

1 Introducción

1.1 Problemática

El proyecto surge en base a la propuesta en la cátedra Comunicaciones de Datos 2023 de encontrar una problemática que, con IOT y la integración de conocimientos de otras materias de la carrera, llevar a cabo una solución práctica utilizando una interfaz codificada en tecnologías como Vue y Django, y como hardware una placa ESP32-WROOM, sensores y relés.

El problema elegido en este caso fue la de conocer el estado de las luces de cada una de las aulas de la Facultad (encendidas o apagadas). Que los usuarios de la APP tengan la posibilidad de encender o apagar las luces de manera remota, facilitando la tarea de las personas que trabajan en área de mantenimiento que, a día de hoy, deben recorrer diariamente a altas horas cada una de las aulas y oficinas procurando que todo quede ordenado, limpio, con las puertas cerradas y luces apagadas. Esto no sólo haría más rápido y práctico las tareas de algunos empleados, sino también el de comenzar a tener un registro histórico del consumo (medido en horas) lumínico, saber cuánto tiempo por día se usan las mismas en cada una de las aulas, permitiendo conocer el uso real y compararlo con el que se estipula. Las ventajas de tener esta información es saber que tan eficiente es el uso lumínico, si se utiliza sólo lo necesario o en exceso para, si esto ocurre, generar conciencia y mejorar en este punto.

1.2 Propuesta

Lo propuesto por el grupo fue una solución integral entre software y hardware. Por parte del software, una aplicación conformada con un FRONT desarrollado en Vue que se comunica con un BACK programado con Django. Todo esto, recibiendo información de lecturas por parte de un sensor fotosensible para definir un umbral en el que defina si la luz está encendida o apagada, además de un relé que se conecte a la red eléctrica y la tecla de la luz, funcionando como un switch de 3 puntos (Generando un circuito para prender o apagar la luz en una escalera), el cual recibe la señal por parte de la aplicación para efectuar su cambio de estado y por ende el de la luz. Más adelante verán un prototipo que explaya lo relatado hasta aquí.

1.3 Propósito

El propósito del proyecto de gestión de luces es el de, en principio, facilitarles el trabajo a los trabajadores del área mantenimiento y el de conocer el consumo energético de la luminaria de la facultad para comprender el uso que se le da.

1.4 Alcance

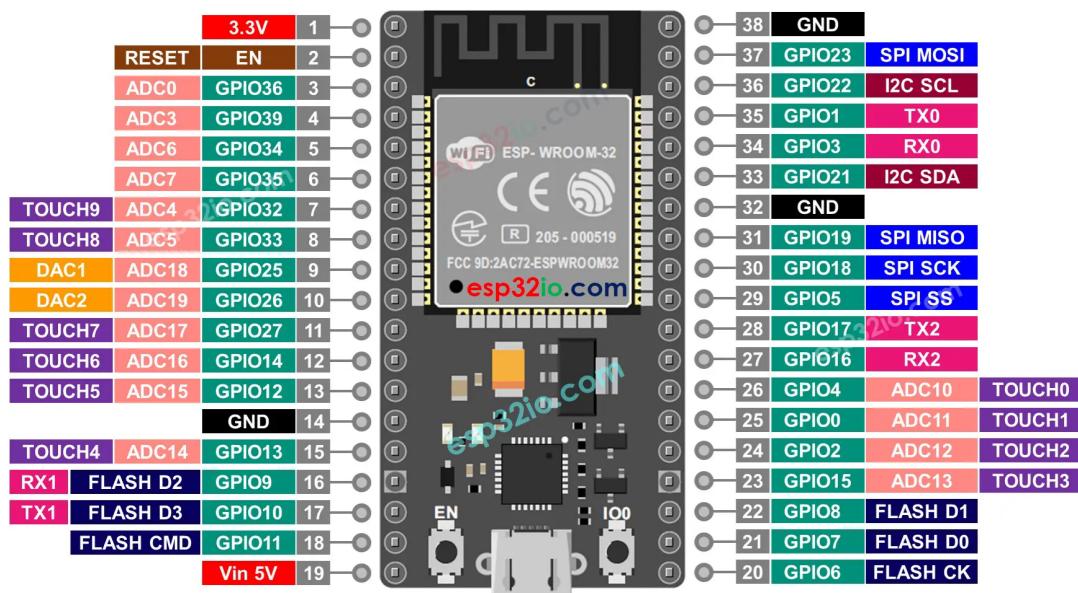
El proyecto pretende ofrecer una solución de automatización para el encendido y apagado de las luces de las aulas de la Facultad, conocer el estado de ellas (encendidas/apagadas) y la generación de datos estadísticos sobre su uso para, a posteriori, tomar decisiones.

1.5 Definiciones, acrónimos y abreviaturas

- Sensor: Dispositivo que se conecta a los pines de la placa ESP32 que cumple con una función como la de tomar valores de luz o cambiar su estado interno.
- Aula: Referencia a cada placa ESP32 en particular que contiene sensores (dentro del código la referencia real será la IP de la placa).
- Interacciones: Es cada acción que el usuario realiza desde la aplicación como puede ser apagar/encender la luz de un aula.
- Usuario: Quien se identifica al ingresar a la aplicación y realiza interacciones con las funciones de esta.
- Estados: Refiere a la lectura realizada por los sensores como desde cuando se encendió la luz, cuando se apagó o cuando el relé pasa de un valor low a un valor high.
- Pinout: Disposición de pines de la placa, de modo que se identifique donde se conectan los sensores y/o la fuente.
- Administrador: Usuario con el poder de crear usuarios, aulas y añadir sensores a cada una de estas.

1.6 Referencias

Imagen referencia PINOUT ESP32:



<https://www.google.com/url?sa=i&url=https%3A%2F%2Fesp32io.com%2F&psig=AOvVaw2uKmISQ6ObNufkb1j5To-7&ust=1699665360371000&source=images&cd=vfe&opi=89978449&ved=0CBMQjhxqFwoTCMjoqlOhuIDFQAAAAAdAAAAABAJ>

Driver para poder conectar la ESP32 a Windows y solucionar problemas de no reconocimiento:

<https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads>

2 Descripción general

2.1 Funciones del producto

- El programa debe permitir visualizar el estado actual de las luces de las diferentes aulas de la facultad
- El programa debe permitir apagar o encender las luces mediante un botón por pantalla
- El programa debe entregar estadísticas acerca de las horas que las luces se encuentran encendidas de forma histórica.

