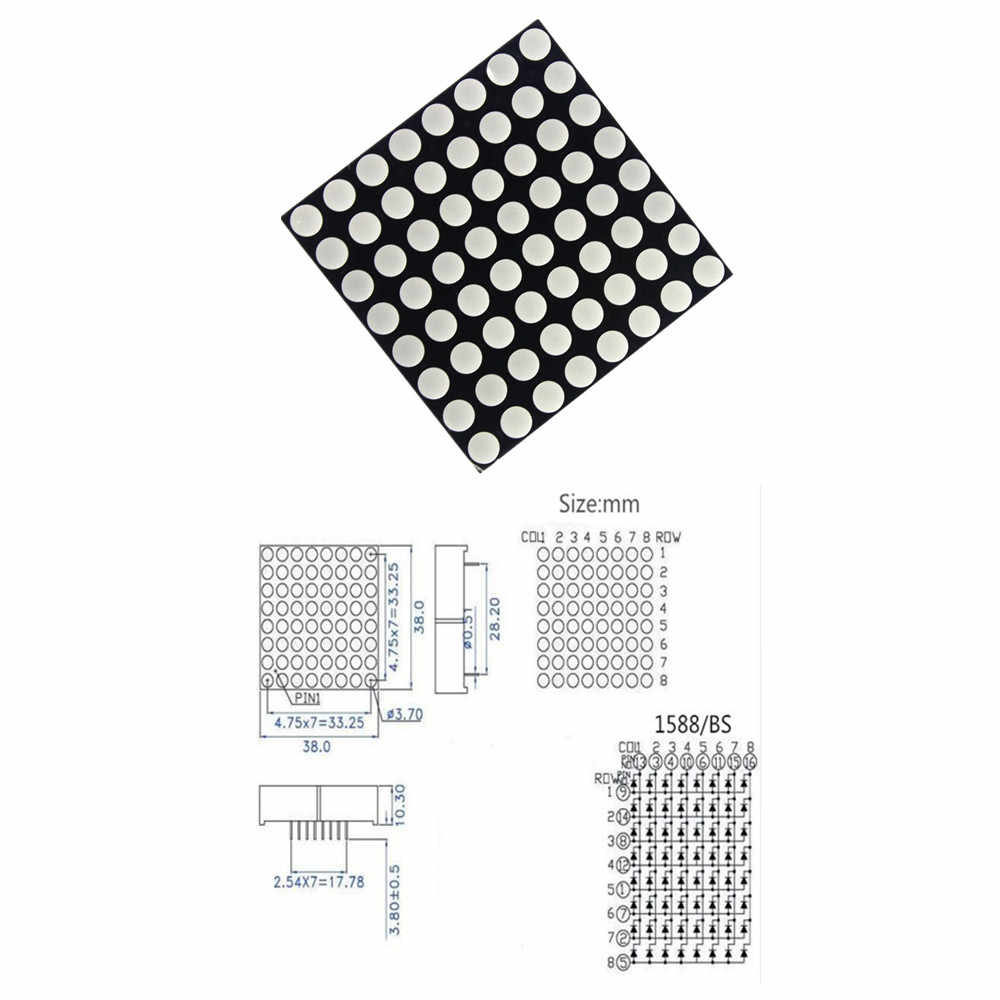
**Pin Descriptions**

* **VCC:** Digital supply voltage.
* **GND:** Ground.
* **Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2** Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit. Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier. If the Internal Calibrated RC Oscillator is used as chip clock source, PB7.6 is used as TOSC2. 1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set. The various special features of Port B are elaborated.
* **Port C (PC5:0)** Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5.0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.
* **PC6/RESET** If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C. If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. The minimum pulse length is given in. Shorter pulses are not guaranteed to generate a Reset. The various special features of Port C are elaborated.
* **Port D (PD7:0)** Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.
* **AVCC** AVCC is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC6..4 use digital supply voltage, VCC.
* **AREF** AREF is the analog reference pin for the A/D Converter.
* **ADC7:6 (TQFP and QFN/MLF Package Only)** In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

**POWER SUPPLY**

The present chapter introduces the operation of power supply circuits built using filters, rectifiers and then voltage regulators. Starting with an ac voltage, then filtering to a dc voltage is obtained by rectifying the ac voltage, then filtering to a dc level and finally, regulating to obtain a desired fixed dc voltage. The regulation is usually obtained from an IC voltage regulator unit, which takes a dc voltage and provides a somewhat lower dc voltage, which remains the same even if the input dc varies, or the output load connected to the dc voltage changes.

**ED DOT MATRIX**

**Fig: LED Dot Matrix**

**LED Matrix**

LED Matrix's are great fun, you can create funky patterns, scroll messages, or create something entirely bizarre. Sadly, controlling one is a tad complicated. But once mastered is easily repeatable.

