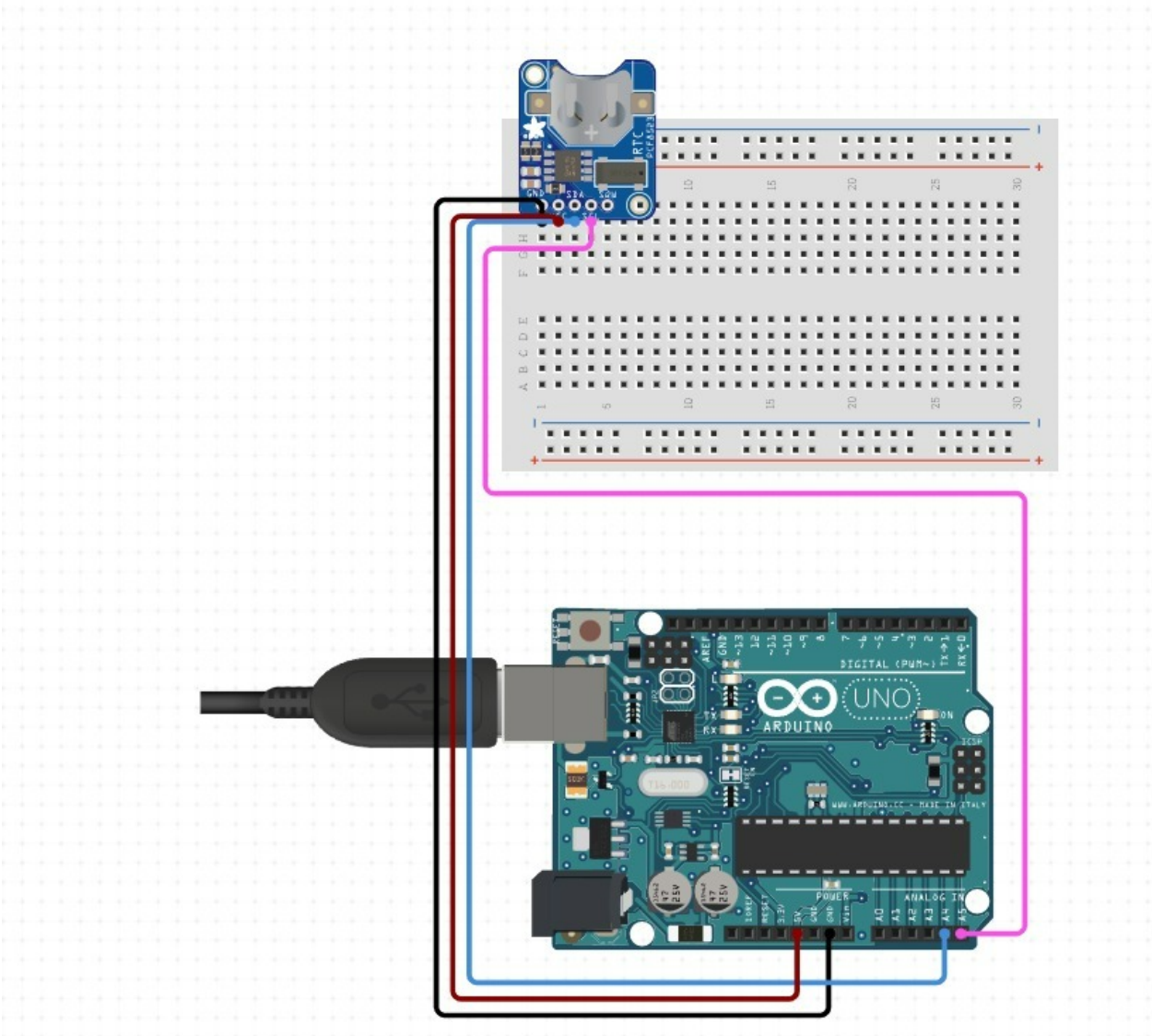
RTC sensor-

The RTC module PCF-8523 has the following 5 pins:

GND	Ground
VCC	Power Pin (3.3V or 5V)
SDA	12C data pin
SCL	12 C clock pin
SQW	Square wave output pin



1. The ground (GND) pin of the sensor is connected to the ground pin of the microcontroller.
2. The sensor VCC pin is connected to the 5V pin of the microcontroller.
3. The sensor SCL pin is connected to the A5/SCL pin of the microcontroller.
4. The sensor SDA pin is connected to theA4/SDA pin of the microcontroller.

