ESCUELA DE EDUCACIÓN SECUNDARIA MODALIDAD TÉCNICO PROFESIONAL PARTICULAR INCORPORADA Nº 2073 “SAN PABLO

Luces ecológicas

Integrantes: - Grassini, Facundo.

- Panichelli, Agustin.

- Ricchetti, Mateo.

-Casetta Augusto.

Índice

Resumen

Nuestro proyecto consiste en crear unas luces ecológicas reguladas con un dispositivo, la cantidad de luz en un cuarto, las cuales podrían ser controladas mediante una aplicación, o página web, para prender, apagar y regular su intensidad, y también consultar detalles como el consumo eléctrico.

Normalmente los ambientes con luz proveniente del exterior, no aprovechan al máximo la misma, y causa de eso se instalan luces que probablemente estén encendidas todo el día, incrementando el consumo eléctrico, desperdiciando energía y gastando dinero, lo que buscamos en este proyecto es solucionar este inconveniente o mal uso, con un dispositivo capaz de controlar y regular la luz artificial según lo necesite un ambiente.

Objetivos y alcances

En los sistemas lumínicos convencionales para interiores, generalmente la luz natural proveniente del exterior, no es aprovechada al máximo, lo que genera un consumo innecesario de energía eléctrica, que obviamente es un recurso que se debe cuidar mucho. En diversas ocasiones; leer, cocinar, trabajar, entre otras, es necesario controlar la cantidad total de luz en un ambiente; sala, cuarto, cocina, etc.

Este proyecto propone una solución para ambas problemáticas utilizando tecnología capaz de sensar la luz natural, y regular el consumo proveniente de la luminaria. También se puede regular la intensidad de la de luz artificial, para llegar a una iluminación ideal, combinando la luz artificial con la natural, y contará con tecnología WiFi para su control. La innovación de este proyecto es el dispositivo en sí mismo, el cual cuenta con la inteligencia suficiente como para una tener la iluminación adecuada con el menor consumo posible.

Cabe destacar que ideamos este proyecto pensando en una escuela, específicamente en la nuestra, ya que sucede este problema que explicamos anteriormente, pero más allá de eso, el dispositivo puede instalarse en cualquier lugar o situación.