2.2 Características de los usuarios

Los usuarios que deberán interactuar con el sistema son:

- Administrador: Tienen acceso a todas las funciones del programa incluyendo la creación de usuarios, aulas y sensores, además de las funciones generales. Son los encargados de configurar nuevos dispositivos y permite o deniega la creación de usuarios.
- Normales: tendrán acceso a las funcionalidades generales del programa, como visualizar los estados y estadísticas, además de tener control sobre el encendido y apagado de las luces.

2.3 Restricciones generales

El producto deberá comunicarse con dispositivos GPS para obtener información posicional.

- El sistema necesitará siempre estar conectado a la red.
- El sistema necesitará contar con alimentación eléctrica para que el hardware se comunique correctamente con el software.
- Debe existir una correcta comunicación entre el hardware ya que sin él la interfaz deja de ser funcional ya que no tiene datos sobre los cuales basarse.

3 Requisitos específicos

3.1 Interfaces externas

En este proyecto, tenemos por un lado la placa con los sensores que funcionarán como servidor y también como cliente. ¿Cómo es esto? Bien, lo que sucede al momento de querer encender o apagar remotamente las luces, por parte de la aplicación se hace una petición a la placa por medio de una comunicación a su IP enviando la orden de que se cambie el estado del relé. Y por otra parte, la placa estará enviando periódicamente al servidor de Django las lecturas del sensor.

El Backend se encargará de recibir estas lecturas para detectar si los datos se encuentran dentro de un umbral en cual se establece si las luces están encendidas o apagadas por medio de estados 1 o 0. Cuando ocurra un cambio de estado (de 1 a 0 o viceversa) se procede a guardar dicho estado, la hora en que finalizó y la hora en que comenzó un nuevo ciclo. Con la duración de estos ciclos, se calculará diariamente el consumo lumínico de cada aula (en horas) y se mostrará en pantalla mediante un gráfico.

Además en medio de los ciclos que indican las luces encendidas, se mostrará por pantalla el tiempo transcurrido desde que justamente se encendieron.

3.2 Funciones

RF1- El sistema debe permitir al administrador crear y administrar usuarios.

RF2- El sistema debe permitir al administrador crear nuevas aulas.

RF3- El sistema debe permitir al administrador administrar sensores de cada aula.

RF4- El sistema debe recibir los estados de las luces.

RF5- El sistema debe permitir cambiar el estado de los relés.

RF6- El sistema debe permitir generar datos estadísticos del consumo diario de las luces (en horas).

RF7- El sistema debe registrar cuando ocurra un cambio de estado de las luces detectado por los sensores.

RF8- El sistema debe registrar el cambio de estado de los relé entendiendo si se accionó el encendido o apagado de las luces.

RF9- El sistema debe registrar las interacciones de los usuarios (quién, cuándo y qué hizo).

RF10- El sistema debe permitir al administrador filtrar las aulas registradas mediante un buscador

RF11- El sistema debe permitir el logueo de usuarios y distinguir su rol.

RF12- El sistema debe tener una barra de navegación que concentre las funcionalidades del mismo en 3 categorías (estadísticas, panel principal y opciones de usuarios).

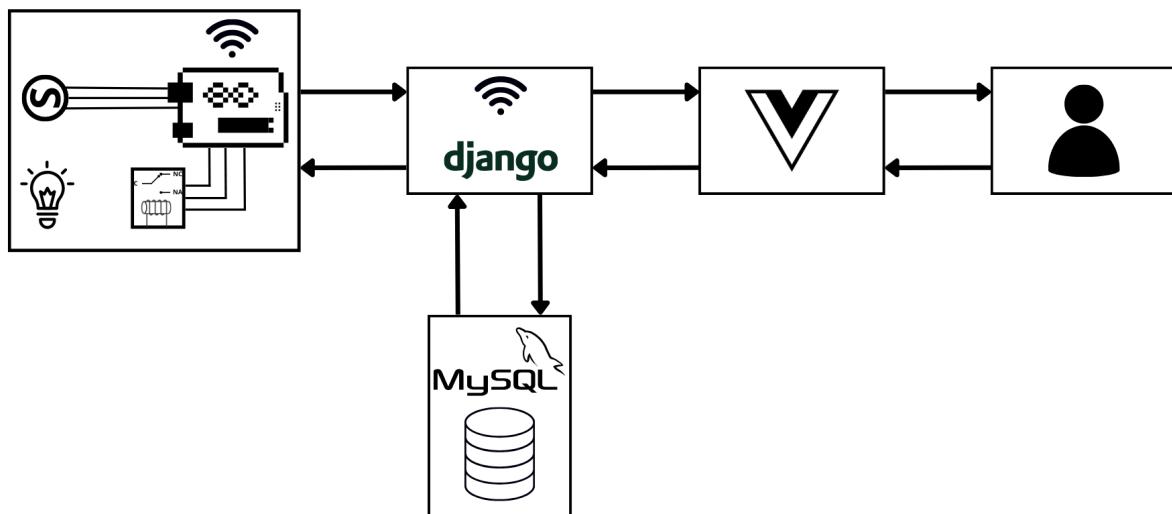
3.3 Requerimientos no funcionales

Rendimiento (velocidad respuesta): La respuesta ante cualquier interacción no debe superar los 10 segundos.

Tanto el servidor Django como la paca deben estar conectadas permanentemente a internet y tener suministro eléctrico.

Es importante que sólo tengan acceso usuarios autorizados.