Application in the project:

Code-

```
#include "RTClib.h"

RTC_PCF8523 rtc;

char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday",
"Thursday", "Friday", "Saturday"};

void setup () {
  Serial.begin(57600);

  if (! rtc.begin()) {
    Serial.println("Couldn't find RTC");
    Serial.flush();
    abort();
  }

  if (! rtc.initialized() || rtc.lostPower()) {
    Serial.println("RTC is NOT initialized, let's set the time!");
    // When time needs to be set on a new device, or after a power loss, the
    // following line sets the RTC to the date & time this sketch was compiled
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
    // This line sets the RTC with an explicit date & time, for example to set
    // January 21, 2014 at 3am you would call:
    // rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
  }
  rtc.start();
}

void loop () {
    DateTime now = rtc.now();
```

```
    Serial.print(now.year(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.day(), DEC);
    Serial.print(" (");
    Serial.print(daysOfTheWeek[now.dayOfTheWeek()]);
    Serial.print(") ");
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.println();

    Serial.println();
    delay(3000);
}
```

Gyro-sensor (MPU-6050)

Programming part is also easy for this project. Here we have used this MPU6050 library to interface it with Arduino. So first of all, we need to download the MPU6050 library from GitHub and install it in the Arduino IDE.

After it, we can find example codes in the example. The user may test that code by directly uploading them to Arduino and can see values over serial monitor. Or the user may use the code given at the end of the article to show values over LCD and serial monitor as well.

In coding, we have included some required libraries like MPU6050 and LCD.

```
#include<LiquidCrystal.h>

LiquidCrystal lcd(8,9,10,11,12,13);

#include <Wire.h>

#include <MPU6050.h>
```

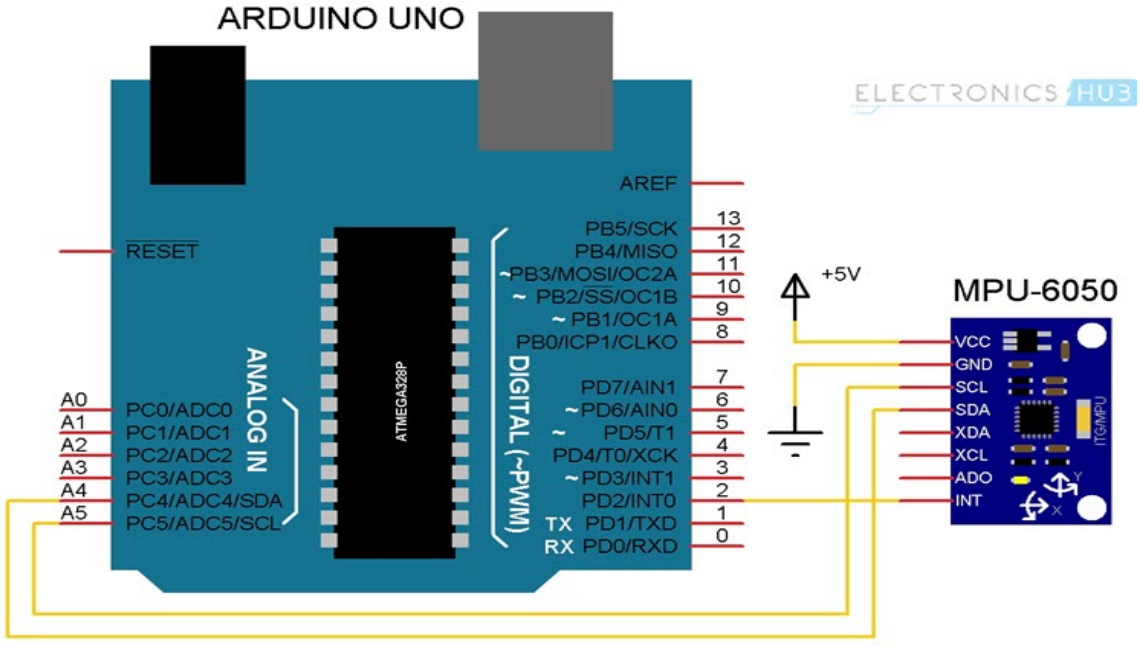
In the setup function, we **initialize both devices** and write a welcome message over LCD.

```
void setup()
{
  lcd.begin(16,2);
  lcd.createChar(0, degree);
  Serial.begin(9600);
  Serial.println("Initialize MPU6050");
  while(!mpu.begin(MPU6050_SCALE_2000DPS, MPU6050_RANGE_2G))
  {
    lcd.clear();
    lcd.print("Device not Found");
    Serial.println("Could not find a valid MPU6050 sensor, check wiring!");
    delay(500);
  }
  count=0;
  mpu.calibrateGyro();
  mpu.setThreshold(3);
```