.: A quick refresher on LED control can be found here tinyurl.com/cmn5nh:

**Matrix Wiring**

Each matrix has 128 LEDs (64 Red & 64 Green) however there is noticeably not 256 leads. Instead the LEDs are wired into a matrix. This matrix has the LED's anodes connected across rows (5 pins) then the red and green LED's cathodes attached across columns (5 pins each). To light an LED, connect its rows anode to +5volts, and through a resistor, it's columns cathode to ground. (you can try this without a micro-controller)

**CHAPTER 5 DETAILS DESIGN AND WORKING PROCESSES**

**5.1 HARDWARE SOFTWARE REQUIREMENTS**

|  |  |
| --- | --- |
| Software | Arduino IDE |
| Hardware | Pins |
| Android smartphone |
| Power Supply |
| LED dot matrix |
| Matrix wiring |

**Table: Hardware software requirements**

**SOFTWARE**

**Arduino IDE**

The [Arduino](https://en.wikipedia.org/wiki/Arduino) integrated development environment ([IDE](https://en.wikipedia.org/wiki/Integrated_development_environment)) is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) application (for [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS), [Linux](https://en.wikipedia.org/wiki/Linux)) that is written in the programming language [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards. The source code for the IDE is released

under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), version 2. [Arduino](http://arduino.cc) is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a [microcontroller](http://en.wikipedia.org/wiki/Microcontroller)) and a piece of [software](http://arduino.cc/en/Main/Software), or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Believe it or not, those 10 lines of code are all you need to blink the on-board LED on your Arduino. The code might not make perfect sense right now, but, after reading this tutorial and the many more Arduino tutorials waiting for you on our site, we'll get you up to speed in no time! The Arduino IDE employs the program *argued* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

**HARDWARE COMPONENTS**

**Pin Descriptions**

* **VCC:** Digital supply voltage.
* **GND:** Ground.
* **Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2**
* **Port C (PC5:0)**
* **PC6**
* **Port D (PD7:0**
* **AREF**
* **ADC7:6**

**POWER SUPPLY**

The present chapter introduces the operation of power supply circuits built using filters, rectifiers and then voltage regulators. The regulation is usually obtained from an IC voltage regulator unit, which takes a dc voltage and provides a somewhat lower dc voltage, which

remains the same even if the input dc varies, or the output load connected to the dc voltage changes.

**LED Matrix**

LED Matrix's are great fun; you can create funky patterns, scroll messages, or create something entirely bizarre.

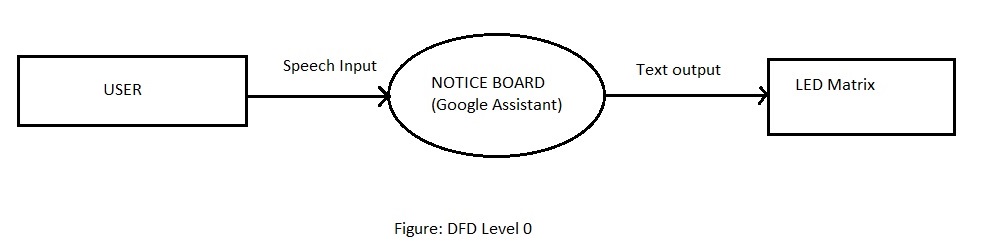
**Matrix Wiring**

Each matrix has 128 LEDs (64 Red & 64 Green) however there is noticeably not 256 leads. Instead the LEDs are wired into a matrix.

**5.2 DFDs (0,1,2)**

DFD 0 Level

A level 0 data flow diagram (DFD), also known as a context diagram, shows a data system as a whole and emphasizes the way it interacts with external entities. This DFD level 0 example shows how such a system might function within a typical retail business.



DFD 1 level

A level 1 data flow diagram (DFD) is more detailed than a level 0 DFD but not as detailed as a level 2 DFD. It breaks down the main processes into subprocesses that can then be analysed and improved on a more intimate level.

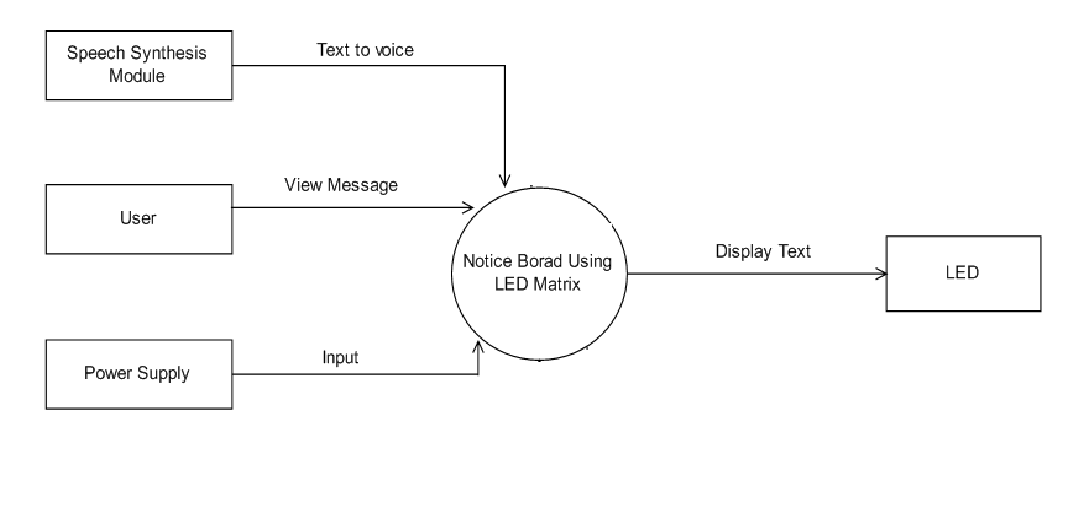


Figure: DFD Level 1

DFD 2 level

A level 2 data flow diagram (DFD) offers a more detailed look at the processes that make up an information system than a level 1 DFD does. It can be used to plan or record the specific makeup of a systeFigure: DFD Level 2