3.4 Arquitectura

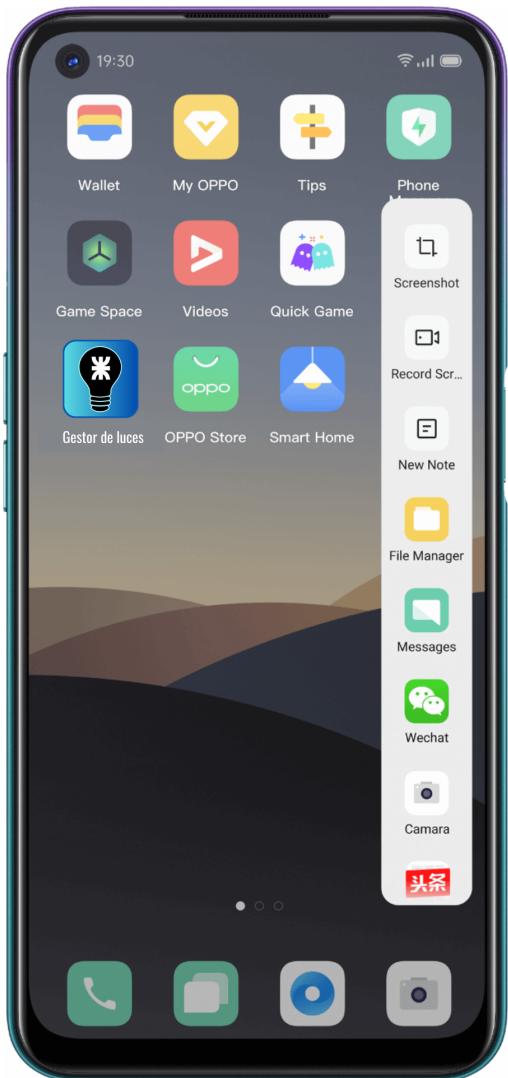


Link:

https://www.canva.com/design/DAFyZg8IMr4/NC9vKk3Bhm1uLCZkuxrs2w/view?utm_content=DAFyZg8IMr4&utm_campaign=designshare&utm_medium=link&utm_source=editor

3.5 Apéndices

Prototipo de interfaz:





Gestion de luces

Buscar

Aula 113 Hace 2:00 hs. Apagar

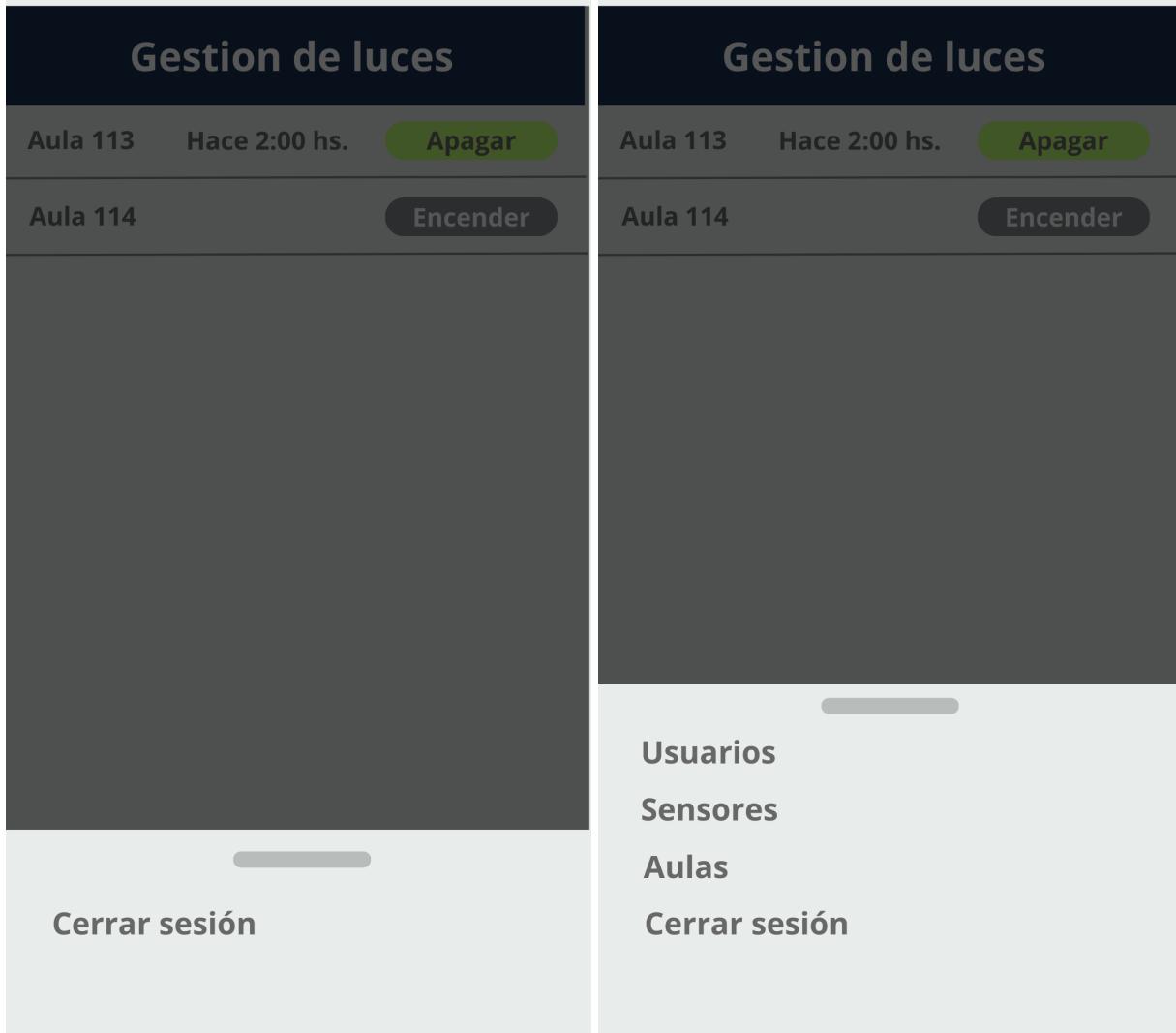
Aula 114 Encender

Aula 115 Hace 2:00 hs. Apagar

Aula 116 Encender

Upss!
Hubo un error
en el sistema

Aceptar



Usuarios

Agregar nuevo usuario +

Buscar 🔍

Guido Brugiatelli	Inactivo
Julian Sanmartino	Activo
Lucia karlen	Inactivo

Nuevo usuario

Nombre de usuario

Contraseña

Nombre

Apellido

Email

Administrador

Si

No

Guardar

Nuevo usuario

Nombre de usuario
Contraseña
Nombre

Error
Datos inválidos

Aceptar

Guardar



Editar usuario

GBrugiatelli
1234
Guido
Brugiatelli
Guidobrugiatelli@example.com

Administrador

Si
 No

Guardar



Editar usuario

GBrugiatelli
1234
Guido

Error Datos inválidos

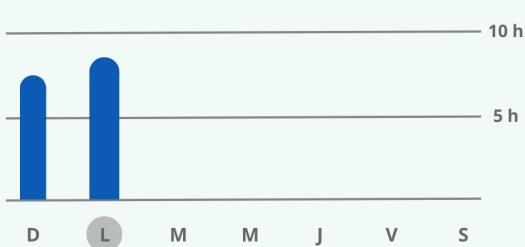
Aceptar

Guardar



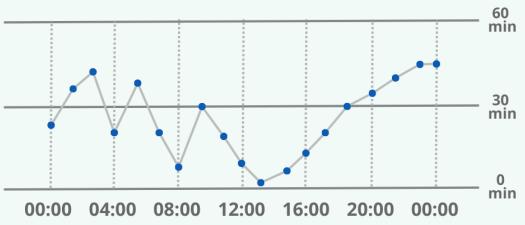
Estadísticas

8 h 20 min
Promedio semanal de uso de luces



Lunes

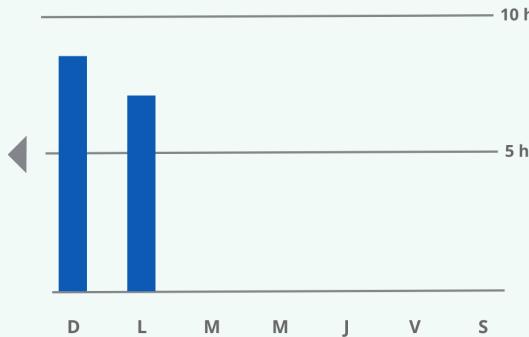
8 h 20 min
Promedio diario de uso de luces



Estadísticas

8 h 20 min

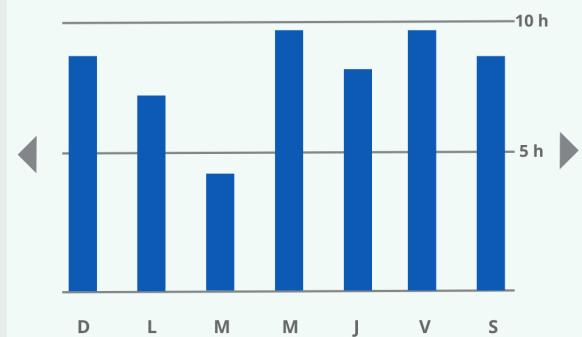
Promedio semanal de uso de luces - Semana actual



Estadísticas

9 h 25 min

Promedio semanal de uso de luces - 05/11/2023 - 11/11/2023



Estadísticas - Aula115

8 h 20 min
Promedio semanal de uso de luces

D	L	M	M	J	V	S
~4.5 h	~9.5 h	~4.5 h	~9.5 h	~4.5 h	~4.5 h	~4.5 h

Lunes

8 h 20 min
Promedio diario de uso de luces

00:00	04:00	08:00	12:00	16:00	20:00	00:00
~15 min	~30 min	~10 min	~5 min	~15 min	~30 min	~45 min

Aulas

Agregar nueva aula

Nombre del aula

Ingrese IP del Arduino asignado

Agregar

Editar o eliminar aula

Seleccione el aula

Nuevo nombre

Aula115

Aula115

Aula115

Aulas

Editar o eliminar aula

Seleccione el aula

Nuevo nombre

Nuevo IP Arduino

- Sólo completar los campos "Nuevo nombre" y "Nuevo IP Arduino" en caso de edición. Si quiere eliminar, sólo seleccione el aula.

Editar **Eliminar**



Sensores

Añadir sensor

Seleccione el aula

Seleccione el tipo de sensor

Agregar

Eliminar sensor

Seleccione el aula

Aula 107

Sensores que tiene el aula:



Sensores

Eliminar sensor

Seleccione el aula

Aula 107 ▾

Sensores que tiene el aula:

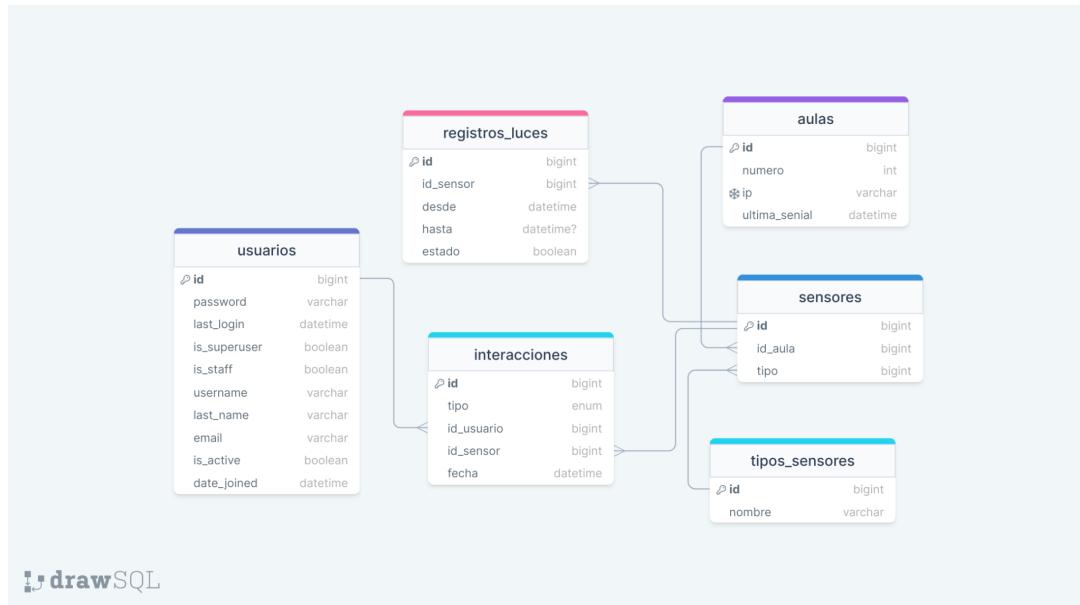
Luz	Eliminar
Relé	Eliminar
Humedad	Eliminar
Ventanas	Eliminar

The interface is a mobile-style card with a dark blue header and footer. The main body is white with rounded corners. It contains a list of sensors with a delete button. The footer has three icons: a grid with two dots, a lightbulb, and a menu icon.