In loop Function, we have called three functions in every 10seconds for **displaying temperature, gyro, and accelerometer reading** on LCD. These three functions are tempShow, gyroShow and accelShow, you can check those functions in the complete Arduino code given at the end of this article:

```
void loop()
{
    lcd.clear();
    lcd.print("Temperature");
```

```
long st=millis();
Serial.println("Temperature");
while(millis()<st+period)
{
  lcd.setCursor(0,1);
  tempShow();
}
lcd.clear();
lcd.print("Gyro");
delay(2000);
st=millis();
Serial.println("Gyro");
while(millis()<st+period)
{
  lcd.setCursor(0,1);
  gyroShow();
}
lcd.clear();
lcd.print("Accelerometer");
delay(2000);
st=millis();
```



1. The MPU-6050 is placed on the breadboard.
2. The ground (GND) pin of the sensor is connected to the ground pin of the microcontroller.

3. The sensor VCC pin is connected to the 5V pin of the microcontroller.
4. The sensor SCL pin is connected to the A5/SCL pin of the microcontroller.
5. The sensor SDA pin is connected to theA4/SDA pin of the microcontroller.
6. The sensor INT pin is connected to the pin2/INT0 of the microcontroller.

Wifi module (nRF24L01)

Code-

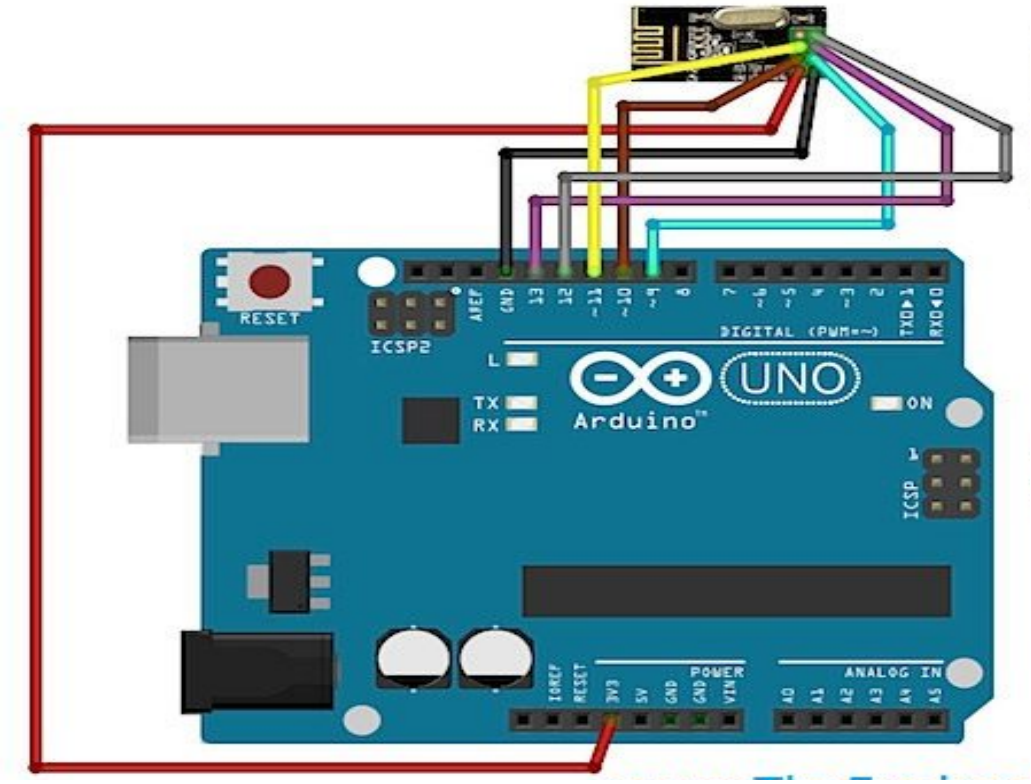
Transmitter code-

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(7, 8); // CE, CSN

const byte address[6] = "00001";

