

ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

SIMULACIÓN DE UN BASURERO AUTOMATIZADO PARA DESARROLLAR HÁBITOS EN PERSONAS CON AUTISMO EN EL CENTRO TERAPÉUTICO VOCES

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO EN ELECTRÓNICA Y TELECOMUNICACIONES

STEVEN ANDRÉS PANCHI ZAPATA

steven.panchi@epn.edu.ec

JEAN ALEJANDRO SANTAFÉ GUALA

Jean.santafe@epn.edu.ec

DIRECTOR: Ing. FANNY PAULINA FLORES ESTÉVEZ, MSc.

fanny.flores@epn.edu.ec

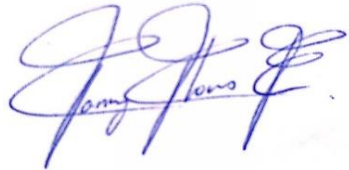
CODIRECTOR: Ing. MÓNICA DE LOURDES VINUEZA RHOR, MSc.

monica.vinueza@epn.edu.ec

Quito, Noviembre 2020

CERTIFICACIÓN

Certificamos que el presente trabajo fue desarrollado por Steven Andrés Panchi Zapata y Jean Alejandro Santafé Guala, bajo supervisión.



Ing. Fanny Flores, Msc.

DIRECTOR DE PROYECTO

Ing. Mónica Vinueza Rhor, Msc.

COORDIRECTOR DE PROYECTO

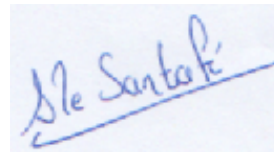
DECLARACIÓN

Nosotros, Steven Andrés Panchi Zapata y Jean Alejandro Santafé Guala, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría, que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

Sin perjuicios de los derechos reconocidos en el primer párrafo del artículo 114 del Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación -COESC-, somos titulares de la obra en mención y otorgamos una licencia gratuita, intransferible y no exclusiva de uso con fines académicos a la Escuela Politécnica Nacional. Entregaremos toda la información técnica pertinente. En el caso de que hubiese una explotación comercial de la obra por parte de la EPN, se negociará los porcentajes de los beneficios conforme lo establece la normativa vigente.



Steven Andrés Panchi Zapata



Jean Alejandro Santafé Guala

DEDICATORIA

El presente trabajo de titulación se lo dedico a las personas que conforman mi familia, mis padres, quienes me han apoyado y guiado durante toda mi carrera estudiantil, a mi hermana por ser mi ejemplo profesional. Y a todos los amigos que he hecho a lo largo de este trayecto.

Steven Panchi

Esta tesis va dedicada con todo mi amor y cariño a mis padres Jorge Santafé y Patricia Guala quienes con su esfuerzo y sacrificio me apoyaron durante toda mi vida estudiantil, gracias a su apoyo incondicional me ayudaron a terminar de mejor manera este proyecto. A mi hermano Josué a quien debo la inspiración en el desarrollo de este dispositivo.

A mi familia paterna y materna que me brindaron apoyo en cada momento. Muchas gracias a todos.

Alejandro Santafé

AGRADECIMIENTO

A Dios que nos brinda guía moral y espiritual en nuestras vidas, por darnos el regalo de la vida y proteger a cada uno de nosotros.

Un agradecimiento especial a la Escuela Politécnica Nacional, por brindarnos la oportunidad de estudiar en esta noble institución y conseguir un título profesional con el cual podremos alcanzar nuestros sueños siendo entes productivos para la sociedad.

A nuestra directora la Ing. Fanny Flores, por la confianza depositada en nosotros, sus enseñanzas y paciencia durante nuestra vida académica. Especialmente por creer en nosotros.

A nuestra codirectora la Ing. Mónica Vinueza, quien nos brindó conocimiento y valores para ser los mejores profesionales.

Steven Panchi, Alejandro Santafé

ÍNDICE DE CONTENIDO

CERTIFICACIÓN	I
DECLARACIÓN	II
DEDICATORIA.....	III
AGRADECIMIENTO	IV
ÍNDICE DE CONTENIDO.....	V
ÍNDICE DE FIGURAS	VIII
ÍNDICE DE TABLAS.....	XI
RESUMEN.....	XII
ABSTRACT	XIII
1 INTRODUCCIÓN	1
1.1 Marco Teórico	2
<i>Software</i>	<i>2</i>
<i>Arduino.....</i>	<i>2</i>
<i>Arduino IDE.....</i>	<i>3</i>
<i>Nextion Editor.....</i>	<i>3</i>
<i>Configure Virtual Serial Port Driver</i>	<i>4</i>
<i>Proteus.....</i>	<i>4</i>
HTML.....	5
SMTP	5
<i>Adobe Illustrator.....</i>	<i>5</i>
<i>Adobe After Effects.....</i>	<i>6</i>
<i>Adobe Photoshop</i>	<i>6</i>
<i>Hardware.....</i>	<i>6</i>
Módulo <i>Arduino Nano</i>	6
ESP8266 <i>NodeMCU V3</i>	7
Pantalla <i>Nextion 7.0"</i>	9
Módulo de 1 Relé.....	10

Servomotor SG90	11
Módulo ultrasónico HC-SR04	12
Módulo sensor LDR	13
Módulo KY-008	14
Luces piloto <i>LED</i>	14
Resistor	15
Transistor 2N2222A.....	15
2 METODOLOGÍA	16
2.1 Método Cualitativo	16
2.2 Método Comparativo	17
2.3 Método Analítico	17
2.4 Método Experimental	17
3 RESULTADOS Y DISCUSIÓN	19
3.1 Autismo y tratamientos	19
Método ABA.....	19
PECS.....	19
Método TEACCH	19
Pictograma	20
3.2 Análisis	20
3.3 Diseño	22
Estructura de los contenedores	23
Diseño de la conexión del sensor de nivel	29
Diseño de la conexión del sensor ultrasónico	31
Diseño de la conexión de la pantalla	32
Diseño de la conexión del ESP8266 <i>NodeMCU V3</i>	33
Diseño de la conexión de los servomotores.....	34
Diseño de la conexión de las luces piloto.....	36
Circuito electrónico completo.....	38
Diagramas de flujo	41

Ruteo del circuito.....	46
3.4 Desarrollo de la simulación.....	50
Uso del sensor ultrasónico para detectar presencia	50
Configuración de los servomotores	51
Librería <i>Nextion Editor</i>	51
Configuración de la pantalla <i>Nextion</i>	52
Estado de activación	60
Estado de interrupción	61
Estado de reposo	62
Librería <i>Gsender</i>	63
Desarrollo del programa principal para el ESP8266 <i>NodeMCU V3</i>	65
Programa adicional para la conexión con la simulación	69
Imágenes presentadas en pantalla.....	69
3.5 Pruebas de funcionamiento	72
Prueba de animación en la pantalla <i>Nextion</i>	72
Estados del sistema automatizado	73
Activación.....	75
Interrupción	79
Reposo.....	81
3.6 Costos de implementación	82
4 CONCLUSIONES Y RECOMENDACIONES.....	83
4.1 Conclusiones	83
4.2 Recomendaciones	85
5 REFERENCIAS BIBLIOGRÁFICAS	87
ANEXOS.....	92

ÍNDICE DE FIGURAS

Figura 1.1: Interfaz del <i>software</i> <i>Nextion Editor</i>	3
Figura 1.2: <i>Virtual Serial Port Driver</i>	4
Figura 1.3: Placa de circuito impreso 3D	4
Figura 1.4: Módulo <i>Arduino Nano</i>	6
Figura 1.5: Módulo <i>Wi-Fi</i> ESP-12E	7
Figura 1.6: SoC ESP8266	8
Figura 1.7: Esquema general <i>NodeMCU</i>	8
Figura 1.8: <i>NodeMCU V3</i>	8
Figura 1.9: Pantalla <i>Nextion 7.0"</i>	9
Figura 1.10: Módulo de 1 relé	10
Figura 1.11: Estructura de un servomotor	11
Figura 1.12: Sensor ultrasónico HC-SR04	12
Figura 1.13: Módulo sensor LDR	13
Figura 1.14: Módulo KY-008	14
Figura 1.15: Luces piloto <i>LED</i>	14
Figura 1.16: Resistencia de película de carbono.....	15
Figura 2.1: Proceso metodológico del proyecto	16
Figura 3.1: Esquema del basurero automatizado.....	21
Figura 3.2: Ubicación física del dispositivo en el Centro Voces.....	22
Figura 3.3: Diagrama general del sistema	22
Figura 3.4: Ubicación de los sensores y actuadores en la parte externa	23
Figura 3.5: Ubicación del sensor de nivel en la parte interna	24
Figura 3.6: Estructura interna.....	24
Figura 3.7: Vista 3D de la estructura	25
Figura 3.8: Plano de la parte superior.....	25
Figura 3.9: Plano de los laterales.....	26
Figura 3.10: Plano de la parte posterior.....	26
Figura 3.11: Plano de la parte frontal.....	27
Figura 3.12: Plano de la base con guía	27
Figura 3.13: Plano vista lateral de la base con guía.....	28
Figura 3.14: Planos superior y frontal de la rampa.....	28
Figura 3.15: Plano de los laterales de la rampa	29
Figura 3.16: Conexión del sensor de nivel de manera física	29
Figura 3.17: Conexión del sensor de nivel en la simulación	30

Figura 3.18: Conexión del sensor ultrasónico de manera física	31
Figura 3.19: Conexión del sensor ultrasónico en la simulación	32
Figura 3.20: Conexión de la pantalla de manera física	32
Figura 3.21: Conexión de la pantalla en la simulación	33
Figura 3.22: Conexión del ESP8266 <i>NodeMCU V3</i> de manera física	33
Figura 3.23: Conexión del ESP8266 <i>NodeMCU V3</i> en la simulación.....	34
Figura 3.24: Conexión de los servomotores de manera física	35
Figura 3.25: Conexión de los servomotores en la simulación.....	35
Figura 3.26: Conexión de las luces piloto de manera física	36
Figura 3.27: Conexión de las luces piloto en la simulación.....	37
Figura 3.28: Circuito electrónico completo usando módulos.....	39
Figura 3.29: Circuito electrónico completo para la simulación	40
Figura 3.30: Diagrama de flujo general del sistema	41
Figura 3.31: Diagrama de flujo de la proyección de imágenes	42
Figura 3.32: Diagrama de flujo de la conexión a la red.....	43
Figura 3.33: Diagrama de flujo del envío de correo electrónico.....	43
Figura 3.34: Diagrama de flujo comunicación entre simulación y <i>NodeMCU V3</i>	44
Figura 3.35: Diagrama de flujo del estado de reposo.....	44
Figura 3.36: Diagrama de flujo del estado de activación	45
Figura 3.37: Diagrama de flujo del estado de bloqueo.....	46
Figura 3.38: Ruteo de la placa	47
Figura 3.39: Visualización 3D de la placa.....	47
Figura 3.40: Placa para el sensor ultrasónico	48
Figura 3.41: Placas para el sensor de nivel.....	48
Figura 3.42: Placas para los servomotores	48
Figura 3.43: Placa para las luces piloto	49
Figura 3.44: Placa para la alimentación.....	49
Figura 3.45: Placa para la pantalla	49
Figura 3.46: Creación de nuevo proyecto en <i>Nextion Editor</i>	53
Figura 3.47: Guardado de proyecto en <i>Nextion Editor</i>	53
Figura 3.48: Selector de pantalla en <i>Nextion Editor</i>	54
Figura 3.49: Selección de la pantalla de 7"	54
Figura 3.50: Configuración de la orientación de la pantalla	55
Figura 3.51: Lugar de trabajo en <i>Nextion Editor</i>	55
Figura 3.52: Adición de páginas en <i>Nextion Editor</i>	56
Figura 3.53: Adición de imágenes en <i>Nextion Editor</i>	56

Figura 3.54: Elección de página para edición en <i>Nextion Editor</i>	57
Figura 3.55: Adición de imagen de fondo en <i>Nextion Editor</i>	57
Figura 3.56: Selección de imagen de fondo en <i>Nextion Editor</i>	58
Figura 3.57: Establecimiento del brillo de la pantalla en <i>Nextion Editor</i>	58
Figura 3.58: Adición de <i>timer</i> en <i>Nextion Editor</i>	59
Figura 3.59: Configuración del <i>timer</i> en <i>Nextion Editor</i>	59
Figura 3.60: Configuración del tiempo de transición de imágenes	60
Figura 3.61: Archivos de la librería <i>Gsender</i>	63
Figura 3.62: Archivos <i>Gsender</i> en <i>Arduino IDE</i>	63
Figura 3.63: Configuración en la cuenta de <i>Gmail</i>	64
Figura 3.64: Acceso del <i>NodeMCU V3</i> a la cuenta de <i>Gmail</i>	64
Figura 3.65: Mesa de trabajo de <i>Adobe Illustrator</i>	70
Figura 3.66: Editor de <i>Adobe After Effects</i>	71
Figura 3.67: Imagen del estado de interrupción	71
Figura 3.68: Mesa de trabajo de <i>Adobe Photoshop</i>	72
Figura 3.69: Ubicación de la herramienta <i>debug</i> en <i>Nextion Editor</i>	72
Figura 3.70: Ventana <i>debug</i> de <i>Nextion Editor</i>	73
Figura 3.71: Virtualización de puertos	74
Figura 3.72: Puertos virtuales pares	74
Figura 3.73: Sensor ultrasónico detecta un usuario en el perímetro	75
Figura 3.74: Motores cerrando la tapa.....	76
Figura 3.75: Motores abriendo la tapa	76
Figura 3.76: Sensor de nivel detecta el haz de luz.....	77
Figura 3.77: <i>COMPIM</i> para la comunicación con la pantalla	77
Figura 3.78: <i>COMPIM</i> para la comunicación con el módulo <i>Wi-Fi</i>	78
Figura 3.79: Luz verde en la etapa de activación	78
Figura 3.80: Luz roja en la etapa de activación	79
Figura 3.81: Sensor de nivel detecta que no hay haz de luz	79
Figura 3.82: Mensaje del monitor serial al enviar el correo.....	80
Figura 3.83: Luz roja en la etapa de interrupción	80
Figura 3.84: Correo que recibe el administrador	81
Figura 3.85: Estado de reposo.....	81
Figura 3.86: Luz verde en el estado de reposo	82

ÍNDICE DE TABLAS

Tabla 1.1: Características del módulo <i>Arduino Nano</i>	7
Tabla 1.2: Características del <i>NodeMCU V3</i>	9
Tabla 1.3: Características de la pantalla <i>Nextion 7 “</i>	10
Tabla 1.4: Características del módulo de 1 relé.....	11
Tabla 1.5: Características del servomotor SG90	11
Tabla 1.6: Características del sensor ultrasónico HC-SR04	13
Tabla 1.7: Características del módulo sensor LDR.....	13
Tabla 1.8: Características del módulo KY-008.....	14
Tabla 1.9: Características de la luz piloto <i>LED</i>	15
Tabla 1.10: Características del transistor 2N2222A	15
Tabla 3.1: Consumo energético	23
Tabla 3.2: Relación porcentaje-distancia de la librería ultrasónico	32
Tabla 3.3: Relación porcentaje-distancia del ultrasónico en la simulación	75
Tabla 3.4: Costos de implementación	82

RESUMEN

En el presente trabajo se exhibe el diseño y simulación de un sistema electrónico orientado a desarrollar hábitos en personas con autismo, el cual ha sido diseñado en base a los requerimientos del Centro Terapéutico Voces. El trabajo comprende el uso de programas tales como *Proteus*, *Nextion Editor*, *Virtual Serial Port Driver* y *Arduino IDE*. El sistema se compone de un sensor de presencia ultrasónico, un sensor de nivel LDR, servomotores para controlar la apertura y cierre de la tapa, luces piloto, un módulo *Wi-Fi* para el envío de correo electrónico, y una pantalla LCD de 7”.

El proyecto se estructura en 5 secciones. Los contenidos de cada una se indican a continuación:

En la primera sección se presenta una introducción que abarca la justificación del proyecto, así como una revisión de la teoría necesaria.

En la segunda sección se presenta la metodología utilizada en cada una de las fases del desarrollo.

En la tercera sección se presenta un análisis de los tratamientos para personas con TEA utilizados en el diseño, los cálculos necesarios de los componentes y la descripción de los algoritmos para cumplir los requerimientos planteados, así como los resultados obtenidos de la simulación.

En la cuarta sección se presentan las conclusiones y recomendaciones obtenidas en el desarrollo del proyecto.

Finalmente, la quinta sección presenta dos manuales, uno de usuario y uno de administrador; detallando en el manual de administrador el procedimiento para efectuar cambios en el proyecto según se requiera.

Palabras clave: Autismo, automatización, correo electrónico, *Nextion Editor*, Arduino.

ABSTRACT

In the present document the design and simulation of an electronic system aimed at developing habits in people with autism, which has been designed based on the requirements of the Voces Therapeutic Center, is presented. The paper includes the use of programs such as Proteus, Nextion Editor, Virtual Serial Port Driver and Arduino IDE. The system consists of an ultrasonic presence sensor, an LDR level sensor, servomotors to control the opening and closing of the lid, pilot lights, a Wi-Fi module for sending e-mails, and a 7" LCD screen.

The project is structured in 5 sections. The contents of each one is indicated below:

The first section presents an introduction, covering the justification for the project, as well as a review of the necessary theory.

The second section presents the methodology used in each of the development phases.

The third section presents an analysis of the treatments for people with ASD used in the design, the necessary calculations of the components and the description of the algorithms to meet the requirements, as well as the results obtained from the simulation.

The fourth section presents the conclusions and recommendations obtained in the development of the project.

Finally, the fifth section presents two manuals, one for the user and one for the administrator; detailing in the administrator manual the procedure to make changes to the project as required.

Key words: *Autism, automation, e-mail, Nextion Editor, Arduino.*

1 INTRODUCCIÓN

En la actualidad, los diagnósticos de personas con TEA (Trastorno del Espectro Autista) permiten tener una perspectiva más real de la población con este síndrome. Uno de los centros en el país que está a la vanguardia, en cuanto a las terapias, es “Voces”. Este centro emplea métodos y técnicas educativas como TEACCH (*Training and Education of Autistic and Related Communication Handicapped Children*), PECS (*Picture Exchange Communication System*) y ABA (*Applied Behavior Analysis*), [1] en los cuales se prioriza el uso de material concreto al momento de desarrollar las terapias conductuales que imparten los especialistas. Por tanto, esta institución cuenta con las condiciones necesarias para que se implemente en ella un sistema automatizado que permitirá a niños de entre 3 y 6 años de edad, adquirir el aprendizaje de la rutina de desechar la basura reciclable en el lugar correspondiente, motivando así el desarrollo de una nueva tarea aprendida y la conservación del medio ambiente.

El sistema automatizado cuenta con componentes como: pictogramas, luces y sensores empleados en la simulación. Los pictogramas establecen la comunicación entre el usuario y el contenedor. Las luces son indicadores del tiempo con que se cuenta para realizar la acción. Los sensores activan las cubiertas e indican al administrador del Centro que el contenedor ha alcanzado su máxima capacidad de almacenamiento. Dicho sistema les permitirá a los niños con TEA desarrollar destrezas de comunicación, orden, clasificación, entre otras, para que puedan mejorar su calidad de vida.

Además, se debe considerar que los aportes de la tecnología para personas con TEA son muy escasos en el país. En la actualidad este tema ha tenido una notable difusión por organismos como APADA (Asociación de Padres para el Apoyo y la Defensa de Personas con Autismo), FEDEA (Federación Ecuatoriana del Espectro Autista), entre otros. [1]

Es imprescindible destacar que la condición de las personas con autismo difiere mucho en relación con las personas “regulares”. Actividades como: amarrarse el calzado, botar o reciclar basura, identificar sitios de uso cotidiano en la casa (baño, cocina, anaquel, entre otros) presentan mayor dificultad y exigen un notable esfuerzo para las personas con TEA. [1]

Hoy se cuenta con muchos estudios sobre el tema, los cuales determinan que los niños con TEA tienen más desarrollado el sentido de la vista. [2] Esta condición permite usar pictogramas en los tratamientos o terapias en muchos aspectos de su vida. El uso del pictograma responde a un diagnóstico individual de cada niño; puesto que, si en este

recurso se modifica significativamente su estructura, se puede originar un obstáculo para que este cumpla su función. [2] [3] Razón por la cual, la selección de los pictogramas en el sistema automatizado se coordinó con los profesionales del centro Voces.

Asimismo, cabe mencionar que el avance vertiginoso de la tecnología ha permitido que se desarrollen nuevos sistemas que ayuden a las personas con TEA, uno de ellos es conocido como “Interacción Natural”, en el cual el usuario usa los movimientos de su cuerpo para comunicarse con la computadora. [2] [3] En este proyecto, cada contenedor cuenta con una pantalla, la cual presenta dos pictogramas que reemplazan textos: el primero con una imagen en movimiento que comunica el tipo de desechos que se pueden depositar y un segundo con una imagen que comunica que el basurero está lleno. Es decir, se emplea la comunicación bidireccional entre el niño con autismo y el sistema automatizado a través de pictogramas, técnica ampliamente utilizada en el tratamiento de TEA.

En definitiva, los profesionales del centro Voces tienen un papel fundamental en el manejo del dispositivo electrónico diseñado. Puesto que, el sistema automatizado se constituye también en parte de una estrategia lúdica que les permitirá a los niños de ese centro familiarizarse con elementos y situaciones del entorno y de su contexto familiar.

El basurero automatizado es un dispositivo nuevo y de fácil utilización. Su costo es relativo, si se consideran los beneficios que brinda dicho dispositivo. [1] [4] Con este proyecto del sistema automatizado se presenta una alternativa nueva de apoyo para el trabajo con los niños con síndrome del espectro autista. Alternativa que, con seguridad, vislumbra un panorama en la investigación y producción de nuevos y mejores sistemas que fortalecerán el desarrollo de las habilidades en estos niños.

1.1 Marco Teórico

Software

- **Arduino**

Es un conjunto de placas de desarrollo y bibliotecas de *software* de código abierto, fáciles de usar en distintas aplicaciones o entornos interactivos. *Arduino*, al ser de código abierto, presenta soporte para una gran gama de sensores y actuadores que pueden ser conectados fácilmente a las placas de desarrollo. [5]

Los módulos Arduino se basan en la familia de microcontroladores ATMEL, donde el microcontrolador es programable por un conjunto de instrucciones que utilizan el *Arduino Language Programming*, basado en *Wiring*, y el *Arduino Development Enviroment*, basado en *Processing*. [5]

- **Arduino IDE**

Es un programa de código abierto ejecutable en los sistemas operativos *Windows*, *Linux* y *Mac OS*. El *Arduino IDE (Integrated Development Environment)*. Utiliza los lenguajes de programación C y C++ para escribir y cargar código en las diversas placas compatibles con *Arduino*; para ello, este *software* utiliza una conexión USB que sirve de interfaz para enviar y recibir datos del módulo mediante la herramienta "Monitor Serie" con la que viene integrado este *software*. [6] [7]

Arduino IDE incluye una gran variedad de librerías para controlar diversos actuadores y sensores compatibles con *Arduino*; además de ello, admite librerías desarrolladas por la comunidad. [5]

- **Nextion Editor**

El editor de *Nextion* es un *software* que otorga el fabricante de las pantallas *Nextion* para diseñar y simular las interfaces gráficas que serán utilizadas en los proyectos. Para una mejor simulación de las interfaces creadas, el editor permite el uso de comunicación serial con *software* y *hardware*, recurso que brinda una mejor apreciación de cómo se verá implementado. [8] En la Figura 1.1 se observa la interfaz del *software* de *Nextion Editor*.

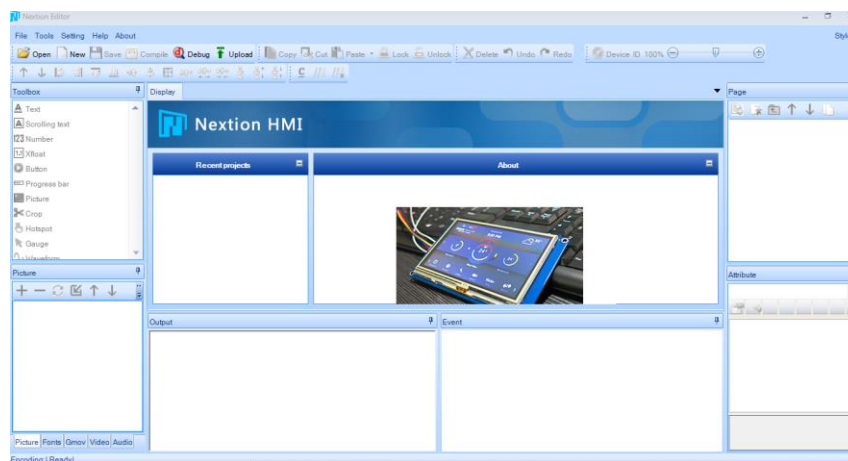


Figura 1.1: Interfaz del *software* *Nextion Editor* [8]

- **Configure Virtual Serial Port Driver**

Es un emulador de puertos seriales, como se observa en la Figura 1.2. Permite al usuario administrar con facilidad, crear y eliminar varios puertos COM virtuales. También, emula totalmente el comportamiento de un puerto serial de comunicaciones. Además, con este programa se pueden crear puertos pares, los cuales se conectan de manera virtual, permitiendo el envío y recepción de datos. [9]

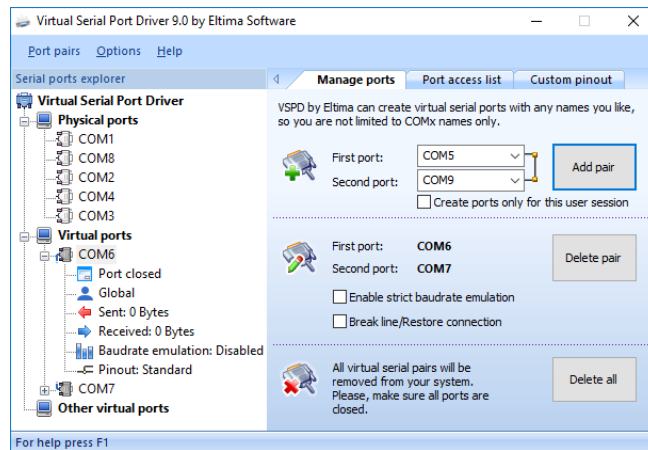


Figura 1.2: Virtual Serial Port Driver [10]

- **Proteus**

Es un entorno de simulación y diseño de circuitos electrónicos desarrollado por *Labcenter Electronics*, el cual permite simular el código de microcontrolador de alto y bajo nivel en el contexto de una simulación de circuito *SPICE (Simulation Program with Integrated Circuits Emphasis)* de modo mixto. [11] Para ello, *Proteus* utiliza los sub-entornos que lo integran, tales como *ISIS (Intelligent Schematic Input System)* para la captura esquemática, *VSM (Virtual System Modelling)* para la simulación en tiempo real y *ARES (Advanced Routing and Editing Software)* para el diseño de placas de circuito impreso. [12] Este último, también permite obtener una imagen del diseño en tres dimensiones, como se muestra en la Figura 1.3.

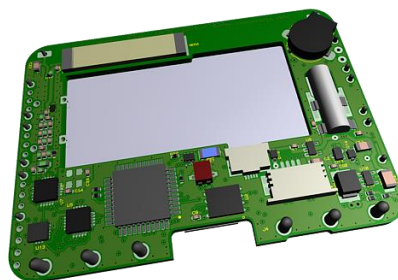


Figura 1.3: Placa de circuito impreso 3D [11]

- **HTML**

HTML (*HyperText Markup Language*), es un lenguaje de marcado utilizado en el desarrollo de páginas *web*. [13]

Sirve para indicar el orden del contenido del sitio *web* por medio de etiquetas conocidas como *tags*; las etiquetas van dentro de los signos mayor que y menor que. En cada una de ellas, se indica el inicio y fin del texto que será generado; esta función permite definir qué sección le corresponde al encabezado, cuerpo del mensaje, entre otros. [13]

- **SMTP**

El SMTP (*Simple Mail Transfer Protocol*) es el estándar básico que usan los servidores de *e-mails* para enviarlos por internet. [14]

Mediante un proceso denominado “almacenar y reenviar”, el SMTP transfiere *e-mail* a través de las redes. Trabaja estrechamente con MTA (*Mail Transport Agent*) para enviar la comunicación al ordenador y a la bandeja de entrada correctos. El MTA es cada aplicación de *software* que usan los clientes de *e-mail* (como *Gmail*, *Outlook*, *Apple Mail*, etc.). [14]

IANA (*Internet Assigned Numbers Authority*) asignó el puerto 465, en un principio, para hacer una versión cifrada del SMTP llamada SMTPS (*Simple Mail Transfer Protocol Secure*). A finales de 1998, la IANA reasignó este número de puerto a un servicio nuevo. Aún así, muchos servicios siguen ofreciendo la interfaz SMTPS en el puerto 465. [14]

El objetivo del puerto 465 es establecer un puerto para que el protocolo SMTP opere con SSL (*Secure Sockets Layer*), que suele usarse para cifrar comunicaciones en Internet. [14]

- **Adobe Illustrator**

Es un programa que permite editar gráficos vectoriales. Esta herramienta opera sobre una mesa llamada “mesa de trabajo”, en la cual se realizan las ediciones artísticas de las imágenes creadas; además, permite modificar tamaño, color, forma, entre otros, según la necesidad de la persona que se encuentre trabajando. [15]

- **Adobe After Effects**

Es una aplicación que permite crear efectos visuales y animaciones. Mediante varios paneles y una línea de tiempo, se pueden agregar imágenes u otros diseños para generar una animación o efecto especial. La superposición de dichas imágenes, produce los efectos deseados que pueden ser *clips* animados, imágenes *gif*, entre otros estilos. [16]

- **Adobe Photoshop**

Es un editor de fotos, el cual permite dar retoques a fotografías y crear imágenes personalizadas con parámetros específicos en cuanto al tamaño de imagen, calidad, resolución, extensión de la misma y peso del archivo. Esta característica es muy importante al trabajar con varios programas que toleren solo cierto tamaño de archivos. [17]

Hardware

- **Módulo Arduino Nano**

Es una placa de desarrollo que utiliza un microcontrolador Atmega328P; este módulo está elaborado en tecnología SMD (*Surface Mount Technology*), tal como se aprecia en la Figura 1.4. El módulo cuenta con 22 pines de entrada y salida digitales, de los cuales 6 son configurables como canales PWM (*Pulse Width Modulation*), 8 admiten entradas analógicas y 2 admiten comunicación UART (*Universal Asynchronous Receiver Transmitter*). [18] El módulo denota los pines de entradas analógicas con la nomenclatura A0-A7 y los pines de comunicación UART como RX0 y TX1. Además, posee un cabezal para conexión ISCP (*In Chip Serial Programmer*), LEDs (*Light Emmiting Diode*) indicadores, un botón de *reset* y un conector mini USB-B hembra. En la Tabla 1.1 se pueden observar más características del módulo. [18] [19]



Figura 1.4: Módulo *Arduino Nano* [18]

Tabla 1.1: Características del módulo *Arduino Nano* [18]

Parámetro	Valor
Voltaje de entrada	7-12 V
Voltaje de funcionamiento	5 V
Memoria <i>flash</i>	32 KB de los cuales 2 KB son utilizados por el gestor de arranque
SRAM	2 KB
Velocidad del reloj	16 MHz
EEPROM	1 KB
Pines digitales E/S	22
Pines de entradas analógicas	8
Salidas PWM	6
Corriente máxima de E/S	40 mA
Tamaño de la placa	18 x 45 mm
Peso	7 g

- **ESP8266 NodeMCU V3**

Es un *kit* de desarrollo de *hardware* abierto con conexión *Wi-Fi*, compatible con el protocolo TCP/IP. Está basado en el módulo *Wi-Fi* ESP-12E, como se puede observar en la Figura 1.5. Está integrado por un SoC (*System on a Chip*) ESP8266, como se presenta en la Figura 1.6; dentro del cual se encuentra un MCU (*Microcontroller Unit*) Tensilica L106. En la Figura 1.7 se observa el esquema general de este kit. [20]



Figura 1.5: Módulo *Wi-Fi* ESP-12E [20]



Figura 1.6: SoC ESP8266 [20]

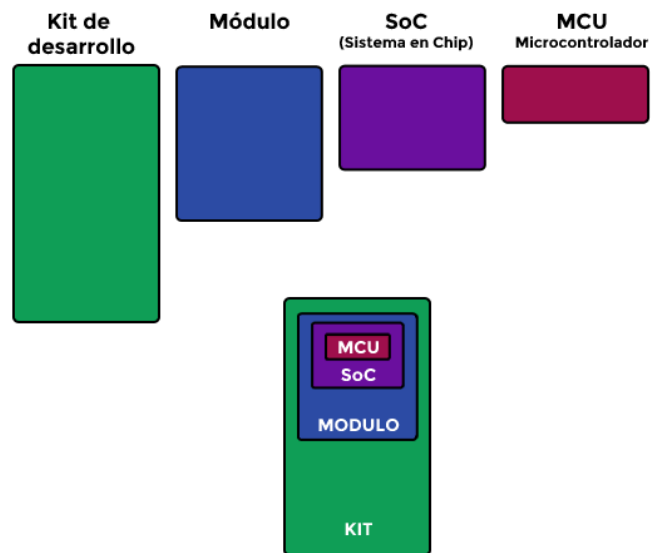


Figura 1.7: Esquema general *NodeMCU* [20]

El ESP8266 *NodeMCU V3*, presentado en la Figura 1.8, cuenta con un puerto micro USB hembra para alimentación y programación, *LEDs* indicadores de estado, un pulsador de *reset* y un pulsador *flash* que debe ser presionado al momento de cargar un *firmware* o código; dicho código puede ser desarrollado mediante el IDE de Arduino. [20] [21] En la Tabla 1.2 se presentan las características del *NodeMCU V3*.