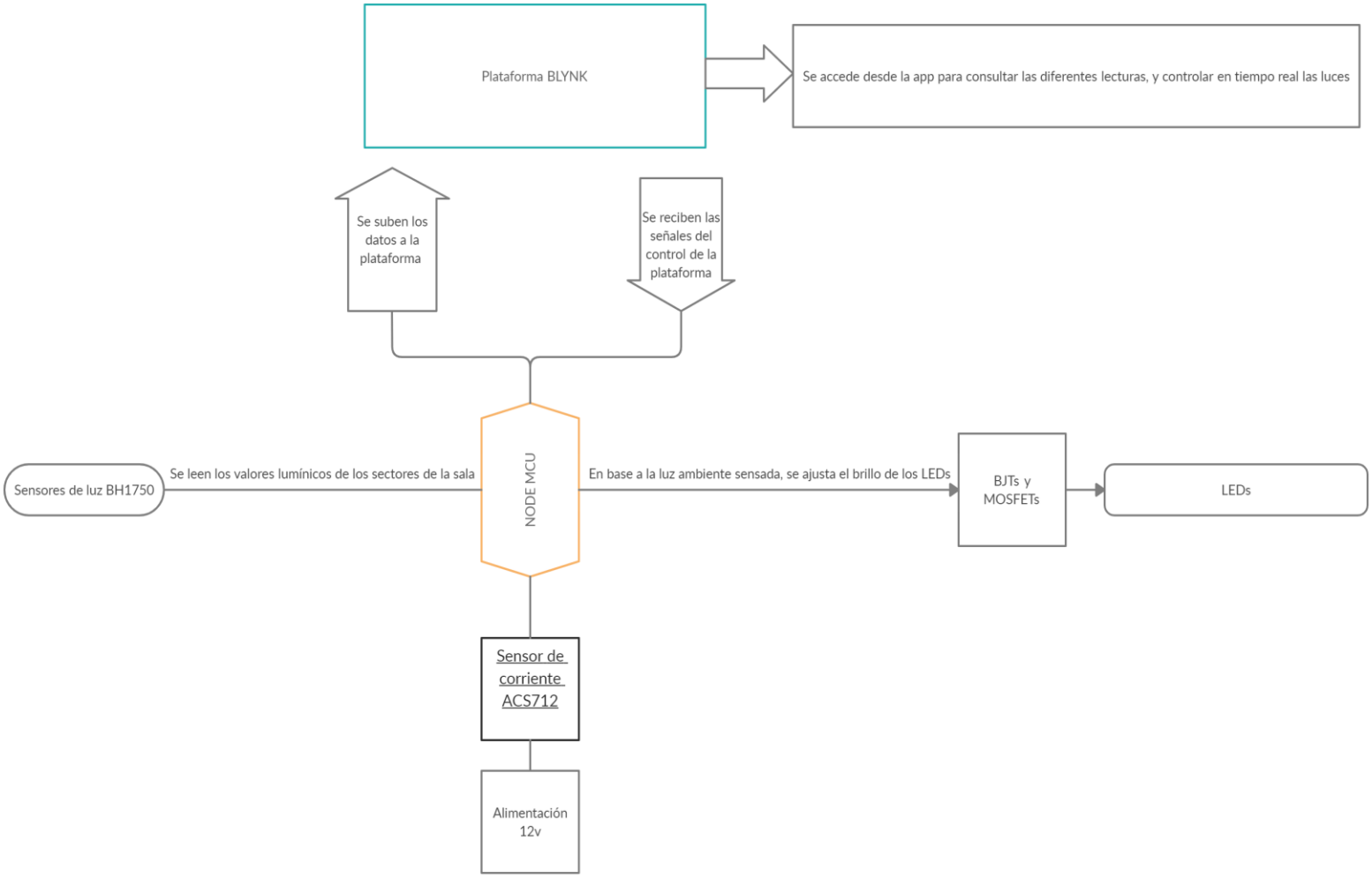
Presupuesto del dispositivo

ELEMENTO	CANTIDAD	PRECIO
Nodemcu Wifi ESP8266	1	$663
Modulo sensor de corriente ACS712	1	$369
Transistores BC547	2	$30
MOSFET IRF840	2	$80
Sensor de iluminación BH1750	2	$290
Resistencias	1KΩ x 2 + 4.7kΩ x 2	$10
Plaqueta montaje	1	$150
Leds	A determinar según el ambiente	$

Descripción general

El dispositivo mencionado, constará de dos sensores digitales de luz de ambiente, encargados de sensar la iluminación total; una placa de desarrollo (Node MCU) y un algoritmo para regular la intensidad de la luz artificial. Esta placa también, es la encargada del control total del sistema, y de la comunicación WiFi con la plataforma Blynk. Para la iluminaria se dispondrá de iluminación LED, que además de ser las indicadas para el control de la intensidad, son las de menor consumo. Para tener un control total de dicho consumo se añadirá un sensor de corriente. Con el cual se podrá controlar y registrar el gasto energético y por ende, la reducción en la factura de consumo.

Desarrollo



- BH1750

El BH1750 es un sensor de luz ambiental digital para I2C. Utilizamos este sensor porque implementa una comunicación serie y tiene la posibilidad de elegir entre dos direcciones I2C. Este IC (circuito integrado) es el más adecuado para medir la luz ambiental. Es posible detectar un amplio rango en alta resolución (de 1 a 65535 lúmenes), también cuenta con una resolución de 0.5lx (modo de alta resolución) o 4lx(modo de baja resolución) y, opera con 3,3V.

- NODEMCU

Se trata de una placa de desarrollo con conexión WiFi, compatible con el protocolo TCP/IP. Su principal función es dar acceso a cualquier microcontrolador a una red de internet.

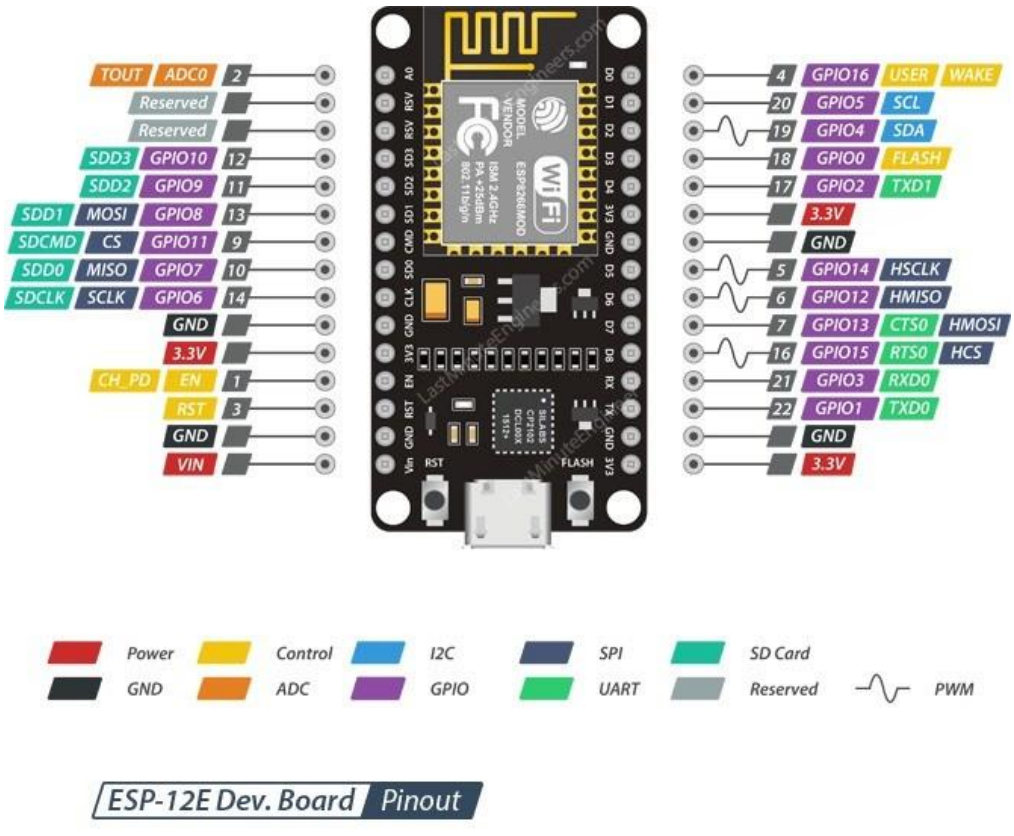
Elegimos esta opción por encima de otras placas por su abierta conectividad y su versatilidad, además de su reducido tamaño y peso.

