# Experiment No:- 1

Aim:- Creating a Darehouse Application in Salesforce.com's force.com.

Theory:-

Step 1

Click on Setup → Create → Object- New Custom Object- ↵

Label - MySale

Plular label - MySales

Object Name - My Sale

Record Name - MySale Desemption

Data Type - Text-

→ click on Save.

Step 2

Under mySale Go to Custom fields and Relationships → click on new Custom field.

Creating 1st field:- Select Data type as Auto Number → next → enter details → field Label: PROD_ID. Display format: MYS-{0000} ↵

Starting Number: 1001 ↵

field Name: PRODID

Next- Save & New.

Creating 2nd field :- Select Data type as Date → next → Enter the details → field Label: Date of Sale → feild Name: Date_of_Sale → Default Value: Today () :-1 → Next → Save & New.

Creating 3rd field :- Select Data type as Number → next → Enter the details → field Label: Quantity Sold → length:3 Decimal places: () → Default Value: Show formulae editor: 1 next → Save & New.

Creating 4th field :- Select Data type as Currency → next → Enter the details → field Label: Rate → field Name: Rate → length:4 → Decimal places:2 → Default value: 10 → Next → Save & New.

Creating 5th field :- Select Data type as Currency → next My Sale field → Quantity Sold_c * Ratec → next → Save.

Now create an App

Setup → Create → App → new → Myshop → Next. → Select an Image → Next → Add Object Mysales

Now Create a Tab

Setup → Create → Tab → New Custom Tab → choose My Sales Object → Select Tab Style → Save.

On the top in the tab bar you can see the tab which has been created by click on the tab you can see your object is opened just click on new button and proride the details mentioned.

Conclusion :- In this We have created a My shop Application on force.com using declarative model.

## Experiment NO: 2

Aim :- Create an Application in Salesforce.com Using Apex programming language.

Theory :- Step 1

Login to your Sandbox or Developeros Orgaanization.

Click on Setup → create → Objects → new custom Objects.

Enter Book label

Enter Books for plural label. click Save.

Step 2

NOW let's create custom field.

In the custom field & relationship section of the Book Object click new. Select Number for the datatype & next. Enter Price for the field label. Enter 16 in the length text box. Enter 2 in the decimal Places & Next, Save.

Step 3

Click Setup → Develop → Apex classes & click New.

Step 4

A trigger is a piece of code that can execute Objects before or after specific data manipulation language events occured.

Click on Setup → Create → Objects → click the object you have created ex: Book scroll down you can see Trigger click on New

In the Trigger Editor enter this class.

```
trigger HelloWorld Trigger on Book__c (before insert)
{
    Book__c[] books = Trigger.new;
    MyHelloWorld.applyDiscount(books);
}
```

Step 5

Click on setup → create → tabs → new custom tab → choose Book → next & next & .. save.

click on tab Books → new → insert a name for Book → insert price for that book → click on Save.

Conclusion:- Thus we have studied how to create and run an application in Salesforce developers site by using APEX programming language.

## Experiment No:- 3

Aim:- To study & implement Web services in SOAP for JAVA Applications.

Theory:-

### Overview Of Web Services

Web Services are application components that are designed to support interoperable machine-to-machine interaction over a network. This interoperability is gained through a set of XML-based open standards, such as the Web Services Description Language (WSDL), the Simple Object Access Protocol (SOAP), and Universal Description, Discovery, and Integration (UDDI). These standards provide a common and interoperable approach for defining, publishing, and using Web services.

### Choosing a Container

You can either deploy your web services in a web container or in EJB containers.

To add an operation to the web service, change the design to view in editor. After you deploy a web service into a server, you can use the IDE to open the service's client.

To test successful deployment to Glass fish or Weblogic server, Right click the project & choose Deploy.

Consuming the Web Service:-

Now that you have developed the web service, you need to create a client to make use of the web service's add method.

Client 1: Java class in Java SE Application

In this section, you create a standard Java Application.

```
Public Static void main (String [] args)
    int i = 3;
    int j = 4;
    int result = add (i, j);
    System.out.println ("Result = " + result);
```

- Right-click the project node and choose from

The Output window now shows the sum: compile

```
ru
n:
Re
sul
t=
7.
BUILD SUCCESSFUL (total: 1 second)
```

Conclusion:- Thus we have studied use of Webservices using SOAP for a Java application.

## Experiment NO:-4

Aim:- The implementation of Para-Virtualization using VM Ware's Workstation / Oracle's Virtual Box and Guest-O.S.

Theory:- Virtual Box is a cross-platform virtualization application. Create on the New botton as the top of the virtual Box Manager window. To create a new, empty virtual hard disk, press the "New" botton. Save the machine state:- With this option, virtual box freezes the virtual machine by completely saving its state to your local disk. When you start the VM again later, you will find the VM continues exactly where it was left off. All your programs will still be open and your computer resources operation, saving the state of a virtual machine is thus in some ways similar to suspending a laptop.

Send the shut-down signal:- This will send an ACPI Shutdown signal to the virtual Machine, which has the same effect as if you had pressed the power botton on a real computer. So long as the VM is running a fairly modern operating system, this should trigger a proper shut-down mechanism from within VM.