**5.3 USE CASE DIAGRAM**

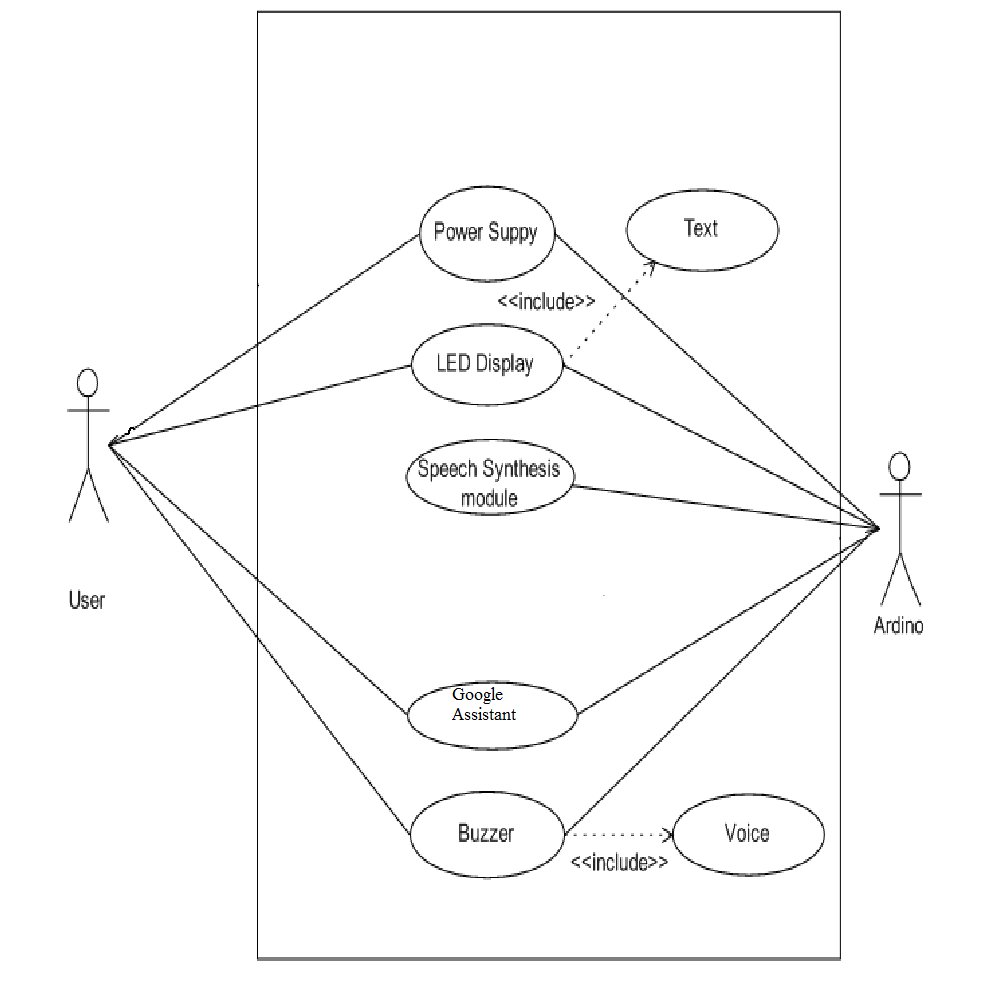


Fig: Use case diagram

**5.4 ACTIVITY DIAGRAM**

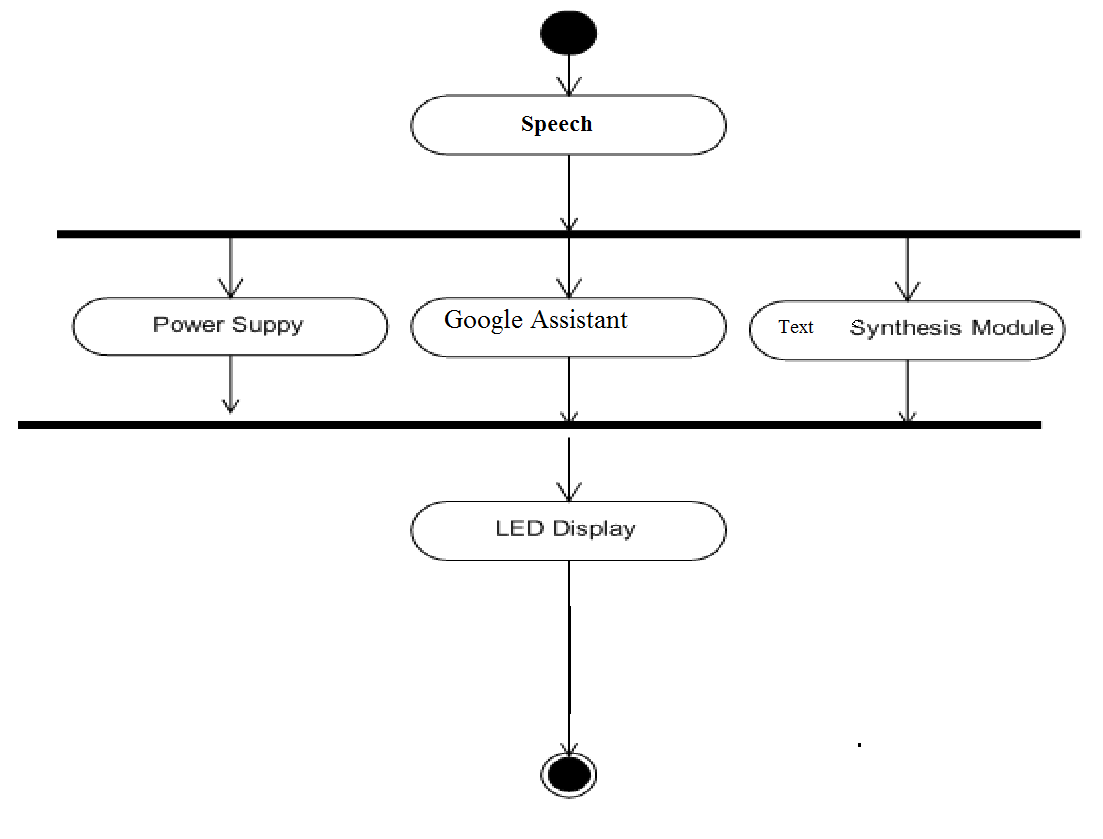
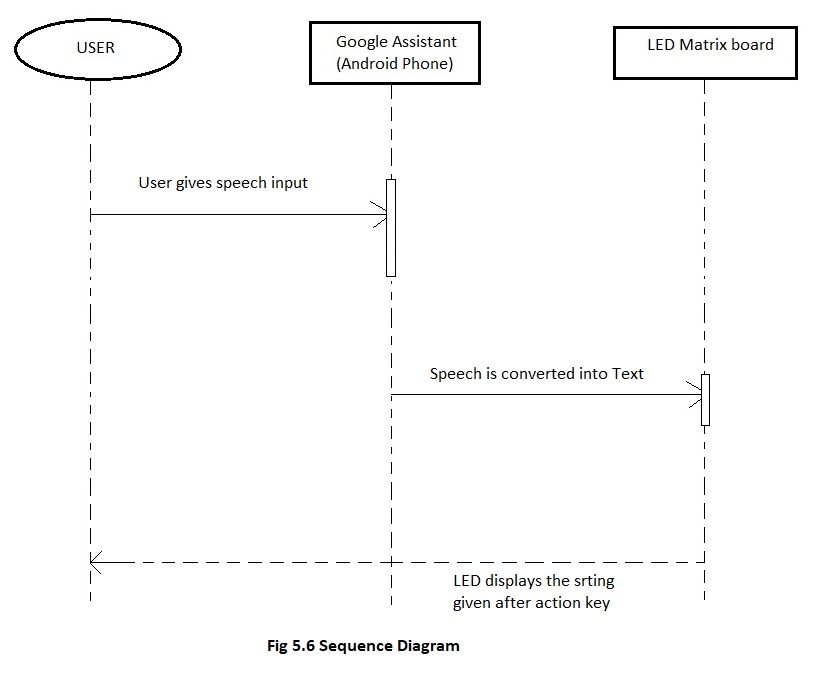


Fig: Activity Diagram

**5.5 SEQUENCE DIAGRAM**

****

**5.6 TIMELINE CHART**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Month/Task | July ’19 | Aug ’19 | Sept ’19 | Oct ’19 | Nov ’19 | Dec ’19 | Jan ’20 | Feb ’20 | Mar ’20 |
|  |  |  |  |  |  |  |  |  |  |
| Requirement Gathering |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Specification |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Literature Survey |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Design Analysis |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Implementation |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Deployment |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Documentation |  |  |  |  |  |  |  |  |  |

**Table 5.1 TIMELINE CHART (GANTT CHART)**

**5.7 IMPLEMENTATION**

**5.7.1 SAMPLE CODE**

#include <MD\_MAX72xx.h>

#include <SPI.h>

#include <ESP8266WiFi.h>

#include <ESP8266WiFiMulti.h>

#include <ESP8266HTTPClient.h>

#include <Adafruit\_MQTT.h>

#include <Adafruit\_MQTT\_Client.h>

#define HARDWARE\_TYPE MD\_MAX72XX::FC16\_HW

#define SCROLL\_DELAY 70

char\* str;

String payload;

uint32\_t present;

bool first\_time;

uint16\_t scrollDelay;

#define CHAR\_SPACING 1

#define BUF\_SIZE 96

char curMessage[BUF\_SIZE];

char newMessage[BUF\_SIZE];

bool newMessageAvailable = false;

ESP8266WiFiMulti WiFiMulti;