Para visualizar el prototipo puede acceder a:

https://www.canva.com/design/DAFv2_PCI9o/WwYouQo-7iTOfYDPbQ7AIQ/view?utm_content=DAFv2_PCI9o&utm_campaign=designshare&utm_medium=link&utm_source=publishsharelink

Esquema de base de datos:



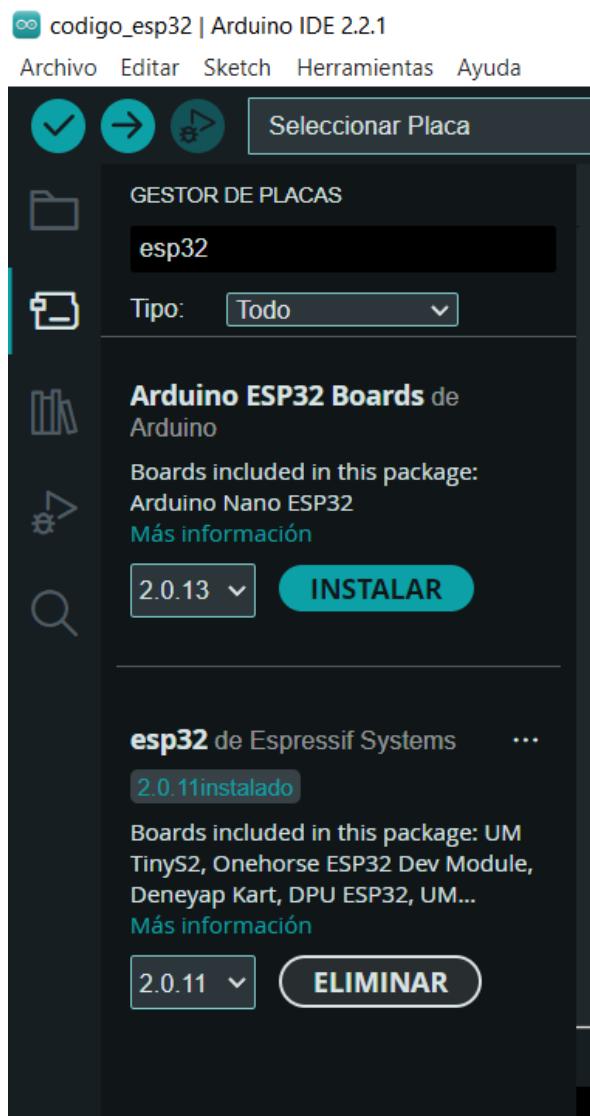
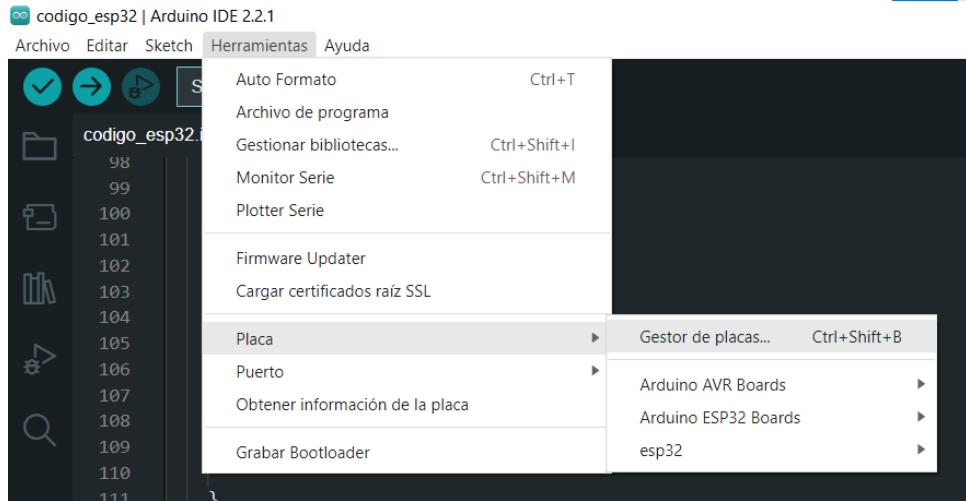
Link: <https://drawsql.app/teams/chulianteam/diagrams/luces>

Problemas y soluciones:

Nos enfrentamos a varios desafíos para llevar adelante el proyecto, a continuación se detallarán los más importantes:

- **Seleccionar placa en IDE Arduino:**

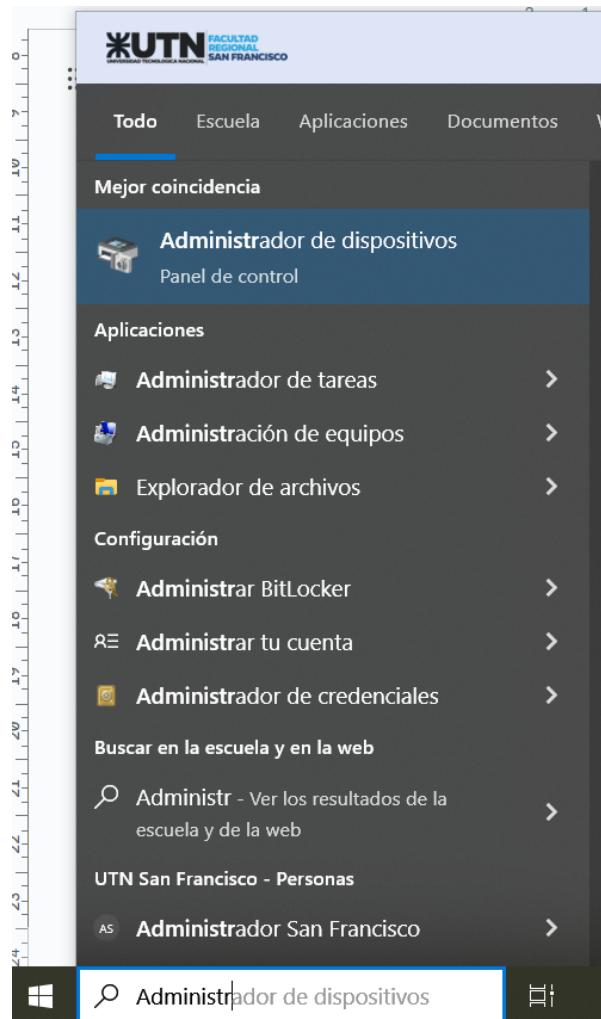
En primer lugar, es necesario saber que de manera predeterminada el IDE de Arduino no reconocerá automáticamente cuando conectemos al puerto usb de nuestro ordenador la ESP32. De modo que tenemos que instalar un complemento que sume nuestra placa a las opciones del programa. Para ello debemos ingresar a nuestro IDE de Arduino, en el menú superior elegimos “Herramientas”, en la ventana que se despliega buscamos “Placas”, allí nos saldrá una opción que se llama “Gestor de placas” (o simplemente presionamos el comando CTRL+SHIFT+B). Ahora un menú lateral aparecerá a la izquierda de la pantalla. En el buscador escribimos “esp32” y entre las opciones que aparezcan instalamos la que tenga como autor a “Espressif Systems”. Ahora cuando ingresemos a la opción “placas” que está en “Herramientas” tendremos muchas más opciones para elegir. En nuestro caso nos importaba “ESP32-WROOM-DA Module”.