Power off the machine:- With this option, virtual box also stops running the virtual machine but without saving its state as an exception, if your virtual Machine has any snapshots, you can reuse this option to quickly restore the current snapshot of this visual. In that case, powering off the machine will not disrupt its state, but any changes made since that snapshot was taken will be lost. The Discard button in the virtual Box Manager window discards a virtual machine's saved state. This has the same effect as powering it off and the same warnings apply.

Importing & Exporting Virtual Machines:-

They can come in several files as one or several disk images, typically in the widely used VMDK format and a textual description file in an XML dialect with an .ovf extension. These files must then reside in the same directory for virtual Box to be able to import them.

Alternatively, the above files can be packed together into a single archive file, typically with .ova extension.

Select file → Export appliance. A different dialog window shows up that allows you to combine several virtual machines into an ovf appliance. Then, select the target location where the target files

should be stored and the conversion process begins. This can again take a while.

Conclusion :-

Thus we have studied use of multiple OS using Virtual Box by virtualizing

## Experiment- NO:- 5

Aim :- Installation & configuration of Hadoop.

Theory :-

- Open ~~Home~~ Home
- Create a folder hadoop.
- Copy from downloads hadoop
- right- click on hadoop.
- cd hadoop.

Standalone operation

- mkdir input-
- cp conf/*.xml index
- bin/hadoop jar hadoop-examples
- cat output/*
- conf/core-site.xml:
  <configuration><property>
- conf/hdfs-site.xml:
- conf/mapred-site.xml:
  <configuration><property>.
- ssh localhost.

- ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
- cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys

- bin/hadoop namenode -format
- bin/start-all.sh

Run the following command to verify that hadoop services are running. If everything was successful you should see following running

2583 Data Node
2970 Resource Manager
3461 Jps
3177 NodeManager
2361 Name Node
2840 Secondary Name Node

Conclusion:-

Thus we have studied how to install and configure hadoop on Ubuntu Operating Systems.

## Experiment :- 6

Aim :- Recapitulation of Python.

a) Write a program to demonstrate working with tuples in Python.

Create an empty Tuple in Python

my_tuple = (); here the variable named my_tuple is the name of the tuple.

Create a Tuple with items in python.

```
my_tuple = ("me", "my friend", "my brothers",
            "my systers"); or

tuple1 = ("python", "tuple", 1952, 2323, 432);
tuple2 = (1, 2, 3, 4, 5);
tuple3 = ("a", "b", "c", "d", "e"); ex:
```

Output :- NOW printing each item in the tuple

me
my friend
my brothers
my systers.

b) Write a program to demonstrate working with dictionaries in python.

How to create a dictionary

```
# empty
dictionary
my_dict = {}
# dictionary with integer
keys my_dict = {1: 'apple', 2: 'ball'}# dictionary
```

With mixed keys

```
my_dict = {'name': 'John', 1: [2,4,3]} # using
dict()
my_dict = dict({1: 'apple', 2: 'ball'})
# from sequence having each item as a
pairmy_dict = dict([(1,'apple'), (2,'ball')]
```

Output:- Jack 26.

c) Write a Python Script that prints prime numbers less than 20.

```
# Python program to display all the prime numbers
upto
# Setting the initial value with 1.
Starting_value = 1
# Taking input from the user
n = int(input("Enter the number: "))
Print("Prime numbers between", Starting_value,
"and", n, "are:") for num in range(Starting_value,
                                   n+1):
    if num > 1:
        for i in range (2, int(num/2)+1): if
        (num % i) == 0:
         break
        else:
         print(num)
```

Output:- Enter the number - 20.

Prime numbers between 1 and 20 are: 2
3
5
7
11
13
17
19.

3) Write a python class to convert an integer to Roman numeral.

```
class py_solution:
    def roman_to_int(self,s):
        rom_val = {'I':1, 'V':5, 'X':10, 'L':50,
                   'C':100, 'D':500, 'M':1000}
        int_val = 0
        for i in range(len(s)):
            if i>0 and rom_val[s[i]] > rom_val
                                        [s[i-1]]:
                int_val += rom_val[s[i]-2 * rom_val[s[i-1]
            else:
                int_val +=
            rom_val[s[i]] return int_val
    Print(py_Solution().roman_to_int('MMCMLXXX
                                     VI'))
    Print(py_solution().roman_to_int('MMMM'))
    Print(py_Solution().roman_to_int('C'))
```

Output: 3986
4000
100.

# Experiment:7

Aim :- SQL Queries by fetching data from database in Raspberry Pi.

Components Required :-

1. Raspberry pi.
2. HDMI
3. Micro USB power input.

Algorithm :- Start the process.

- Connect micro USB power input to Raspberry Pi.
- Connect HDMI to the system to act as monitor.
- Connect USB port 2.0 mouse & keyboard.
- When enter the coding in the terminal to Update and Upgrade package.
- Create database in MySQL and basic SQL Queries.
- Stop the process.

Coding :-

```
Create database example.db;
Create User
create Table
Insert into Books
Update Books
Delete from Books.
```

Output:- Id | Title | Author

Result:- The output to fetch data from database using SQL Queries in Raspberry Pi has successfully executed.

## Experiment No:- 8

Aim:- Write a program for LED Blink using Raspberry Pi.

Components:-
1. Raspberry Pi.
2. Bread board.
3. Jumper wires.
4. Resistor.
5. LED.