void setup() {
  radio.begin();
  radio.openWritingPipe(address);
  radio.setPALevel(RF24_PA_MIN);
  radio.stopListening();
}

void loop() {
  const char text[] = "Hello World";
  radio.write(&text, sizeof(text));
  delay(1000);
}
```

Receiver Code-

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(7, 8); // CE, CSN

const byte address[6] = "00001";

void setup() {
```

```
  Serial.begin(9600);
  radio.begin();
  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_MIN);
  radio.startListening();
}

void loop() {
  if (radio.available()) {
    char text[32] = "";
    radio.read(&text, sizeof(text));
    Serial.println(text);
  }
}
```



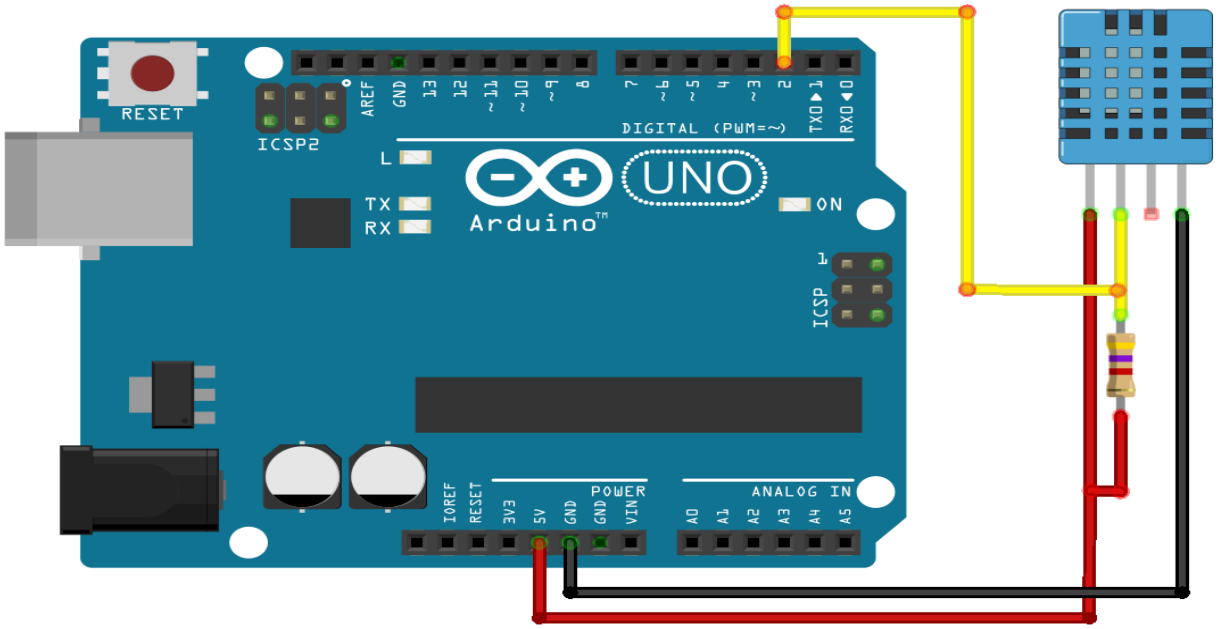
1. The nRF24L01 Wi-fi module is placed on the breadboard.
2. The GND pin of the module is connected to the GND of the microcontroller.

3. The VCC pin of the module is connected to the 3.3V pin of the microcontroller.
4. The CE pin of the module is connected to pin9 of the microcontroller.
5. The CSN pin of the module is connected to pin10 of the microcontroller.
6. The SCK pin of the module is connected to pin13 of the microcontroller.
7. The MISO pin of the module is connected to pin12 of the microcontroller.
8. The MOSI pin of the module is connected to pin11 of the microcontroller.