Algunas de sus características son:

- Voltaje de operación entre 3V y 3,6V consumiendo sólo 80 mA en uso normal.
- Soporta IPv4 y los protocolos TCP/UDP/HTTP/FTP

El NodeMCU es el módulo más característico de este tipo. Su precio ronda los 6€. A diferencia de los otros módulos, viene con todo lo necesario para empezar a trabajar de forma sencilla. Incluye un adaptador serie/USB y se alimenta a través del microusb. Está basado en el ESP-12 y la última versión oficial es la 2.

Disposición de pines del Node MCU:



Para nuestro proyecto, los pines D1 y D2 serán utilizados para la comunicación I2C. Luego el pin A0 (El cual es analógico) para el sensor de corriente, los pines D4 y D5 (Con salida PWM) se utilizan para controlar la etapa de potencia que controla el brillo de la iluminación de los LEDs.

- ACS712

ACS712 es un pequeño sensor de corriente de efecto hall, que devuelve una señal entre 0 y 5v proporcional a la corriente que circula por él. Osea que este sensor sensa el campo producido por la circulación de corriente y devuelve un valor proporcional a este.

Este elemento fue elegido por su económico precio.

- RESISTENCIAS

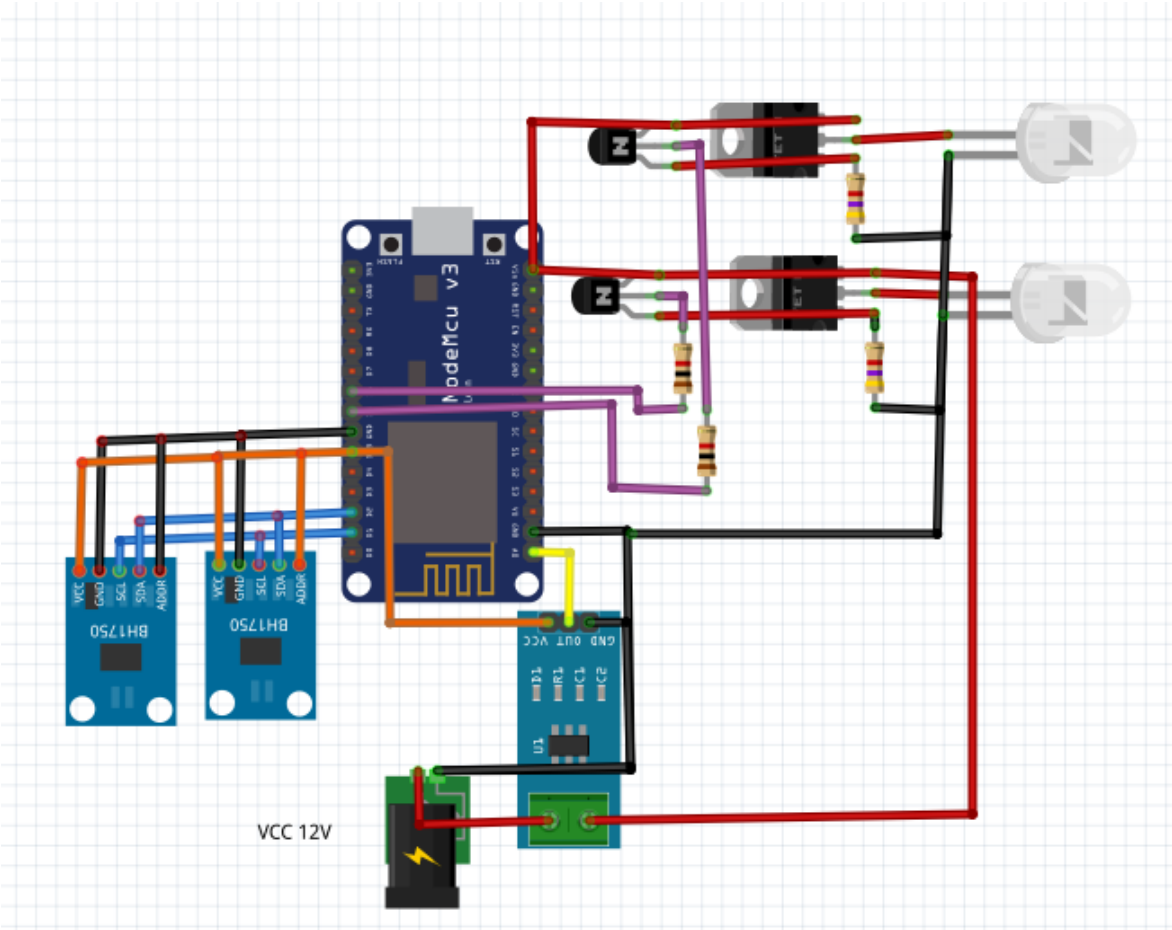
Estas son las encargadas de frenar el paso de la corriente:

- Las resistencias de 1KΩ limitan la corriente de base del transistor y así evitan que este se queme.
- Las resistencias de 4.7KΩ limitan la corriente del transistor y también se encargan de apagar los MOSFETs

- TRANSISTORES BJT

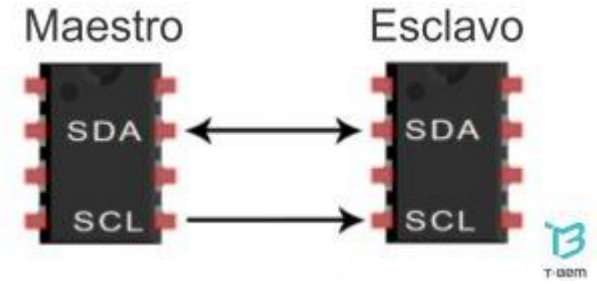
Estos son los encargados de acoplar los MOSFETs con la placa de desarrollo, permitiéndole manejar tensiones mas elevadas y mayores corrientes.

Esquema del hardware



EL PROTOCOLO I2C

El protocolo I2C es un protocolo de comunicación sincrónico y serial. Es decir, envía y recibe datos por un solo canal y estos datos están sincronizados por un ciclo de reloj. Se basa en el sistema maestro-esclavo, es decir que podemos tener un maestro controlando varios esclavos.



SDA

(Serial data) Es la vía de comunicación entre el maestro y el esclavo para transmitir información

SCL (serial clock) es la vía por dónde viaja la señal de reloj. Nosotros lo utilizamos para comunicar los sensores de luz con la placa de desarrollo.

IDE Arduino

IDE – entorno de desarrollo integrado, llamado IDE (sigla en inglés de integrated development environment), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios.

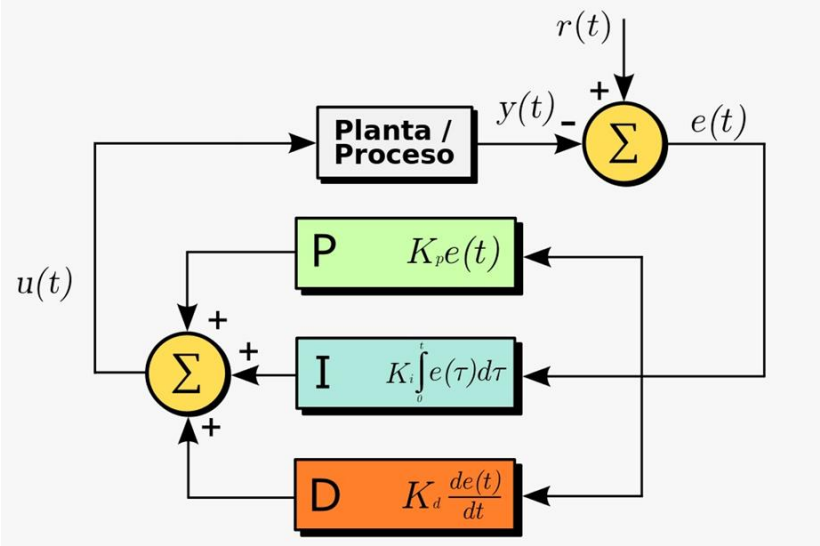
Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI).

EL CODIGO

El código se encarga de:

1. Leer los valores de los sensores con respecto a la luz del ambiente.
2. Leer el valor de corriente, sacar la potencia consumida y guardarla en la memoria.
3. Sacar el promedio del consumo de las pasadas 24hs con los valores guardados.
4. Leer el valor de brillo enviado mediante la aplicación "Blynk".
5. Regular la intensidad de la luminaria mediante los parámetros leídos (valor del brillo, el control PID y la luz ambiente).
6. Enviar los valores de luz, potencia en tiempo real y el promedio diario de la potencia.
7. Leer el estado de la luz enviado mediante la aplicación "Blynk".
8. Enciende o apaga la luz según el estado leído .

CONTROL PID



Un controlador PID (controlador proporcional, integral y derivativo) es un mecanismo de control simultáneo por realimentación ampliamente usado en

sistemas de control industrial. Este calcula la desviación o error entre un valor medido y un valor deseado.