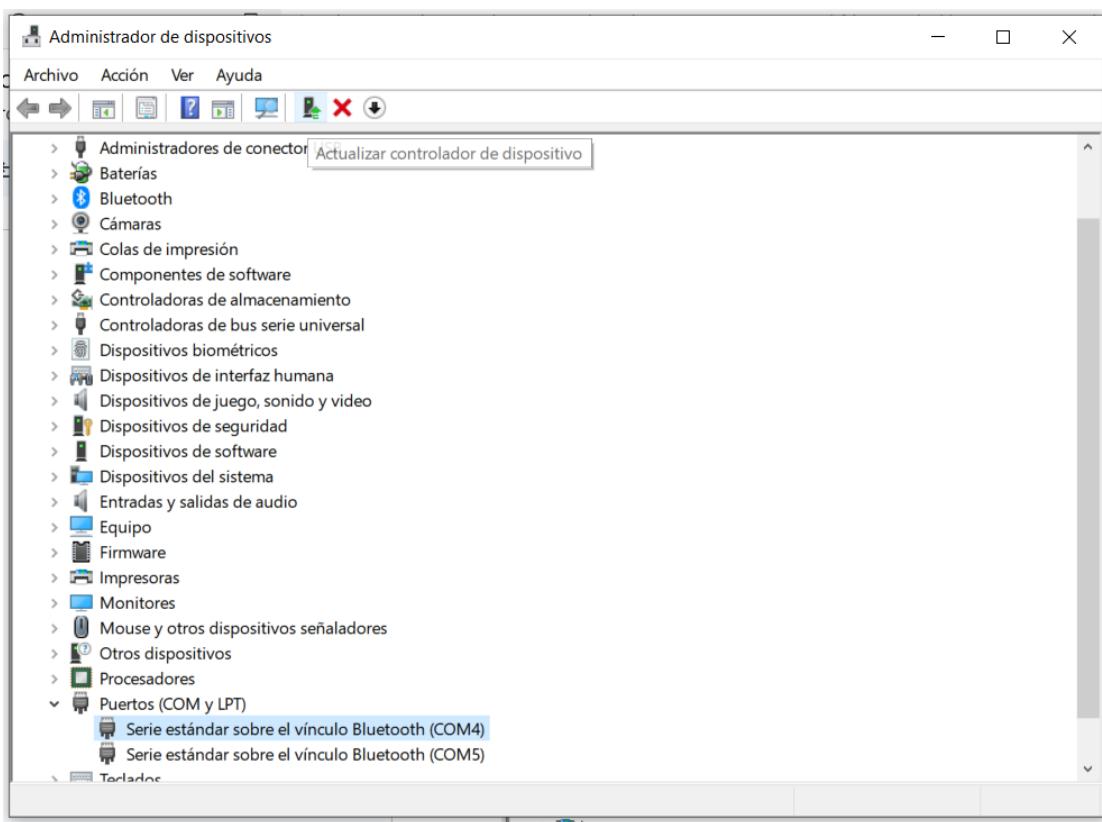
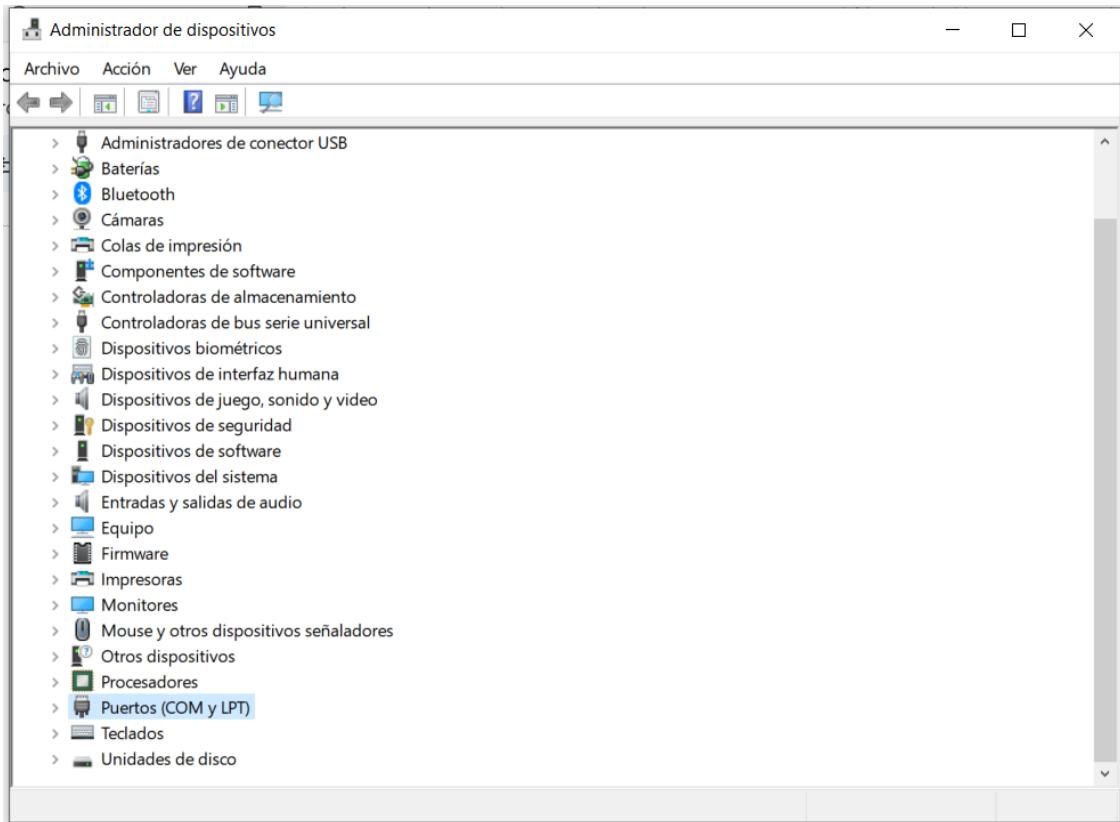