Figura 1.8: *NodeMCU V3* [20]

Tabla 1.2: Características del *NodeMCU V3* [20] [22]

Parámetro	Valor
MCU	Tensilica L106 de 32-bits
SoC	ESP8266 @ 80MHz (3.3V) (ESP-12E)
RAM	50 KB
<i>Flash</i>	4 MB
<i>Wi-Fi</i>	802.11 b/n/g 2.4GHz
Pines de entradas analógicas	Uno de 10 bits (A0)
Pines digitales E/S	9 pines digitales (D0-D8)
Tipo de comunicación	Comunicación UART (Rx-Tx)
Voltaje de entrada	5V
Voltaje de funcionamiento	3.3 V (500mA)
Convertor USB-serial	CH340G
Corriente nominal por pin	12mA
Consumo de corriente de trabajo	90mA
Consumo de corriente modo <i>sleep</i>	8mA

- **Pantalla *Nextion 7.0***

Como se presenta en la Figura 1.9, es una pantalla *LED HMI (Human Machine Interface)*, que une un procesador integrado y una pantalla táctil de memoria junto con el *software Nextion Editor* para el desarrollo de proyectos HMI GUI. La pantalla táctil es del tipo resistivo (RTP), cuenta con una resolución de 800x480 pixeles y se conecta al periférico MCU vía TTL serial (5V, Tx, Rx, GND) para notificar eventos en los que el periférico MCU puede actuar. [23] En la Tabla 1.3 se muestran más características de la pantalla.



Figura 1.9: Pantalla *Nextion 7.0* [23]

Tabla 1.3: Características de la pantalla *Nextion 7* “ [23]

Parámetro	Valor
Color	65536 colores
Tiempo de vida promedio	30000 horas mínimo
Voltaje de operación mínimo	4.75 V
Voltaje de operación recomendado	5 V
Voltaje de operación máximo	7 V
Corriente de operación normal	510 mA
Corriente de operación modo <i>sleep</i>	15 mA
Memoria <i>flash</i>	16 MB
Memoria RAM	3584 Bytes
Dimensiones	181 × 108 × 9 mm
Brillo	230 nits
Peso	268 g

- **Módulo de 1 Relé**

En la Figura 1.10 se aprecia el módulo de 1 relevador, mejor conocido como relé, el cual es capaz de conmutar cargas de potencia de hasta 10 A y 250 VCA (30 VCD). La entrada de control (IN1), está aislada eléctricamente mediante un opto acoplador para minimizar el ruido que percibe el circuito de control, al momento de realizar la conmutación de cargas. Además, posee un LED indicador de estado, que enciende cuando la bobina está activa; y un *jumper* que, al sacarlo, permitirá alimentar el relé independientemente de la parte electrónica del módulo. [24] En la Tabla 1.4 se muestran más características del módulo de un relé.



Figura 1.10: Módulo de 1 relé [24]

Tabla 1.4: Características del módulo de 1 relé [24]

Parámetro	Valor
Voltaje de alimentación de la bobina	5 VCD
Señal de control	3.3 V o 5 V
Activación de la salida NA	0 V
Tiempo de acción NA	10 ms
Tiempo de acción NC	5 ms
Resistencia de bobina	55Ω

- **Servomotor SG90**

Un servomotor, presentado en la Figura 1.11, es un dispositivo conformado por un motor DC, un sistema de engranaje y un circuito de control electrónico que se encarga de manejar el movimiento y la posición del motor DC, mediante una señal PWM a 50HZ proveniente de un microcontrolador o cualquier otro circuito que la pueda generar. [25] [26] En la Tabla 1.5 se muestran las características del motor SG90.

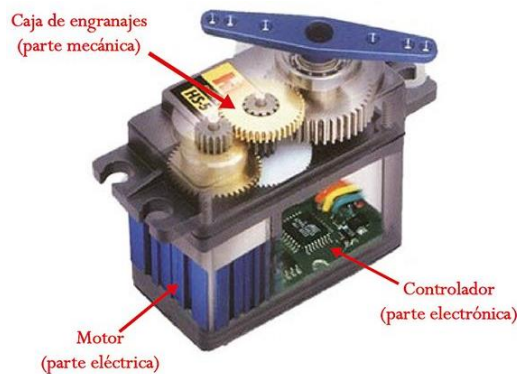


Figura 1.11: Estructura de un servomotor [25]

Tabla 1.5: Características del servomotor SG90 [26]

Parámetro	Valor
Voltaje de funcionamiento	5V
Torque	2.5 Kg/cm
Velocidad de operación	0.1s / 60°
Capacidad de rotación	180°
Peso	9g
Banda muerta	10us

- **Módulo ultrasónico HC-SR04**

Es un sensor para medir distancia por medio de ondas sonoras ultrasónicas, las cuales son emitidas mediante el cabezal emisor T y al ser reflejadas en algún objeto son receptadas por el cabezal receptor R; obteniendo así la medida de la distancia basada en el tiempo que se da entre la emisión y la recepción de la onda; para ello, se conoce que la velocidad del sonido es de 343m/s a una temperatura de 20°C. [27] [28] Si esto se lo transforma a unidades en cm/us da como resultado la Ecuación 1.1:

$$343 \frac{m}{s} \cdot 100 \frac{cm}{m} \cdot \frac{1}{1000000} \frac{s}{us} = \frac{1}{29.2} \frac{cm}{us}$$

Ecuación 1.1: Velocidad del sonido en cm/us [28]

El resultado demuestra que para recorrer un centímetro, el pulso se demora 29,2 microsegundos. [28] Por tanto, se puede obtener la distancia en función del tiempo que tarda entre la emisión y recepción del pulso sonoro utilizando la Ecuación 1.2:

$$Distancia(cm) = \frac{Tiempo(us)}{29.2 * 2}$$

Ecuación 1.2: Distancia medida por el sensor ultrasónico [28]

La razón por la que se divide el tiempo a la mitad, se debe a la duración que tarda el vaivén del pulso. [28] Sin esta división, la distancia recorrida sería el doble de la real. Esta característica es muy importante, porque al momento de configurar el sensor ultrasónico en la tarjeta *Arduino Nano* permite, mediante la fórmula, calcular distancia en centímetros. Cabe mencionar que, el sensor posee 4 pines: dos para alimentación (VCC y GND) y dos TTL (*Trigger* y *Echo*). El *Trigger* es para el pulso de disparo y el *Echo* para el pulso de retorno. El sensor HC-SR04 se muestra en la Figura 1.12 y sus características se presentan en la Tabla 1.6.



Figura 1.12: Sensor ultrasónico HC-SR04 [27]

Tabla 1.6: Características del sensor ultrasónico HC-SR04 [27]

Parámetro	Valor
Voltaje de funcionamiento	5 V
Corriente de funcionamiento	15 mA
Frecuencia de trabajo	40 KHz
Rango de funcionamiento	2 – 400 cm
Ángulo de detección	15°
Pulso de disparo TTL	10 us
Dimensiones	45 x 20 x 15 mm

- **Módulo sensor LDR**

Dispositivo que detecta la luz emitida sobre la superficie del panel de su fotorresistencia (LDR); la cantidad de luz percibida provoca variación en la resistencia LDR, originando un cambio de voltaje que será comparado con la caída de voltaje de un potenciómetro mediante un LM333, dando finalmente como resultado una salida digital. El módulo sensor LDR se presenta en la Figura 1.13.

El potenciómetro permite ajustar la sensibilidad de la salida digital; de tal manera que se obtendrá un 0 lógico cuando la intensidad de la luz exceda el valor fijado por el potenciómetro. [29] Las características del módulo sensor LDR se presentan en la Tabla 1.7.



Figura 1.13: Módulo sensor LDR [29]

Tabla 1.7: Características del módulo sensor LDR [29]

Parámetro	Valor
Voltaje de operación	3.3 V y 5 V
Número de pines	3 (GND, VCC, SIGNAL-D0)
Salida digital	0 y 1 por D0
Dimensiones	32 mm x 14 mm x 7 mm

- **Módulo KY-008**

El KY-008 es un módulo emisor de luz láser roja de alta potencia, como se presenta en la Figura 1.14. En la Tabla 1.8 se muestran las características del módulo KY-008 [30]:



Figura 1.14: Módulo KY-008 [30]

Tabla 1.8: Características del módulo KY-008 [30]

Parámetro	Valor
Alimentación	5 VCD 30 mA
Longitud de onda	650 nm
Dimensiones	24 mm x 15 mm x 8 mm

- **Luces piloto LED**

Luz indicadora LED con tecnología *multi-chip*, presentada en la Figura 1.15, para proporcionar un color uniforme y brillante; posee una tuerca de bloqueo en la parte posterior del panel, y utiliza abrazaderas roscadas para la conexión. [31] Las características de la luz piloto LED se presentan en la Tabla 1.9.



Figura 1.15: Luces piloto LED [31]

Tabla 1.9: Características de la luz piloto *LED* [31]

Parámetro	Valor
Alimentación	12 VCD
Consumo	≤ 20 mA
Tiempo de vida	30000 horas

- **Resistor**

Un resistor o resistencia es un elemento del tipo pasivo sin polaridad, que produce una oposición al paso de corriente eléctrica, causando en sus terminales una diferencia de voltaje. [32] La Figura 1.16 presenta una resistencia de película de carbono.

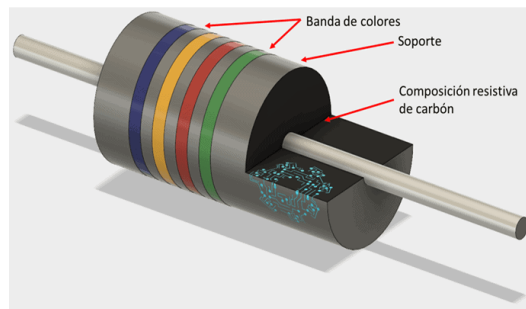


Figura 1.16: Resistencia de película de carbono [32]

- **Transistor 2N2222A**

El 2N2222A es un transistor bipolar de juntura tipo NPN; este transistor de silicio de baja potencia está diseñado para amplificación lineal y conmutación. [33] Las características del transistor 2N2222A se presentan en la Tabla 1.10.

Tabla 1.10: Características del transistor 2N2222A [33]

Parámetro	Valor
Voltaje colector-emisor	40 VCD
Voltaje colector-base	75 VCD
Voltaje emisor-base	6.0 VCD
Corriente máxima de colector	600 mA
Factor de amplificación HFE	100
Frecuencia máxima de trabajo	250 KHz

2 METODOLOGÍA

Este trabajo se fundamenta en distintos métodos de investigación que permitieron diseñar la simulación del sistema automatizado, objeto de esta tesis y cumplir con los objetivos planteados en la misma.

Entre los antecedentes para implementar el sistema automatizado están, en primer término, la visita al Centro Voces, en el cual se recabó información pertinente para verificar las condiciones que se requieren para la simulación del dispositivo. Luego, se entrevistaron a los especialistas del Centro para conocer las características del contenedor de reciclaje. Posteriormente, con los datos obtenidos se diseñó el sistema a través de *software* de simulación y se realizó la programación. Finalmente, se efectuaron las pruebas de funcionamiento. La Figura 2.1 presenta un gráfico de procesos de la metodología utilizada.

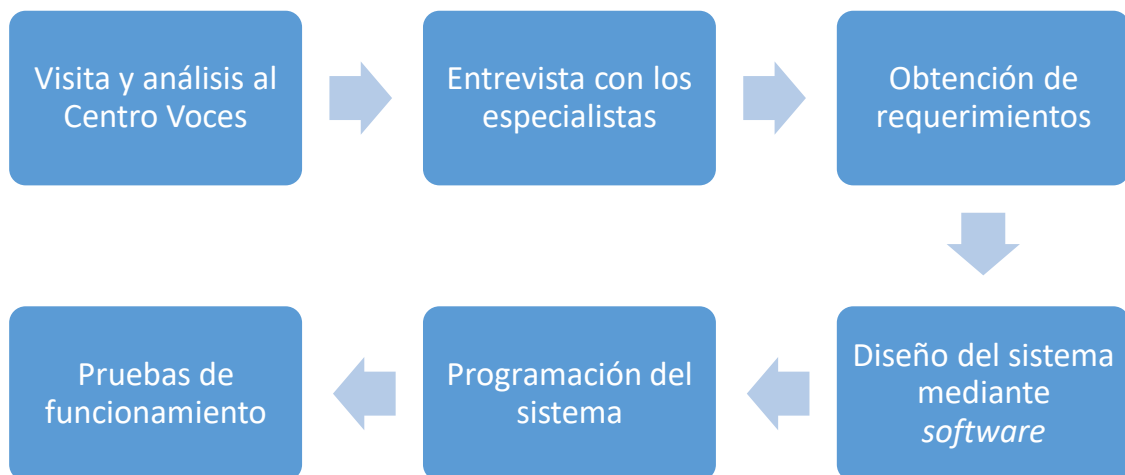


Figura 2.1: Proceso metodológico del proyecto

2.1 Método Cualitativo

Mediante el método cualitativo se logró recabar información para determinar las características y requerimientos que debía tener el dispositivo automatizado en el Centro Terapéutico Voces; tales como el dimensionamiento de la estructura, los tiempos para realizar la acción correspondiente al depósito de la basura, y el uso de indicadores visuales. Por tanto, cabe reiterar que los niños con TEA aprenden mediante los sentidos con material concreto, datos que se obtuvieron mediante la observación.

2.2 Método Comparativo

El sistema automatizado maneja sensores y una pantalla que es compatible con *Arduino*; de modo que, se ha evaluado cada versión de este microcontrolador, tomando en cuenta el número de pines, puertos de comunicación y compatibilidad con la versión de pantalla utilizada. En efecto, se concluye que la versión Nano de *Arduino* es la mejor opción para el desarrollo del sistema.

Para la comunicación con el administrador, se optó por seleccionar un módulo *Wi-Fi* conocido como ESP8266 *NodeMCU V3*, el cual permite enviar correo electrónico de manera rápida y sencilla, en comparación con sus versiones anteriores; las cuales requieren un microcontrolador externo para su manejo, pero que ralentiza la transmisión de datos.

2.3 Método Analítico

Mediante el método analítico se determinaron los conocimientos específicos sobre el uso correcto de la tarjeta *Arduino Nano*. Se estudió su compatibilidad y funcionalidad con otros módulos externos que conforman el sistema automatizado para conseguir un funcionamiento eficiente.

Asimismo, a través de este método se eligió el *software* de simulación y programación más adecuado como: *Proteus* y *Arduino IDE*, los cuales permiten simular los circuitos electrónicos e integrar librerías que permiten la comunicación entre el sistema y la pantalla. Además, se cuenta con la función para poder enviar el correo de administrador mediante el módulo *Wi-Fi*.

2.4 Método Experimental

Mediante el método experimental se pudo acceder a repositorios en línea, en los cuales se encuentran códigos de libre uso y acceso. Dichos repositorios pueden ser empleados en proyectos posteriores para el desarrollo de nuevas tecnologías.

Se utilizó el *software Proteus* para desarrollar la simulación del sistema automatizado, el cual permite simular elementos reales para poder comprobar su funcionamiento. Este programa posee un elemento llamado *COMPIN* que permite comunicación serial con *software* y *hardware*. Debido a la versatilidad de este elemento de simulación y en conjunto con un programa externo, que virtualiza puertos físicos de la computadora, se

pudo conectar el editor de *Nextion* con la simulación realizada en *Proteus*. Además, permitió incluir a la simulación el módulo *Wi-Fi* que no posee librerías u otra forma de emularlo de manera virtual.

En la simulación realizada en *Proteus* se incluyeron todos los elementos que tendría el sistema real, tales como: tarjeta *Arduino Nano*, sensor ultrasónico, LDR, resistencias, luces *LEDs*, transistores, servomotores, relé, adicional el uso del *COMPIN* para la conexión con el *software* y *hardware* externo.

Tanto las tarjetas *Arduino Nano* como el sensor ultrasónico utilizados en la simulación, fueron programados con el *software Arduino IDE*. Mediante el uso de librerías se puede también configurar periféricos como los servomotores y el módulo *Wi-Fi*.

Se utilizó el *COMPIN* para la conexión entre la simulación en el programa *Proteus* y el editor de *Nextion* con el *software Configure Virtual Serial Port Driver*, que conecta virtualmente dos puertos físicos de la computadora y permite la comunicación entre ellos. Dentro del editor de *Nextion* se escogió la opción de comunicación serial para poder enlazar ambos programas en tiempo real, y mediante una librería se pudo establecer comunicación. Esto a fin de mostrar las imágenes correspondientes a cada etapa de funcionamiento del sistema automatizado. De igual manera, se realizó un procedimiento similar para la inclusión del módulo *Wi-Fi* a la simulación; en el cual, mediante el *COMPIN* y el *software* de virtualización de puertos, se logró conectar a un microcontrolador dentro de la simulación mediante comunicación serial.

Se realizaron las pruebas de funcionamiento de cada uno de los elementos utilizados, así como de los tiempos de respuesta y duración en cada etapa que comprende el sistema automatizado. Se determinó que muchos de los tiempos de duración programados en la simulación podrían operar con lentitud. Esta condición se presenta debido a las características de la capacidad del CPU, que se evidencia al momento de llevar a cabo una simulación completa con los programas empleados simultáneamente.

El *software Arduino IDE* permitió configurar el módulo *Wi-Fi* el cual, mediante librerías, logra conectarse a un correo de *Gmail* y a la red local donde será utilizado. De esta manera, envía el mensaje al administrador. Además, el *software* permite la comprobación del funcionamiento, mediante el monitor serial. Indica, por último, si el mensaje fue enviado y si hubo acceso a la red.

3 RESULTADOS Y DISCUSIÓN

3.1 Autismo y tratamientos

El autismo puede definirse como un trastorno neurobiológico (trastorno genético a nivel del cerebro), que se presenta durante los primeros años de vida. [34] Este presenta características propias en el comportamiento de la persona que ha sido identificada con TEA, tales como la deficiencia en la comunicación e interacción social, y en la expresión de patrones repetitivos en cuanto a su participación en actividades cotidianas. [35] [36]

En Ecuador, el diagnóstico que determina la condición de autismo en una persona, por lo general, es desconocido o no se establece en los primeros años del niño. [4] Un mínimo porcentaje de personas con TEA son diagnosticadas a tiempo; en el 2016, el MSP (Ministerio de Salud Pública) reportó a 1258 personas que fueron diagnosticadas con algún Trastorno del Espectro Autista. [4] [36] Esta situación genera dificultades en cuanto a la adaptación del niño en el contexto familiar, social o educativo. Por lo tanto, solo con terapias u otras actividades (deporte, música, entre otros) se podría mejorar su calidad de vida. A continuación, se detallan algunos tratamientos utilizados en terapias de niños con TEA:

- **Método ABA**

El método ABA emplea varias técnicas; tales como el ensayo discreto, la presencia y retirada de apoyos, la búsqueda de reforzadores afectivos y la realización sistemática de registros; las cuales, tienen como objetivo disminuir los comportamientos inadecuados y desarrollar conductas apropiadas en los niños con autismo. [37]

- **PECS**

Los PECS tienen como objetivo enseñar comunicación funcional mediante el uso de imágenes. Además, con el empleo de este método se logra la interacción comunicativa entre el niño con TEA y las personas de su contexto social. [38]

- **Método TEACCH**

El método TEACCH tiende a aprovechar las capacidades de aprendizaje visual de los niños con TEA; esta característica es un punto estratégico en los primeros años

de vida. [39] En el proceso se exige el uso de material concreto para que los niños puedan realizar de manera autónoma sus actividades cotidianas. Por consiguiente, pueden mejorar las relaciones interpersonales al momento de comunicarse: formular una oración de manera correcta para expresar un requerimiento o para establecer un diálogo conciso.

- **Pictograma**

Los pictogramas son representaciones visuales con información específica de acciones, conceptos u objetos, transmitiendo un mensaje que debe ser traducido de manera rápida, directa y concreta. Por eso, son ampliamente utilizadas en las terapias conductuales impartidas por los profesionales en el autismo. [3]

3.2 Análisis

El diseño del basurero automatizado se fundamentó en el método TEACCH y en los requerimientos del Centro Terapéutico Voces para crear un dispositivo electrónico que sirva como material concreto para enseñar el reciclaje de desechos a los niños con TEA.

Luego de utilizar el método TEACCH, se buscaron las mejores herramientas para crear un producto que sirviera como material de enseñanza que complemente las terapias brindadas por las especialistas del Centro. El elemento principal seleccionado fue una pantalla de 7 pulgadas, la cual tiene como función principal proyectar imágenes de fácil comprensión para los niños con TEA. En la pantalla se presentan pictogramas o imágenes del tipo de basura que puede ser depositado en el respectivo contenedor.

Tanto los PECS y el método TEACCH explican que, mediante el uso de imágenes o pictogramas de fácil comprensión, se puede enseñar conductas e identificar objetos. Este aprendizaje se debe empezar durante las etapas iniciales de vida, cuando el aprendizaje es mayormente visual. Razón por la cual, se seleccionó una pantalla cuyo tamaño es óptimo para que se puedan apreciar las imágenes de forma clara.

Se usan también dos luces piloto, las cuales sirven como señales de alerta y como indicadores al momento de usar el basurero. Las características de las dos luces son específicas. La verde indica que el sistema se puede usar con normalidad; es decir, depositar basura. La roja indica que no se puede usar el dispositivo; es decir, que se encuentra lleno. Por lo tanto, el usuario deberá esperar a que el dispositivo tenga capacidad para reutilizarlo. Las luces tienen dos funciones adicionales. La primera se

emplea al momento en que el basurero entra en estado de activación; se concede un tiempo de 10 segundos para poder realizar la acción de depositar la basura. Para lo cual, es importante considerar que en los niños con TEA se cuenta con un tiempo de 5 segundos para captar su atención. [1] De modo que, si se sobrepasa ese tiempo, el niño puede empezar a perder el interés o puede distraerse; por tanto, no completaría la actividad. Una vez que el tiempo asignado para depositar la basura culmine, la luz verde se apaga y se enciende la luz roja. Esta función indica que el tiempo ha terminado y que se deben esperar 3 segundos para que el sistema se reactive y se pueda realizar la acción.

Se plantea la automatización de la apertura y cierre de la tapa del basurero mediante el uso de servomotores y un sensor de presencia. La distancia de activación está comprendida en un rango de 60 cm. Cuando el niño u otro usuario se encuentre dentro del perímetro, el sensor de presencia activa el basurero. Entonces, los servomotores hacen una apertura automática de la tapa del dispositivo, se deposita la basura y, transcurridos los 10 segundos, la tapa se cierra automáticamente.

Además, se considera un sensor de nivel en el interior del dispositivo, cuya función es informar si el basurero se encuentra lleno o vacío. Cuando se encuentre en estado lleno, la tapa no se abrirá; esta función contribuye con la protección del sistema.

Por último, el módulo *Wi-Fi* se encarga de enviar un correo electrónico al administrador del Centro Voces, con información que le permitirá conocer el límite de capacidad de cada contenedor. La Figura 3.1 presenta un esquema del basurero automatizado.

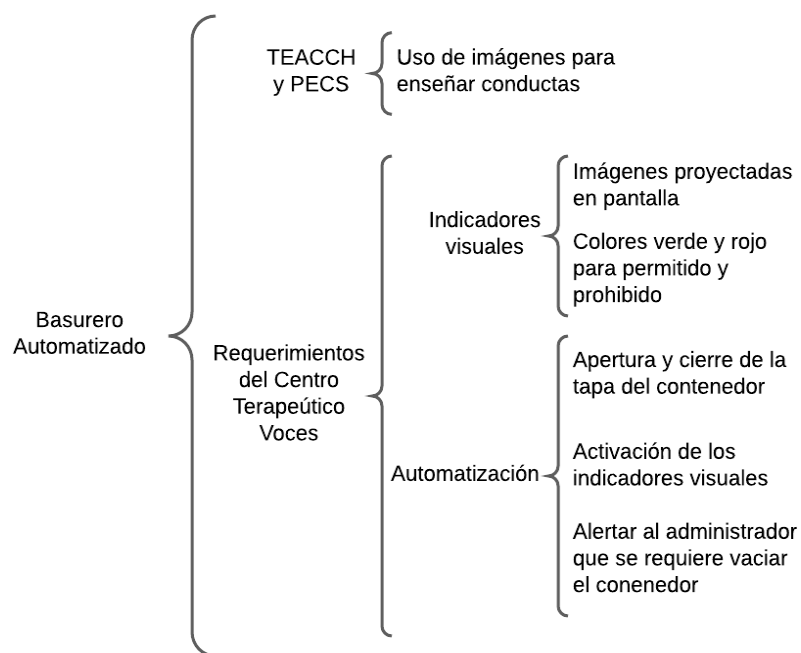


Figura 3.1: Esquema del basurero automatizado

Finalmente, para diseñar la estructura del sistema, se analizó el espacio físico disponible del Centro; como resultado de ello, se consideró ubicarlo en el patio del establecimiento, sitio que se muestra en la Figura 3.2.



Figura 3.2: Ubicación física del dispositivo en el Centro Voces

3.3 Diseño

El sistema se encuentra constituido de dos sensores y cuatro actuadores; un sensor ultrasónico para detectar presencia y un sensor LDR en conjunto con un módulo láser para detectar nivel, dos servomotores para levantar la tapa, una pantalla de 7" para proyectar pictogramas y un módulo Wi-Fi para enviar e-mail. En la Figura 3.3 se muestra el diagrama general del sistema.

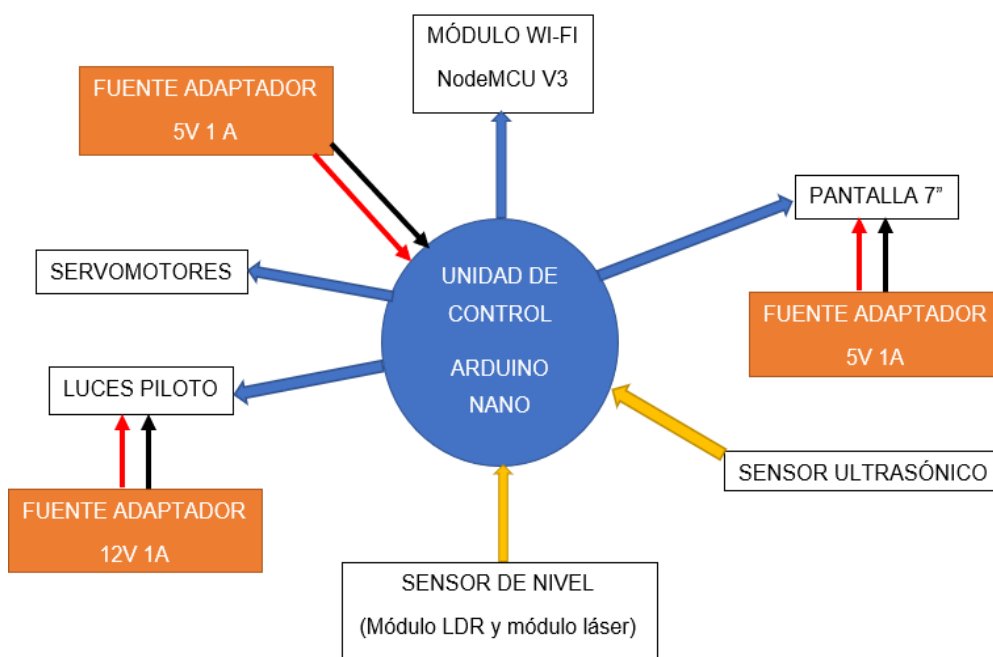


Figura 3.3: Diagrama general del sistema

Como se pudo observar en la Figura 3.3, se hace uso de dos fuentes de 5V 1A y una fuente de 12V 1A. Considerando que la pantalla de 7" opera a 5V 510mA y las luces piloto led trabajan a 12V; en la Tabla 3.1 se muestra el consumo energético de los elementos restantes, dando como resultado una fuente de 5V 1A para alimentar a la unidad de control y los módulos.