Algorithm:-
- Start the process.
- Connect micro USB power input to Raspberry Pi
- Connect HDMI to the system to act as monitor for Rasberry pi.
- Connect USB port 2.0 to mouse and keyboard.
- Enter the coding in the terminal for installing Python and GPIO.

- Open notepad → enter coding → save as] file extension .python or .py
- Copy file location → open terminal → paste file location in terminal → press enter.
- In the terminal window to get output enter 0 or 1, to switch light ON when the input is 1.
- Stop the process.

Coding :- Sudo - apt get install python pip sudo apt-get install python.

```
def Sudo pip install
import RP GPIO as GPIO import time
GPIO.setmode (GPIO.BCM)
GPIO.setwarnings (false)
Print "LED" on "GPIO" .Output
Print "LED" off.
```

Output :-

```
pi@raspberrypi:~ $ cd Desktop/
pi@raspberrypi:~/Desktop $ sudo apt-get install python-rpi.gpio
Reading package lists... Done
Building dependency tree
Reading state information... Done
python-rpi.gpio is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
pi@raspberrypi:~/Desktop $ nano ledexample.py
pi@raspberrypi:~/Desktop $ python ledexample.py
```

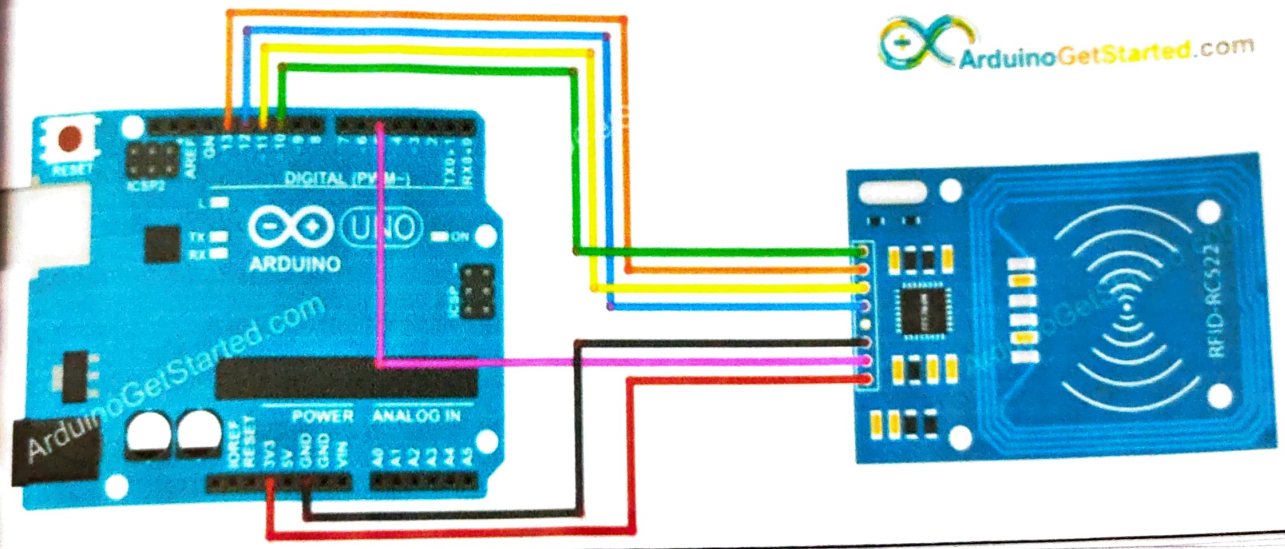
Result :- Thus the output to switch Light ON/OFF using Raspberry Pi has been succesfully executed.

# Experiment No:-9

**Aim:-** Study and Implement RFID, NFC using Arduino.

**Wiring table of RFID/NFC RC522 Module and Arduino**

| RFID/NFC RC522 | Arduino |
|---|---|
| SS | → 10 |
| SCK | → 13 |
| MOSI | → 11 |
| MISO | → 12 |
| IRQ(not connected) | |
| GND | → **GND** |
| RST | → 5 |
| VCC | → 3.3V |



```
#include <SPI.h>
#include <MFRC522.h>

#define SS_PIN 10
#define RST_PIN 5

MFRC522 rfid(SS_PIN, RST_PIN);

void setup() {
  Serial.begin(9600);
  SPI.begin(); // init SPI bus
  rfid.PCD_Init(); //init MFRC522
  Serial.println("Tap RFID/NFC Tag on reader");
}
// rfid.PCD
void loop() {
  if (rfid.PICC_IsNewCardPresent()) { //new tag is available
    if (rfid.PICC_ReadCardSerial()) { //NUID has been readed
      MFRC522::PICC_Type piccType = rfid.PICC_GetType(rfid.uid.sak);
      //Serial.print("RFID/NFC Tag Type: ");
      // Serial.println(rfid.PICC_GetTypeName(piccType));
      // Print NUID in Serial Monitor in the hex format
      Serial.print("UID:");
      for (int i = 0; i < rfid.uid.size; i++) {
        Serial.print(rfid.uid.uidByte[i] < 0x10 ? " 0" : " ");
        Serial.print(rfid.uid.uidByte[i], HEX);
      }
      Serial.println();
      rfid.PICC_HaltA(); //halt PICC
      rfid.PCD_StopCrypto1(); //Stop encryption on PCD
    }
  }
```