Humidity sensor (DHT-11)-

Code-

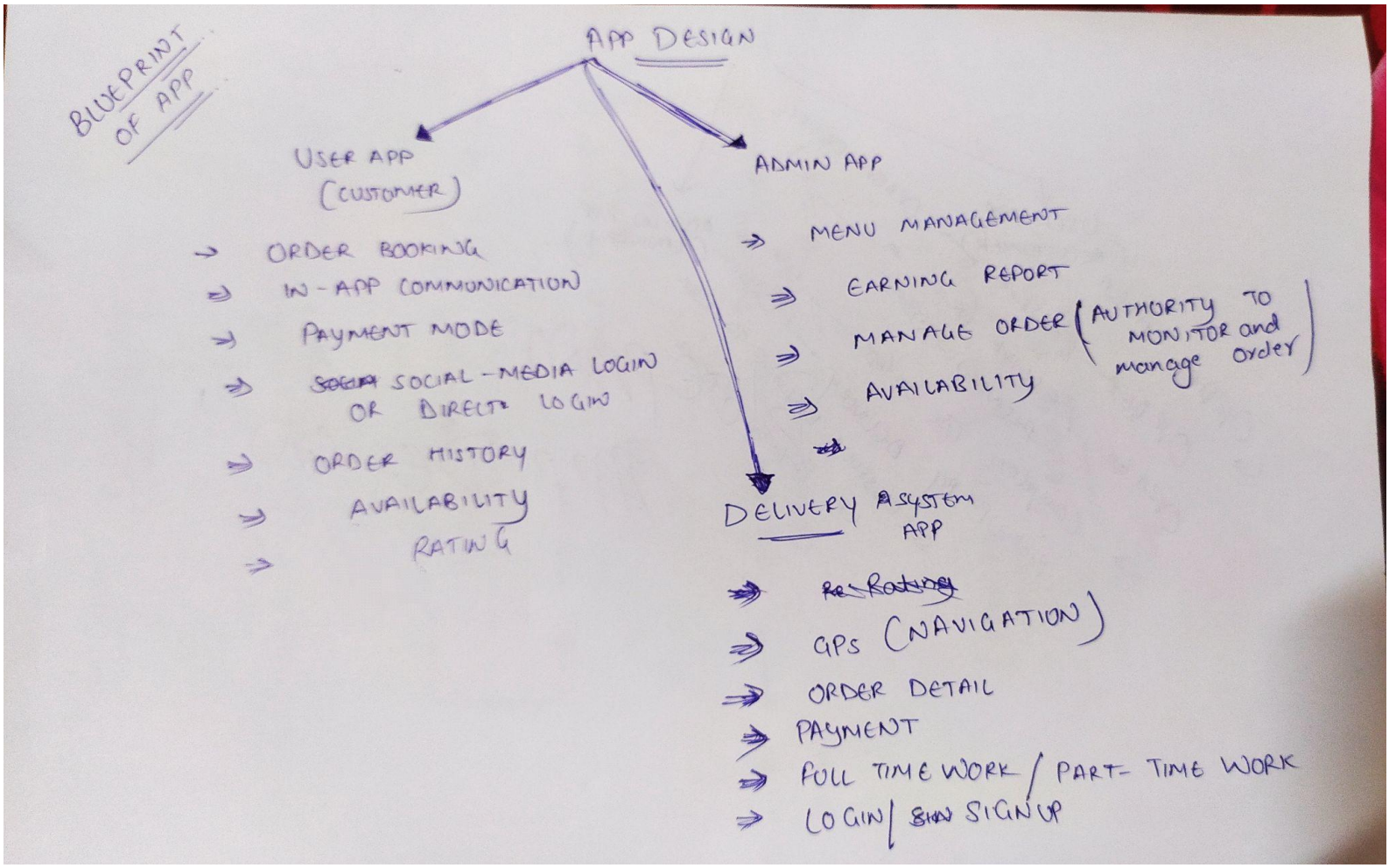
```
#include <dht.h>

#define dht_apin A0 // Analog Pin sensor is connected to
dht DHT;
void setup(){

 Serial.begin(9600);
 delay(500);//Delay to let system boot
 Serial.println("DHT11 Humidity & temperature Sensor\n\n");
 delay(1000);//Wait before accessing Sensor

}//end "setup()"

void loop(){
 //Start of Program

   DHT.read11(dht_apin);

   Serial.print("Current humidity = ");
   Serial.print(DHT.humidity);
   Serial.print("%  ");
   Serial.print("temperature = ");
   Serial.print(DHT.temperature);
   Serial.println("C  ");

   delay(5000);//Wait 5 seconds before accessing the sensor again.

 //Fastest should be once every two seconds.

}// end loop(
```