Tabla 3.1: Consumo energético

Elemento	Voltaje	Amperaje	Cantidad	Amperaje total
<i>Arduino Nano</i>	5 V	19 mA	1	19 mA
<i>NodeMCU V3</i>	3.6 V	90 mA	1	90 mA
Servomotor SG90	5 V	130 mA	2	260 mA
Módulo Laser	5V	30 mA	1	30 mA
Módulo LDR	5V	50 mA	1	50 mA
Módulo ultrasónico	5V	15 mA	1	15 mA
Total				464 mA

Estructura de los contenedores

La estructura cuenta con tres contenedores estándar; cada uno para material específico: papel, plástico o desechos orgánicos, los cuales se ubican dentro de un solo armazón con su respectivo sistema independiente. Cabe mencionar que los colores utilizados para diferenciar el tipo de basura son gris para papel, azul para plásticos y verde para residuos orgánicos.

Como se observa en la Figura 3.4, se ha considerado una pantalla por basurero, sobre la cual iría instalado el sensor ultrasónico. Los servomotores se encontrarían en la parte posterior y a los laterales de la apertura para depositar la basura; y las luces piloto se ubicarían en la parte superior de la apertura donde se depositaría la basura.

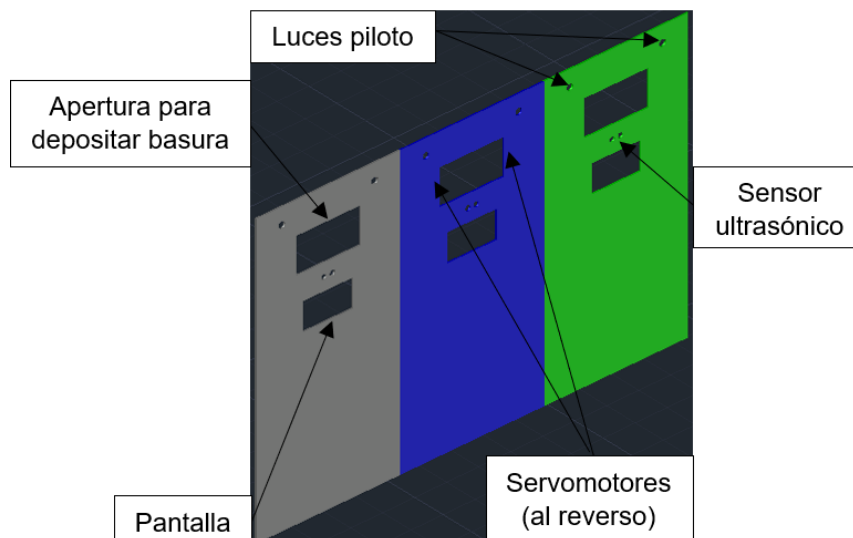


Figura 3.4: Ubicación de los sensores y actuadores en la parte externa

Como se observa en la Figura 3.5, se ha considerado la presencia de orificios para facilitar la instalación de los sensores de nivel en la estructura interna; mientras que la parte delantera está destinada para mantener a salvo la pantalla, al momento en que depositen la basura por la apertura.

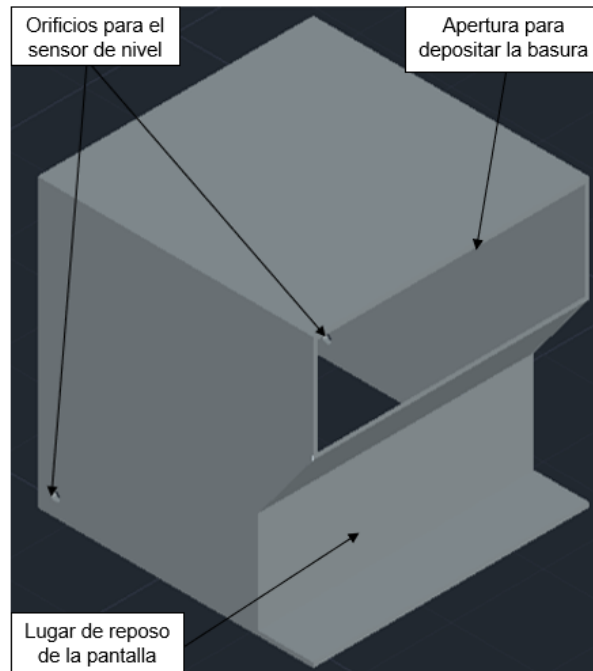


Figura 3.5: Ubicación del sensor de nivel en la parte interna

La Figura 3.6 muestra la ubicación de los contenedores y el lugar destinado para la ubicación de los circuitos electrónicos en la estructura interna, la cual se la muestra sin los laterales, la parte superior, la parte frontal, y la parte trasera del armazón.

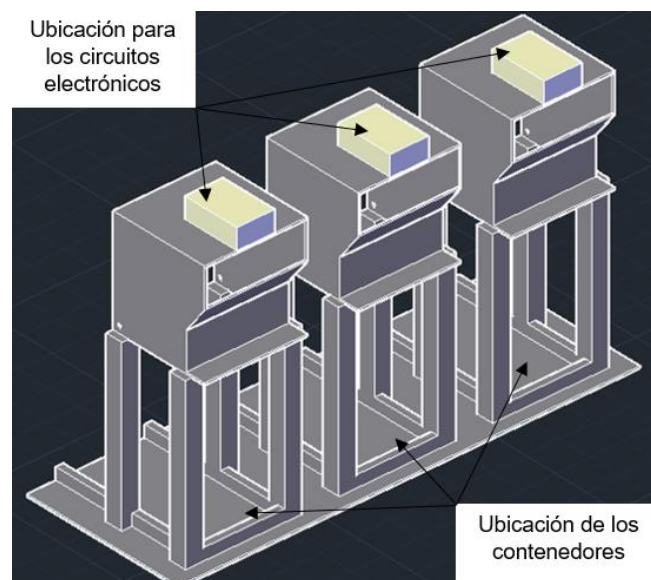


Figura 3.6: Estructura interna

La Figura 3.7 presenta la vista en tres dimensiones de la estructura; en donde, se utiliza tres colores para identificar a qué contenedor corresponde cada tipo de basura. El color gris corresponde a papel, el azul a plásticos y el verde a desperdicios orgánicos.

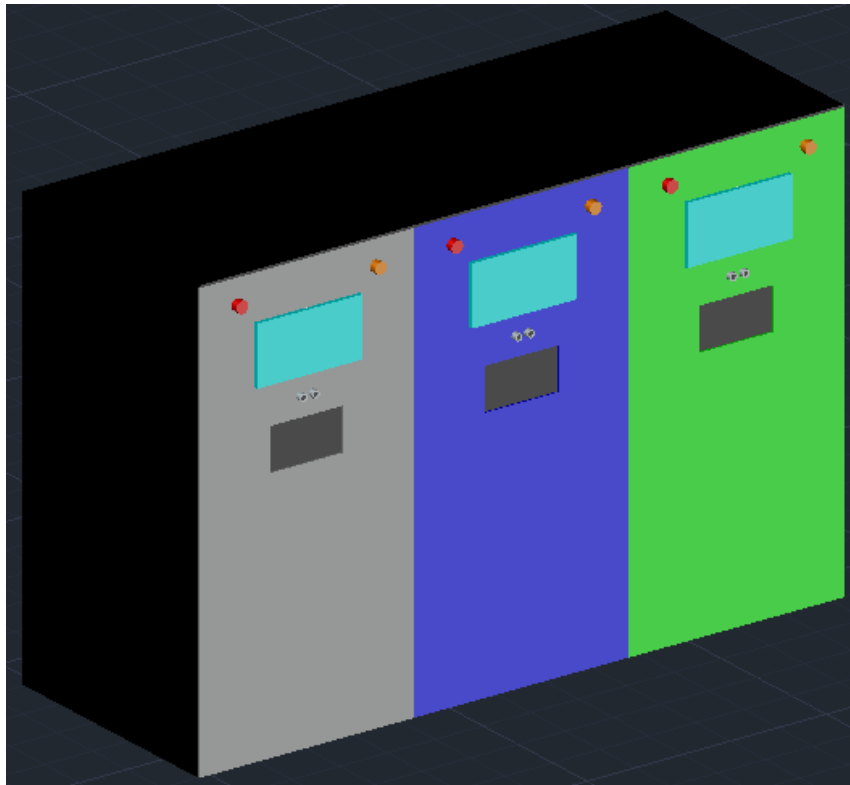


Figura 3.7: Vista 3D de la estructura

A partir de la Figura 3.8 hasta la 3.15 se aprecian los planos de la estructura.

La Figura 3.8 presenta el plano de la parte superior de la estructura, el cual se conforma de una sola pieza de 50x135 cm.

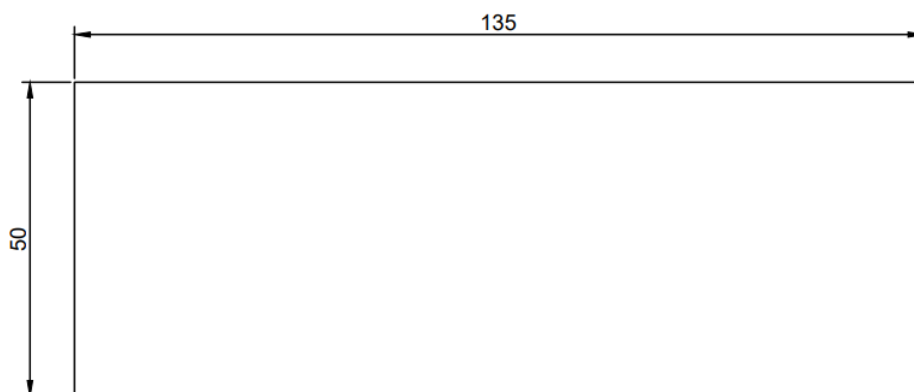


Figura 3.8: Plano de la parte superior

La Figura 3.9 presenta el plano de los laterales de la estructura, los cuales se conforman de una sola pieza de 90x50 cm.

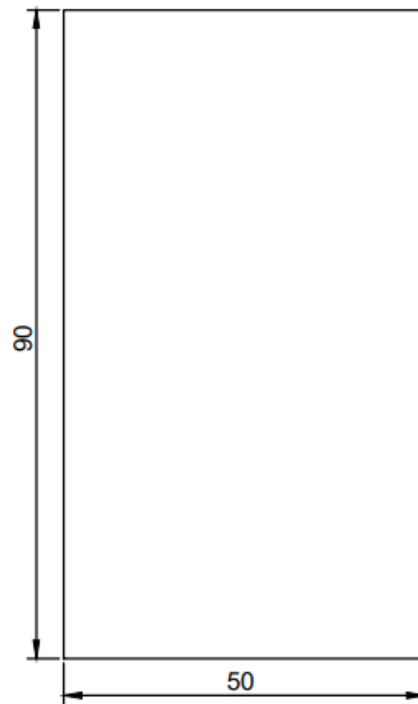


Figura 3.9: Plano de los laterales

La Figura 3.10 presenta el plano de la parte posterior de la estructura, el cual es una pieza de 90x135 cm. Las líneas azules representan aperturas en la pieza, donde van las puertas con medidas 82x32 cm; y las líneas punteadas son de referencia.

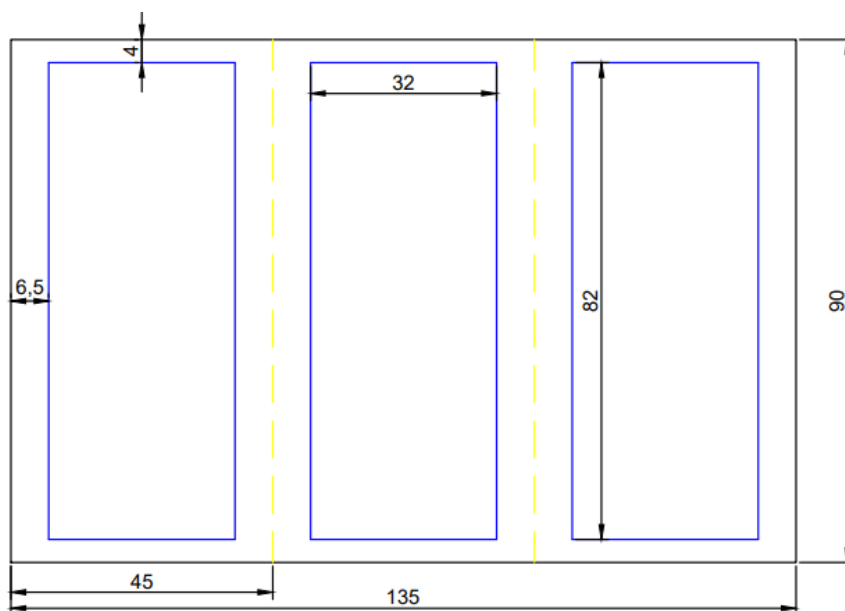


Figura 3.10: Plano de la parte posterior

La Figura 3.11 presenta el plano de la parte frontal de la estructura, el cual es una pieza de 90x135 cm. Las líneas azules representan aperturas en la pieza, donde se ubican algunos sensores y actuadores, además del lugar por donde se deposita la basura; y las líneas punteadas son de referencia.

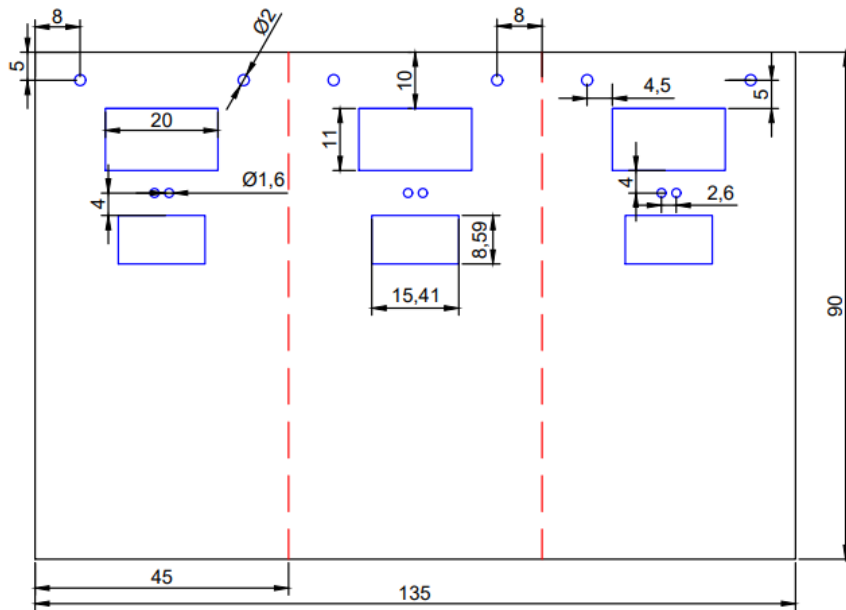


Figura 3.11: Plano de la parte frontal

La Figura 3.12 presenta el plano de la base de la estructura, la cual es una pieza de 50x135 cm y sobre la misma, siendo parte de una sola pieza, se encuentran unas guías que ayudan a mantener el contenedor interno en su lugar. Cabe mencionar que las líneas punteadas son solo para referencia

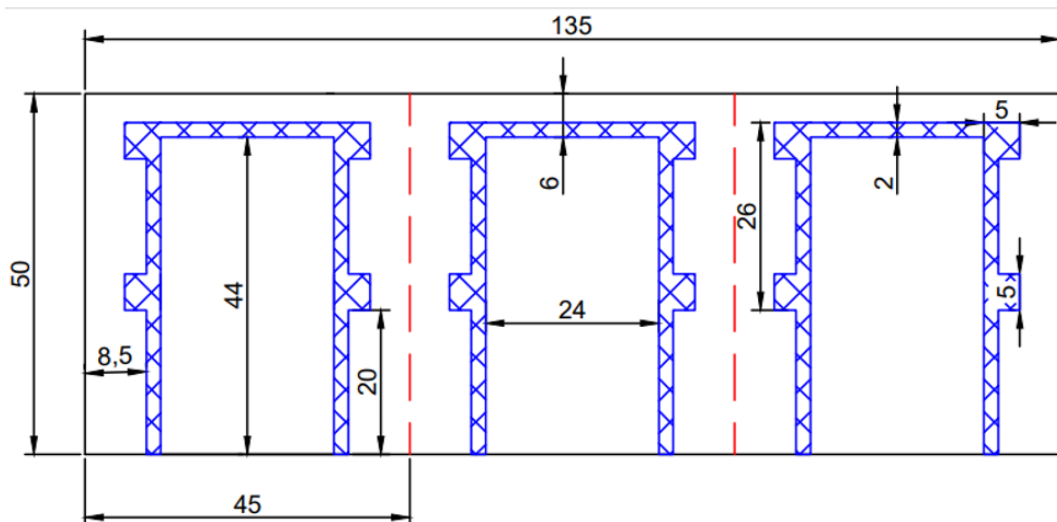


Figura 3.12: Plano de la base con guía

La Figura 3.13 presenta una vista lateral del plano de la base; en donde, sobre la parte más alta va colocada una pequeña estructura interna a la que se la ha denominado como “la rampa”.

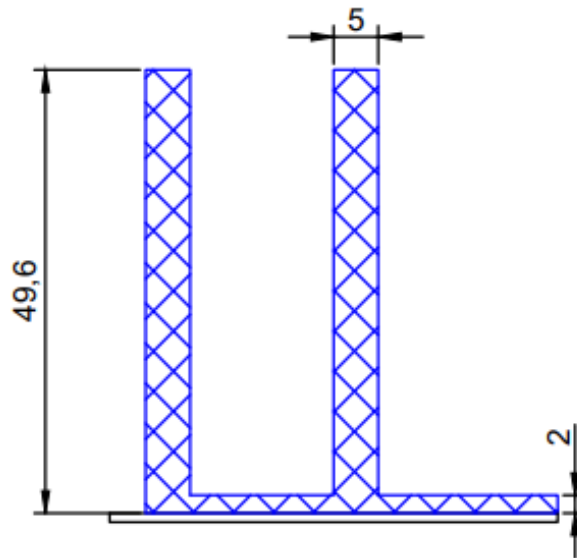


Figura 3.13: Plano vista lateral de la base con guía

La Figura 3.14 presenta los planos de la parte superior y una vista frontal del plano de la rampa. La tapa superior de la rampa es una pieza de 30x30 cm, mientras que, el frente de la rampa se compone de la unión de dos piezas, empezando por la parte superior hacia abajo son: una apertura de 11x30 cm, una pieza de 10x30 cm y una pieza de 11.4x30 cm. Cabe mencionar que la pieza de 10x30 cm posee una inclinación de 53° con respecto a la horizontal, como se observa en la Figura 3.14,

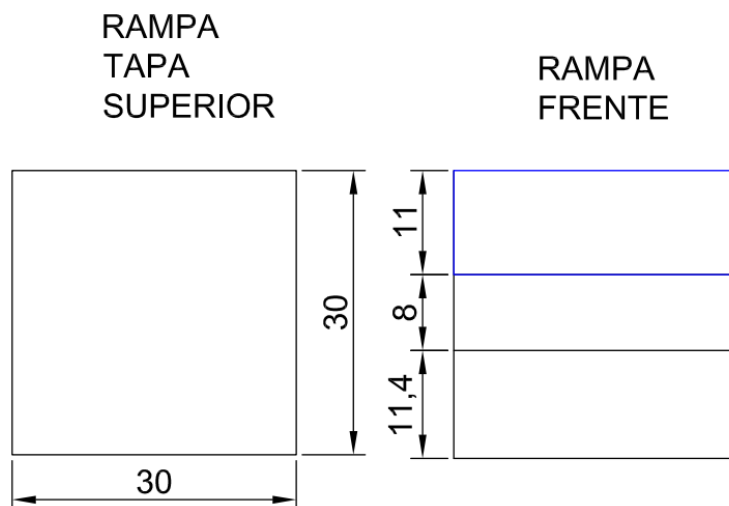


Figura 3.14: Planos superior y frontal de la rampa

fijado por el potenciómetro, se considera que ha sido cortado, se enciende el *LED* D3 y se produce un 1 lógico a la salida de la compuerta *NOT*. La salida de la compuerta se conecta hacia el *Arduino* Nano en el pin D2.

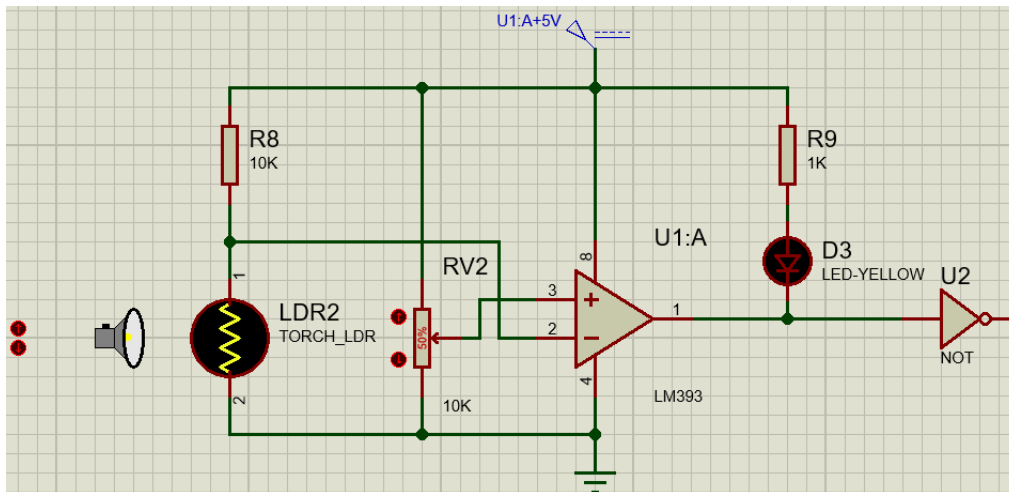


Figura 3.17: Conexión del sensor de nivel en la simulación

Dado que, el *LED yellow* solo se lo requiere como indicador, y la corriente mínima de trabajo de los *LEDS* es 5mA, se ha aplicado el despeje de la resistencia, a partir de la Ley de Ohm, presentado en la Ecuación 3.1.

$$R = \frac{V}{I} = \frac{5V}{5[mA]} = 1000[ohms]$$

Ecuación 3.1: Despeje de resistencia de la ley de Ohm para R9

Dando una resistencia de 1KΩ para R9, el cual es un valor comercial utilizable.

La salida del LM393 (1), otorga +5V o 0V, dependiendo del voltaje en la entrada positiva (3) y la entrada negativa (2), como se muestra en las siguientes ecuaciones:

$$V(3) > V(2) \quad \text{entonces,} \quad V(1) = +5[V]$$

$$V(2) > V(3) \quad \text{entonces,} \quad V(1) = 0[V]$$

para ello, se ha dispuesto la resistencia de 10KΩ, el potenciómetro (RV2 = 10KΩ) y el *LDR* conforme a las fórmulas de divisor de tensión, presentadas en las Ecuaciones 3.2 y 3.3.

$$V(3) = \left\{ \frac{RV2\left(\frac{x}{100}\right)}{\left[RV2\left(1 - \frac{x}{100}\right)\right] + \left[RV2\left(\frac{x}{100}\right)\right]} \right\} * 5$$

Ecuación 3.2: Divisor de tensión en la entrada positiva del LM393

$$V(2) = \left\{ \frac{LDR}{10 [K\Omega] + LDR} \right\} * 5$$

Ecuación 3.3: Divisor de tensión en la entrada negativa del LM393

En donde, “x” representa el porcentaje entre 0 y 100 del potenciómetro RV2, y el LDR puede tomar un valor entre 100Ω y 1MΩ.

Por ejemplo, si el potenciómetro se encuentra al 40%, y la incidencia de luz en el LDR provoca que el LDR tome un valor de 20KΩ, la entrada positiva (3) será igual a 2V y la entrada negativa (2) será igual a 3.33V; por lo cual la salida del LM393 (1) será igual a 0V. El *LED yellow* se enciende y la salida de la compuerta NOT será 1 lógico, lo que indica que el haz de luz ha sido cortado.

Diseño de la conexión del sensor ultrasónico

Para medir la proximidad de un usuario, se ha utilizado un sensor ultrasónico, el cual conecta sus pines *Trigger* y *Echo* al *Arduino Nano*; este sensor funciona con +5V. En la Figura 3.18 se muestra la conexión del sensor ultrasónico; cabe mencionar que el pin *SIMPIN* del ultrasónico se lo usa solo para la simulación. Pero, dicho pin no existe en un ultrasónico real.

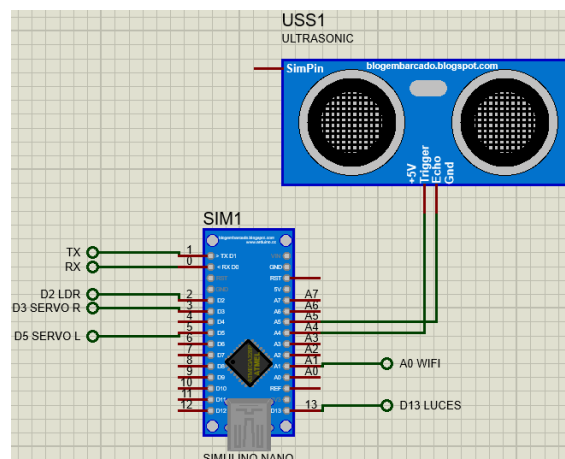


Figura 3.18: Conexión del sensor ultrasónico de manera física

Como se puede observar en la Figura 3.19, para la simulación se utiliza un potenciómetro conectado al *SIMPIN* del ultrasónico. La relación del porcentaje del potenciómetro con la distancia se muestra en la Tabla 3.2.

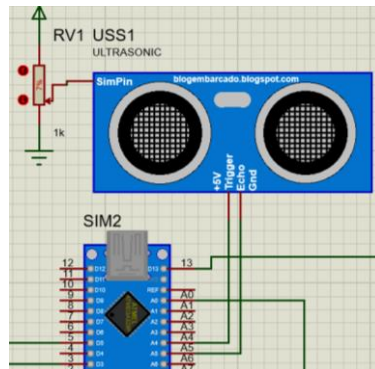


Figura 3.19: Conexión del sensor ultrasónico en la simulación

Tabla 3.2: Relación porcentaje-distancia de la librería ultrasónico

PORCENTAJE	DISTANCIA
0%	4 cm
1%	10 cm
2%	26 cm
3%	33 cm
4%	44 cm
5%	55 cm
6%	66 cm
7%	83 cm
8%	89 cm
9%	104 cm

Diseño de la conexión de la pantalla

La conexión de la pantalla de 7" se muestra en la Figura 3.20; cabe recalcar que, para energizarla, se usa una fuente de alimentación independiente al sistema. La conexión con el *Arduino Nano* se realiza mediante comunicación UART y se utiliza para indicar a la pantalla cuándo actuar.

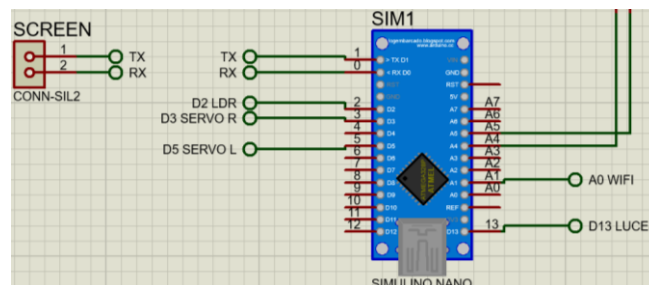


Figura 3.20: Conexión de la pantalla de manera física

En la simulación, la conexión de la pantalla se realiza mediante un *COMPIM*, como se observa en la Figura 3.21, el cual mediante el software *Configure Virtual Serial Port Driver* se conecta al *Nextion Editor*, permitiendo así simular la pantalla.

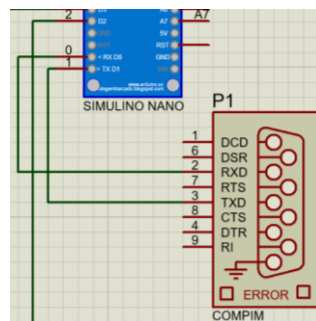


Figura 3.21: Conexión de la pantalla en la simulación

Diseño de la conexión del ESP8266 *NodeMCU V3*

Para el funcionamiento del módulo *Wi-Fi* ESP8266 *NodeMCU V3* se utiliza un solo pin de *Arduino Nano*, como se muestra en la Figura 3.22. El *NodeMCU V3* funciona con +5V pertenecientes a la alimentación general del sistema, y es programable mediante el *Arduino IDE*. Cabe indicar que el módulo cuenta con un programa independiente, el cual se activa al recibir una señal digital proveniente del *Arduino Nano*.

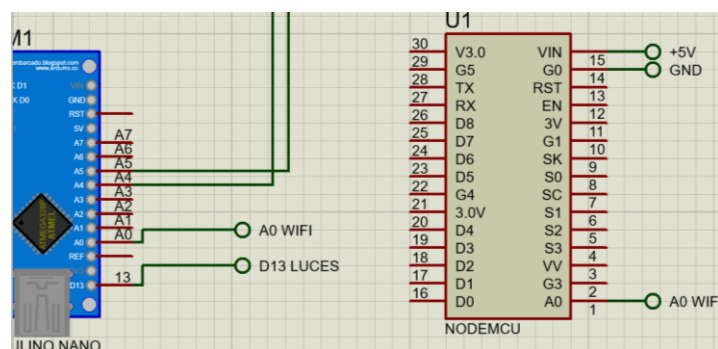


Figura 3.22: Conexión del ESP8266 *NodeMCU V3* de manera física

Dado que no existe una librería para simular el ESP8266 *NodeMCU V3* en *Proteus*, se utiliza un *COMPIM* junto con el software *Configure Virtual Serial Port Driver* para comunicar la simulación en *Proteus* con el *NodeMCU V3* físico mediante un puerto COM del computador. En este caso, se utiliza un *Arduino Nano* más, como se muestra en la Figura 3.23, el cual recepta la señal proveniente del *Arduino Nano* principal y mediante comunicación UART se conecta con el *NodeMCU V3* físico. Adicionalmente, se emplea un *LED* para indicar que se está produciendo la comunicación.