# Experiment NO: 10

**Aim:-** Study the temparature sensor and write program for moniter temparature using Arduino

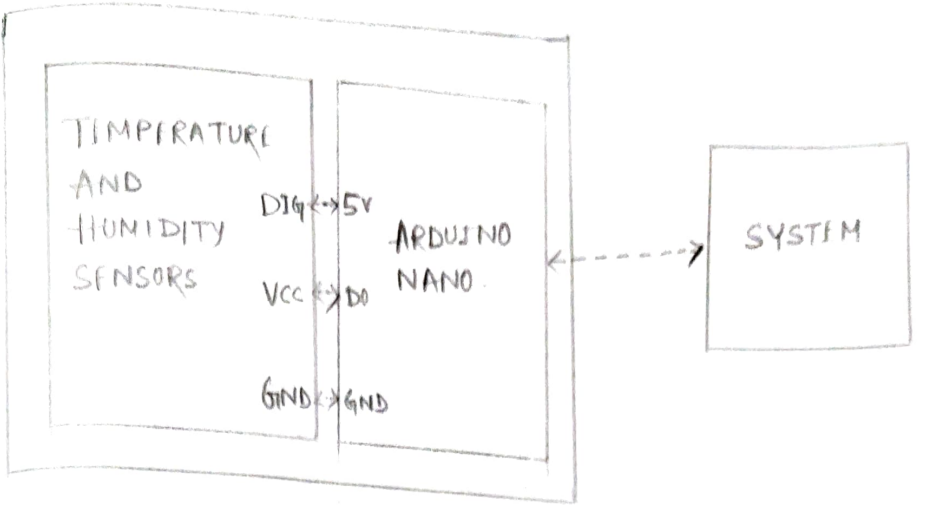
**Components Required:-**

- Temparature and humidity sensor.
- Jumpwe wires.
- Connectivity cable or USB cable.

**Algorithm:-**

Step 1:- Start the proces.
Step 2:- Start → Arduino 1.8.8
Step 3:- Include the DHT library to the Arduino software.
Step 4:- Then enter the coding in Arduino software.
Step 5:- complete the coding in Arduino.
Step 6:- In Arduino board connect VCC to the power supply 5V and connect STG to digital signal DT and connect SND to ground GND using Jumper wires.
Step 7:- Connect the arduino board with USB cable to the system.
Step 8:- Select tools → Selected.
Step 9:- Upload the coding to arduino board. Then the output will be displayed in the serial moniter.
Step 10:- Stop the proces.

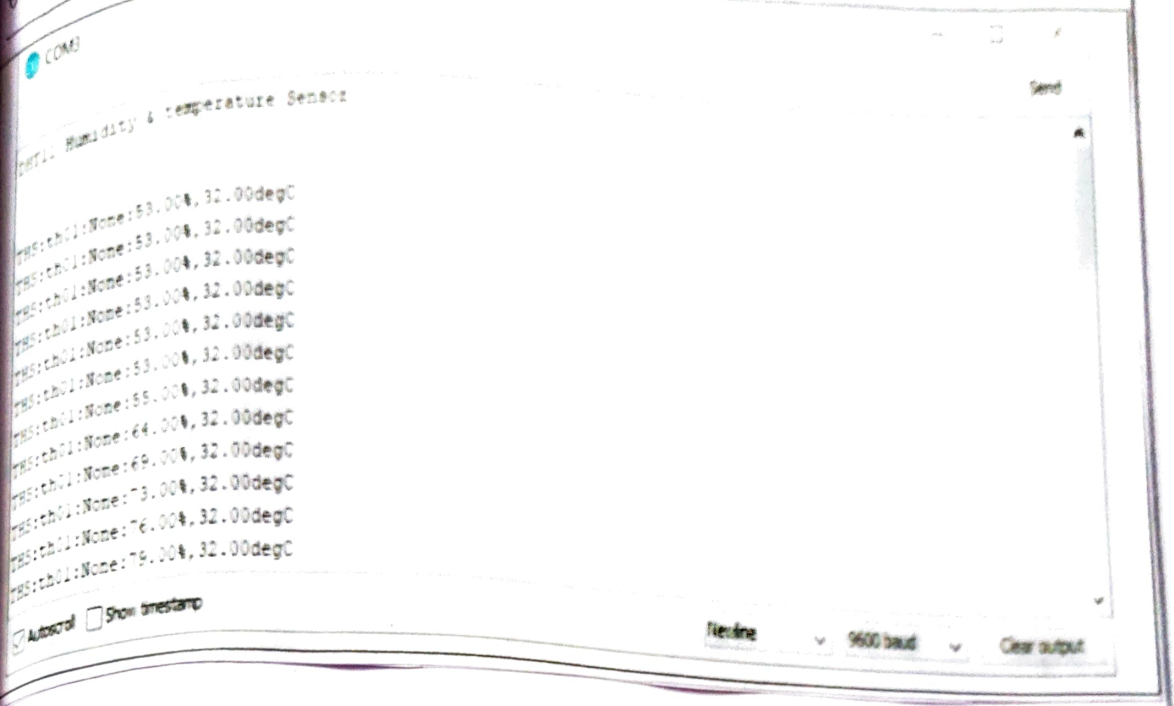
Block Diagram :-




Coding :-

```
#include <dht.h>
#define dht_apin A0 // Analog Pin sensor is
                    Connected to dht-DHT;
Void setup()
{
 pinMode (A0, INPUT); Serial.begin (9600);
 delay (500);
Serial.println ("DHT11 Humidity & temperature
                 Sensor \n\n"); delay (1000);
}
Void loop ()
{
 DHT.read11 (dht_apin); Serial.print ("THS: th01:
                                      None");
 Serial.print (DHT.humidity); Serial.print ("%,");
 // Serial.print ("temperature ="); Serial.print
                                   (DHT.Temperature);
 Serial.println ("deg C");
 delay (2000); // Wait 5 seconds before accessing
 sensor again
}.
```