El algoritmo del control PID consta de tres parámetros distintos: el proporcional, el integral, y el derivativo. El valor proporcional depende del error actual, el integral depende de los errores pasados y el derivativo es una predicción de los errores futuros. La suma de estas tres acciones es usada para ajustar el proceso por medio de un elemento de control, como la posición de una válvula de control o la potencia suministrada a un calentador, en nuestro caso el control de la intensidad de la luz. En nuestro caso los valores de PID los obtuvimos mediante el proceso empírico, llega así al valor deseado.

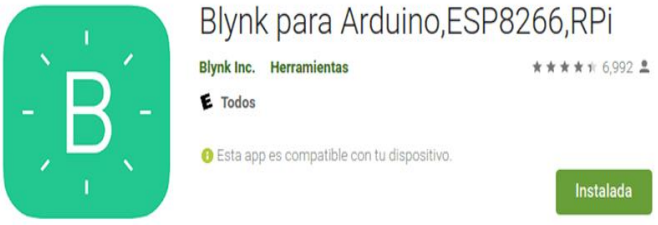
Para el correcto funcionamiento de un controlador PID que regule un proceso o sistema se necesita, al menos:

- Un sensor, que determine el estado del sistema (el BH1750 en nuestro caso).
- Un controlador, que genere la señal que gobierna al actuador (La placa de desarrollo).
- Un actuador, que modifique al sistema de manera controlada (La iluminaria).

El sensor proporciona una señal analógica o digital al controlador, la cual representa el punto actual en el que se encuentra el proceso o sistema. La señal puede representar ese valor en tensión eléctrica, intensidad de corriente eléctrica o frecuencia.

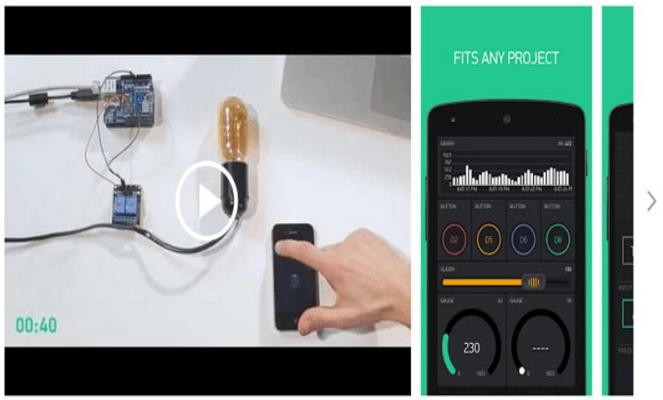
BLYNK

Blynk es una plataforma que provee soluciones para el desarrollo de aplicaciones de IoT. Su funcionamiento se basa en una app, que puede utilizarse tanto en teléfonos Android como iOS, que se comunica con el hardware a través de los servidores de Blynk. La app permite diseñar un “panel de control” con una amplia variedad de elementos que posibilitan enviar y recibir información hacia y desde el hardware.



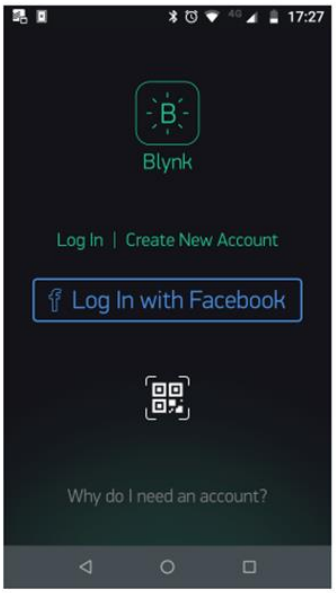
Paso 1: Instalar la app Blynk

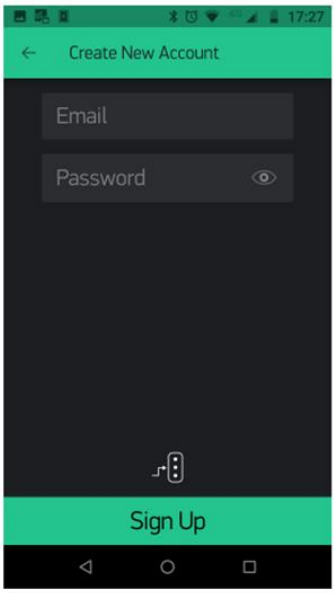
Para ello debemos ir a la tienda de apps correspondiente a nuestro teléfono (Android o iOS), descargar e instalar la app.