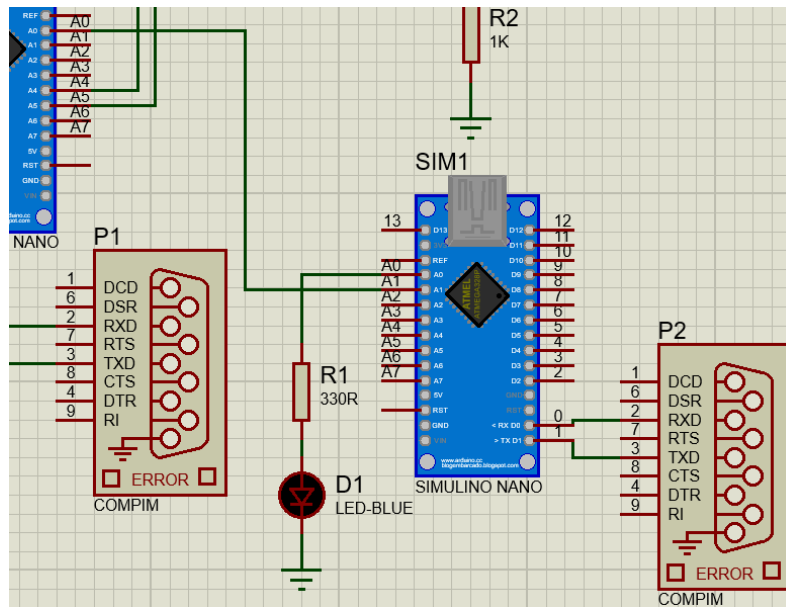


Figura 3.23: Conexión del ESP8266 *NodeMCU V3* en la simulación

Dado que, el LED azul debe llamar la atención, y la corriente máxima de trabajo de un LED es 15-20 mA, se ha aplicado la Ecuación 3.4, que corresponde al despeje de la ecuación de Ohm.:

$$R = \frac{V}{I} = \frac{5V}{15[mA]} = 333.33[ohms]$$

Ecuación 3.4: Despeje de resistencia de la ley de Ohm para R1

El resultado, en la misma ecuación es una resistencia de 333.33Ω. Considerando que, el valor comercial más próximo es de 330Ω, se ha optado por seleccionar una resistencia de 330Ω.

Diseño de la conexión de los servomotores

Los servomotores están conectados a la salida de la señal PWM proveniente del *Arduino Nano* mediante un pequeño circuito que utiliza un transistor 2N2222A, el cual se ha configurado en colector común o como un seguidor de emisor para no invertir el pulso PWM. Tanto el transistor como los servomotores funcionan con +5V de la alimentación general del sistema. En las Figuras 3.24 y 3.25 se observa la conexión física y simulada, las cuales no difieren entre sí en los elementos utilizados.

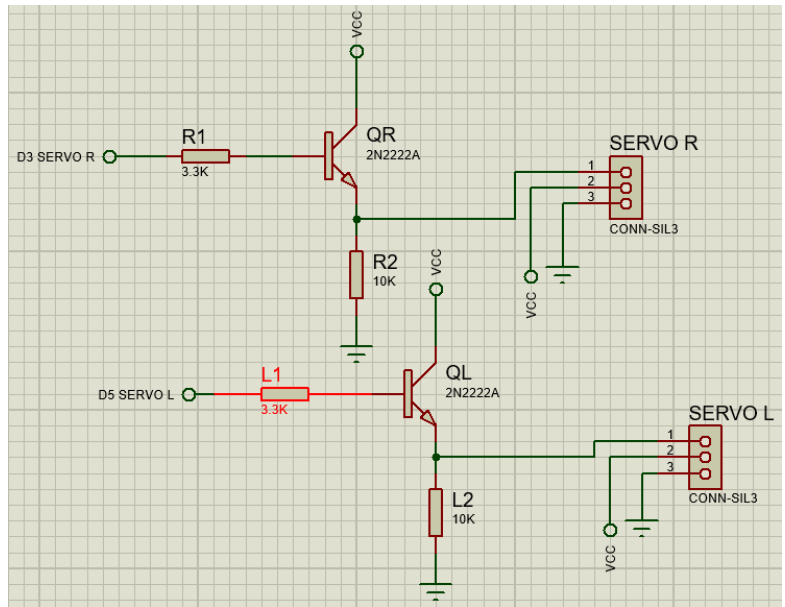


Figura 3.24: Conexión de los servomotores de manera física

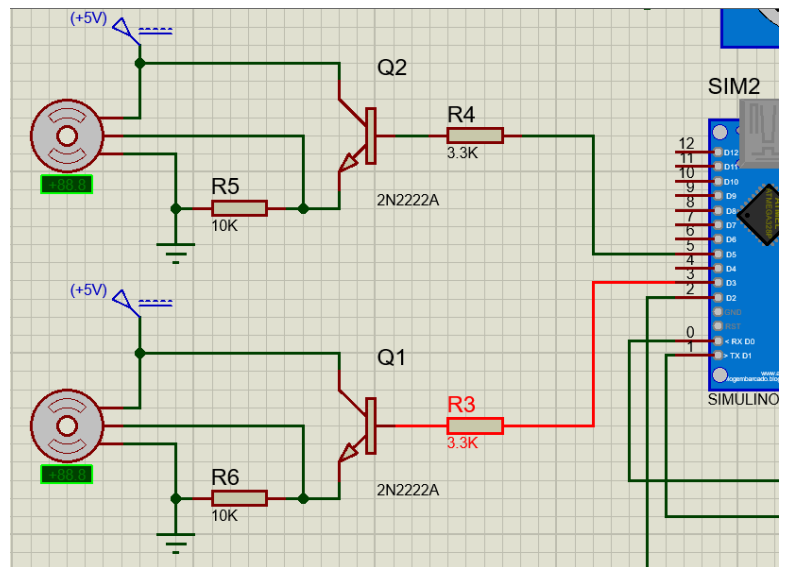


Figura 3.25: Conexión de los servomotores en la simulación

Para el dimensionamiento de las resistencias de las Figuras 3.24 y 3.25, considerando que, se encuentran en una misma configuración, se ha procedido a utilizar la siguiente nomenclatura:

Rb para la resistencia de la base del transistor.

Re para la resistencia del emisor del transistor.

Considerando, que el servomotor consume un máximo de 150mA y el hfe del transistor es 100; aplicando la Ecuación 3.5 se tiene:

$$I_b = \frac{I_e}{h_{fe} + 1} = \frac{150[mA]}{100 + 1} = 1.48[mA]$$

Ecuación 3.5: Corriente de base en el transistor 2N2222A

y sabiendo que, el voltaje en la resistencia de base es igual a la Ecuación 3.6:

$$V_{Rb} = V_b - V_{be} = 5[V] - 0.7[V] = 4.3[V]$$

Ecuación 3.6: Voltaje en la resistencia de base del transistor 2N2222A

aplicando la Ecuación 3.7 se obtiene lo siguiente:

$$R_b = \frac{V_{Rb}}{I_b} = \frac{4.3[V]}{1.48[mA]} = 2905\Omega$$

Ecuación 3.7: Resistencia de base en el transistor 2N2222A

Considerando los valores comerciales se determina $R_b=3.3K\Omega$.

Para la resistencia de emisor, se utiliza el principio de *pull-down*; utilizando el valor estándar de 10K Ω , esto para evitar que, cuando la PWM esté en amplitud cero, se produzca un cortocircuito.

Diseño de la conexión de las luces piloto

Las luces piloto son controladas por un módulo de 1 relé, cuya señal de control es proveniente del pin D13 del *Arduino Nano*, como se muestra en la Figura 3.26; dicho módulo funciona con +5 voltios pertenecientes a la alimentación general del sistema, pero se usa una fuente externa regulada de 12V para alimentar a las luces piloto.

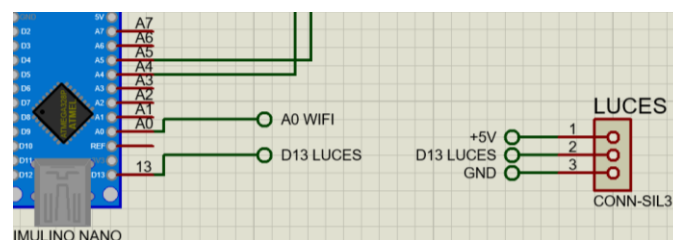


Figura 3.26: Conexión de las luces piloto de manera física

En la simulación se emplea un circuito basado en un relé y un opto acoplador, como se muestra en la Figura 3.27, tal como en el módulo físico de 1 relé. La salida normalmente abierta del relevador se conecta al *LED* rojo que representa a la luz piloto roja, y la salida normalmente cerrada se conecta al *LED* verde que simboliza a la luz piloto verde.

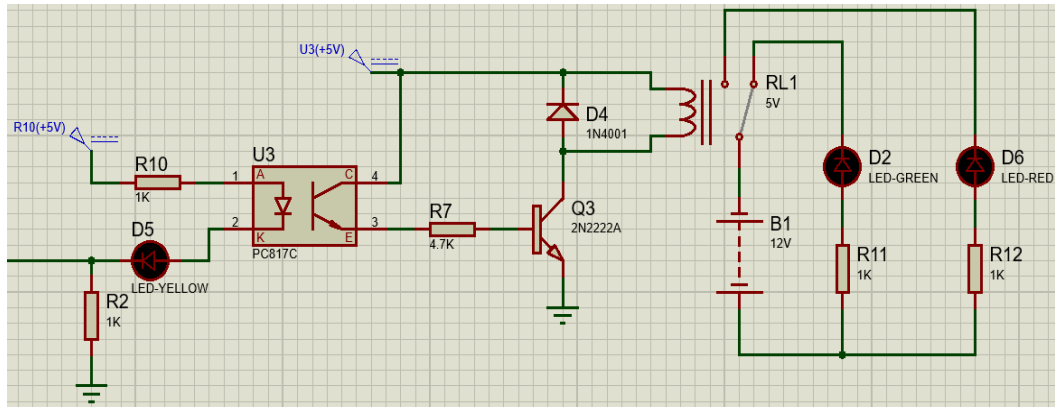


Figura 3.27: Conexión de las luces piloto en la simulación

Tanto para el dimensionamiento de R2 como de R10, se ha considerado la corriente mínima de trabajo del *LED* del optoacoplador, la cual es 5mA; aplicada la Ecuación 3.8:

$$R = \frac{V}{I} = \frac{5V}{5[mA]} = 1000[ohms]$$

Ecuación 3.8: Despeje de resistencia de la ley de Ohm para R2 y R10

Se obtiene como resultado el uso de una resistencia de 1KΩ, el cual es un valor comercial utilizable.

En el dimensionamiento de la resistencia R7 (Rb), se toma en cuenta que, la resistencia de la bobina del relé es 55Ω cuando es alimentado con 5V, el hfe del transistor 2N2222A es 100 y que la caída de voltaje base-emisor (Vbe) es 0.7V; aplicando las Ecuación 3.9, 3.10 y 3.11 se obtiene:

$$I_c = \frac{5[V]}{55[\Omega]} = 90.90[mA]$$

Ecuación 3.9: Corriente de colector

$$I_b = \frac{I_c}{h_{fc}} = \frac{90.90[mA]}{100} = 909[uA]$$

Ecuación 3.10: Corriente de base

$$R_b = \frac{V_b - V_{be}}{I_b} = \frac{5[V] - 0.7[V]}{909[\mu A]} = 4730.47[\text{ohms}]$$

Ecuación 3.11: Resistencia de base

Dando como resultado una resistencia de 4730Ω; usándose en este caso el valor comercial más próximo, el cual es 4.7KΩ.

Para dimensionar las resistencias R11 y R12, se considera, la máxima corriente de trabajo de los *LEDS* D2 y D6, la cual es 15mA. Aplicando la Ecuación 3.12 se obtiene:

$$R = \frac{V}{I} = \frac{12V}{15[mA]} = 800[\text{ohms}]$$

Ecuación 3.12: Despeje de resistencia de la ley de Ohm para R11 y R12

En donde, R representa tanto a R11 y R12; se obtiene como resultado una resistencia de 800Ω; usándose en este caso el valor comercial próximo más común de 1KΩ.

Circuito electrónico completo

La Figura 3.28 corresponde al circuito electrónico completo con el uso de módulos, aquí se integran las diversas conexiones que lo comprenden tales como: *Arduino Nano* el cual controla todo el sistema, *NodeMCU V3* que se encarga de enviar el *e-mail*, módulo ultrasónico, módulo LDR y módulo laser KY-008 que son sensores de presencia y nivel respectivamente, servomotores SG90, luces piloto y la pantalla *Nextion* que son actuadores para los diversos estados del sistema; las señales de los sensores de nivel y presencia son importantes para definir el estado del sistema.

La Figura 3.29 corresponde al circuito electrónico utilizado para la simulación, aquí se integran las diferentes conexiones que lo conforman como es el uso de dos *Arduino Nano*, uno para controlar todo el sistema y otro exclusivo en la simulación como interfaz para comunicación con el *NodeMCU V3*, los diagramas esquemáticos del sensor de nivel y de las luces piloto, la conexión de los servomotores y el sensor de presencia, y el uso de los elementos *COMPIM* para simular comunicación serial en *Proteus*.

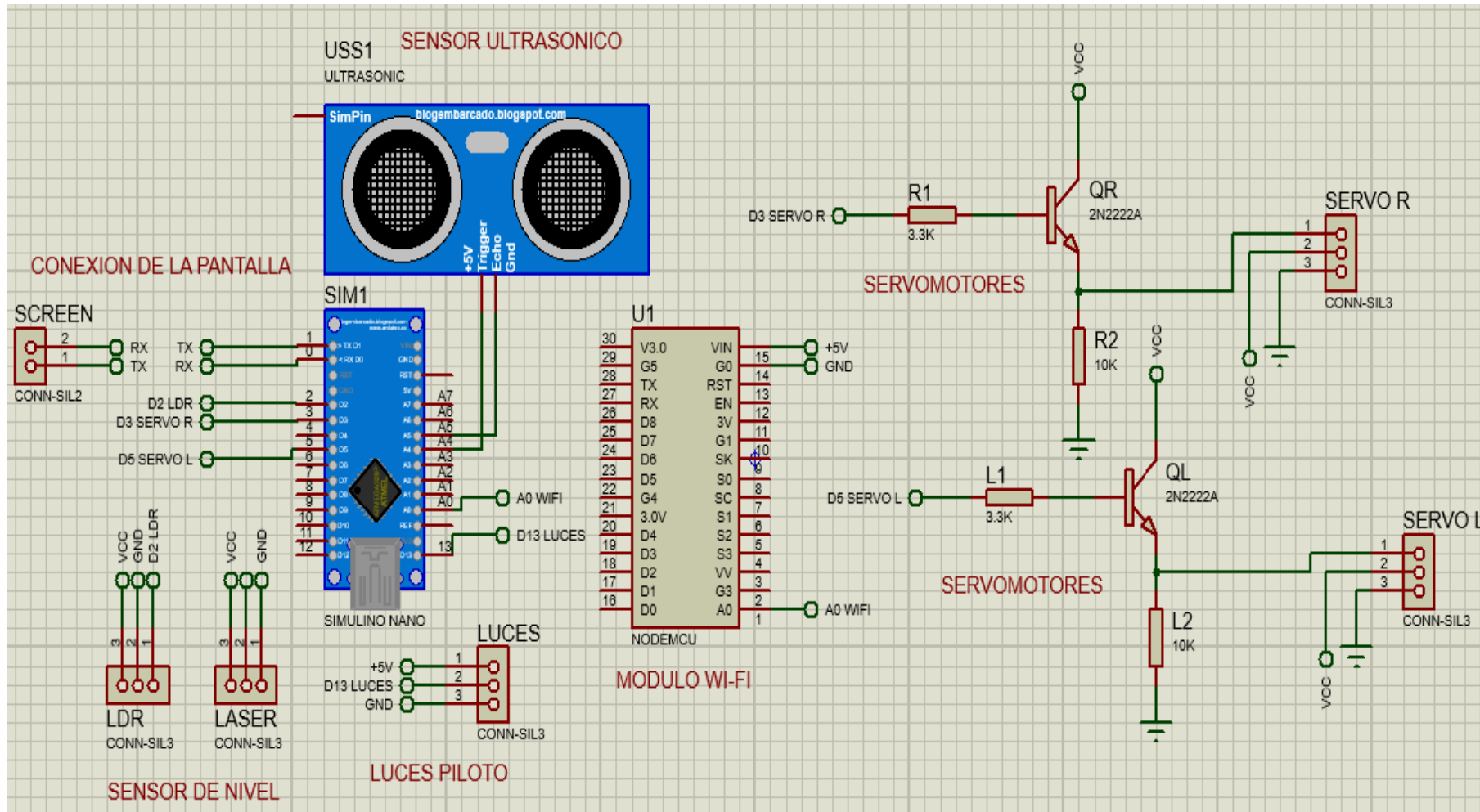


Figura 3.28: Circuito electrónico completo usando módulos

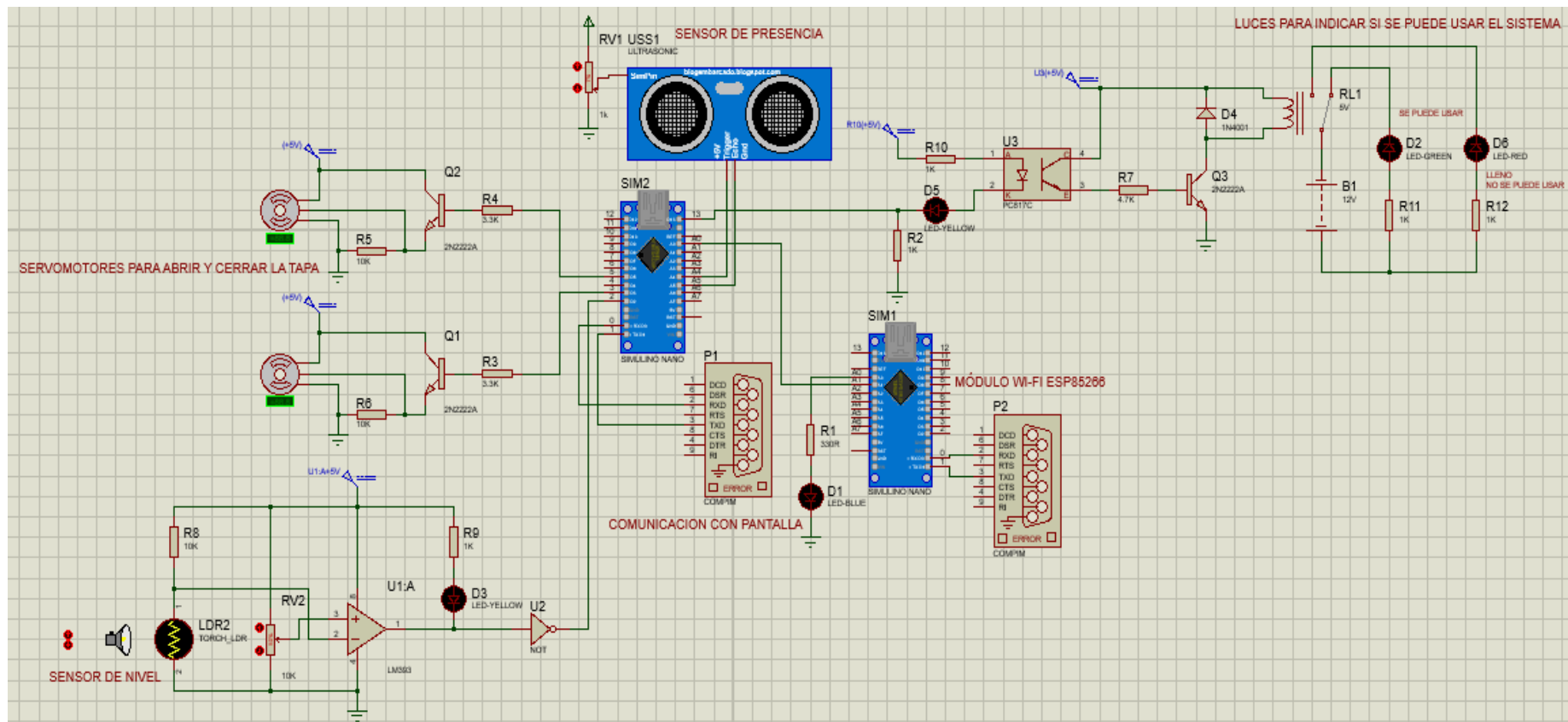


Figura 3.29: Circuito electrónico completo para la simulación

Diagramas de flujo

Previo a la programación, se realizó un diagrama de flujo general que abarque los estados del sistema principal; además, se realizaron diagramas para los subprocesos tales como el envío de correo electrónico y la proyección de imágenes en la pantalla.

Como se aprecia en la Figura 3.30, el diagrama general presenta los tres estados del sistema: reposo, activación y bloqueo; dependiendo del estado, se utiliza los subprocesos: envío de correo electrónico y proyección de imágenes en pantalla.

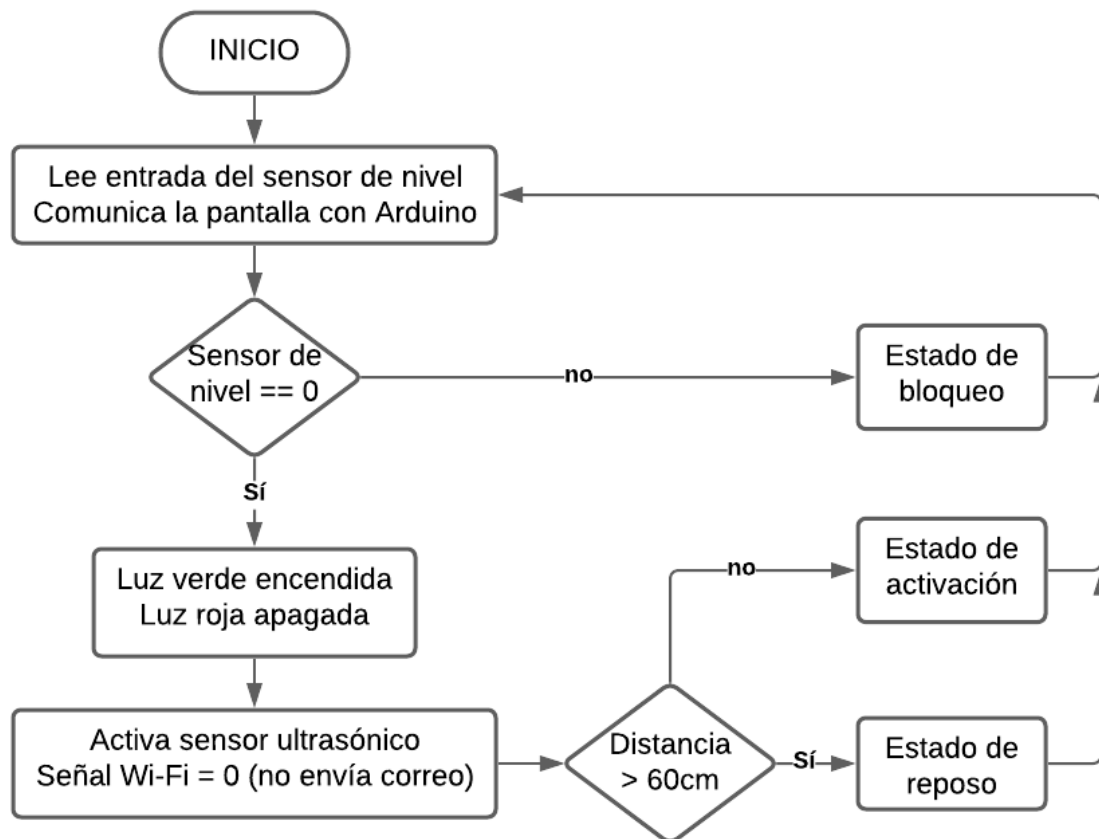


Figura 3.30: Diagrama de flujo general del sistema

En la Figura 3.31 se muestra el diagrama del subproceso para proyectar imágenes en la pantalla, el cual va programado en la pantalla y se basa en la transición rápida de imágenes.

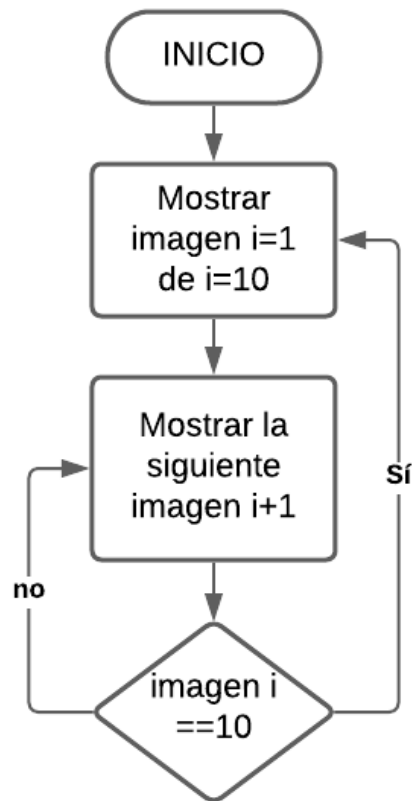


Figura 3.31: Diagrama de flujo de la proyección de imágenes

Para el subproceso de enviar correo electrónico, se debe conectar el *módulo Wi-Fi* a la red; la Figura 3.32 corresponde al diagrama de flujo de conexión a la red, en donde, una vez ingresadas las credenciales, se comprueban que estas sean válidas y se testea el estado de la conexión. Si la conexión fue exitosa, se procede a verificar si se recibe una señal de entrada proveniente del *Arduino Nano*; caso contrario, cuando la conexión ha fallado, se debe ingresar nuevamente las credenciales de red. La Figura 3.33 corresponde al diagrama de flujo del envío de correo electrónico; para el envío del *e-mail*, el *módulo Wi-Fi* verifica si se recibe una señal de entrada proveniente del *Arduino Nano*; cuando se recibe la señal, se procede a enviar el correo electrónico, y a imprimir en el monitor serial si hubo éxito o no al enviar el *e-mail*. Tanto la conexión a la red como el envío de correo electrónico son subprocesos programados en el *NodeMCU V3*.

Adicionalmente, se desarrolló un programa para la comunicación entre la simulación y el *NodeMCU V3*, debido a que no existen librerías para simular el *NodeMCU V3* en Proteus; la Figura 3.34 muestra el diagrama para la comunicación entre la simulación y el *NodeMCU V3*.

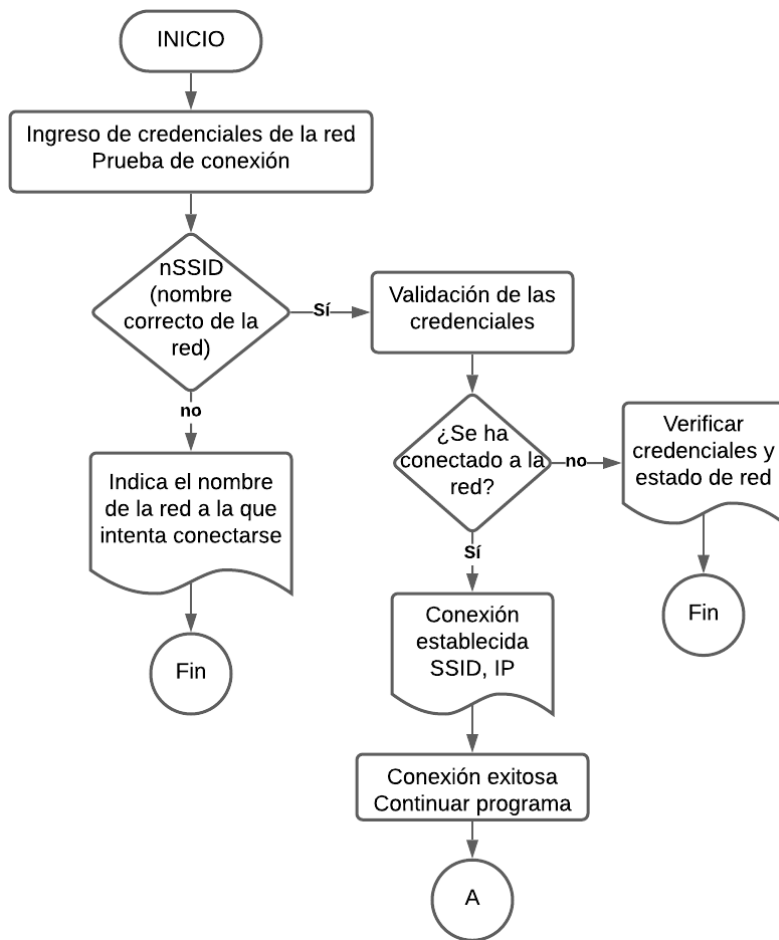


Figura 3.32: Diagrama de flujo de la conexión a la red

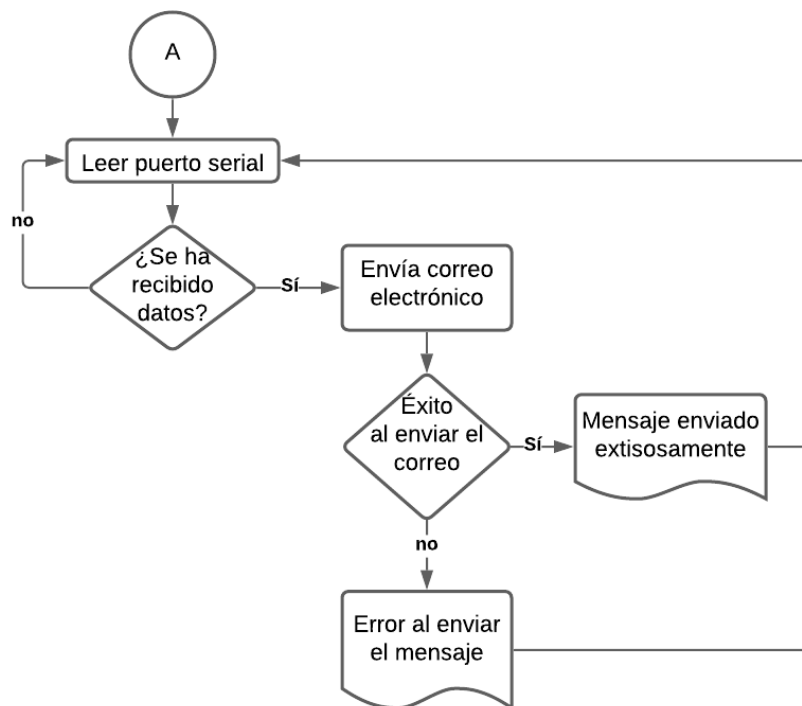


Figura 3.33: Diagrama de flujo del envío de correo electrónico

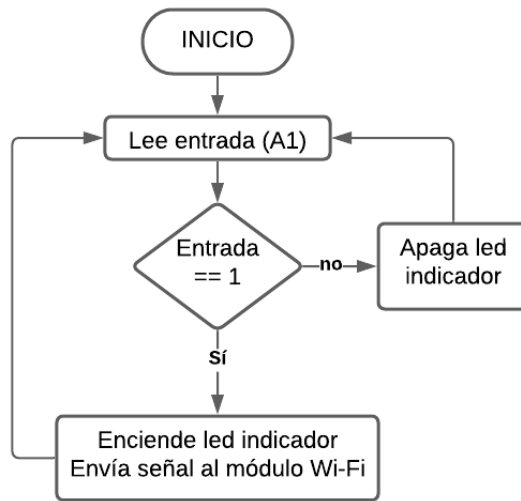


Figura 3.34: Diagrama de flujo comunicación entre simulación y *NodeMCU V3*

La Figura 3.35 muestra el diagrama de flujo del estado de reposo, en el cual la pantalla permanece con brillo mínimo y un fondo negro; además, permanece encendida la luz piloto verde y la tapa cerrada. En este estado, el sensor de presencia permanece atento a que alguien se acerque dentro del perímetro de detección, el cual es menor a 60 cm.