Output:-




Result:-

Thus the Output to get temparature notification using Arduino has successfully executed.

# Experiment No:- 11

**Aim:-** Sense a finger when it is placed on Board Using Arduino.

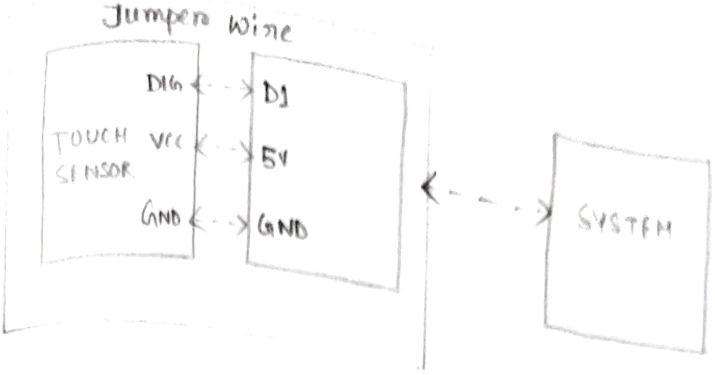
**Components Required:-**

- Touch Sensor
- Jumper Wire.
- USB Cable.

**Algorithm:-**

Step 1:- Start the process.
Step 2:- Arduino.
Step 3:- Then enter the coding in arduino software.
Step 4:- Compile the coding in the arduino software.
Step 5:- In arduino board, connect Vcc to power supply 5V and connect SIG to electrical signal DT and connect to ground and wing Jumper wires.
Step 6:- Connect the arduino board with USB cable to the System.
Step 7:- select tools → Select processor board and Port.
Step 8:- Upload the coding to arduino board. Then the output will be displayed in the serial monitor.
Step 9:- Stop the process.

Block Diagram :-




Coding :-

```
int LED = 13;  // define LED interface.
int buttonpin = 7; // define Metal touch sensor Interfacient val; // define numeric
variables val
void setup()
{
 Serial.begin(9600);
PinMode (Led, OUTPUT); // define LED as Output interface pinMODE (buttonpin, INPUT); //
define metal touch sensor output interface.
}
void loop()
{
 val = digitalRead (buttonpin);
 // Serial.println(val);
if (val == 1) // When the metal touch sensor detects a signal, LED flashes
{
 digitalWrite (Led, HIGH); Serial.println(val);
 delay (1000);
}
else
 { digitalWrite (Led, LOW); Serial.println(val); delay (1000);
  }}
```

Output :-


Result :- Thus the output for sensing a finger when it is placed in board Arduino has been successfully executed.

# Experiment NO:-12

**Aim:-** Connect with the Available Wi-fi Using Arduino.

**Components Required:-**

- ESP 8266 module or Wifi module
- Connecting cables or USB cables.

**Algorithms:-**

Step1:- Start the process.

Step2:- Start → Arduino IDE 1.8.8.

Step3:- Include the file directory ESP 8266 in Arduino.

Step4:- Then enter the coding to Wifi module or ESP 8266 module.

Step5:- Then enter the coding in Arduino Software.

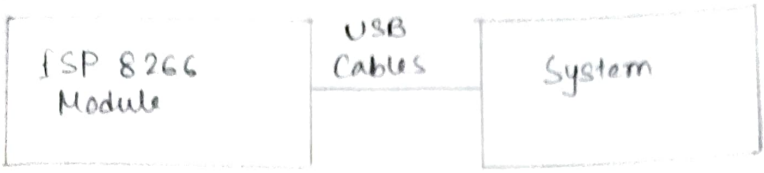
Step6:- Connect the USB cable to the Wi-fi module and the Arduino connected system with available Software.

Step7:- Select tools → Select board → Node MCU 0.9C ESP-12 module and the Select → Port.

Step8:- Upload the coding to ESP 8266 module and open serial monitor to view the available network connects IP address.

Step 9:- Stop the process.