Paso 2: Crear una cuenta en Blynk

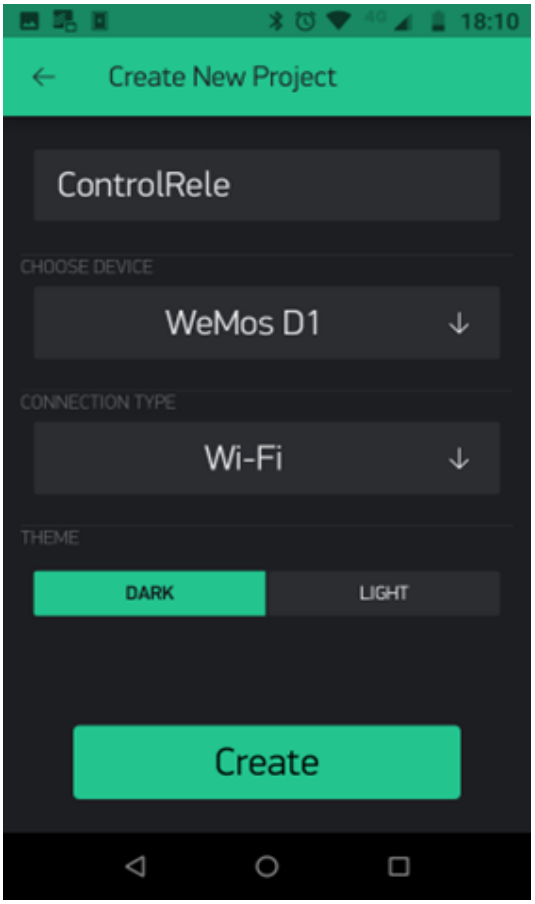
Una vez descargada e instalada la app, la abrimos y creamos una cuenta, con una dirección de email a la que podamos acceder y una contraseña.



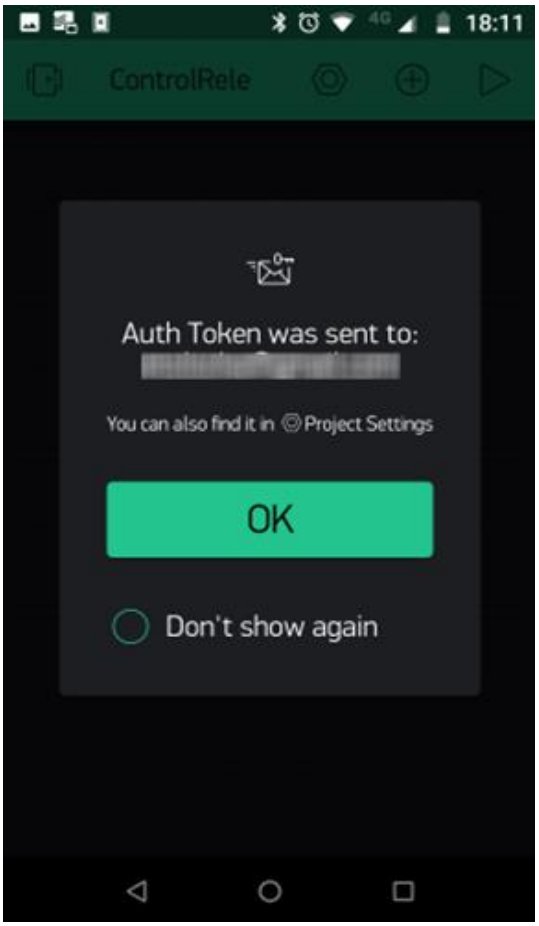


Paso 3: Crear un proyecto

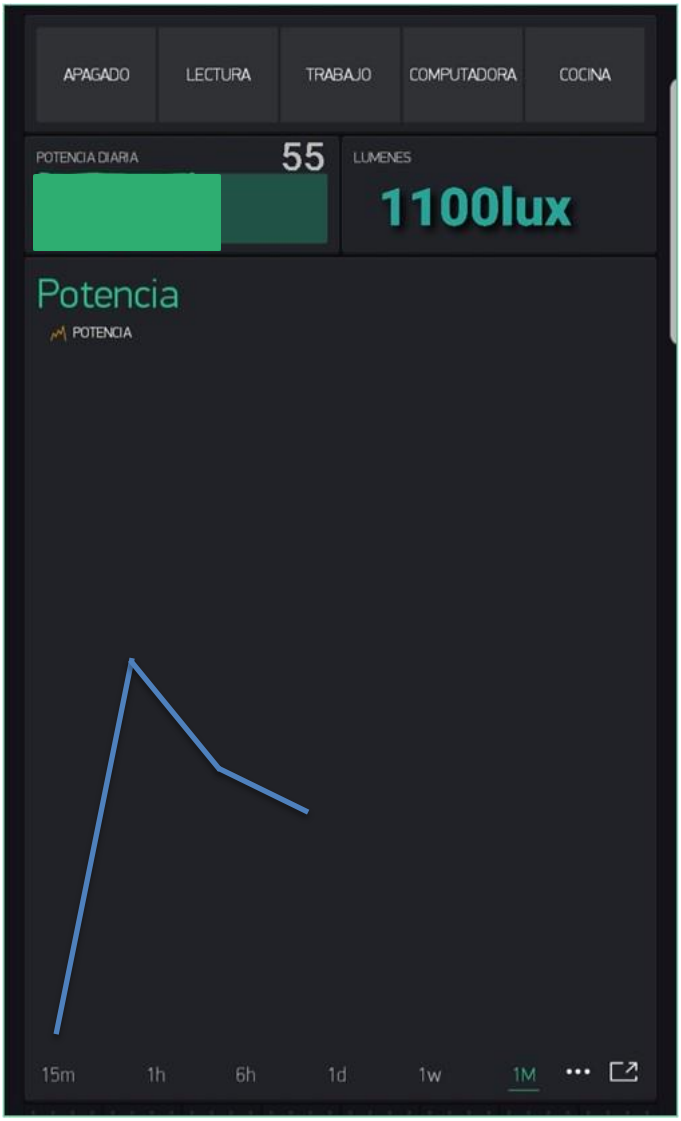
Vamos ahora a crear un proyecto que contenga los elementos necesarios para controlar nuestro hardware. En la parte superior de la app tocamos el símbolo “+” para crear el proyecto. Debemos darle un nombre y configurarlo según los detalles del hardware que utilicemos.