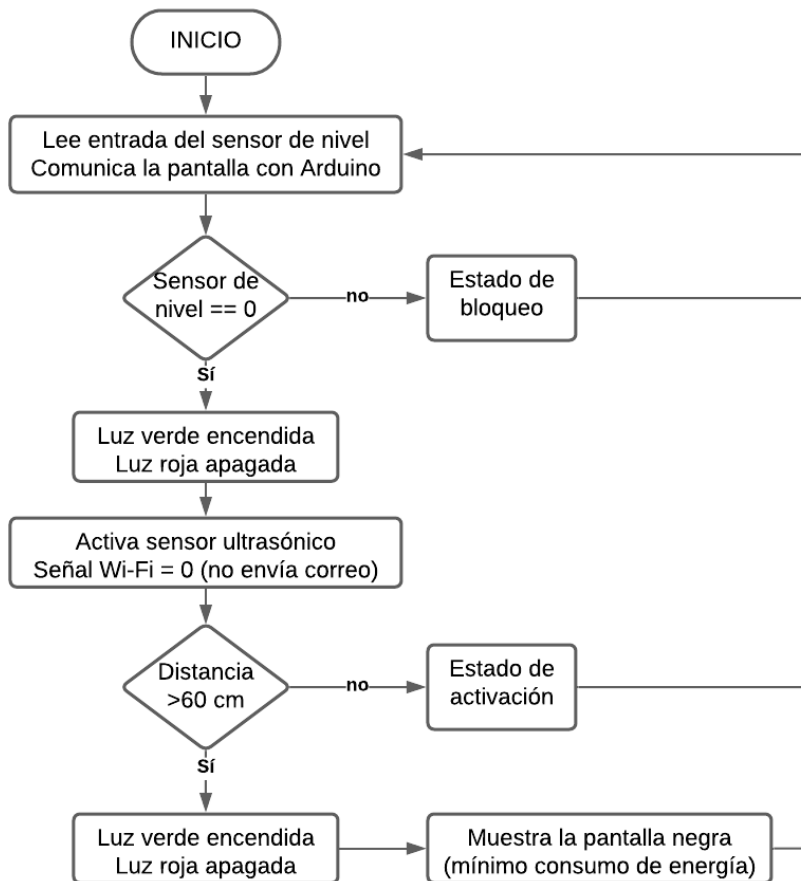


Figura 3.35: Diagrama de flujo del estado de reposo

La Figura 3.36 muestra el diagrama de flujo del estado de activación, en el cual la pantalla presenta una animación, la tapa se abre y permanece en dicho estado por un tiempo de diez segundos; después de ello, la tapa se cierra y enciende la luz piloto roja por un tiempo de cinco segundos; luego de ello, se enciende la luz piloto verde y el sistema vuelve al estado de reposo.

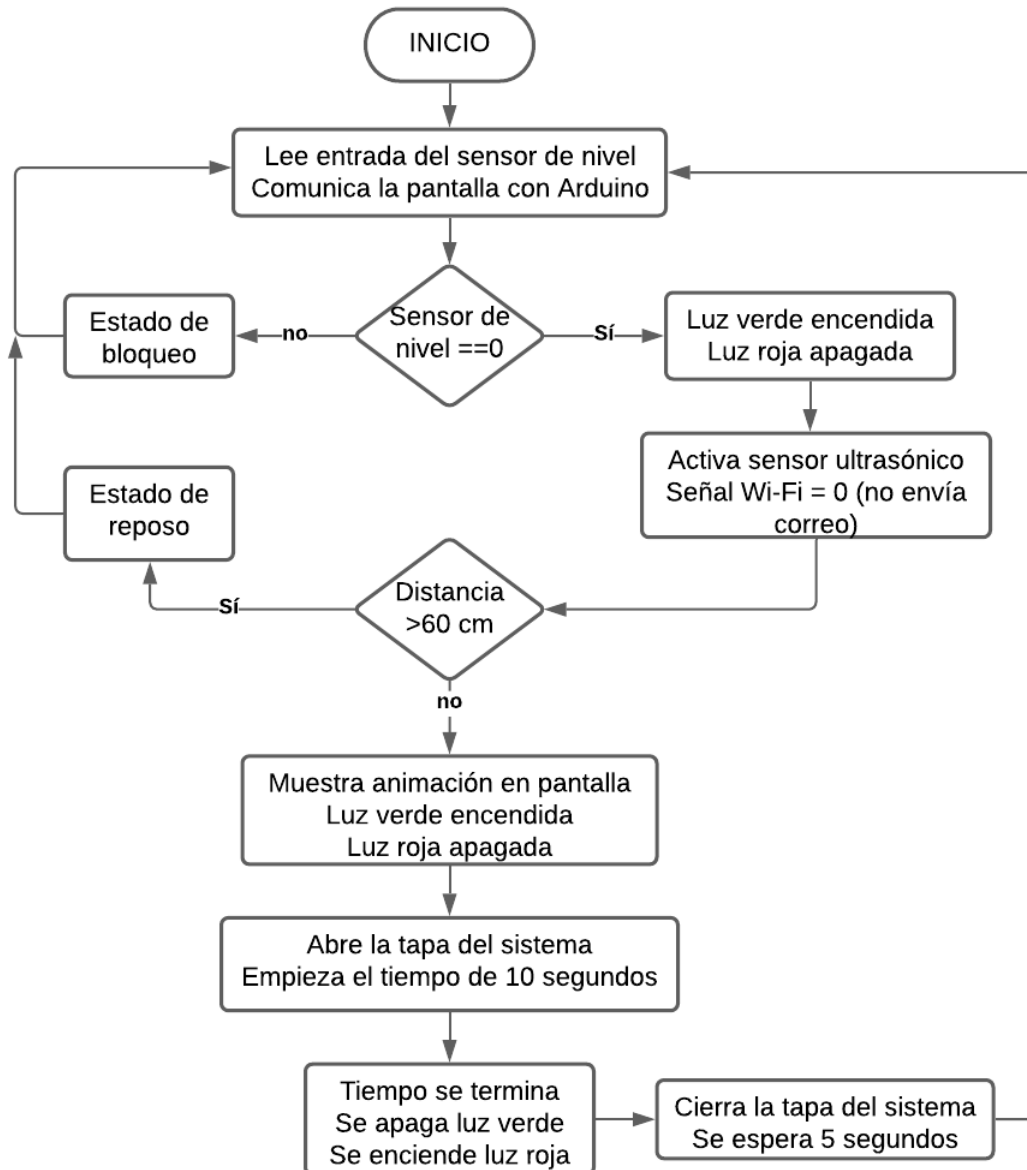


Figura 3.36: Diagrama de flujo del estado de activación

La Figura 3.37 muestra el diagrama de flujo del estado de bloqueo, en el cual la pantalla presenta una imagen estática, se enciende la luz piloto roja y se envía un correo electrónico al *e-mail* establecido mediante programación.

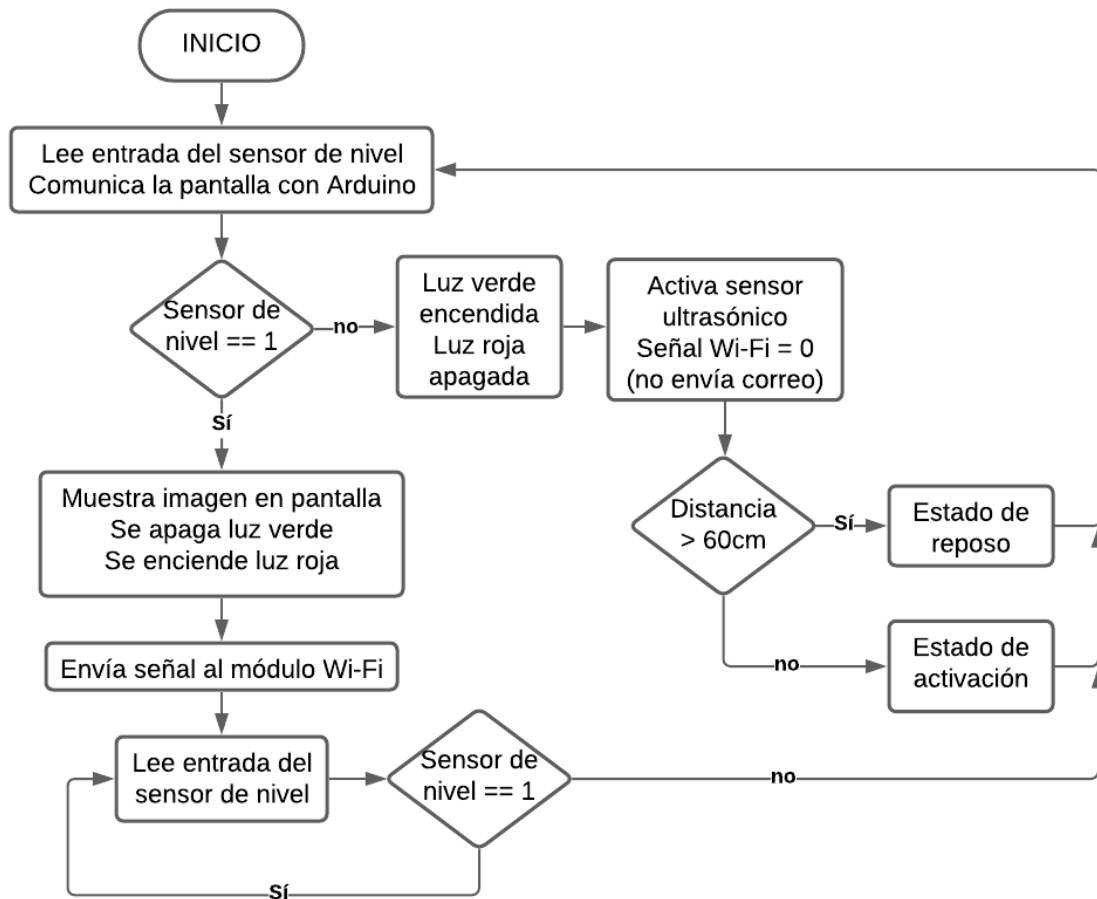


Figura 3.37: Diagrama de flujo del estado de bloqueo

Ruteo del circuito

El ruteo del circuito se realizó mediante el programa ARES, el cual permite ubicar elementos, trazar pistas, dimensionar la placa y una visualización 3D del mismo.

El circuito impreso principal realizado sólo en la *bottom copper* se puede apreciar en la Figura 3.38, y en la Figura 3.39 se observa la visualización 3D del mismo.

Adicionalmente, se presentan desde las Figuras 3.40 a 3.45 pequeños circuitos impresos 3D en donde se ubicarían los sensores y actuadores físicos.

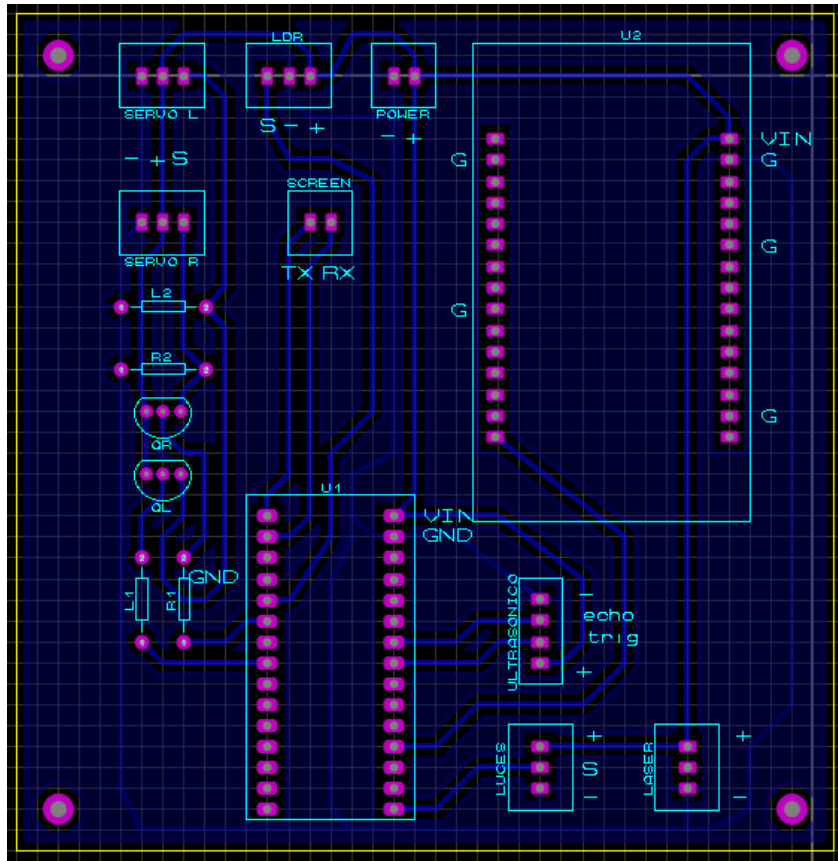


Figura 3.38: Ruteo de la placa

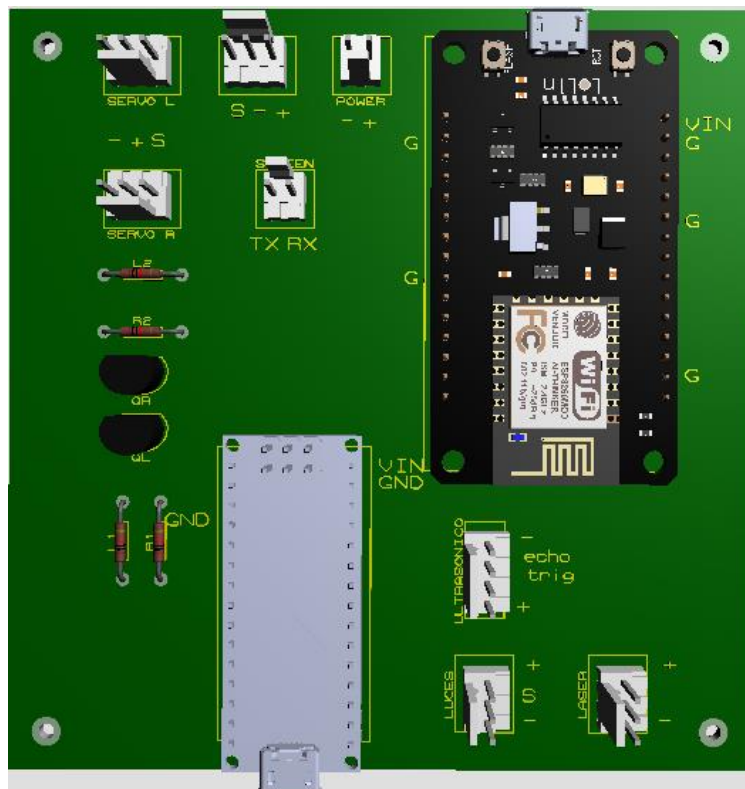


Figura 3.39: Visualización 3D de la placa

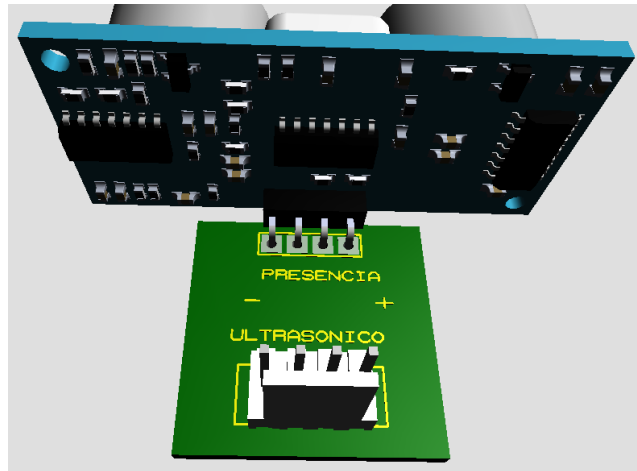


Figura 3.40: Placa para el sensor ultrasónico

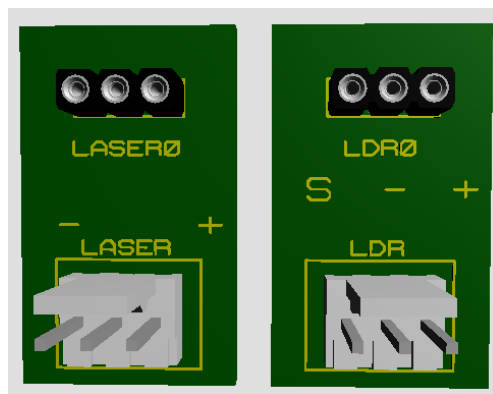


Figura 3.41: Placas para el sensor de nivel

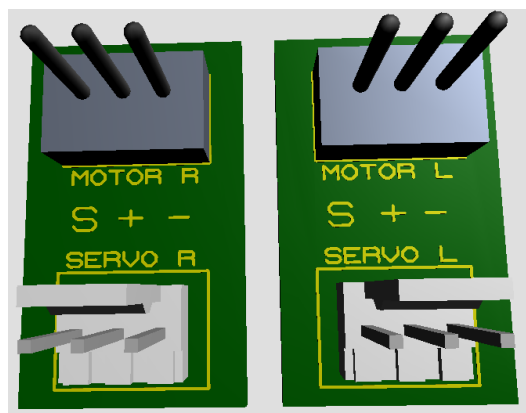


Figura 3.42: Placas para los servomotores

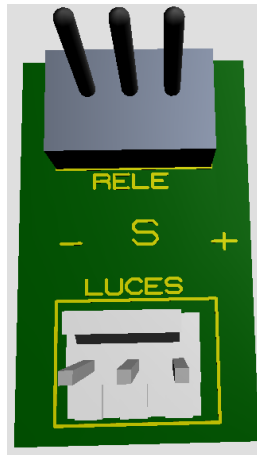


Figura 3.43: Placa para las luces piloto

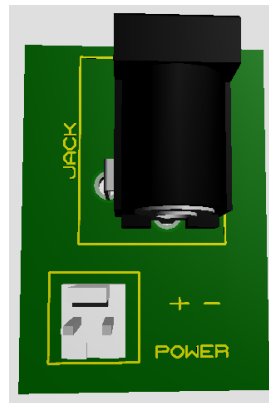


Figura 3.44: Placa para la alimentación

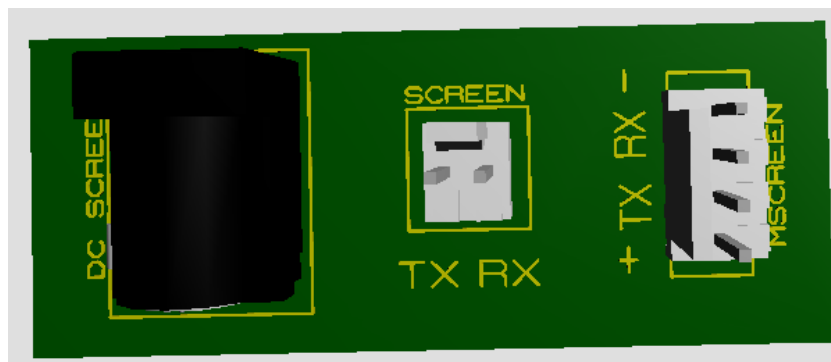


Figura 3.45: Placa para la pantalla

3.4 Desarrollo de la simulación

Uso del sensor ultrasónico para detectar presencia

Para poder llevar a cabo su configuración, primero se deben crear dos variables para almacenar el tiempo y distancia medida por el sensor ultrasónico. Esto se consigue usando la función “*long*” dentro del programa *Arduino IDE*.

La referencia se expresa en la siguiente sintaxis:

```
long tiempo, distancia;
```

Una vez creadas las variables, se debe configurar dos pines de la tarjeta. Uno de ellos se encarga de enviar el pulso sonoro; y el otro, de la recepción. De manera que, una vez obtenido el valor del tiempo de respuesta, puedan ser almacenados los valores en las otras variables creadas anteriormente. La referencia se expresa en la siguiente sintaxis:

- Declaración de pines para el envío y recepción del pulso sonoro

```
int EchoPin = A5;  
int TrigPin = A4;
```

- Configuración de los pines de la tarjeta

```
pinMode(TrigPin, OUTPUT); // Declaro el pin de trigger como salida  
pinMode(EchoPin, INPUT); //Declaro el pin de echo como entrada
```

Cuando se termina el proceso de declaración de pines, se debe incluir la fórmula para medir la distancia en centímetros en el programa principal. Adicionalmente, en la variable de tiempo se debe indicar que el pin “*Echo*” debe recibir el pulso sonoro cuando el pin “*Trigger*” lo haya enviado. La referencia se expresa en la siguiente sintaxis:

```
tiempo = pulseIn(EchoPin, HIGH);  
distancia = (tiempo /2) / 29;
```

Esta fórmula configura de manera correcta el sensor ultrasónico, el cual detectará la presencia del usuario.

Configuración de los servomotores

En el proceso de automatización para la apertura y cierre de la tapa del dispositivo automatizado, se utilizaron servomotores, los cuales son compatibles con la tarjeta *Arduino Nano*. El programa *Arduino IDE* cuenta con una librería integrada, la cual facilita la configuración de los servomotores al momento de controlar la dirección en la que se desplazarán.

Primero, se debe activar la librería en el programa, luego crear dos variables para controlar los servomotores que cumplen la función de abrir y cerrar la tapa. La referencia se expresa en la siguiente sintaxis:

```
#include <Servo.h>
Servo servoMotor1; //variable servol
Servo servoMotor2; //variable servo2
```

Los servomotores trabajan con señales PWM, características que permiten seleccionar los pines 3 y 5 que son los que entregan dichas señales. También, permiten el correcto funcionamiento de los mismos. Además, se debe especificar el estado inicial en el que se encontrarán los servomotores. La referencia se expresa en la siguiente sintaxis:

```
servoMotor1.attach(3); //servo1 trabaja pin 3
servoMotor2.attach(5); //servo2 trabaja pin 5

servoMotor1.write(180); // cerrado
servoMotor2.write(0); // cerrado
```

Una vez finalizada la configuración de los servomotores, estos están listos para realizar la apertura y cierre automático de la tapa.

Librería *Nextion Editor*

Para que la pantalla *Nextion* y la tarjeta de *Arduino Nano* puedan comunicarse a través del puerto serial, es necesario utilizar una librería especial que otorga el fabricante de la pantalla. Al tratarse de una librería oficial, esta posee ejemplos de cómo debe ser utilizada para ciertas aplicaciones. En el sistema automatizado se requiere manejar el cambio de pantallas para poder mostrar la imagen adecuada. La referencia se expresa en la siguiente sintaxis:

```
#include "Nextion.h" //se añade la libreria para la pantalla nextion

NexPage page0 = NexPage(1, 0, "page0"); //Se indica las paginas a ser usadas
NexPage page1 = NexPage(0, 0, "page1"); //de la siguiente manera (posicion, ID, Nombre)
NexPage page2 = NexPage(2, 0, "page2");
```

Una vez incorporada la librería, se procede a agregar las páginas que fueron configuradas en el editor de *Nextion*. Cada una de ellas posee un identificador y nombre único, el cual se debe copiar literalmente del editor.

Para poder establecer comunicación entre la pantalla y la tarjeta es necesario utilizar la siguiente sintaxis:

```
nexInit(); //Comunica la pantalla con el arduino por el puerto serial
```

La referencia anterior es la línea de código que se encarga de iniciar la comunicación por el puerto serial. Para mostrar la imagen en pantalla que se desee, se usa la siguiente sintaxis.

```
page0.show(); //se muestra la pagina 0 de la pantalla nextion
```

La referencia anterior, corresponde a la línea de código que permite mostrar, en cualquier instante, la página con la imagen adecuada.

Configuración de la pantalla *Nextion*

La pantalla *Nextion* posee su *software* oficial propio de configuración llamado *Nextion Editor*, el cual permite crear interfaces gráficas. Para el sistema automatizado únicamente se necesitó manejar cambios de pantalla. En cada pantalla se ha colocado imágenes específicas para cada estado que comprende el dispositivo.

Existen diferentes tamaños de pantalla que el usuario puede elegir, según su necesidad. Para la elaboración del sistema automatizado se eligió el modelo de 7 pulgadas, el cual permite una mejor visualización de imágenes.

Cuando se crea un nuevo proyecto en el editor de *Nextion*, se debe nombrar al archivo y guardarlo en un lugar seguro donde pueda ser encontrado con facilidad. Las Figuras 3.46 y 3.47 muestran el procedimiento a seguir.

Debido a que existen tres modelos diferentes de pantalla en los que cada versión superior ofrece mejores características y mayor capacidad de memoria para crear proyectos el tamaño y formato de imagen es un aspecto importante a considerar. Los formatos de imagen soportados por la pantalla son JPG (*Joint Photographic Experts Group*) y PNG (*Portable Network Graphics*).



Figura 3.46: Creación de nuevo proyecto en *Nextion Editor*

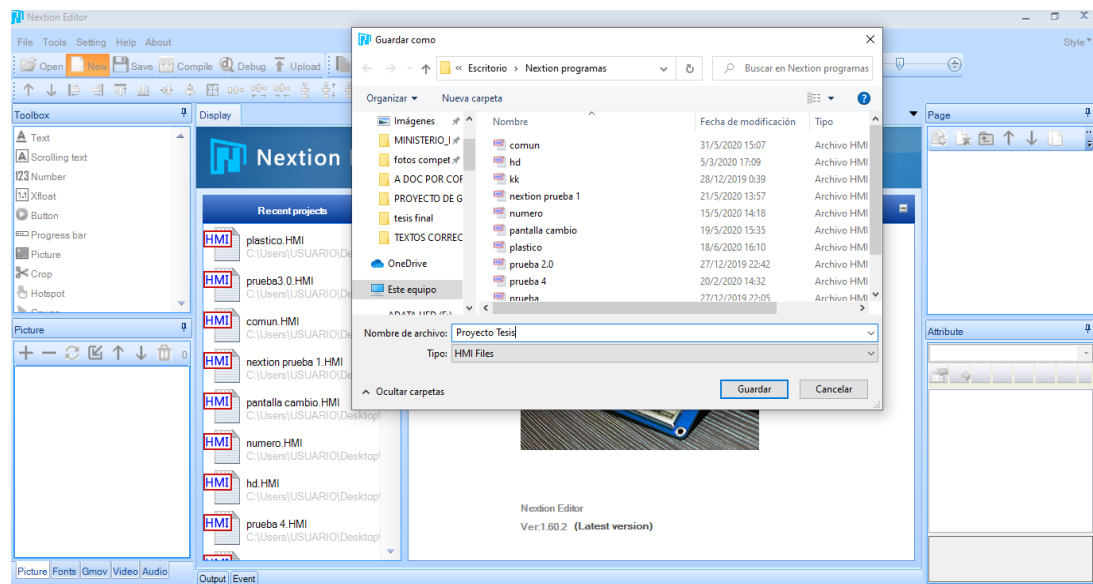


Figura 3.47: Guardado de proyecto en *Nextion Editor*

Cuando se haya creado el archivo, aparece una ventana como la que se muestra en la Figura 3.48. Inmediatamente, se debe seleccionar el tipo de pantalla, su resolución y la orientación de la hoja donde se trabaja.

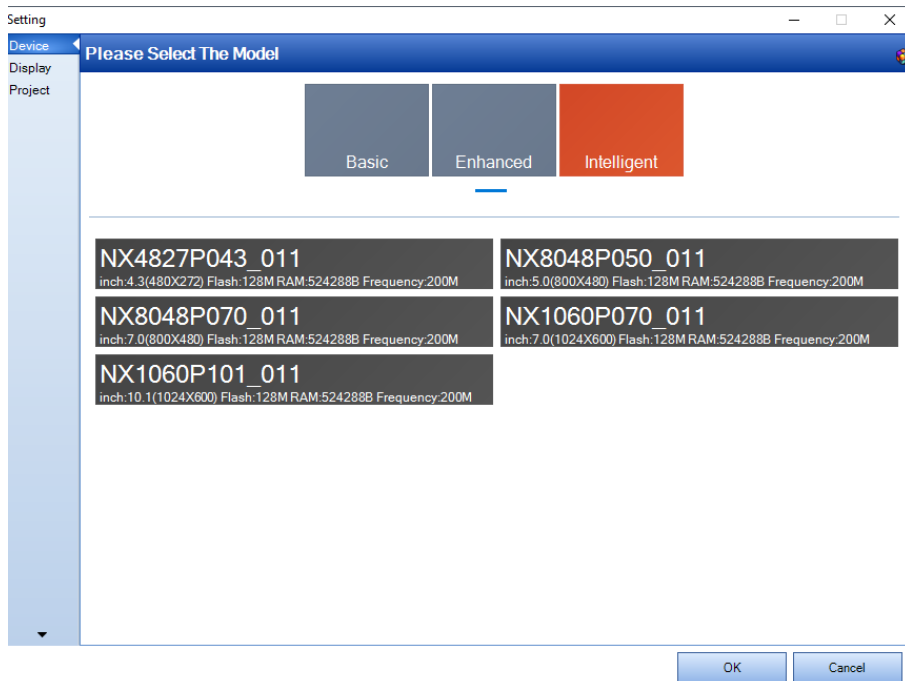


Figura 3.48: Selector de pantalla en *Nextion Editor*

El sitio *web* oficial de *Nextion*, posee una lista detallada con los diferentes tipos de pantalla que se ofrecen en el mercado. Para el diseño del sistema automatizado se ha elegido una versión básica de 7 pulgadas. En las Figuras 3.49 y 3.50 se puede apreciar la configuración que se utiliza para el desarrollo del proyecto.