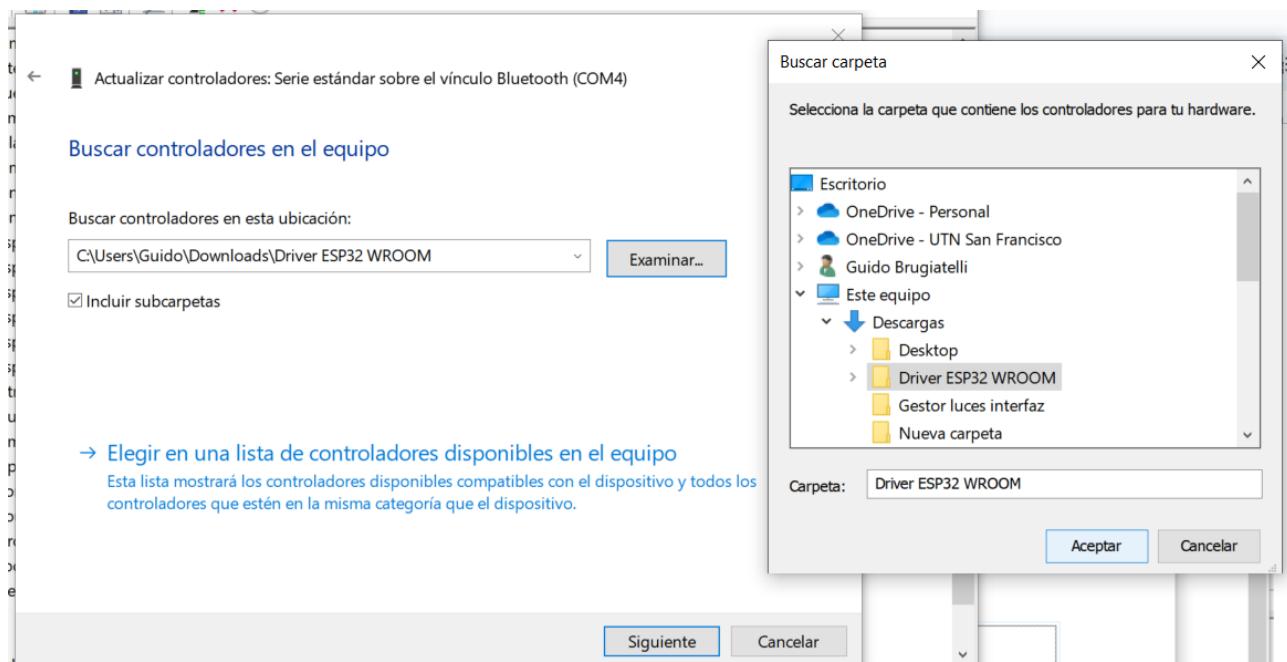
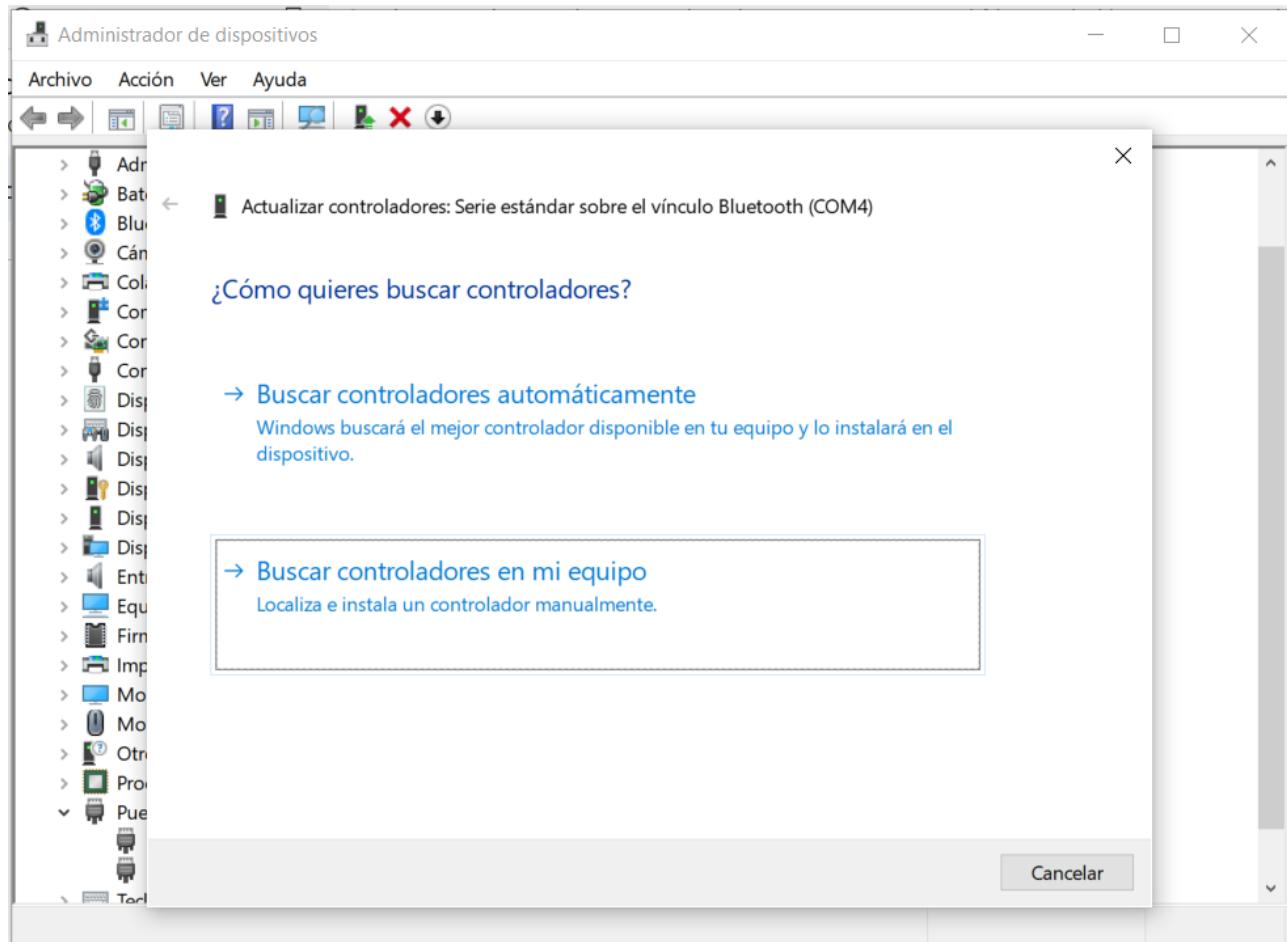
En este caso ya está instalado, pero debe aparecer la opción con el botón azul de instalar.
Otra aclaración importante es que la versión del IDE de Arduino es la 2.2.1