Una vez hecho esto, Blynk nos enviará un mail a la cuenta ingresada al principio con un “token”, una clave que vinculará la app con el hardware y que deberemos copiar luego en nuestro programa.



Paso 4: Diseñar la interfaz, seguidamente en el marco del proyecto incluiremos:



Conclusión

En conclusión, creemos que este proyecto es muy viable, ya que reduce muchísimo el desperdicio de energía eléctrica, beneficiando así al medio ambiente. También sabemos que si se quieren emplear muchos de estos dispositivos se deberá hacer una inversión bastante importante, sin embargo, ese gasto se amortizará con el paso del tiempo, gracias al ahorro energético que mencionamos anteriormente.

El mayor inconveniente que tuvimos al crear el proyecto, fue encontrar un sensor de luz que tenga salida digital, ya que el Node MCU solo tiene una entrada analógica, y además encontrar un sensor que utilice un protocolo de dos direcciones como el I2C. También tuvimos algunos problemas para encontrar la manera de hacer un registro mensual del consumo.

Agradecimientos.

- A la profesora Sandra Caruso, por impulsar nuestro proyecto inicialmente, y enseñarnos a confeccionar el informe.
- Al profesor Jorge Oglietti, por aconsejarnos también con algunas opciones en cuanto al ensamblaje del hardware.
- A los profesores Rodrigo Ortolá, y a Marcos Pirani, por guiarnos para mejorar el código del algoritmo y por darnos tips para mejorar el hardware.

Bibliografía.