#define MAX\_DEVICES 4

#define CLK\_PIN D5 // or SCK

#define DATA\_PIN D7 // or MOSI

#define CS\_PIN D8 // or SS

#define AIO\_SERVER "io.adafruit.com"

#define AIO\_SERVERPORT 1883 // use 8883 for SSL

#define AIO\_USERNAME "Maddy\_1234"

#define AIO\_KEY "aio\_zHYC65IrGbaYpGDkk6FJaXco2CKk" // Madiha Feed

WiFiClient client;

Adafruit\_MQTT\_Client mqtt(&client, AIO\_SERVER, AIO\_SERVERPORT, AIO\_USERNAME, AIO\_KEY);

Adafruit\_MQTT\_Subscribe message = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/led-matrix");

void MQTT\_connect();

MD\_MAX72XX mx = MD\_MAX72XX(HARDWARE\_TYPE, CS\_PIN, MAX\_DEVICES);

uint8\_t scrollDataSource(uint8\_t dev, MD\_MAX72XX::transformType\_t t)

{

static char \*p = curMessage;

static uint8\_t state = 0;

static uint8\_t curLen, showLen;

static uint8\_t cBuf[8];

uint8\_t colData;

switch

{

case 0:

showLen = mx.getChar(\*p++, sizeof(cBuf) / sizeof(cBuf[0]), cBuf);

curLen = 0;

state++;

if (\*p == '\0')

{

p = curMessage;

if (newMessageAvailable)

{

strcpy(curMessage, str);

newMessageAvailable = false;

}

}

case 1:

colData = cBuf[curLen++];

if (curLen == showLen)

showLen = CHAR\_SPACING;

curLen = 0;

state = 2;

}

break;

case 2:

colData = 0;

curLen++;

if (curLen == showLen)

state = 0;

break;

default:

state = 0;

}

return (colData);

}

void scrollText(void)

{

static uint32\_t prevTime = 0;

if (millis() - prevTime >= scrollDelay)

{

mx.transform(MD\_MAX72XX::TSL);

prevTime = millis();

}

}

void no\_connection(void)

{

newMessageAvailable = 1;

strcpy(curMessage, "No Internet! ");

scrollText();

}

void setup()

{

mx.begin();

mx.setShiftDataInCallback(scrollDataSource);

scrollDelay = SCROLL\_DELAY;

strcpy(curMessage, "Hello! ");

newMessage[0] = '\0';

Serial.begin(57600);

Serial.print("\n[MD\_MAX72XX Message Display]\nType a message for the scrolling display\nEnd message line with a newline");

// Serial.setDebugOutput(true);

Serial.println();

Serial.println();

Serial.println();

for (uint8\_t t = 4; t > 0; t--) {

Serial.printf("[SETUP] WAIT %d...\n", t);

Serial.flush();

delay(1000);

}

WiFi.mode(WIFI\_STA);

WiFiMulti.addAP("ATUL","heimdall@1@2@3");

Serial.println("Connecting");

newMessageAvailable = 1;

present = millis();

first\_time = 1;

// Setup MQTT subscription for onoff feed.

mqtt.subscribe(&message);

str = "Ask Assistant ";

}

void loop()

{

while (WiFiMulti.run() != WL\_CONNECTED) {

Serial.println("WiFi not connected!");

delay(1000);

}

MQTT\_connect();

Adafruit\_MQTT\_Subscribe \*subscription;

while ((subscription = mqtt.readSubscription(1))) {

if (subscription == &message) {

payload ="";

Serial.print(F("Got: "));

Serial.println((char \*)message.lastread);

str = (char\*)message.lastread;

payload = (String) str;

payload += " ";

str = &payload[0];

newMessageAvailable = 1;

}

}

scrollText();

}

void MQTT\_connect()

{

int8\_t ret;

if (mqtt.connected())

{

return;

}

Serial.print("Connecting to MQTT... ");

uint8\_t retries = 3;

while ((ret = mqtt.connect()) != 0)

{

Serial.println(mqtt.connectErrorString(ret));

Serial.println("Retrying MQTT connection in 5 seconds...");

mqtt.disconnect();

delay(5000);

retries--;

if (retries == 0) {

while (1);

}

}

Serial.println("MQTT Connected!");

}

**5.7.2 TYPES OF TESTING**

**TESTING**

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest.

UNIT TESTING

Unit testing is a software development process in which the smallest testable parts of an application called units, are individually and independently inspected for proper operation. Unit testing is the testing of an individual unit or group of relates units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing excepted output given input.

INTEGRATION TESTING

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing black box testing.

FUNCTION TESTING

Functional testing is the testing to ensure that the species functional-it required in the system requirements works. It falls under the class of black box testing.

SYSTEM TESTING

System testing is the testing to ensure that by putting the software in die rent environments (e.g., Operating Systems) it still works. System testing is done with full system implantation and environment. It falls under the class of black box testing.

**USABILITY TESTING**

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how pro anciently and the client perform? How pleasing is it to use its design.

**STRESS TESTING**

Stress testing helps in finding out the product capability to handle a certain n load and when the system will break down due to overload of records. Loading of record can be steadily increased to see when the system will fail.