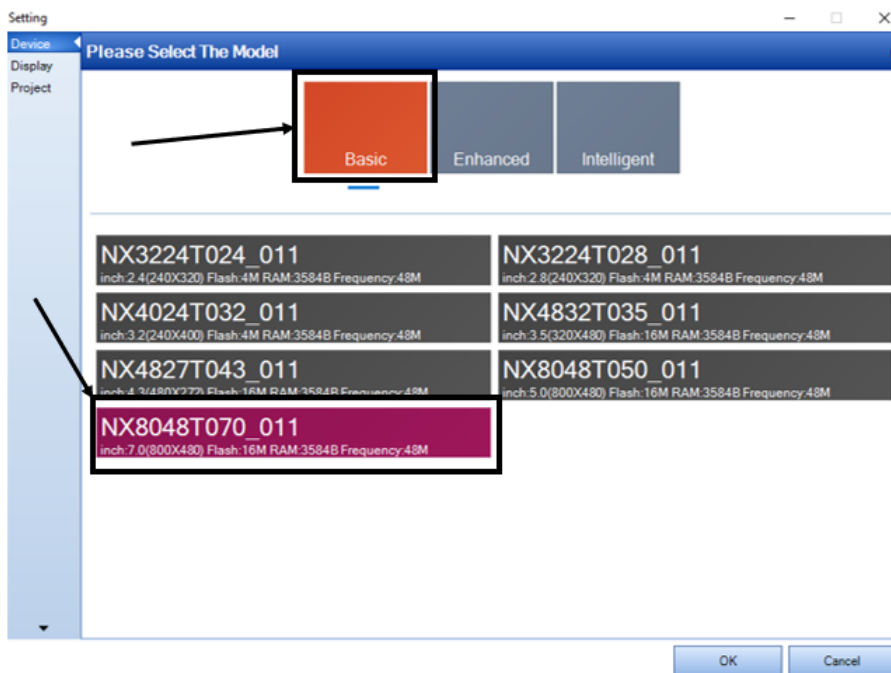


Figura 3.49: Selección de la pantalla de 7"

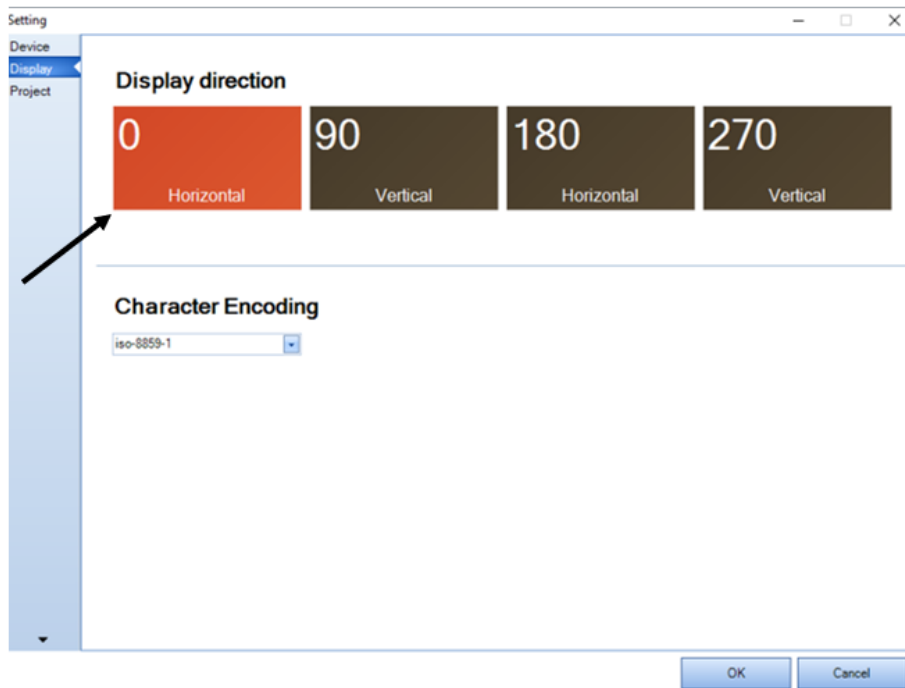


Figura 3.50: Configuración de la orientación de la pantalla

Finalizada la configuración, se presenta la ventana de trabajo mostrada en la Figura 3.51.

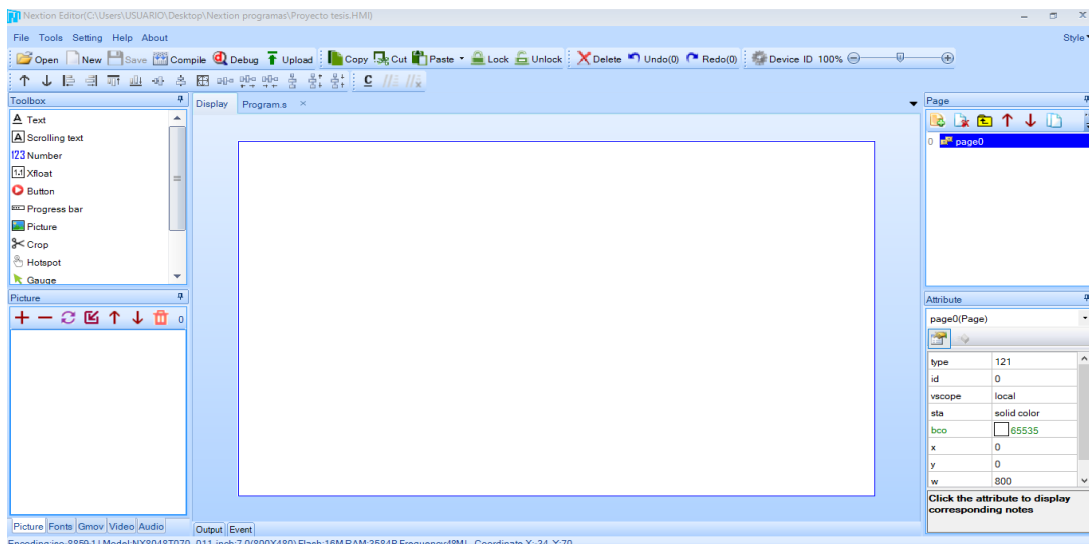


Figura 3.51: Lugar de trabajo en *Nextion Editor*

Se debe añadir dos páginas más, como se muestra en la Figura 3.52; en total, se tiene tres páginas, una para el estado de reposo, una para el estado de activación y una para el estado de bloqueo.



Figura 3.52: Adición de páginas en *Nextion Editor*

Una vez agregadas las páginas, se debe añadir las imágenes que van a ser utilizadas para todo el proyecto. Como se muestra en la Figura 3.53 se da clic en el ícono “más” para poder agregar imágenes al programa.

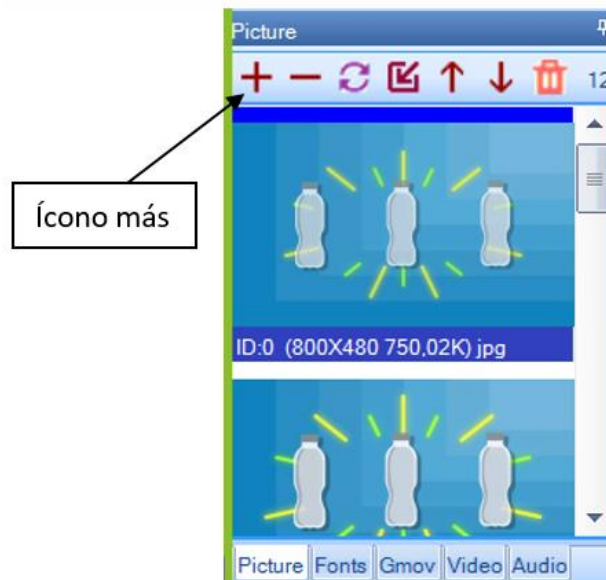


Figura 3.53: Adición de imágenes en *Nextion Editor*

En la parte inferior derecha del programa se encuentra la sección “*Attribute*”, como se muestra en la Figura 3.54; en donde, se escoge la página para proceder a editarla. Cada página tiene tres opciones para seleccionar el fondo de pantalla. Para insertar una imagen, como se muestra en la figura 3.55, se debe seleccionar la opción “*image*”.

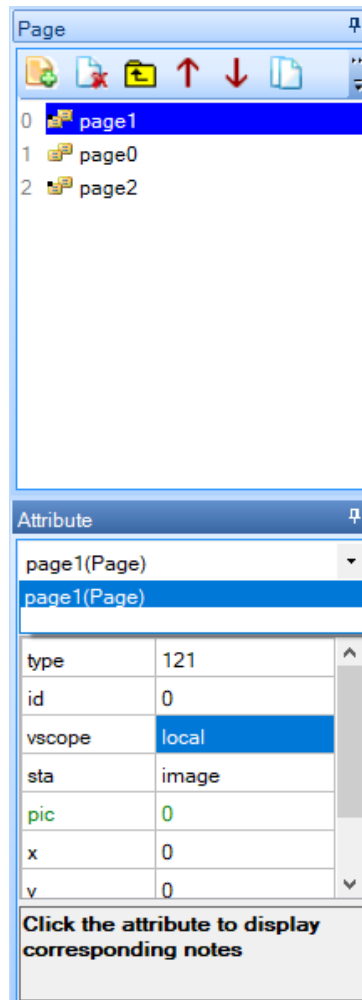


Figura 3.54: Elección de página para edición en *Nextion Editor*

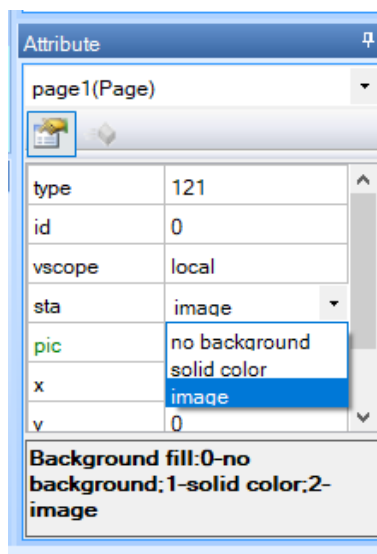


Figura 3.55: Adición de imagen de fondo en *Nextion Editor*

Como se muestra en la Figura 3.56, se selecciona “browse” para buscar la imagen de fondo deseada entre las que fueron añadidas. Se debe seleccionar la primera imagen para crear la animación.

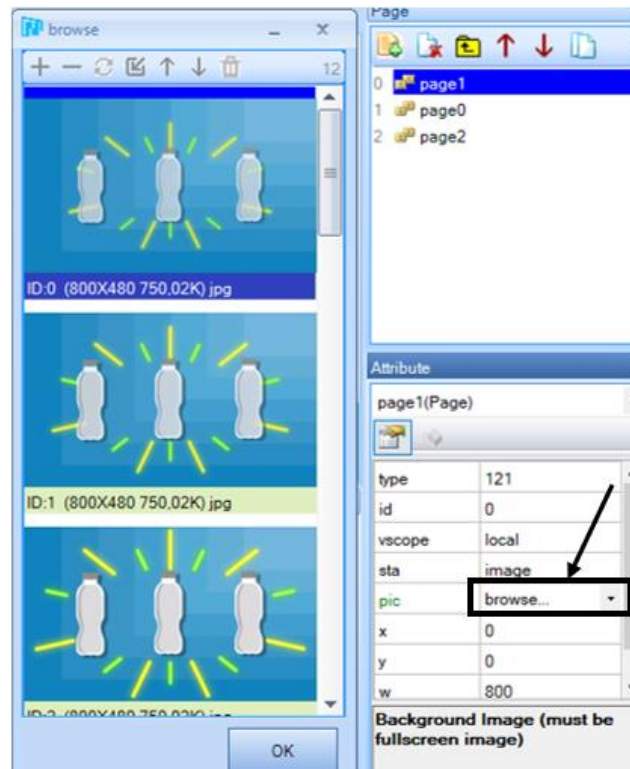


Figura 3.56: Selección de imagen de fondo en *Nextion Editor*

Cada una de las páginas añadidas cumple un rol diferente en el sistema automatizado. La página uno muestra la animación; la página cero, la imagen de lleno. La página dos usa un fondo totalmente negro con brillo al mínimo que simula un estado de reposo, en el cual el consumo de energía es mínimo, esto con el fin de salvaguardar la pantalla.

La Figura 3.57 muestra el uso del comando “dim”, el cual permite configurar el brillo de la pantalla, siendo cero el mínimo y cien el máximo. Para añadir esta línea de código se debe dar clic en “page” para abrir la ventana de configuración.

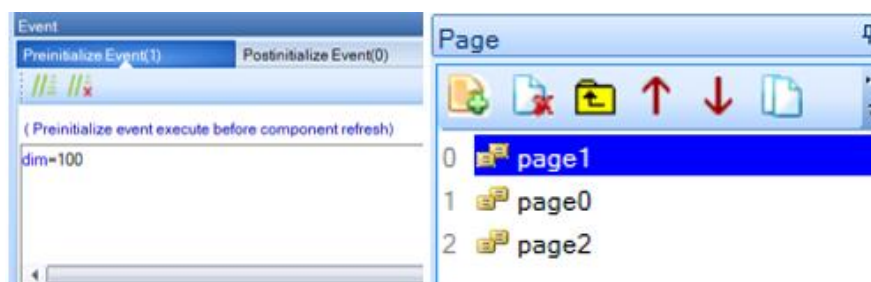


Figura 3.57: Establecimiento del brillo de la pantalla en *Nextion Editor*

El sistema automatizado cuenta con dos estados principales. En el primer estado, de activación, la imagen de la basura, que puede ser depositada, tendrá una animación simulando ser la de un *gif* con el propósito de llamar la atención de los niños con TEA. En el segundo estado, cuando el basurero se encuentre lleno, se mostrará una imagen estática indicando que el sistema se encuentra lleno.

Para poder llevar a cabo la animación, es necesario añadir el elemento *timer*, como se indica en la Figura 3.58; este permite realizar la transición de varias imágenes con efectos de animación.

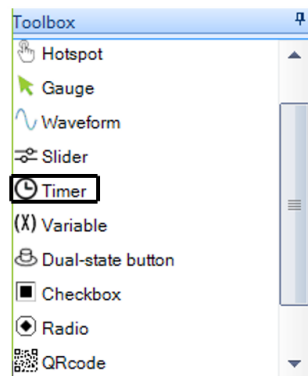


Figura 3.58: Adición de *timer* en *Nextion Editor*

Como se muestra en la Figura 3.59, una vez agregado el elemento *timer*, se despliega un ícono con el nombre *tm0*. Se debe dar clic en él para poder desplegar la ventana de configuración que desarrolla el código del *timer*.

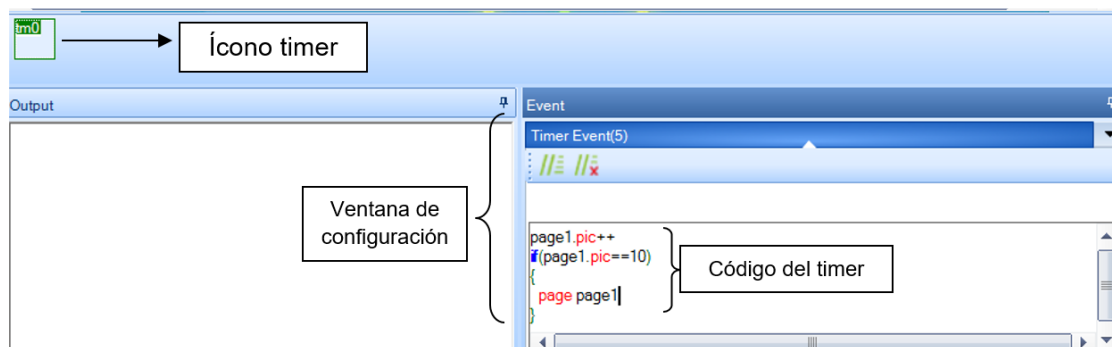


Figura 3.59: Configuración del *timer* en *Nextion Editor*

La velocidad de la transición de las imágenes se puede configurar en la ventana *Attribute*, como se muestra en la Figura 3.60. Para este caso, se usó un tiempo de 100

milisegundos que permite obtener una animación óptima al momento de ser presentada en la pantalla.

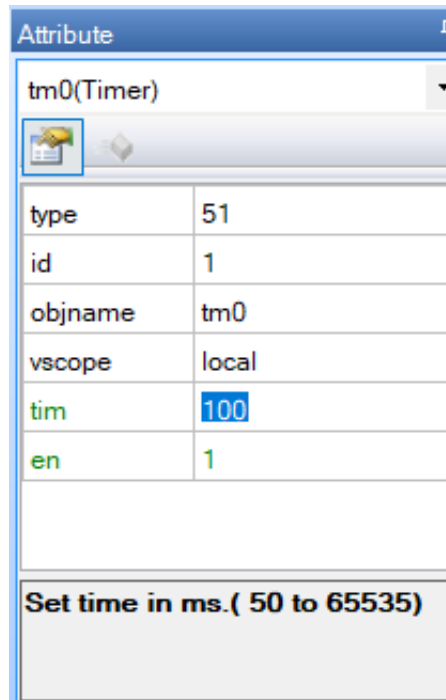


Figura 3.60: Configuración del tiempo de transición de imágenes

Estado de activación

Durante el estado de activación, el sistema automatizado realizará varias acciones para que los niños con TEA puedan colocar la basura en el contenedor respectivo. Esta acción cuenta con un tiempo de 10 segundos, desde que se abre hasta que se cierra la cubierta del sistema. Previamente, para llegar a este estado, el usuario debe encontrarse dentro del perímetro de activación de 60 centímetros, entonces el sensor de nivel debe detectar que el dispositivo no se encuentra lleno. Una vez ahí, la tapa del basurero se accionará de manera automática, gracias a los servomotores que fueron configurados para que den un giro de 180 grados, abriéndola en su totalidad.

Cuando se cumple la acción de apertura de la cubierta, las luces piloto funcionan como señales de alerta. La luz verde permanece encendida indicando que se cuenta con tiempo suficiente para botar la basura. Una vez que finaliza esa acción, la luz roja se activa e indica que el tiempo se ha terminado y que se deben esperar 3 segundos para reiniciar el proceso.

La pantalla permanece activa durante todo el proceso de uso del sistema automatizado e indica el contenido (imagen o pictograma) que se depositará en el respectivo contenedor: papel, plástico o basura orgánica.

Para configurar los periféricos que permiten realizar todas estas acciones, se ha utilizado la siguiente sintaxis:

```

void loop() {                                     //Inicia el programa
dat=digitalRead(in_sensor);                       //Lee la entrada in_sensor
nexInit();                                        //Comunica la pantalla con el arduino
while (dat==0) {
  digitalWrite(luces, HIGH);                       //Enciende luz verde
  digitalWrite(wifi, LOW);                          //Salida pin Wi-Fi cero
  digitalWrite(TrigPin, LOW);                       //Envio 0 logico por trigger
  delayMicroseconds(4);                             //Realizo un retraso de 4 microsegundos
  digitalWrite(TrigPin, HIGH);                      //Envio 1 logico por trigger
  delayMicroseconds(10);                            //Retraso de 10 us
  digitalWrite(TrigPin, LOW);                       //Envio 0 logico por trigger

  tiempo = pulseIn(EchoPin, HIGH); //Detecta llegada de señal por echo cuando este da l
  distancia = (tiempo /2) / 29; //Formula para obtener la distancia en cm

  if (distancia > 60){                             //Condicional if delimitando el rango de medida
    page2.show();                                   //Muestra pag2 de la pantalla
    delay(100);                                     //Retardo para vizualizar la imagen
    digitalWrite(luces, HIGH);                     //Enciende luz verde
    dat=digitalRead(in_sensor);                     //Lee el pin del sensor de nivel
  }
  else {                                             //Condicional dentro del rango de medida
    pagel.show();                                   //Muestra pag1 de la pantalla
    digitalWrite(luces, HIGH);                       //Enciende luz verde
    servoMotor1.write(180);                          //180 grados cerrado
    servoMotor2.write(0);                            //0 grados cerrado
    delay(100);
    servoMotor1.write(0);                             //0 grados abierto
    servoMotor2.write(180);                          //180 grados abierto
    delay(7000);
    digitalWrite(luces, LOW);                        //Enciende luz roja
    delay(3000);
    servoMotor1.write(180);                          //180 grados cerrado
    servoMotor2.write(0);                            //0 grados cerrado
    delay(5000);
    dat=digitalRead(in_sensor);                       //Lee el sensor de nivel
  }
}
}

```

Estado de interrupción

Durante este estado, el sistema automatizado permanece bloqueado; es decir, que no se podrá depositar basura. Esta función se cumple cuando el sensor de nivel ha detectado que el basurero ha llegado a su capacidad máxima.

La pantalla proyecta una imagen, la cual indica que no se puede arrojar basura. Además, la luz piloto roja se activa e indica que el dispositivo se podrá reutilizar cuando se haya vaciado su contenido.

Al final del proceso, la tarjeta central envía la señal digital al módulo *Wi-Fi* para que éste, a su vez, envíe información sobre el estado del sistema automatizado en un correo electrónico al administrador. Se utilizó la siguiente sintaxis:

```
if(dat==1){                                     //Si la variable recibe 1 lógico
  digitalWrite(luces, LOW);                     //Enciende luz roja
  page0.show();                                 //Muestra la pag0 de la pantalla
  delay(3000);                                  //Retardo para vizualizar la imagen
  digitalWrite(wifi,HIGH);                     //Led indicador encendido wi-fi
  delay(3000);
  digitalWrite(wifi,LOW);                       //Led indicador apagado wi-fi
  dat=digitalRead(in_sensor);                  //Testea entrada sensor de nivel
  while(dat==1){                                //Bucle detecta cambio de estado sensor de nivel
    delay(100);
    dat=digitalRead(in_sensor);
  }
}
```

Estado de reposo

Este estado depende de dos factores. El primero, si el sensor de nivel indica que el basurero no ha alcanzado el límite de su capacidad. El segundo, si ningún usuario se encuentra dentro del perímetro de activación.

Si se cumplen las dos condiciones anteriores, el sistema automatizado se encuentra en *stand by*. La pantalla entra en estado de ahorro de energía y muestra una pantalla negra, la cual tiene el nivel de brillo al mínimo que permite consumir la menor energía posible.

Los dispositivos periféricos complementarios se hallan inactivos, a excepción del sensor ultrasónico y la luz piloto verde. El sensor ultrasónico, constantemente está atento si un usuario se halla dentro del perímetro de activación; mientras que, la luz verde indica que el sistema puede ser utilizado. Se ha utilizado la siguiente sintaxis:

```
if (distancia > 60){                            //Condicional if delimitando el rango de medida
  page2.show();                                  //Muestra pag2 de la pantalla
  delay(100);                                    //Retardo para vizualizar la imagen
  digitalWrite(luces, HIGH);                    //Enciende luz verde
  dat=digitalRead(in_sensor);                  //Lee el pin del sensor de nivel
}
```


Librería *Gsender*

Es una librería creada para enviar correo electrónico a través de *Gmail*. Es compatible con el módulo *WiFi ESP8266 NodeMCU V3*, el cual debe estar conectado a internet para enviar el correo.

Se debe ingresar las credenciales de acceso del correo *Gmail* que será utilizado para enviar el *e-mail*. Además, se indica que, el puerto SMTP 465 utilizado para enviar correos electrónicos, es el más seguro porque trabaja con capa de *sockets* seguros.

Para iniciar con la configuración de la librería, se debe contar con tres archivos, el primero será de *Arduino IDE*, el segundo un tipo C++ *Source* y el tercero un archivo “.H”, que es la librería.

En la Figura 3.61 se puede apreciar los tres archivos que deben encontrarse en la misma carpeta para que pueda ser utilizada la librería.

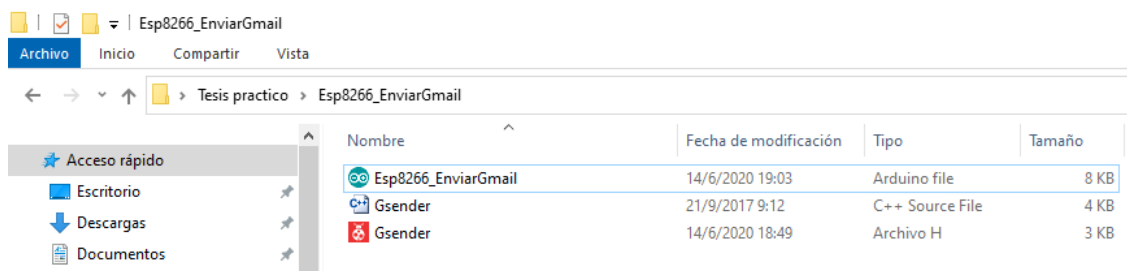


Figura 3.61: Archivos de la librería *Gsender*

Se debe ingresar al archivo de *Arudino IDE* para realizar una primera configuración de la librería, la cual cumple varias funciones: acceso a la cuenta de *Gmail*, asignación del puerto SMTP y detección de errores cuando no se pueda enviar el mensaje.

La Figura 3.62 muestra cómo se ven los archivos dentro de *Arduino IDE*. Se debe seleccionar la pestaña *Gsender.h*, en la cual se realiza la configuración de la librería para los accesos mencionados anteriormente.

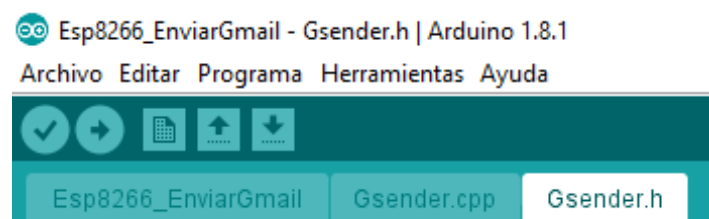


Figura 3.62: Archivos *Gsender* en *Arduino IDE*

Se debe activar la librería antes mencionada en la pestaña seleccionada e incluir la librería *WifiClientSecure*, la cual establece las características del modo cliente del dispositivo electrónico. Esto quiere decir que, busca conectarse a un servidor de manera segura. Se ha utilizado la siguiente sintaxis:

```
#ifndef G_SENDER //Inicia la librería
#define G_SENDER //Indica la librería a usarse
#define GS_SERIAL_LOG_1 // Muestra en el monitor serial las respuestas del servidor
#include <WifiClientSecure.h> //Establece las características del modo cliente del módulo wifi
```

Es necesario activar el acceso de aplicaciones poco seguras para establecer la conexión con la cuenta *Gmail*, esto se encuentra en la opción de seguridad de la cuenta.

Las Figuras 3.63 y 3.64 muestran el procedimiento para realizar la acción.

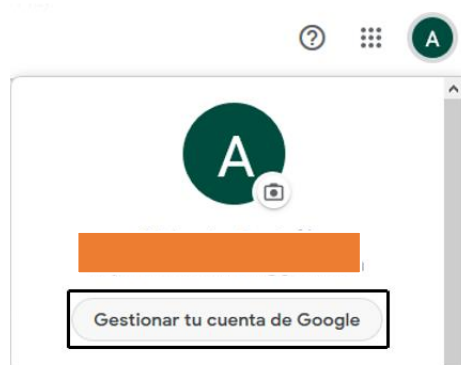


Figura 3.63: Configuración en la cuenta de *Gmail*

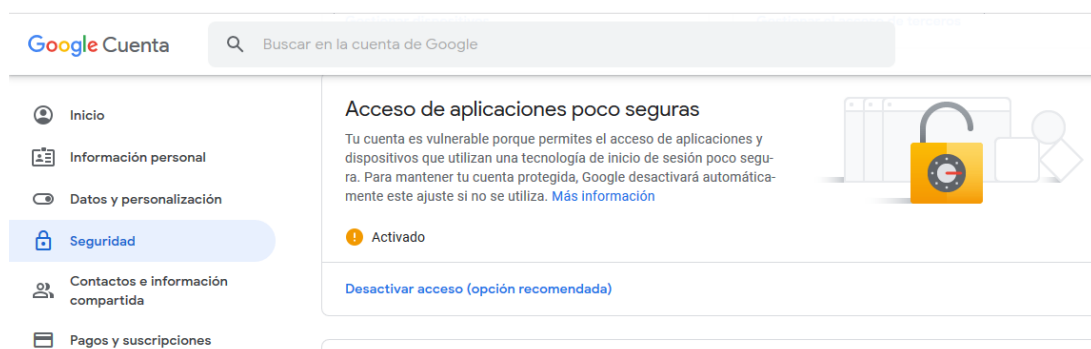


Figura 3.64: Acceso del *NodeMCU V3* a la cuenta de *Gmail*

Se debe escribir en código base 64 el correo y contraseña para acceder de manera remota a la cuenta *Gmail*, este es un requerimiento de la librería. En otra línea de configuración, el correo se digita en español; además, se añaden comandos para la conexión con el servidor y la detección de errores en el envío del mensaje.

Esta sintaxis es un ejemplo modificable de la librería que se adaptó al requerimiento del sistema automatizado. También, es necesario indicar que esta configuración forma parte del programa principal. Es decir, mediante otros comandos y variables, que se explican posteriormente, se complementa el envío del *e-mail*. Para esta sección se ha utilizado la siguiente sintaxis:

```
class Gsender //inicia el programa
{
protected: //usa de manera segura la librería
    Gsender();
private:
    const int SMTP_PORT = 465; //Uso del puerto 465 de SMTP para enviar correo
    const char* SMTP_SERVER = "smtp.gmail.com"; //conexión al servidor SMTP
    const char* EMAILBASE64_LOGIN = "XXXXXXXXXXXXXXXXXXXX"; //E-mail base 64 remitente
    const char* EMAILBASE64_PASSWORD = "XXXXXXXXXXXXXXXXXXXX"; //Contraseña base 64
    const char* FROM = "basutronpapel@gmail.com"; //E-mail del remitente
    const char* _error = nullptr; //Comprobación de acceso
    char* _subject = nullptr;
    String _serverResponse; //Respuesta del servidor
    static Gsender* _instance;
    bool AwaitSMTPResponse(WiFiClientSecure &client, const String &resp = "", uint16_t timeOut = 10000); //Tiempo de espera para respuesta del servidor
public: //Comandos para conocer errores al momento //de enviar el correo y las respuestas //del servidor
    static Gsender* Instance();
    Gsender* Subject(const char* subject);
    Gsender* Subject(const String &subject);
    bool Send(const String &to, const String &message);
    String getLastResponse();
    const char* getError();
};
#endif // G_SENDER //Finaliza el programa
```

Desarrollo del programa principal para el ESP8266 NodeMCU V3

Para configurar el módulo *Wi-Fi* es necesario descargar la librería ESP8266WiFi.h, disponible en el repositorio digital de *GitHub*. Esta posee los comandos necesarios para conectarse a una red local y comunicarse, a la vez, con la librería *Gsender*.