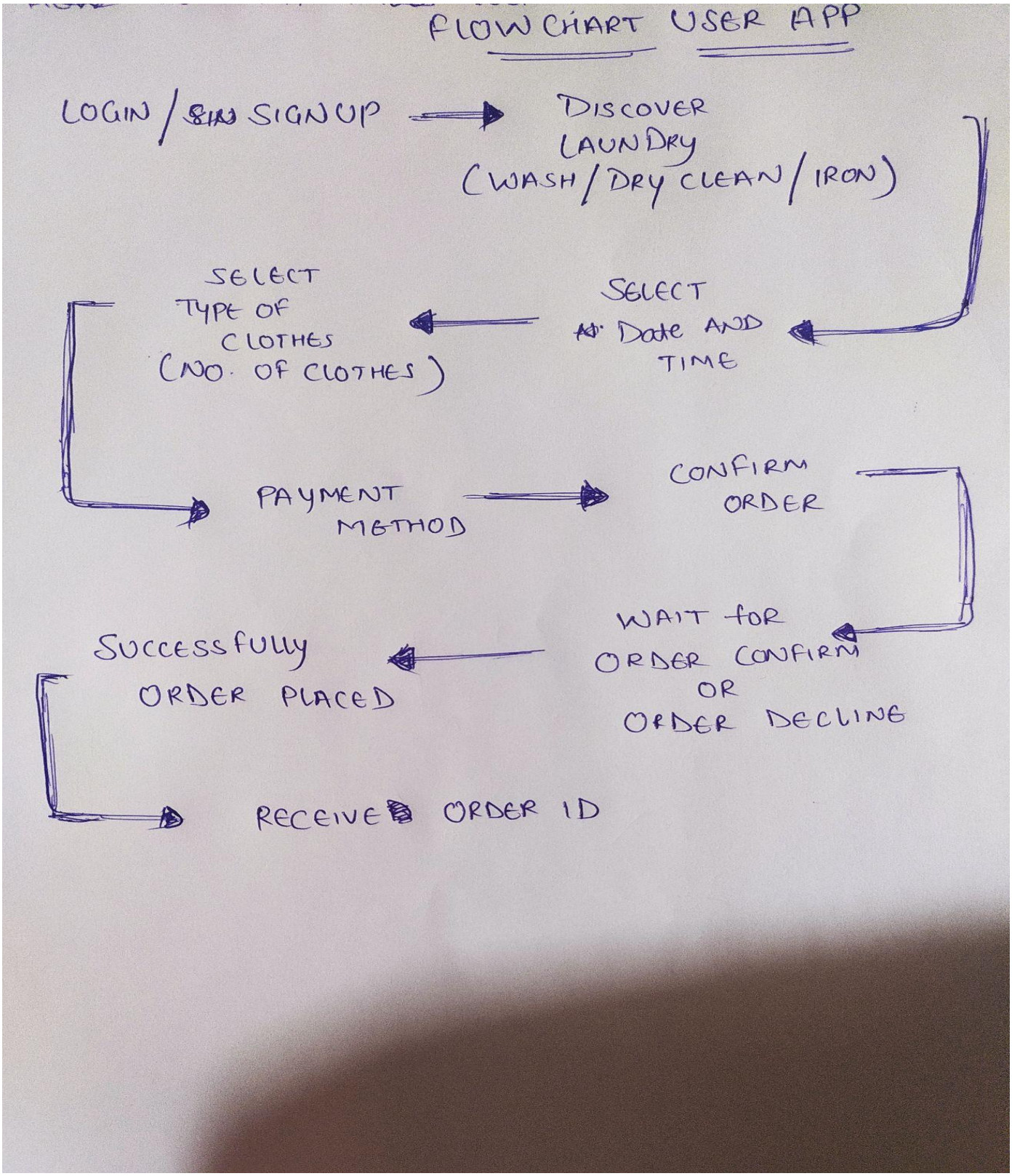
1. The DHT-11 sensor is placed on the breadboard.
2. The GND pin of the sensor is connected to the GND pin of the microcontroller.
3. The VCC pin of the sensor is connected to the 5V pin of the microcontroller.
4. The DATA pin is connected to the pin2 of the microcontroller.
5. A 10k ohm resistor is placed between the VCC and the DATA pin to act as a medium strength pull up on the data line.