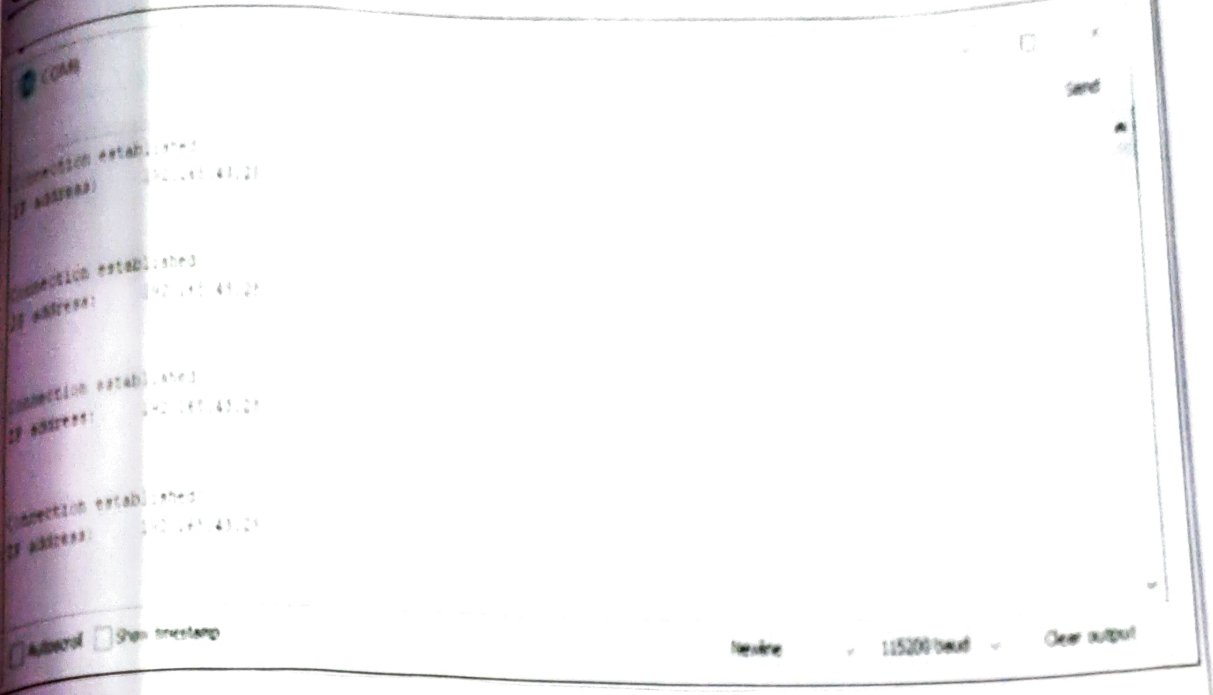
Block Diagram :-




Coding :-

```
#include <ESP8266WiFi.h> // include WiFi library
const char* ssid = "error"; // The SSID (name) of the WiFi network you want to connect to
const char* password = "network error"; // password
}
Serial.begin(115200);
delay(10); Serial.println('\n');
WiFi.begin(ssd, password);
Serial.print(ssid); Serial.print("....") int i=0;
While (WiFi.status()!= WL-CONNECTED) { // Wait for the Wi-Fi to connect
delay(1000);
Serial.print(++i); Serial.print(' ');
}
Void loop() { serial.println('\n');
Serial.println("connection established!");
Serial.print("IP address:\t");
Serial.println(WiFi.localIP()); // Send the IP address of the ESP8266 to the computer
}
}
```

Output:-




Result:-

Thus the output for connecting with the available Wifi using Arduino has been successfully executed.

## Experiment NO:-13

**Aim:-** Detect the vibration of an Object Using Arduino.

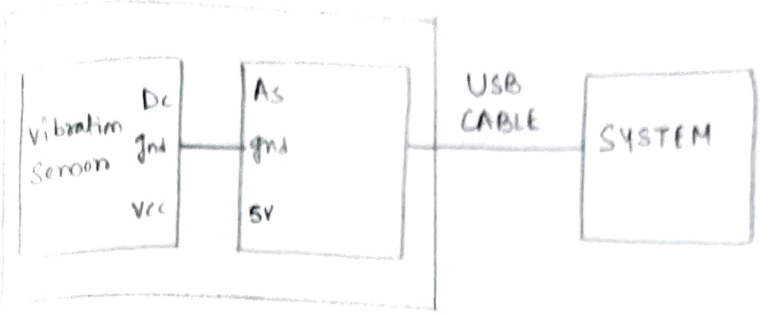
**Components Required:-**

- Vibration sensor
- Jumper wires
- USB Cable.

**Algorithm:-**

Step 1:- Start the process.
Step 2:- Start → Arduino 1.8.8.
Step 3:- Then enter the coding in Arduino software.
Step 4:- In Arduino board, connect vcc to power supply 5V and connect do to analog pin A0 and connect gnd to ground gnd using jumper wires.
Step 5:- Connect the arduino board withe the USB cable to the system.
Step 6:- Select tools → select board → Arduino Nano gnd → Select processor → AT mega 8 23 p and then select the port.
tep 7:- Upload the coding to the Arduino board.
tep 8:- Then the output will be displayed in the serial monitor.
tep 9:- stop the process.