Se debe añadir las librerías *Gsender* y ESP8266WiFi.h al programa principal. También, se crean variables en las cuales se guardan datos como: nombre de la red inalámbrica a la que se conectará el módulo, contraseña de la red, estado inicial para conexión a la red, tiempo para intentar una reconexión y el mensaje que se envía por correo electrónico. Se ha utilizado la siguiente sintaxis:

```

#include <ESP8266WiFi.h>           //Añade la librería para el ESP8266
#include "Gsender.h"              //Añade la librería Gsender
#pragma region Globals           //Inicia la extensión de código de la librería

const char* ssid = " ";          //Red local a la que se conecta
const char* password = " ";     //Contraseña de la red

uint8_t connection_state = 0;    //Determina el estado de la conexión
uint16_t reconnect_interval = 10000; //Si no se conecta espera este tiempo para volver a intentar
#pragma endregion Globals        //Termina la extensión de código de la librería
String TramaMensajeGmail = "";   //Variable usada para guardar el mensaje que se enviará

```

La siguiente sintaxis muestra el procedimiento que debe realizarse para conectar el dispositivo a la red. Primero, se debe comprobar que las credenciales (Nombre de red y contraseña) sean correctas; en el monitor serial se informa al usuario, periódicamente, si se está logrando establecer conexión.

```

uint8_t WiFiConnect(const char* nSSID = nullptr, const char* nPassword = nullptr)
{
    //Inicialización para conectarse a la red
    static uint16_t attempt = 0; //Primer intento para conectarse
    Serial.print("Conectando a "); //Indica en el monitor serial a que red se está conectando
    if(nSSID) { //Condición IF para la conexión a la red
        WiFi.begin(nSSID, nPassword); //Identifica la red e ingresa la contraseña
        Serial.println(nSSID); //Muestra en el monitor serial el SSID de la red
    } else { //Si la primera condición no se cumple
        WiFi.begin(ssid, password); //Comprueba si la red y contraseña son correctas
        Serial.println(ssid); //Indica la red a la que se quiere conectar
    }
}

```

Si la información ingresada es correcta, se establece conexión a la red inalámbrica local. Mediante los comandos *WiFi.status* y *WL_CONNECTED*, se mantiene activo al enlace.

Se ha creado una variable "i", la cual permite limitar el número de intentos que hará el módulo para conectarse a la red; de esta manera, se evita un bucle infinito, cuando no sea posible enlazarse a la red. Además, el condicional *attempt*, utilizado en la sintaxis anterior junto con la variable "i", informa al usuario, si se ha establecido el enlace.

El funcionamiento se basa en dos condiciones:

- Si "i ++ <50 y attempt = 0"
 Cuando se cumplan estas dos condiciones, se ha tenido éxito en la conexión del dispositivo electrónico con la red inalámbrica local. Además, se observa en el monitor serial el SSID de la red y los mensajes "conexión establecida", seguido de la dirección IP que se le ha asignado al módulo *Wi-Fi*.
- Si "i== 51 y attempt % 2 ==0"
 Si se da esta condición, quiere decir que se ha excedido el número de intentos para enlazarse a la red. En el monitor serial se informa al usuario

que se ha agotado el tiempo de espera para establecer conexión. Razón por la cual, se le recomienda verificar, tanto la disponibilidad de la red como la validez de las credenciales introducidas. La sintaxis utilizada es la siguiente:

```
uint8_t i = 0; //Inicia variable i en cero
while(WiFi.status() != WL_CONNECTED && i++ < 50) //Condiciona para mantener la conexión a la red
{
    delay(200); //Retardo para poder comprobar la conexión
    Serial.print("."); //imprime un punto en el monitor serial para informar la conexión
}
++attempt; //contador para detectar la conexión
Serial.println(""); //imprime mensaje de estado de conexión
if(i == 51) { //Si la variable i es igual a 51
    Serial.print("Conexion: supero el limite de tiempo TIMEOUT: ");
    Serial.println(attempt); //Muestra el tiempo que exedió en la espera de conectarse a la red
    if(attempt % 2 == 0) //Si el contador supera el 0 inicial de conexión
        Serial.println("Verifique si el Access Point esta disponible o verifique el SSID y el Password\r\n");
    return false; //Finaliza la acción
}

Serial.println("Conexion: ESTABLECIDA");
Serial.print("Direccion IP Leida: ");
Serial.println(WiFi.localIP());
return true; //Mantiene la conexión
}
```

A continuación, se indica la configuración que se debe realizar para mantener estable el enlace. Se considera un tiempo de 50 milisegundos para comprobar el estado actual de la conexión y se da un margen de tiempo para poder reconectarse en caso de ocurrir algún error inesperado.

```
void Awaits() //Estado para esperar tiempos de conexión con la red
{
    uint32_t ts = millis(); //Inicializa la variable ts en milisegundos
    while(!connection_state) //Mientras se encuentre en el estado de conexión a la red
    {
        delay(50); //Retraso de 50ms para comprobar
        if(millis() > (ts + reconnect_interval) && !connection_state){
            connection_state = WiFiConnect();
            ts = millis(); //Condiciona If para comprobar
        } //que el módulo se encuentra conectado
    }
}
```

En la siguiente sintaxis se puede observar la activación del monitor serial y del comando que permite mantener conectado el módulo *Wi-Fi* a la red.

```
void setup() //declaracion de variables
{
    Serial.begin(9600);
    connection_state = WiFiConnect(); //Mantiene el modulo conectado a la red
}
```

Cuando el módulo *Wi-Fi* reciba un dato por el puerto serial, se envía el correo electrónico al administrador. La librería *Gsender* permite enviar el *e-mail* de manera rápida, al haber configurado el remitente con anterioridad y permitir almacenar el mensaje programado en una variable llamada `EnviarMensajeGMAIL`. Se ha utilizado la siguiente sintaxis:

```
void loop()                                //Inicia el programa
{
  if (Serial.available()){
    char car = Serial.read(); //Espera caracter para poder enviar el correo
    Serial.println(car);      //Lee caracter enviado y lo borra del buffer
    delay(1000);              //Retardo para enviar el correo
    EnviarMensajeGMAIL();     //Se envia el correo
  }
}
```

Se debe crear el mensaje que será enviado en el correo electrónico. Se recomienda, en esta sección del programa, verificar si el módulo se encuentra conectado a la red antes de enviar el *e-mail*.

Una vez comprobado el enlace, se debe indicar que, para esta sección, se utiliza la librería *Gsender*, cuyas funciones son acceder a las credenciales del remitente del mensaje, añadir el correo del receptor e informar si hubo éxito en el envío.

Para la redacción del correo se ha usado HTML, el cual es más recomendable al momento de enviar un mensaje, ya que permite colocar título principal, cuerpo y mensaje de despedida.

La librería permite conocer si el mensaje fue enviado. Esta información está disponible en el monitor serial. Se ha usado la siguiente sintaxis:

```
void EnviarMensajeGMAIL (void)
{
  if(!connection_state) //Si no se encuentra conectado a la red
    Awaits();           //se encuentra constantemente intentando conectarse a la red

  Gsender *gsender = Gsender::Instance(); //Comunicacion con la librería Gsender

  String subject = "MENSAJE - ESTADO BASURERO PAPEL";
                                     //Asunto del mensaje
  TramaMensajeGmail += "<html>";      //Inicio del cuerpo del mensaje en HTML
  TramaMensajeGmail += "<body>";

  TramaMensajeGmail += "<h1>EL BASURERO QUE CONTIENE PAPEL SE ENCUENTRA LLENO</h1>";
  TramaMensajeGmail += "<br>";

  TramaMensajeGmail += "<p>";
  TramaMensajeGmail += "<b>VACIAR</b>.";
  TramaMensajeGmail += "<br>";
  TramaMensajeGmail += "<b>POR FAVOR</b>.";
}
```

```

TramaMensajeGmail += "</p>";
TramaMensajeGmail += "</body>";
TramaMensajeGmail += "</html>";

if(gsender->Subject(subject)->Send("leonkennedy1297@gmail.com", TramaMensajeGmail)) {
                                                                    //correo del destinatario
    delay(1000);
    Serial.println("MENSAJE ENVIADO EXITOSAMENTE"); //Mensaje mostrado en el monitor serial
} else {

    Serial.print("ERROR AL ENVIAR EL MENSAJE: "); //Mensaje mostrado si hubo algun error al enviar el correo
    Serial.println(gsender->getError());          //Indica posibles errores dentro de la libreria Gsender
}

```

Programa adicional para la conexión con la simulación

El módulo *Wi-Fi* no posee librerías que permitan simularlo virtualmente. De modo que, para solventar este problema se ha incluido en la simulación un *Arduino Nano*, el cual envía, por el puerto serial, un dato que será leído por el dispositivo electrónico, así cumple la acción correspondiente. La sintaxis es la siguiente:

```

int entrada = A1;
int led = A0;
int dat = 0;

void setup() {
    Serial.begin(9600);
    pinMode(entrada, INPUT);
    pinMode(led, OUTPUT);
}

void loop() {
    dat=digitalRead(entrada);

    if(dat==1){
        Serial.print(5);
        digitalWrite(led,HIGH);
        delay(5500);
    }
    else {
        digitalWrite(led,LOW);
    }
}

```

Imágenes presentadas en pantalla

En los programas *Adobe Illustrator*, *Adobe After Effects* y *Adobe Photoshop* se elaboraron, tanto la animación como la imagen del estado de interrupción. Se trabajó en

coordinación con el centro terapéutico Voces para elegir el mejor diseño, el cual es de fácil comprensión para los niños con TEA.

La animación programada en la pantalla no es más que la sucesión rápida de varias imágenes que fueron agregadas al programa cargado en la pantalla *Nextion*.

Mediante el uso de *Adobe Illustrator* se crearon las imágenes individuales para la animación en la pantalla.

En la Figura 3.65 se observa la mesa de trabajo que presenta este programa, la cual permite la elaboración de imágenes, en las que se puede elegir el color de cada elemento, forma, tamaño, entre otros.

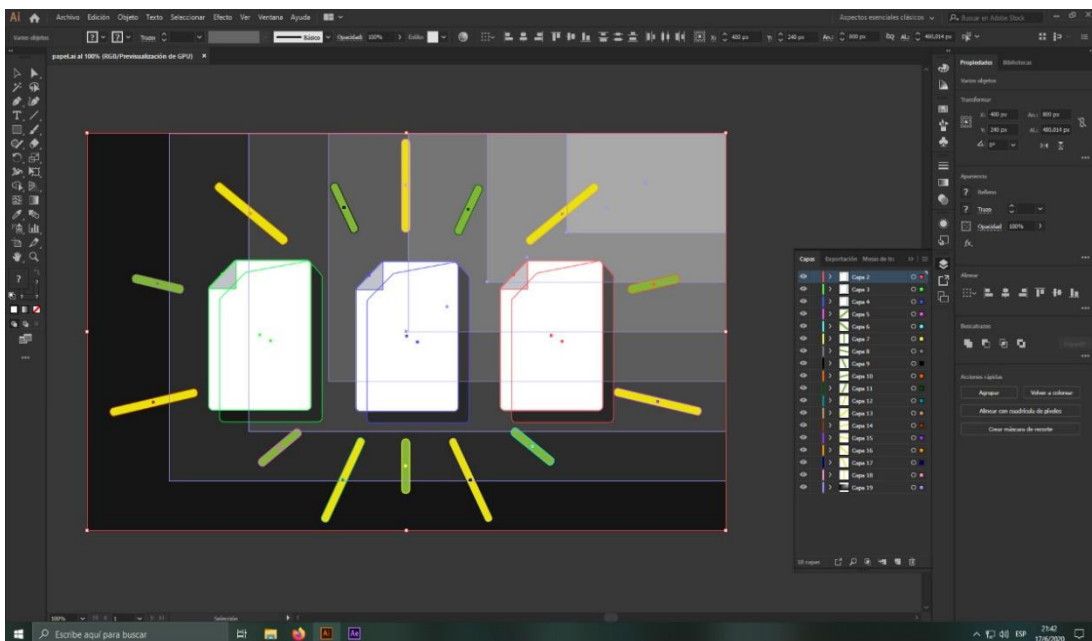


Figura 3.65: Mesa de trabajo de *Adobe Illustrator*

Se crearon 10 imágenes para la animación, que, al pasar por una transición rápida, simulan una animación. Mediante el *software Adobe After Effects*, se pudo comprobar, si las imágenes creadas anteriormente generaban dicho efecto.

En la Figura 3.66 se observa el editor de *Adobe After Effects*, el cual es muy semejante a un editor de video. Se debe añadir, en dicho editor, una por una las imágenes que se deseen observar. Además, se programa un tiempo óptimo de transición de imágenes que, para esta aplicación, es 100 milisegundos.

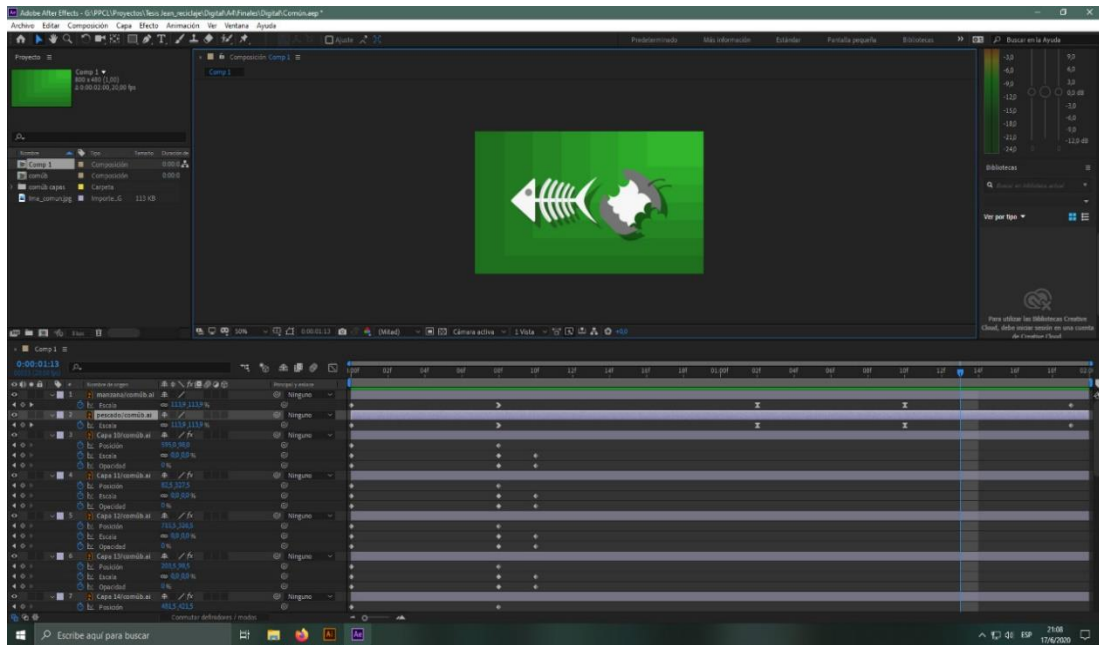


Figura 3.66: Editor de *Adobe After Effects*

La imagen del estado de interrupción es estática, así que no es necesario añadir otras para generar la animación. El diseño de la misma se puede apreciar en la Figura 3.67



Figura 3.67: Imagen del estado de interrupción

Se utilizó el programa *Adobe Photoshop* para realizar retoques a las imágenes. Este programa presenta la mesa de trabajo mostrada en la Figura 3.68, en la cual se pueden realizar cambios de resolución, tamaño del archivo, entre otros aspectos.



Figura 3.68: Mesa de trabajo de *Adobe Photoshop*

3.5 Pruebas de funcionamiento

Las pruebas realizadas para comprobar el correcto funcionamiento de la simulación están detalladas a continuación:

Prueba de animación en la pantalla *Nextion*

El editor de *Nextion* cuenta con una herramienta llamada *debug*, la cual permite simular las interfaces creadas antes de grabarlas en la memoria de la pantalla. En la Figura 3.69 se observa la localización del elemento *debug* en la barra de herramientas en la parte superior.

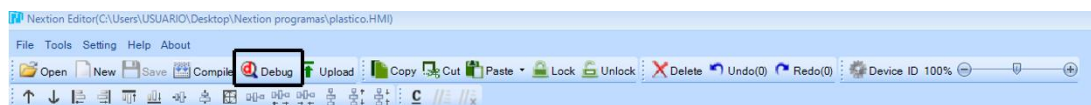


Figura 3.69: Ubicación de la herramienta *debug* en *Nextion Editor*

Una vez se haya iniciado la simulación, aparece una ventana como se puede apreciar en la Figura 3.70. Se debe elegir la opción *User MCU input* para comprobar el cambio de pantallas, lo cual permite comunicación serial con otro *software* externo.

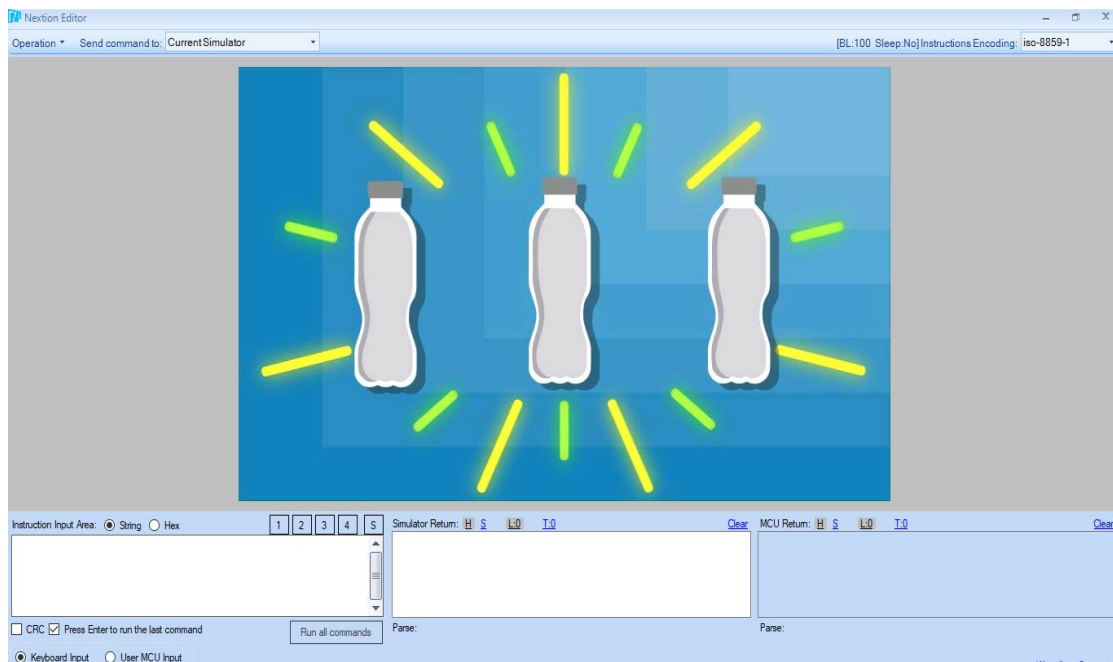


Figura 3.70: Ventana *debug* de *Nextion Editor*

Estados del sistema automatizado

La simulación del sistema automatizado se realizó en el *software Proteus*. Sin embargo, el programa no cuenta con librerías para poder simular la pantalla *Nextion* ni el módulo *Wi-Fi ESP8266 NodeMCU V3*.

Se utilizó un elemento llamado *COMPIN* para solventar el inconveniente antes mencionado, debido a que esta herramienta permite comunicación serial de manera virtual con *software* y *hardware* externo a *Proteus*. Además, fue necesario utilizar un programa adicional llamado *Configure Virtual Serial Port Driver*, cuya función es virtualizar los puertos físicos de la computadora y conectarlos a través de un cable virtual, permitiendo la transferencia de información entre los puertos pares.

La Figura 3.71 muestra el *software* y los puertos que han sido virtualizados para llevar a cabo una completa simulación de todos los elementos.

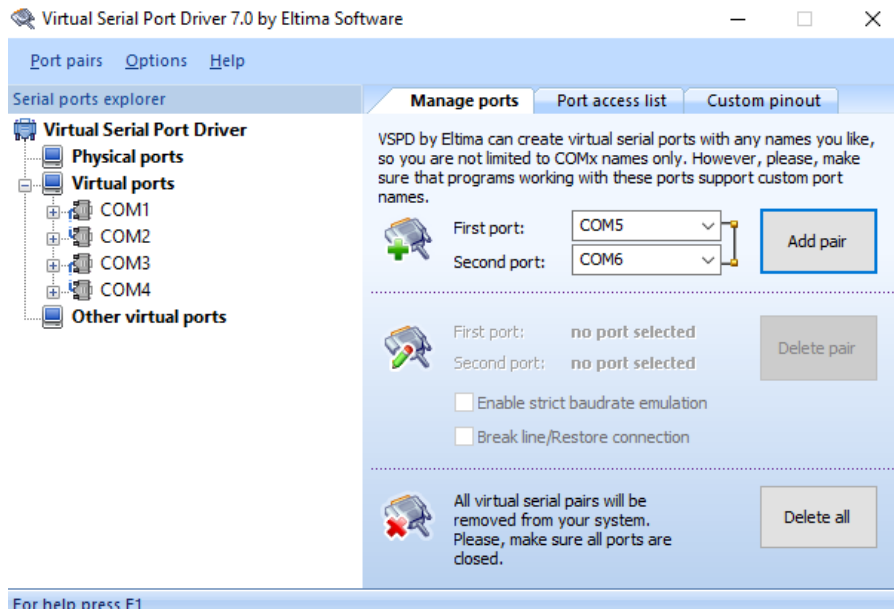


Figura 3.71: Virtualización de puertos

La Figura 3.72 indica que los puertos 1 y 2, 3 y 4 son pares; es decir, están conectados virtualmente entre ellos y es posible la transmisión de datos.

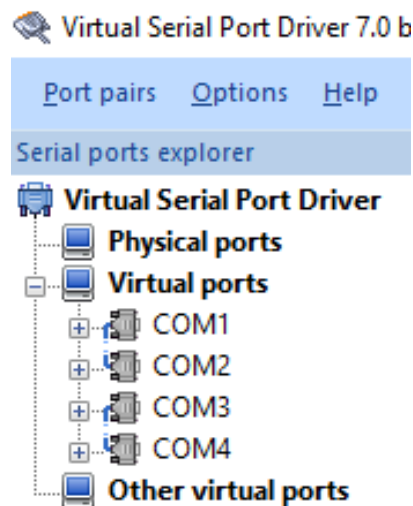


Figura 3.72: Puertos virtuales pares

Virtualizar los puertos fue imprescindible para enlazar el *software* de simulación de la pantalla *Nextion* con *Proteus*, permitiendo comprobar el cambio de pantalla en tiempo real. También, permite verificar el funcionamiento del módulo *Wi-Fi*, cuando éste actúe en conjunto con el programa principal.

Se realizó la comprobación de cada etapa de la siguiente manera:

- **Activación**

En la Tabla 3.3 se muestra la relación entre porcentaje-distancia; ya que, el sensor ultrasónico utilizado usa un potenciómetro para variar entre estos valores en el *software Proteus*. Este procedimiento es necesario para comprobar las etapas del sistema automatizado.

Tabla 3.3: Relación porcentaje-distancia del ultrasónico en la simulación

PORCENTAJE	DISTANCIA
0%	4 cm
1%	10 cm
2%	26 cm
3%	33 cm
4%	44 cm
5%	55 cm
6%	66 cm
7%	83 cm
8%	89 cm
9%	104 cm

La figura 3.73 indica que el sensor ultrasónico detecta un usuario dentro del perímetro de activación. Además, muestra la animación en pantalla de lo que puede ser depositado en el basurero.

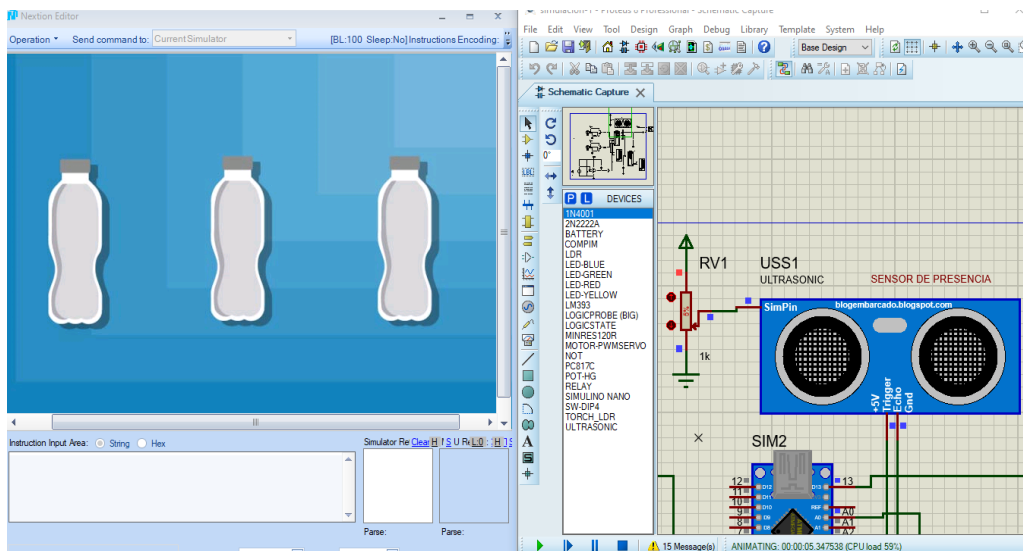


Figura 3.73: Sensor ultrasónico detecta un usuario en el perímetro

A continuación, se puede apreciar cómo los servomotores cambian del estado inicial cerrado, al de apertura; como se muestran en las Figuras 3.74 y 3.75.

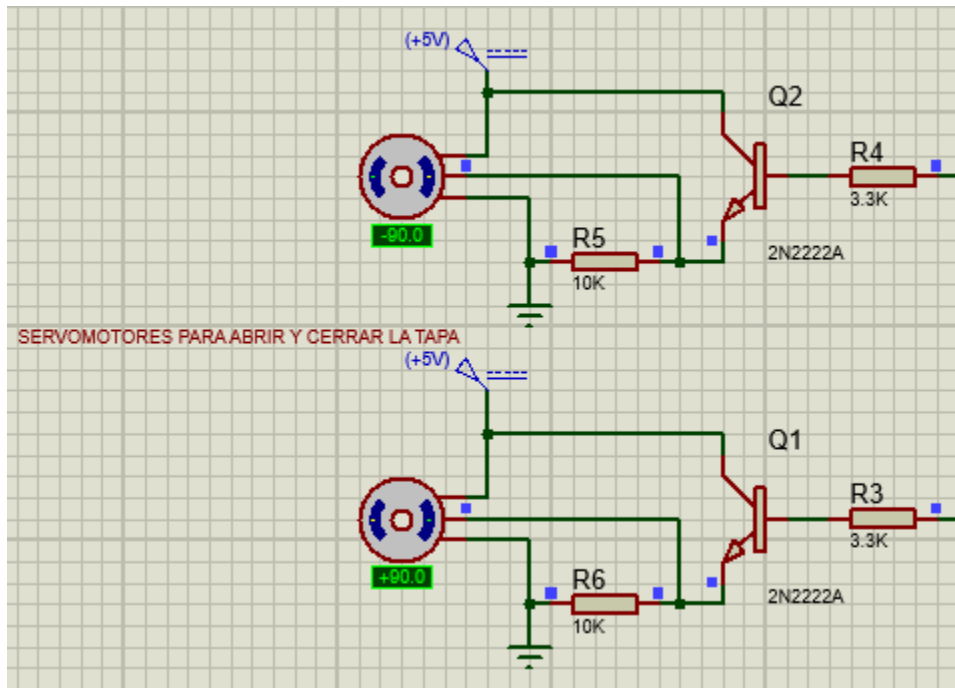


Figura 3.74: Motores cerrando la tapa

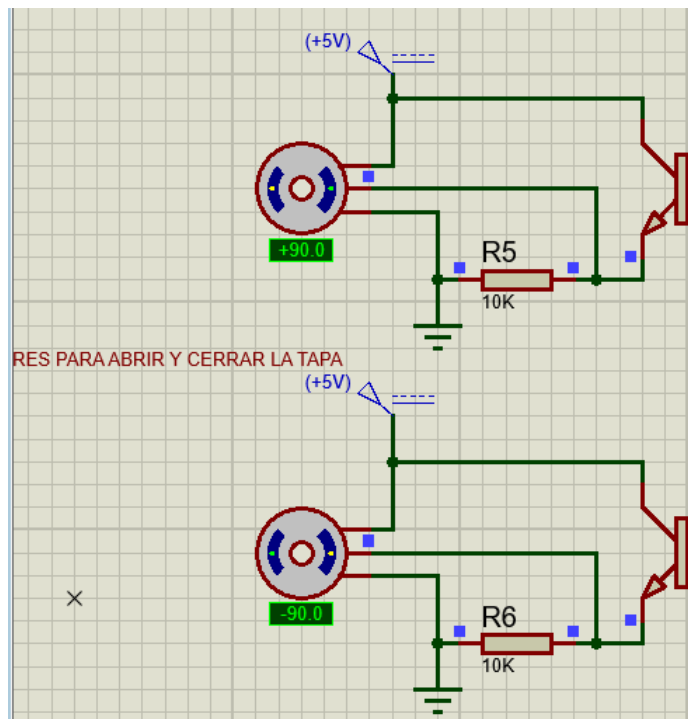


Figura 3.75: Motores abriendo la tapa

El sensor de nivel indica que el sistema automatizado se encuentra bajo el límite de su capacidad máxima; por tanto, el LED indicador no se enciende, y permite el normal funcionamiento de la etapa de activación, como se muestra en la Figura 3.76.

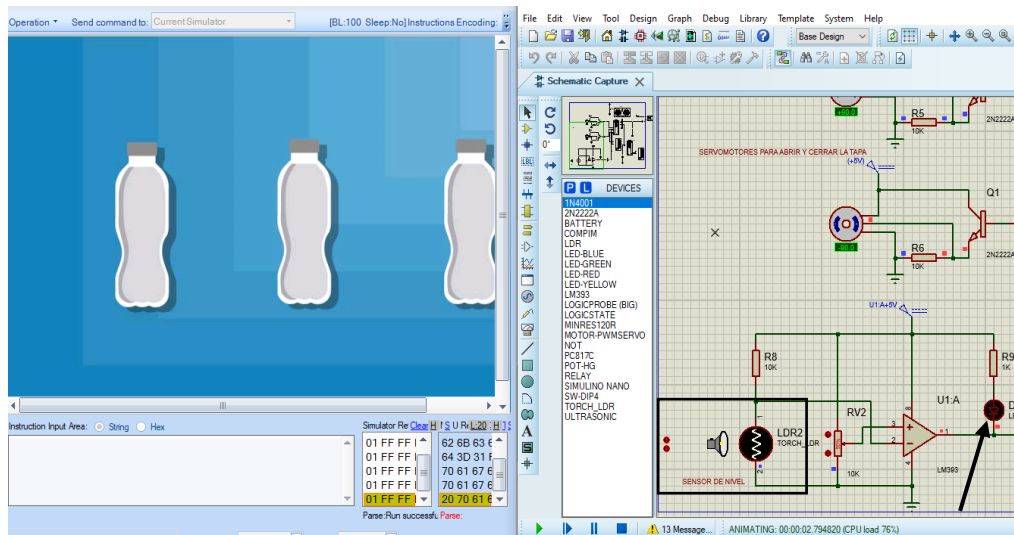


Figura 3.76: Sensor de nivel detecta el haz de luz

En las Figuras 3.77 y 3.78 se observa el elemento COMPIM, el cual permite la conexión entre el *software* de *Nextion*, el programa *Proteus* y el módulo *Wi-Fi*. La comunicación con la pantalla siempre está activa, porque debe presentar la información de la etapa en la que se encuentre el basurero. Durante este estado, el módulo *Wi-Fi* permanece inactivo.