**LOAD TESTING**

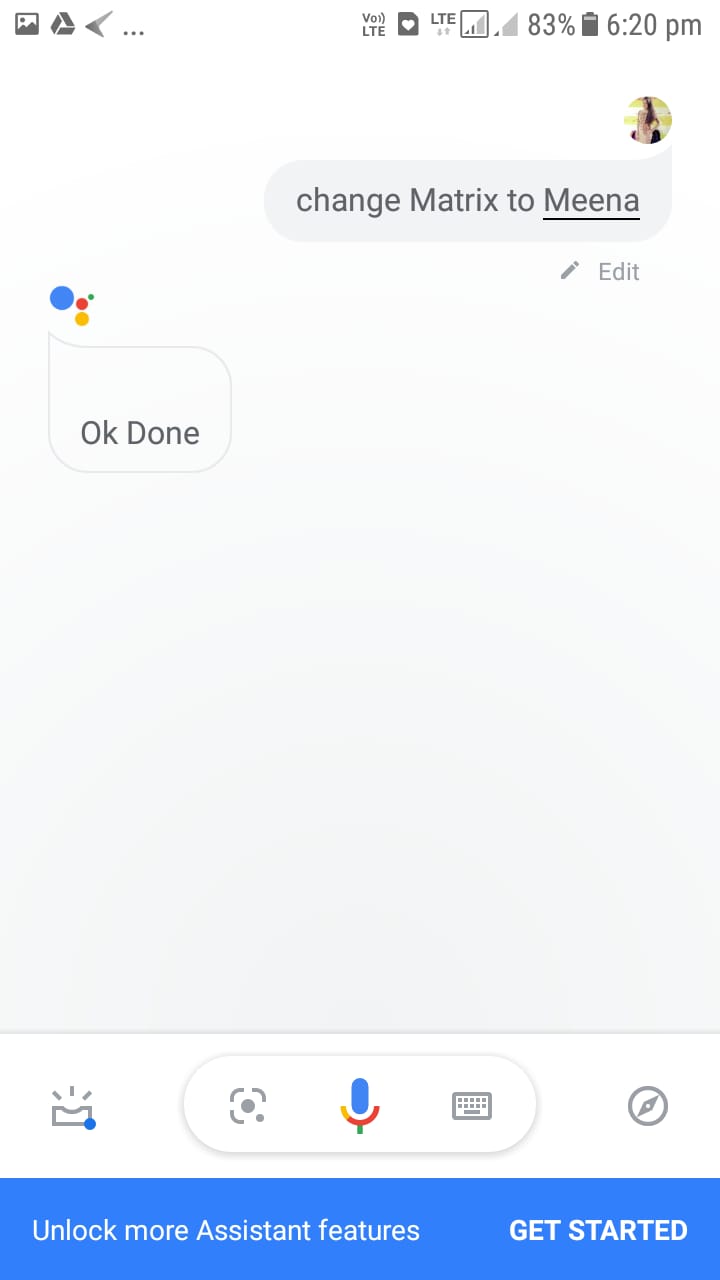
Load testing is a type of non-functional testing. A load test is type of software testing which is conducted to understand the behaviour of the application under a specific expected load. Load testing is performed to determine a system's behaviour under both normal and at peak conditions.