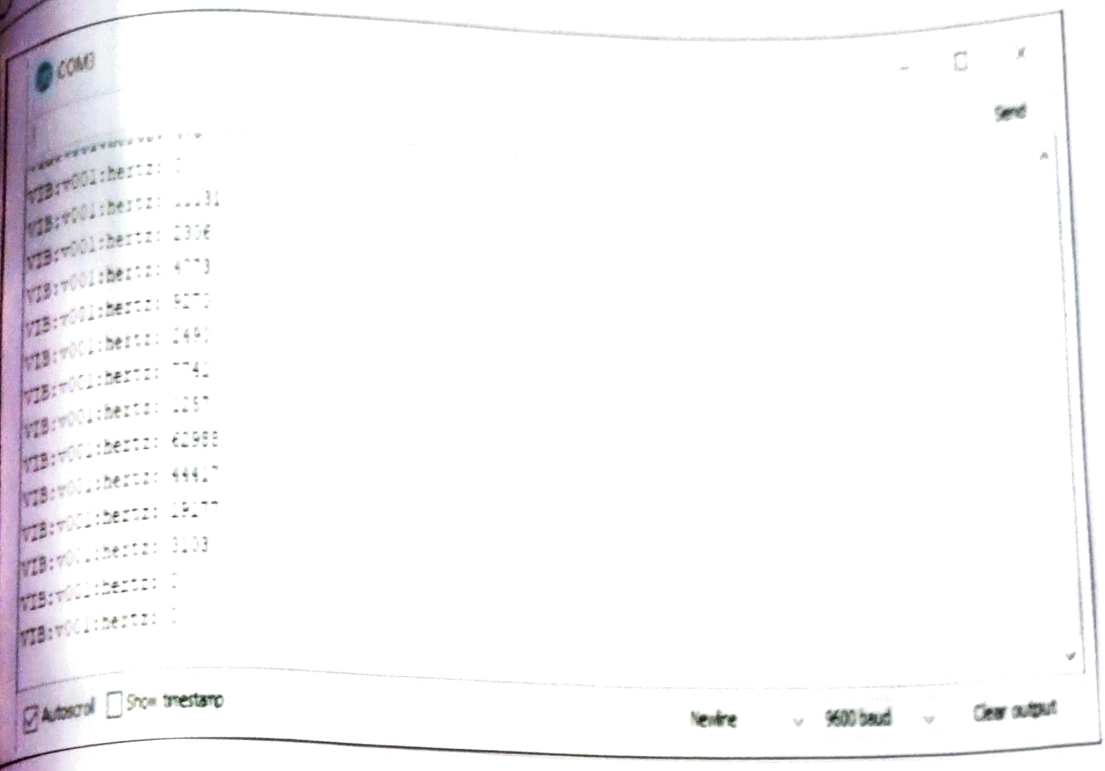
Block Diagram :-




Coding :-

```
Int ledPin = 13; Int vib = A0; void setup()
{
Pin Mode (ledPin, Output);
Pin Mode (vib, Input); // set fp input from
measurement-Serial.begin (9600); // init serial 9600
}
Void loop()
{ Long measurement = pulseIn (vib, HIGH); delayMicro
                                          seconds (50);
Serial.print ("VIB:VOOI:hent-z;"); Serial.println
                                   (measurement);
```

Output :-



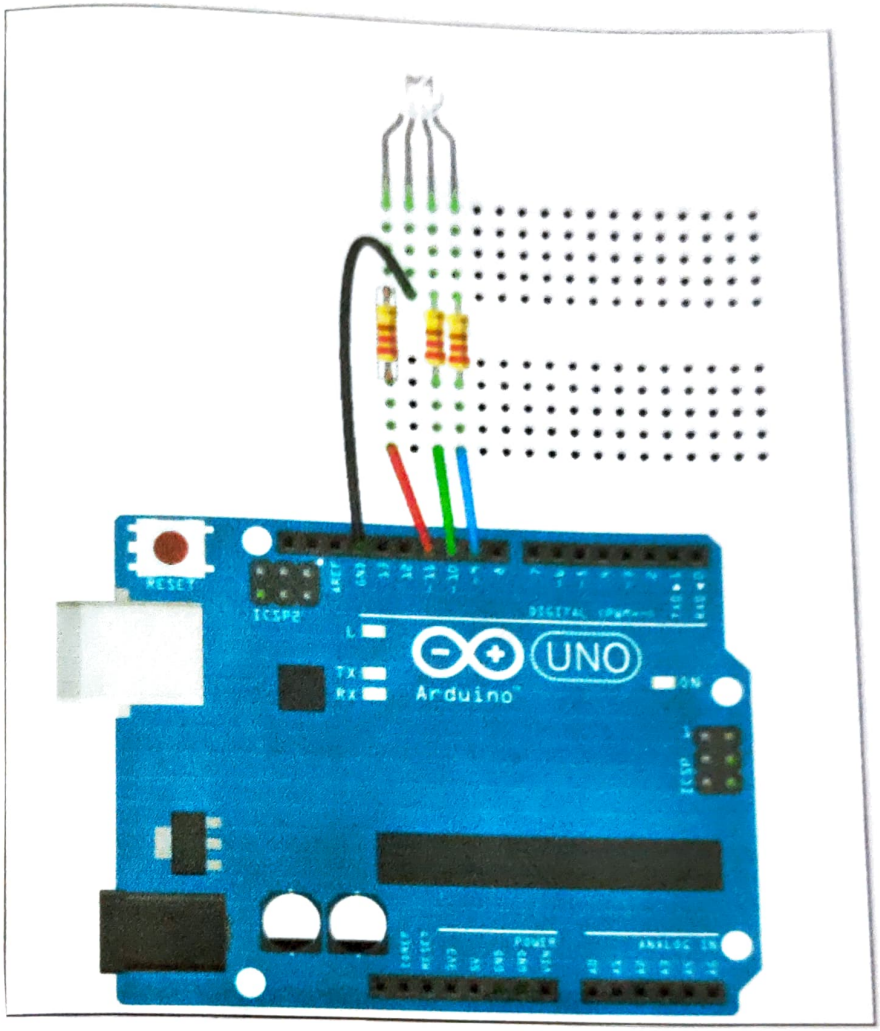

Result :-

Thus the output for detecting the vibrations of an object with vibration sensors using Arduino has been successfully executed.