A 10K Ohm pull-up resistor is needed between the signal line and 5V line **to make sure the signal level stays high by default.**

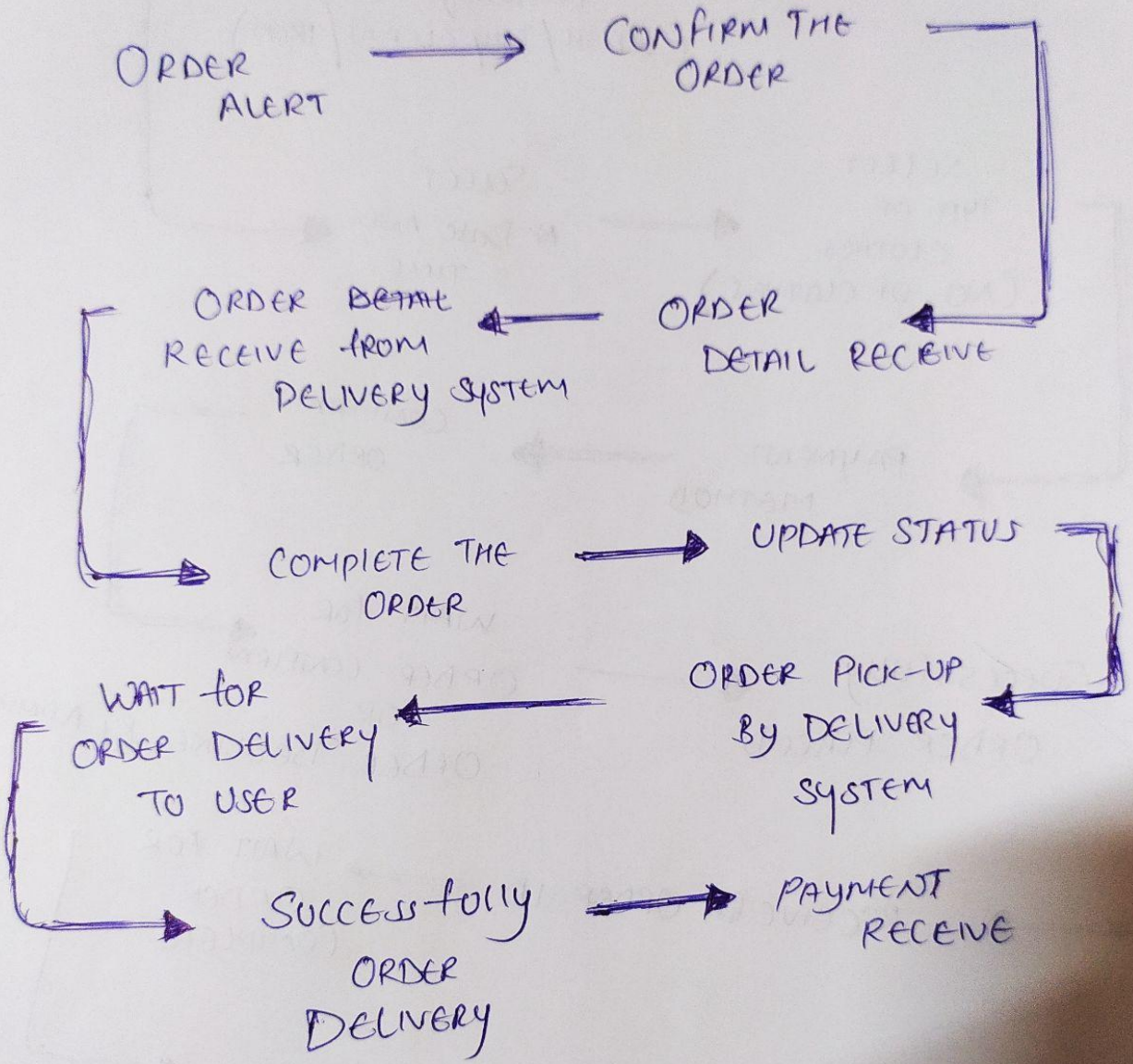