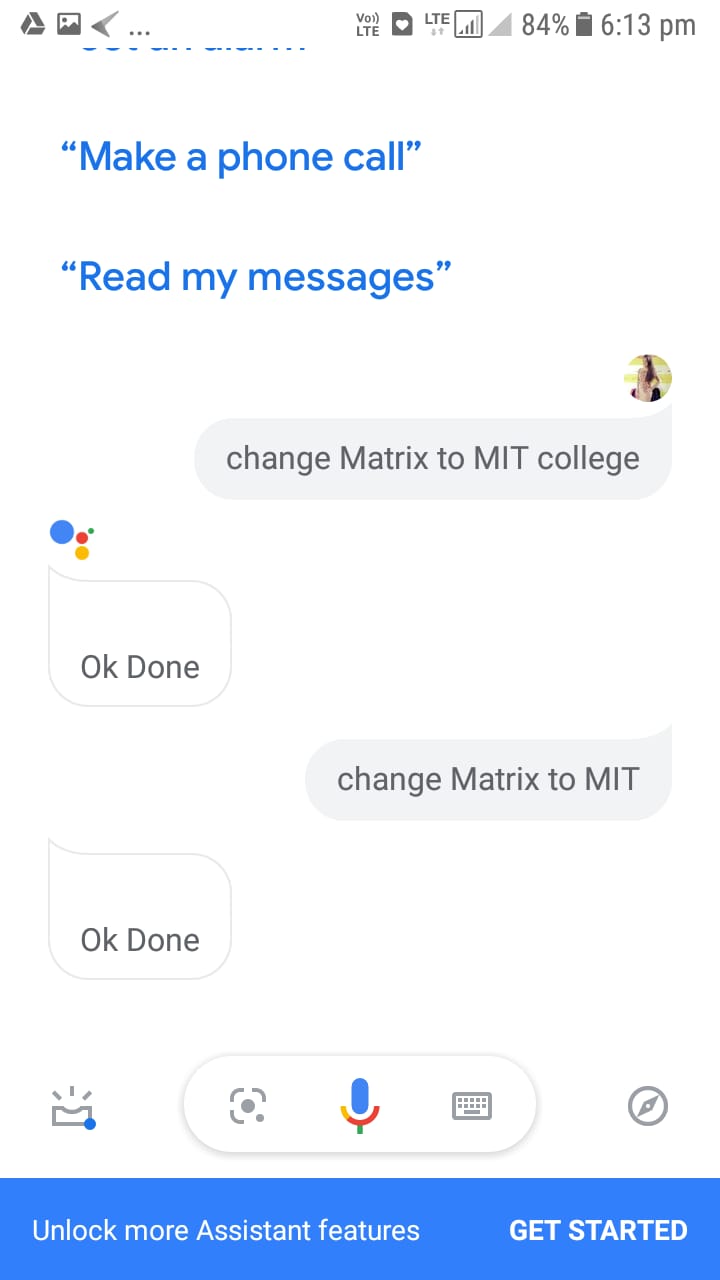
**5.7.3 TEST CASE**

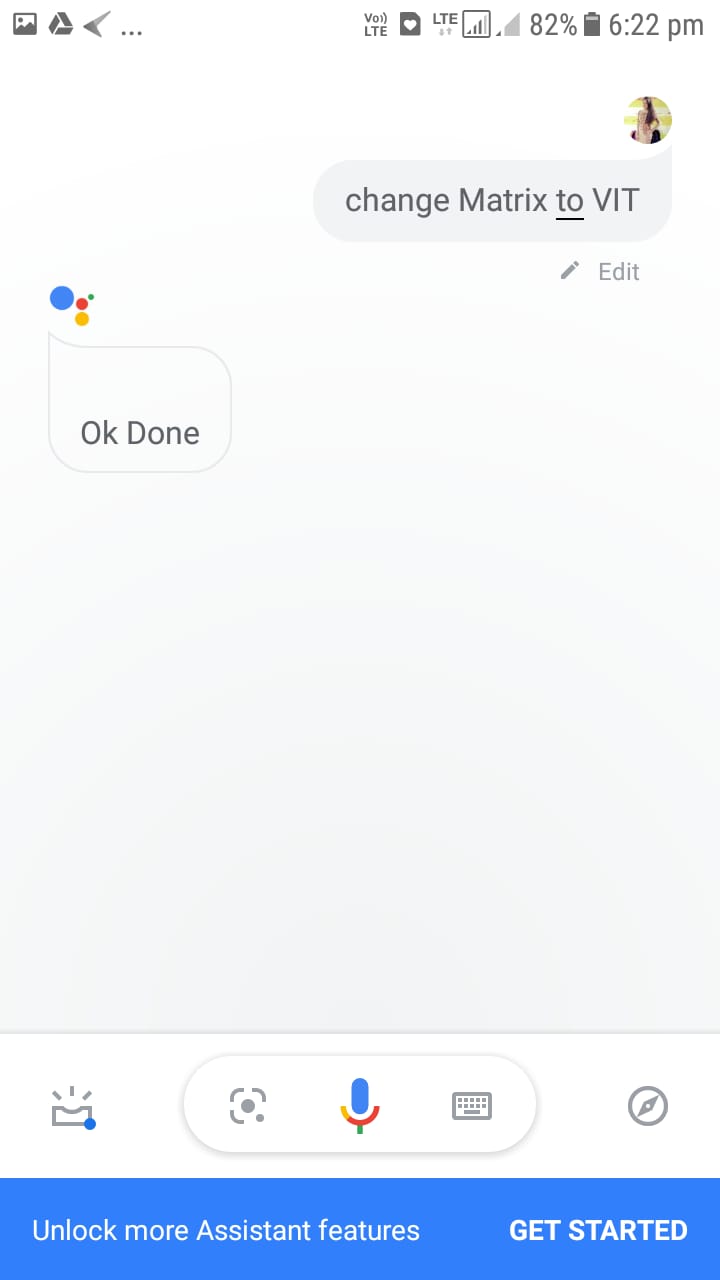
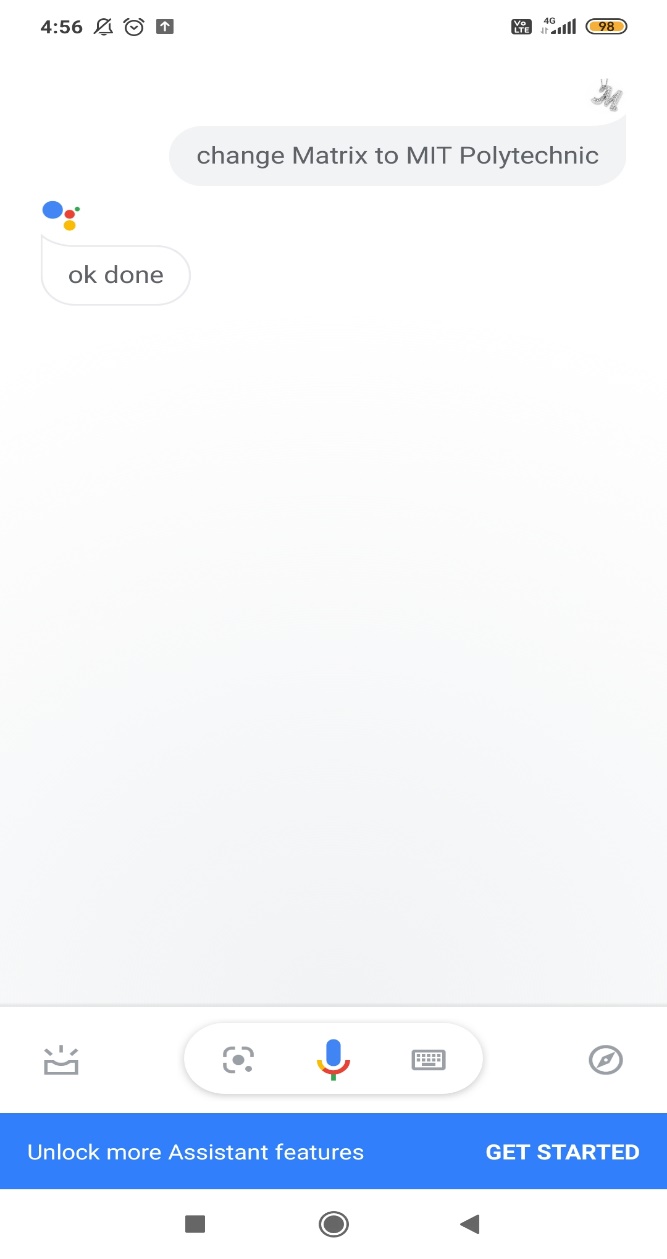
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status (Pass/ Fail)** |
| **1** | Set up Hotspot connection | Hotspot name & password | LED matrix should be able to access the internet. | Internet access enabled. | Pass |
| **2** | Compile code | Run | Code should compile without errors | Code compiled without errors | Pass |
| **3** | Give speech input through Google Assistant of Admin’s account. | “Change matrix to \_\_\_\_”  (Variable display inputs) | LED matrix should display the string said after the action key. | LED matrix displays the string said after the action key. | Pass |