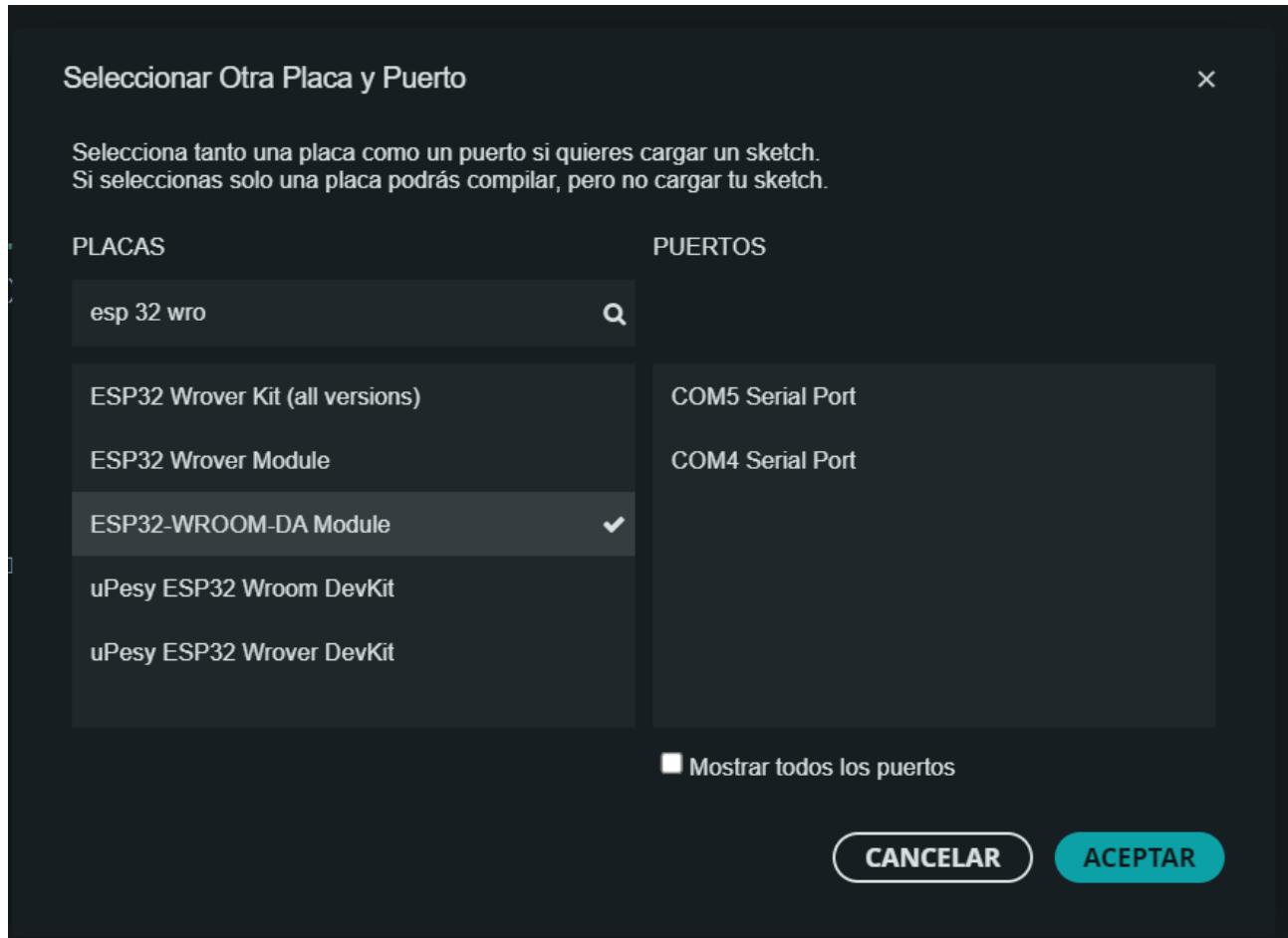
● **Problema de drivers con Windows:**

Al conectar la placa en el IDE de Arduino, no es reconocida, por lo que debemos descargar los controladores que mencionamos en “referencias”. Una vez descargados, conectamos la placa al puerto USB de nuestro ordenador. Abrimos el “Administrador de dispositivos” y allí se despliegan la mayoría de los controladores presentes. En la sección “Puertos(COM y LP)” clickeamos y vemos las opciones, entre ellas debería aparecer una con un icono triangular amarillo en símbolo de que algo no funciona bien. Hacemos un click sobre ese dispositivo y elegimos el botón “Actualizar controlador de dispositivo”. La ventana emergente nos preguntará si queremos que se busque automáticamente o buscamos en nuestro ordenador, seleccionamos esta última. Allí debemos buscar la ubicación donde tenemos los controladores descargados al comienzo del proceso (previamente descomprimido en una carpeta). Luego damos en aceptar y se instalará sólo. A partir de allí debemos configurar un puerto con nuestra placa eligiendo “ESP32-WROOM-DA Module” y ya con eso será suficiente para solucionar el problema.









Código:

Para acceder al código del proyecto, deben ingresar al siguiente link de github: <https://github.com/jisanmartino03/gestion-luces.git>

Aquí está todo el material de Backend, Frontend y Arduino. La metodología de trabajo fue en división de áreas (Back, Front y Arduino) con asignación de tareas que debían revisarse y aprobarse por parte de algún miembro del grupo antes de que los cambios se vuelvan parte del producto final.