Application interface blueprint-



FLOW CHART USER APP

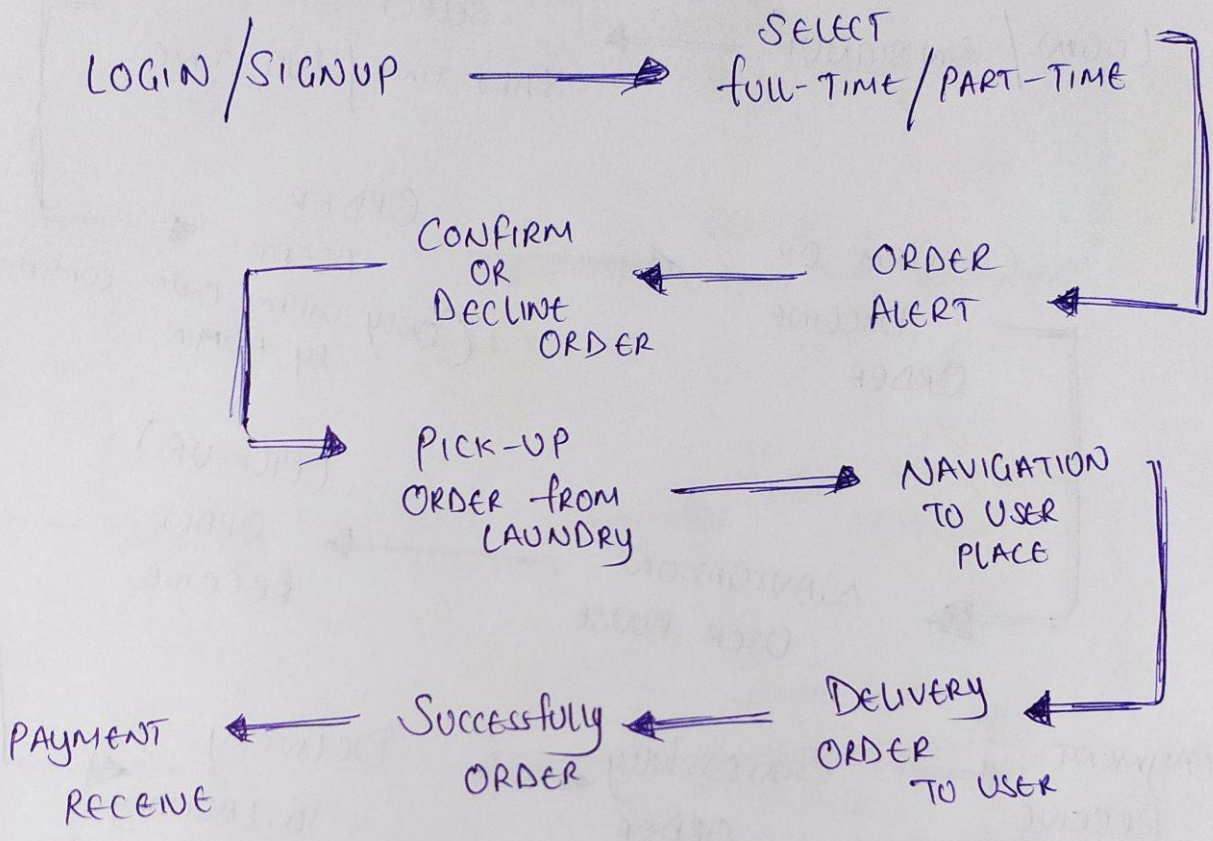


FLOW CHART ADMIN APP



of

flow chart Delivery System



LAUNDRY TO USER PLACE

DELIVERY SYSTEM