**CHAPTER 6 Results and applicatioN**

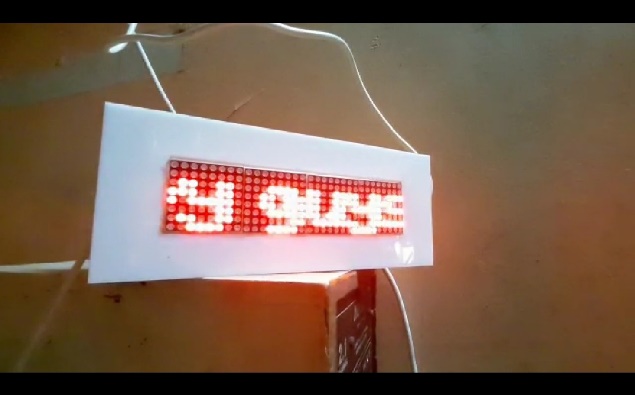
* 1. **SCREENSHOTS**

****

****

****

* 1. **RESULTS**









**6.3 APPLICATIONS**

* Notice Board.
* School and Colleges.
* Railway Station.
* Advertisements

**CHAPTER 7 ­­­­­­­­­­­­­­­­CONCLUSION AND FUTURE SCOPE**

**Conclusion:**

This is a Notice that uses LED Matrix Display. In this project we are going to develop a display, to which speech input will be provided, using Google Assistant. The speech input will be converted to text output using Google’s speech-to-text feature & subsequently will be viewed on the LED display. Our goal with this project is to revolutionize, rather digitalize the traditional use of Paper-Pin notice boards in schools & colleges, etc.

**Future Scope:**

This project is currently developed for digitalizing the traditional use of Paper-Pin notice boards in schools & colleges, offices, educational institutions, etc., but in the future more features can be added to make the board wireless and thus more feasible to use. With newer technologies being developed we may be able to these boards more cos-effective and portable.

We can make modifications to add one or more local languages according to the specific regions as also emoticons can be added. According to a person or organizations need the size of the LED matrix can be varied form small to huge screens for optimum space use and thus the font size also can be varied. Another variation can be added where the notice is forwarded to all contacts stored in the database via a text message using GSM module.sss

**REFERENCES & BIBLIOGRAPHY**

1. IJSR-CSEIT, 17 May 2017: - **Smart rolling LED Display using Arduino and Bluetooth by** Diptanuprasad Chakraborty, Shubham Yadav, Sonal Rathore, Sunil Kumar, Ruchita Agarwal, Pallavi Chandrakar.
2. IRJET, 3 March 2019- **A paper on IOT based digital notice board using Arduino ATMega** 328 by Pooja Pawar, Suvarna Langade, Mohini Bandgar.
3. IJEDR, 2014: - **Scrolling LED Display using wireless transmission** by Anuradha Mujumdar, Vaishali Niranjane, Deepika Sange
4. November, 2015: **- A survey on digital notice board** by Jaiswal Rohit, Kalawade Sanket, Kore Amod, Lagad Sanket.
5. Internet.