https://blynk.io/
https://www.instructables.com/BH1750-Digital-Light-Sensor/

Anexo

Códigos

```
#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>        //Aqui se incluye la libreria que permite conectarse a una red wifi
#include <BlynkSimpleEsp8266.h> //Aqui se incluye la libreria que permite comunicarse con la app Blynk

#include <Wire.h>               //Aqui se incluye la libreria que permite la comunicacion serial mediante I2C
#include <BH1750.h>             //Aqui se incluye la libreria que permite leer el sensor de luz bh1750

BH1750 sensorL;                 //Aqui declaro "sensorL" como un sensor bh1750
BH1750 sensorR;                 //Aqui declaro "sensorR" como un sensor bh1750
int estado;                     //Aqui declaro el numero "estado"
float luxL;                     //Aqui declaro el numero flotante "luxL"
float luxR;                     //Aqui declaro el numero flotante "luxR"
float prom;                     //Aqui declaro el numero flotante "prom"
float Ampers;                   //Aqui declaro el numero flotante "Ampers"
float P;                        //Aqui declaro el numero flotante "P"
double pinValue;                //Aqui declaro el numero doble "pinValue"
unsigned long t=0;              //Aqui declaro el numero long "t"
unsigned long t2=0;             //Aqui declaro el numero long "t2"
///*
```

```
float Kp = 2;                 //Aqui declaro el numero flotante "Kp"
float Ki = 5;                 //Aqui declaro el numero flotante "Ki"
float Kd = 1;                 //Aqui declaro el numero flotante "Kd"

unsigned long currentTime, previousTime;
float elapsedTime;            //Aqui declaro el numero flotante "elapsedTime"
float errorR, errorL;         //Aqui declaro los numeros flotantes "errorR", "errorL"
float lastErrorR, lastErrorL;  //Aqui declaro los numeros flotantes "lastErrorR", "lastErrorL"
float setPoint;               //Aqui declaro el numero flotante "P"
float cumErrorR, rateErrorR;   //Aqui declaro los numeros flotantes "cumErrorR", "rateErrorR"
float cumErrorL, rateErrorL;   //Aqui declaro los numeros flotantes "rateErrorL", "cumErrorL"
//*/
/*
int k= 255;
*/

char auth[] = "wJ2MYIvMEA567b-JilPfA7XhAb8U1G1F";//Aqui se pone el token de la app
Blynk

char ssid[] = "SPEEDY-D77E7F"; //Aqui declaro la red wifi a conectar
char pass[] = "8914299027";    //Aqui declaro la contraseña del wifi

void setup() {
  Blynk.begin(auth, ssid, pass);//Aqui me conecto con la red wifi y con la app Blynk
  Serial.begin(115200);         //Aqui inicio la comunicacion serial (para poder ver los datos
con la computadora)
  Wire.begin(4, 5);             //Aca se inicia la comunicacion I2C por los pines 4,5
  pinMode(12, OUTPUT);          //Aqui declaro el pin 12 como salida
  pinMode(14, OUTPUT);          //Aqui declaro el pin 14 como salida
  sensorL.begin(BH1750::CONTINUOUS_HIGH_RES_MODE, 0x23, &Wire);//Aca se inicia
la comunicacion I2C con el "sensorL" en la direccion 0x23
  sensorR.begin(BH1750::CONTINUOUS_HIGH_RES_MODE, 0x5C, &Wire);//Aca se inicia
la comunicacion I2C con el "sensorR" en la direccion 0x5c
}
void loop() {
  t=millis();               //Aca se guarda el tiempo actual en "t"
  if(t<t2){                 //Aca se compara el tiempo "t" con el tiempo"t2"
    t=t2;                   //si por alguna razon extraordinaria "t" es mayor que "t2" estos se igualan
  }
  Blynk.run();              //Aca corre la app Blynk
  delay(130);               //Aca espero 130ms para que los sensores logren sensar la luz
  if(t-t2>60000){           //si la diferencia entre "t" y "t2" es mayor que 60000ms o 1min pasa
lo siguiente
    Ampers = (3.3*analogRead(A0)/1023)/0.04; //leo los ampers que consume
    P= Ampers*12;                          //mediate formula saco la potencia (A*V=P)
    t=t2;                                  //igualo los tiempos "t" y "t2" para que espere otro min
  }
  luxL = sensorL.readLightLevel();//Aca se lee el sensor "sensorL"
  luxR = sensorR.readLightLevel();//Aca se lee el sensor "sensorR"
  prom = (luxL+luxR)/2;         //Aca se saca el promedio de los lumenes medidos
   luces();                     //Aca se ejecuta la rutina luces
}
BLYNK_READ(V0) {                //Aca si la app Blynk solicita el valor "V0"
  Blynk.virtualWrite(V0, P);    //escribo "P" en "V0" y lo envio
```

```
}
BLYNK_WRITE(V1){                //Aca si la app Blynk envia el valor "V1"
  swich (param.asInt()){
          case 1: {pinValue=0;
                    break;  }
          case 1: {pinValue=500;
                    break;  }
          case 1: {pinValue=300;
                    break;  }
          case 1: {pinValue=50;
                    break;  }
          case 1: {pinValue=750;
                    break;  }

}
 BLYNK_READ(V2) {               //Aca si la app Blynk solicita el valor "V2"
  Blynk.virtualWrite(V2, prom);     //escribo "y" en "V2" y lo envio
}
void luces(){                   //aca escribo la rutina "luces"
///*
if(estado==1){ //si la luz esta prendida
setPoint= pinValue*655.35;//paso el porcentaje a lumenes para poder sacar las cuentas
currentTime = millis();//Mido el tiempó
  if(currentTime<previousTime){//Aca se compara el tiempo "currentTime" con el tiempo
"previousTime"
    currentTime=previousTime;//si por alguna razon extraordinaria "currentTime" es mayor
que "previousTime" estos se igualan
    delay(12);//espero 12ms
    currentTime = millis();//Mido el tiempó devuelta asi la integral y la deribada no me dan
error
  }
elapsedTime = currentTime - previousTime;// veo cuanto tiempo paso
previousTime = currentTime;//guardo el ultimo tiempo
errorL = setPoint - luxL;//veo suanto hay que correjir(error)
errorR = setPoint - luxR;//veo suanto hay que correjir(error)
cumErrorL += errorL * elapsedTime;// integro el error
cumErrorR += errorR * elapsedTime;// integro el error
rateErrorL = (errorL - lastErrorL)/elapsedTime;// derivo el error
rateErrorR = (errorR - lastErrorR)/elapsedTime;// derivo el error
int left = Kp * errorL + Ki * cumErrorL + Kd * rateErrorL;//sumo el PID
int right = Kp * errorR + Ki * cumErrorR + Kd * rateErrorR;//sumo el PID
//*/

 /*
int left=pinValue*luxL/k;
int right=pinValue*luxR/k;
*/
// ACA LO QUE HACEMOS ES FIJARNOS QUE NO SE ESCAPE DE LOS LIMITES(0-255)
    if (right < 0){
      right= 0;
      }
    if (right > 255){
      right= 255;
      }
    if (left < 0){
```

```
      left = 0;
      }
   if (left > 255){
      left = 255;
      }
  analogWrite(14, left);    //Aca ""escribo"" el valor en la salida
  analogWrite(12, right);   //Aca ""escribo"" el valor en la salida
}
 }
```

Modo de uso

Actividad	lectura	Trabajo	Compu	Cocina
Lúmenes	300lux	750lux	50lux	500lux

En este lugar se indican los lúmenes actuales del ambiente

Con estos botones se selecciona si se quiere la luz apagada o prendida y a para que actividad

Este es el indicador del consumo

En esta sección se grafica el consumo en diferentes intervalos de tiempo (15m, 1h, 6h, 1d, 1w, 1M, 2M)