# Experiment NO:-14

Aim:- Write a program for RGB LED using Arduino

Circuit Diagram:-




```
int red_light_pin = 11;
int green_light_pin = 10;
int blue_light_pin = 9;
Void setup () {
pinMode (red_light_pin, OUTPUT);
pinMode (green_light_pin, OUTPUT);
```

```
pinMode (blue_light_pin, OUTPUT);
}
void loop () {
  RGB_Color (255, 0, 0); // Red
  delay (1000);
  RGB_color (0, 255, 0); // Green
  delay (1000);
  RGB_color (0, 255, 0); // Blue
  delay (1000);
  RGB_color (255, 255, 125); //Raspberry
  delay (1000);
  RGB Color (0, 255, 255); // Cyan
  delay (1000);
  RGB color ( 255, 0, 255); // Magenta
  delay (1000);
  RGB color (255, 255, 0); // Yellow
  delay (1000);
  RGB color (255, 255, 255); // White
  delay (1000);
}
void RGB color (int red light value, int green light-
value, int blue light- value)
{ analog Write (red light-pin, red light value);
  analog Write (green light- pin, green light-value);
  analog Write (blue-light-pin, blue-light-
                value);
}
```

# Experiment No:- 15

**Aim:-** Write a program using Arduino IDE for Blink LED.

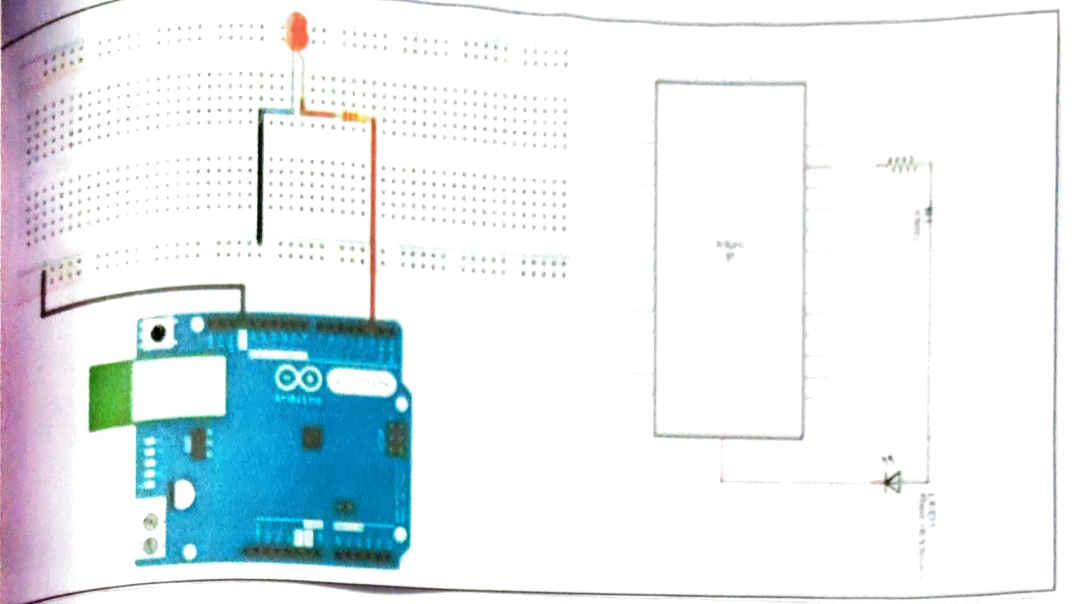
LEDs are small, powerful lights that are used in many different applications. To start, we will work on blinking an LED, the Hello World of microcontrollers. It is as simple as turning a light on and off. Establishing this important baseline will give you a solid foundation as we work towards experiments that are more complex.

## Components Required:-

You will need the following components—

- 1 X Breadboard
- 1 X Arduino Uno R3.
- 1 X LED
- 1 X 330Ω Resistor
- 2 X Jumper.

Procedure :- follow the circuit diagram and hook up the components on the breadboard as shown in the image given below.



Note - to find out the polarity of an LED, look at it closely. The shorter of the two legs towards the flat edge of the bulb indicates the negative terminal.

Arduino Code :- /* Blink

Turns on an LED on for one second, then off for one second, repeatedly. */

the setup function runs once when you press reset or power the board

```
void setup() { // initialize digital pin 13 as an output.
  pinMode(2, OUTPUT);
}
```

```
// the loop function runs over and over again
// forever.
void loop () {
digitalWrite (2, HIGH); // turn the LED on
                          (HIGH is the voltage level)
delay (1000) ; // Wait for a second.
digitalWrite (2, LOW); // turn the LED off by
                          making the voltage LOW)
delay (1000); // Wait for a second.

}
```

Result:-

If you should see your LED turn and off.
If the required output is not seen, make sure you have assembled the circuit correctly, and verified and uploaded the code to your board.