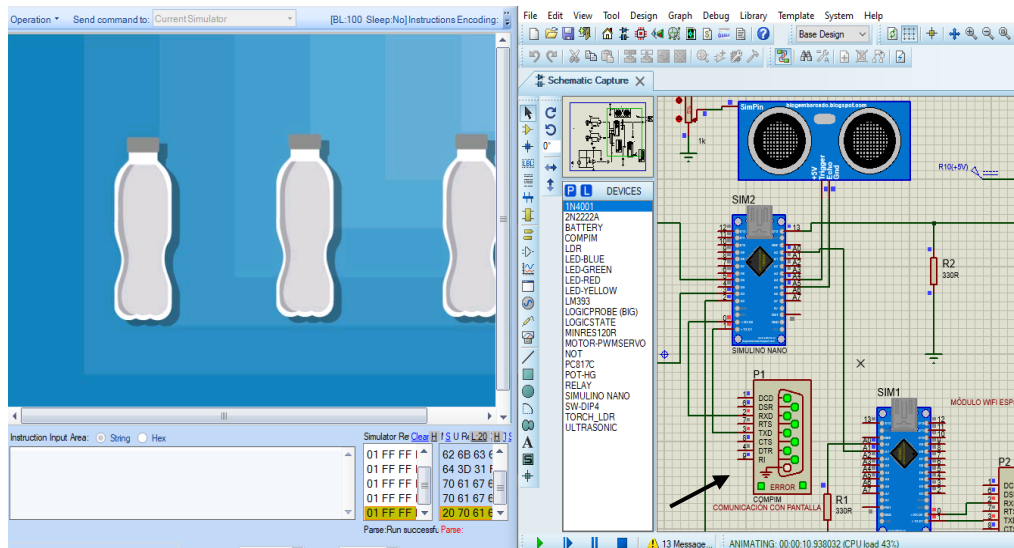


Figura 3.77: COMPIM para la comunicación con la pantalla

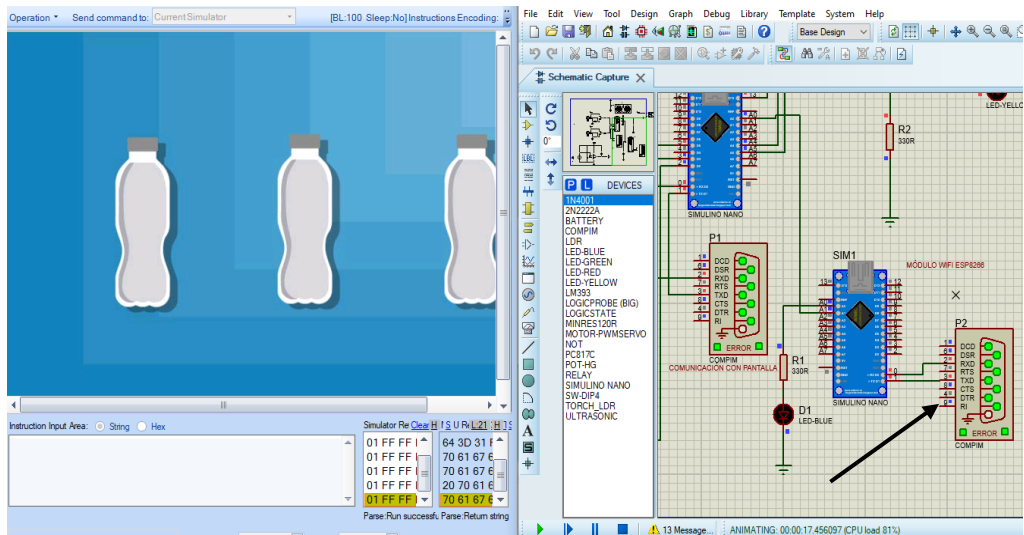


Figura 3.78: COMPIM para la comunicación con el módulo Wi-Fi

Se puede observar en la Figura 3.79 la luz verde, que indica que aún se cuenta con tiempo para realizar la acción correspondiente.

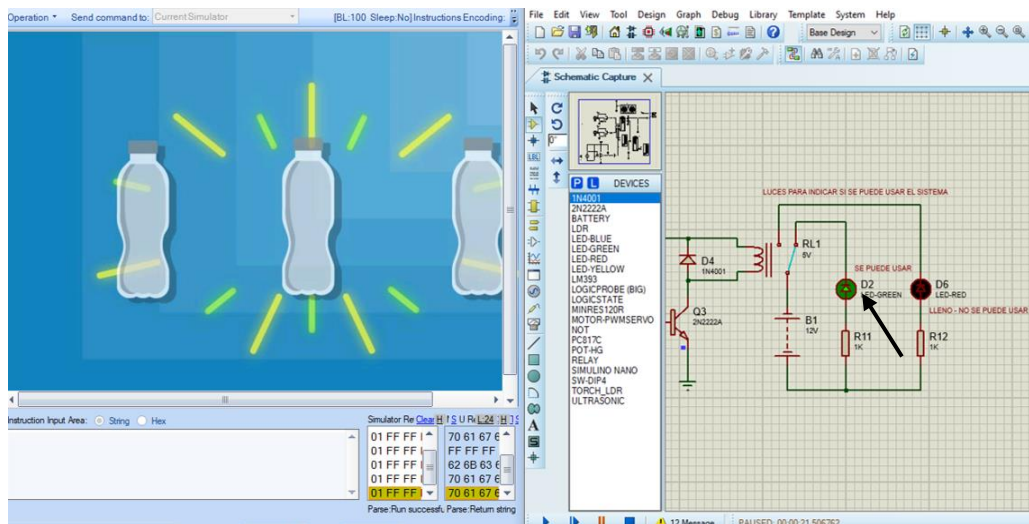


Figura 3.79: Luz verde en la etapa de activación

Una vez que el tiempo concluya, la luz verde se apaga; entonces, la luz roja se enciende y los servomotores vuelven a su estado inicial, como se muestra en la Figura 3.80; esta acción indica que se agotó el tiempo.

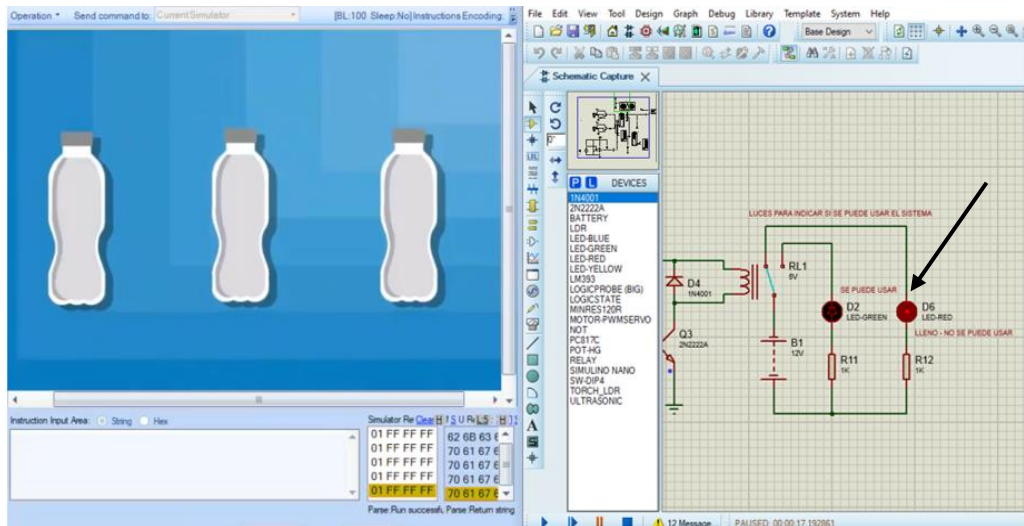


Figura 3.80: Luz roja en la etapa de activación

- **Interrupción**

En la Figura 3.81 se observa el sensor de nivel que ha detectado el basurero al límite de su capacidad. El monitor serial y el LED de la Figura 3.82, informan al usuario que el mensaje ha sido enviado.

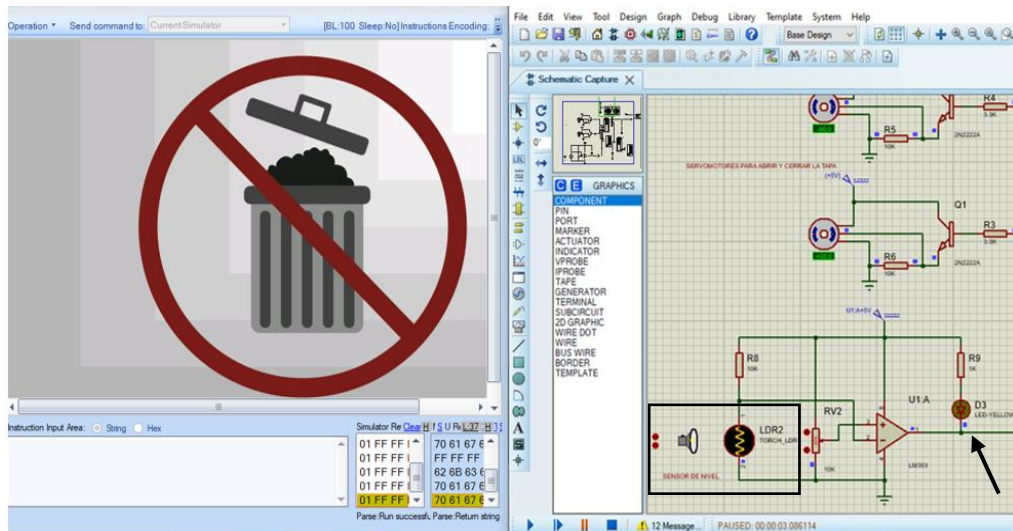


Figura 3.81: Sensor de nivel detecta que no hay haz de luz

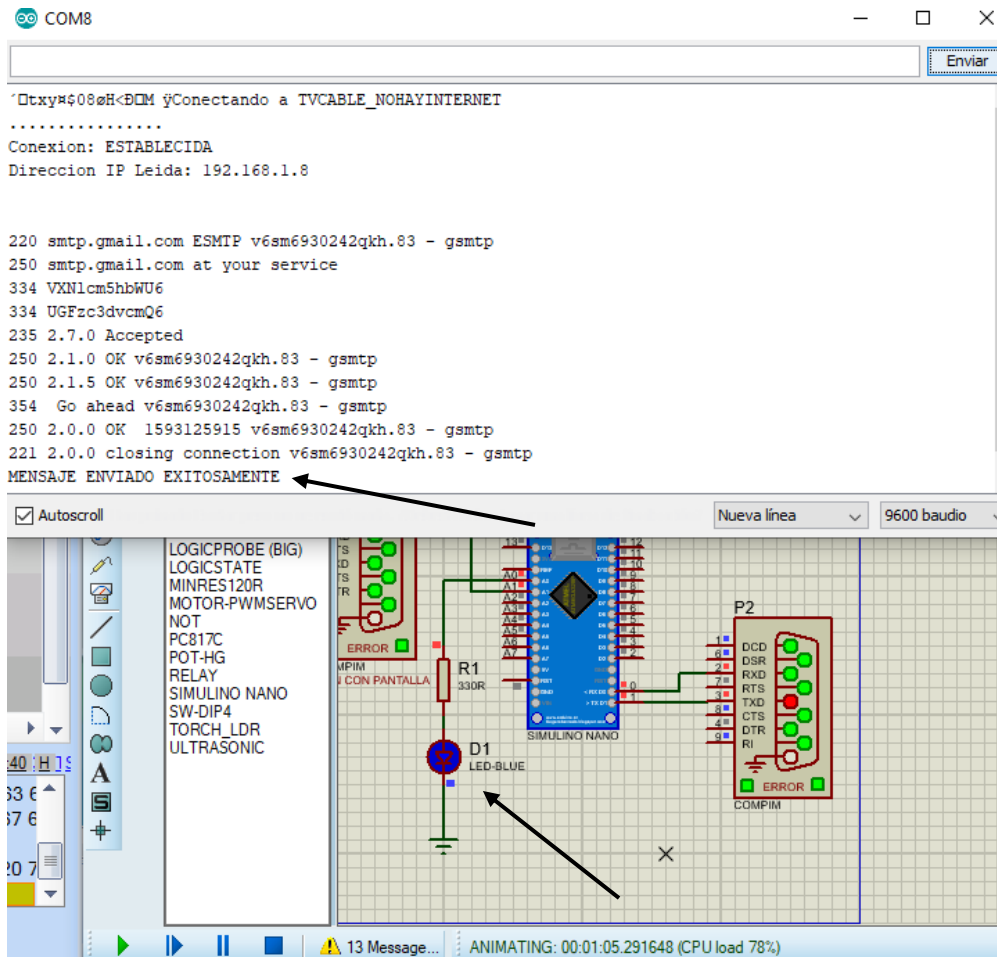


Figura 3.82: Mensaje del monitor serial al enviar el correo

La luz roja informa al usuario que no puede utilizar el sistema automatizado hasta que el administrador vacíe el contenedor, como se muestra en la Figura 3.83.

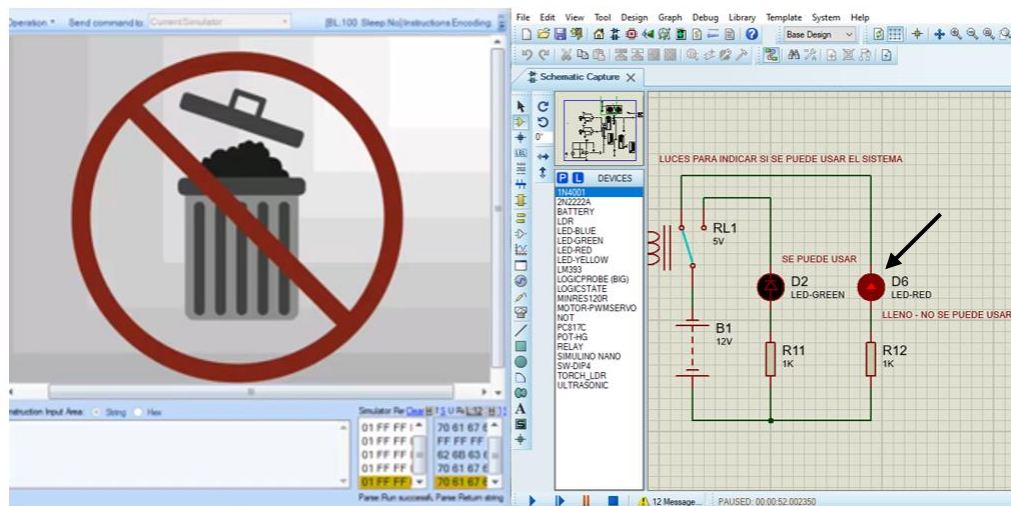


Figura 3.83: Luz roja en la etapa de interrupción

El administrador recibe un correo electrónico como el que se indica a continuación en la Figura 3.84.

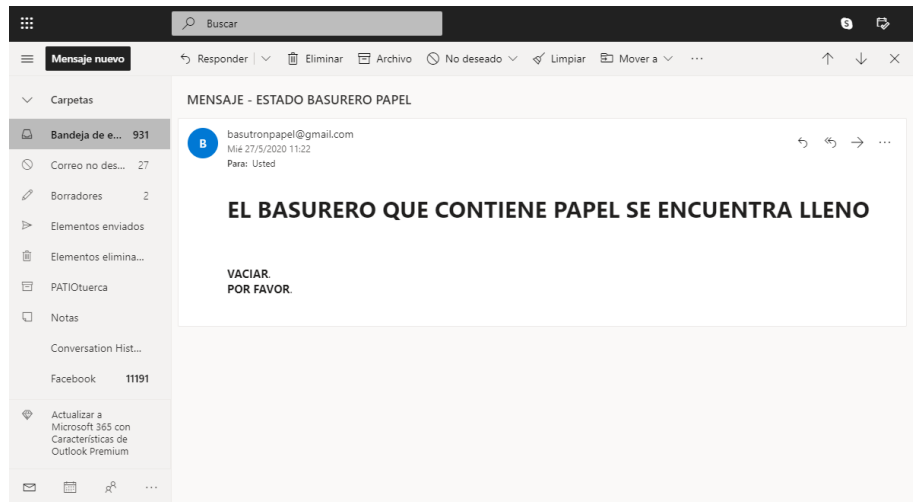


Figura 3.84: Correo que recibe el administrador

- **Reposo**

Cuando el usuario se encuentre fuera del perímetro de activación, la pantalla exhibe un fondo negro como se observa en la Figura 3.85, esta función indica que está en mínimo consumo de energía.

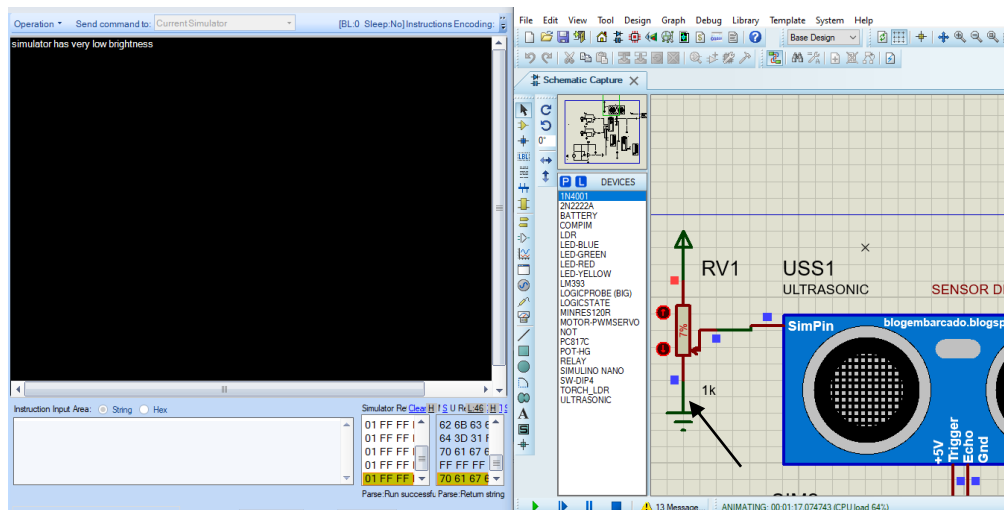


Figura 3.85: Estado de reposo

La luz verde indica que solo la persona que se encuentre dentro del perímetro de activación, puede utilizar el sistema automatizado, como se observa en la Figura 3.86.

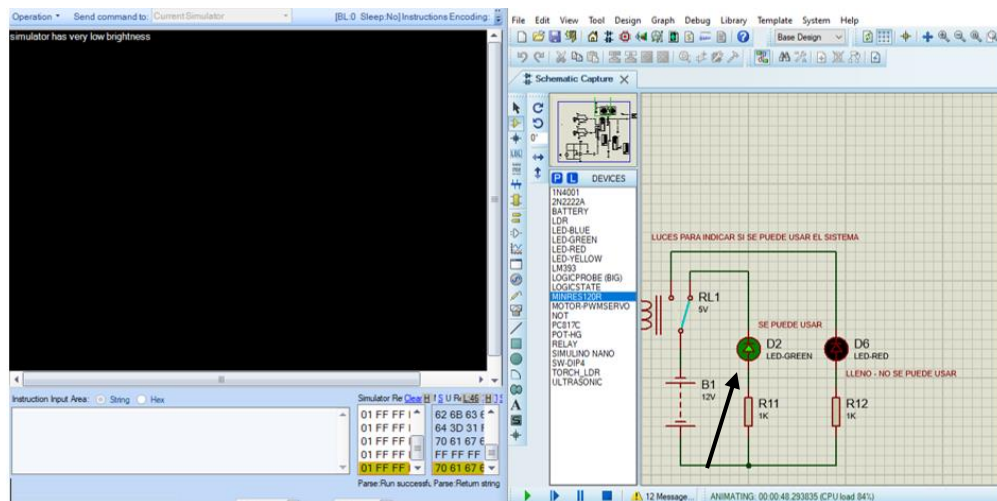


Figura 3.86: Luz verde en el estado de reposo

3.6 Costos de implementación

Los costos de implementación son estimaciones y pueden variar, los mismos fueron consultados en el sitio *web* de compra y venta “Mercado Libre”; en el presente presupuesto presentado en la Tabla 3.4, no se considera rubros por mano de obra ni por desarrollo intelectual.

Tabla 3.4: Costos de implementación

DETALLE	PRECIO	CANTIDAD	TOTAL
Adaptador 12 V 1A	\$ 5,00	1	\$ 5,00
Adaptador 5V 1A	\$ 2,00	6	\$ 12,00
Baquelitas	\$ 3,50	4	\$ 14,00
Caja estanca	\$ 5,00	3	\$ 15,00
ESP8266 <i>NodeMCU</i> V3	\$ 8,00	3	\$ 24,00
Estructura	\$ 350,00	1	\$ 350,00
Luces piloto LED	\$ 2,35	6	\$ 14,10
Módulo <i>Arduino</i> Nano	\$ 6,50	3	\$ 19,20
Módulo de 1 Relé	\$ 2,00	3	\$ 6,00
Módulo KY-008	\$ 2,00	3	\$ 6,00
Módulo sensor LDR	\$ 2,00	3	\$ 6,00
Módulo ultrasónico HC-SR04	\$ 2,00	3	\$ 6,00
Pantalla <i>Nextion</i> 7.0”	\$ 100,00	3	\$ 300,00
Servomotor SG90	\$ 2,50	6	\$ 15,00
Tacho reciclaje	\$ 17,49	3	\$ 52,47
Dispositivos y artículos varios	\$ 30,00	-	\$ 30,00
TOTAL			\$ 874,77

4 CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- A través del presente proyecto, las herramientas tecnológicas motivan la atención y el desempeño autónomo de las personas con TEA en cuanto al reciclaje y la conservación del medio ambiente.
- El uso de material concreto, como recomienda el método TEACCH ayuda a enseñar hábitos a niños con TEA para que desarrollen destrezas comunicativas; el sistema automatizado simulado a través del presente proyecto es, precisamente, un recurso tangible que permitirá alcanzar ese objetivo.
- Los pictogramas son imágenes que se utilizan en muchos centros que brindan terapias a niños con TEA. Estos recursos facilitan el contacto entre el dispositivo automatizado y el niño mediante una comunicación bidireccional.
- La asignación de un tiempo determinado para el uso del dispositivo es un requerimiento imprescindible; porque los niños con TEA solo prestan atención o concentración por espacios mínimos de tiempo. Por tanto, se establecieron 5 segundos para depositar basura en el contenedor, de acuerdo con los requerimientos de las terapeutas del centro Voces.
- En el diseño del sistema se utilizaron varios programas, tanto de simulación como de comunicación entre interfaces *software – software* y *hardware – software*. Además, se observa que la velocidad de simulación se encuentra limitada por la cantidad de recursos disponibles en el computador.
- Las versiones básicas de las pantallas *Nextion* permiten el uso de limitadas herramientas que corresponden a su software de programación; tal como la herramienta *timer*, usada para generar una animación mediante transición de imágenes a una velocidad imperceptible al ojo humano.
- *Arduino (software y hardware)* es ampliamente utilizado, debido a que existe una gran cantidad de información sobre su correcto uso y óptima programación; además, permite la compatibilidad entre varios periféricos y módulos *Arduino* para elaborar proyectos.

- La versión *NodeMCUV3* del módulo *Wi-Fi ESP8266* facilita la conexión a internet y permite usar la opción de acceso remoto a correos *Gmail*; ya que, es un dispositivo creado para usarse en aplicaciones de IoT (*Internet of Things*) y gracias a las librerías que posee para realizar el proceso de conexión remota, no es necesario crear un servidor de correo electrónico para enviar un *e-mail* desde dicho módulo.
- El modelo de 7 pulgadas de la pantalla *Nextion* brinda una resolución y calidad de imagen que se puede apreciar con claridad en interiores y exteriores de un ambiente. Esta característica le facilita al programador del sistema automatizado determinar la intensidad luminosa con la que se proyecta el contenido de la pantalla.
- El *Arduino Nano* es el módulo que más se ajusta a los requerimientos del sistema automatizado, debido a que, es más compacto que el *Arduino Uno* y comparte las mismas características técnicas.
- Mediante el uso de *software* de simulación, se obtienen vistas preliminares del funcionamiento de los estados del sistema; esta característica permite comprobar el funcionamiento de los algoritmos, y la depuración de estos en los casos que lo ameriten.
- Los programas de dibujo asistido por computadora facilitan la elaboración y la visualización de la estructura del basurero automatizado; además permiten realizar una selección del tipo de material que se necesita para fabricar la estructura del dispositivo con un beneficio costo-calidad.
- El *datasheet* proporciona información sobre voltaje y corriente que soporta cada elemento, entre otras características, que conforma el basurero. Sobre la base de esa información se realizaron cálculos que determinaron los elementos de protección pertinentes para evitar posibles daños por picos de voltaje o corriente en el circuito.
- El desarrollo de manuales facilita a los usuarios y administradores un correcto uso del basurero automatizado; además, ese instructivo permite al administrador modificar el sistema según los requerimientos que exigen las terapias.

- El sistema automatizado presentado no tiene la intención de sustituir ninguna terapia impartida por especialistas que trabajan con niños que padecen TEA; en cambio, es un complemento tecnológico desarrollado para trabajar en conjunto con dichos especialistas.

4.2 Recomendaciones

- Se debe considerar las características técnicas del ordenador antes de utilizar *softwares* específicos para el diseño del sistema automatizado. Para la simulación eficiente de este proyecto se recomienda un sistema operativo de 64 bits con un procesador Intel Core i5-3450 y 8GB de memoria RAM, además de 2GB de espacio libre en el disco duro o disco de estado sólido
- Los pictogramas son recursos que se deben utilizar en las terapias y en el empleo de sistemas como el dispositivo diseñado; porque se requiere motivar a los niños con TEA para que se logren objetivos en su proceso de formación personal y en su relación con su entorno familiar y social.
- Las dimensiones del basurero, así como las imágenes y tiempos de uso en cada etapa deben ser adaptados a los requerimientos del lugar donde vaya a ser utilizado.
- Se debe considerar el tipo de población que se beneficiará con la aplicación del proyecto; puesto que, la ejecución del mismo se desarrolla en base de la simulación del sistema automatizado para trabajar con niños de 3 a 6 años en el Centro Terapéutico Voces.
- Para determinar el rango de detección del sistema automatizado se debe considerar el área en la cual se van a ubicar los contenedores, considerando obstáculos u otros agentes que pudieran dificultar su correcto funcionamiento.
- Es necesario que los sistemas automatizados cuenten con un manual de funcionamiento, sus instrucciones deben ser de fácil comprensión para los usuarios y deben informar sobre el uso adecuado de los dispositivos.

- Para la implementación física con módulos, se debe considerar el consumo energético de cada uno, mediante una tabla de consumo energético se puede analizar qué tipo de fuente es recomendable utilizar.
- Si se desea presentar más información en la pantalla o un video, se recomienda utilizar las series “mejoradas” o “inteligentes” de las pantallas *Nextion*, las cuales poseen mayor memoria, presentan un *hardware* mejorado, entre otros beneficios. Para ello se debe considerar una relación costo-beneficio.
- La implementación física de la estructura se recomienda hacerla en plástico polietileno de alta densidad, debido a su resistencia y ligereza. Otra opción puede ser de láminas de lata, tomando en cuenta las medidas de protección para evitar descargas eléctricas.
- El sistema automatizado podría implementar mejoras entre la interacción del usuario con TEA y una retroalimentación positiva por parte del basurero automatizado, como utilizar voces pre-grabadas, efectos de sonido o animaciones; como parte de un aprendizaje lúdico.

5 REFERENCIAS BIBLIOGRÁFICAS

- [1] M. Vaca, Interviewee, *Métodos de enseñanza a niños con TEA*. [Entrevista]. 09 01 2020.
- [2] G. Herrera , C. Xavier, S. Javier , R. Luis , P. Carlos y P. Javier , «Research Gate,» Enero 2012. [En línea]. Available: https://www.researchgate.net/publication/266078731_Pictogram_Room_Natural_Interaction_Technologies_to_Aid_in_the_Development_of_Children_with_Autism#read. [Último acceso: 25 Agosto 2020].
- [3] O. C. Acosta, «Acceda Cris,» Mayo 2017. [En línea]. Available: https://accedacris.ulpgc.es/bitstream/10553/54026/2/0750462_00000_0000.pdf. [Último acceso: 25 Agosto 2020].
- [4] C. López y Larrea, «Autismo en Ecuador: un Grupo Social en espera de atención,» *Revista Ecuatoriana de Neurología*, vol. 26, nº 3, pp. 1-12, 2017.
- [5] Arduino, «What is Arduino?,» [En línea]. Available: <https://www.arduino.cc/en/Guide/Introduction>. [Último acceso: 24 Junio 2020].
- [6] Arduino, «Arduino,» [En línea]. Available: <https://www.arduino.cc/en/main/software>. [Último acceso: 08 Junio 2020].
- [7] F. Martínez, «Tutorial Arduino: IDE Arduino,» Open Webinars , 03 Febrero 2015. [En línea]. Available: <https://openwebinars.net/blog/tutorial-arduino-ide-arduino/>. [Último acceso: 08 Junio 2020].
- [8] ITEAD STUDIO , «Nextion Tech,» ITEAD STUDIO, 2011. [En línea]. Available: https://nextion.tech/nextion-editor/#_section1. [Último acceso: 19 06 2020].
- [9] Eltima IBC, «Virtual Serial Port,» Eltima, 2020. [En línea]. Available: <https://www.virtual-serial-port.org/es/>. [Último acceso: 20 06 2020].
- [10] Eltima IBC, «Capturas de pantalla de Virtual Serial Port Driver,» Eltima, 2020. [En línea]. Available: <https://www.virtual-serial-port.org/es/screenshots.html>. [Último acceso: 28 Junio 2020].

- [11] Labcenter Electronics , «Proteus,» [En línea]. Available: <https://www.labcenter.com/>. [Último acceso: 06 Junio 2020].
- [12] E. G. Breijo, Compilador C CCS y simulador Proteus para microcontroladores PIC, Barcelona: Marcombo Ediciones Técnicas, 2009.
- [13] J. Flores, «codigofacilito,» 25 08 2015. [En línea]. Available: <https://codigofacilito.com/articulos/que-es-html>. [Último acceso: 02 07 2020].
- [14] G. Gavrilova, «mailjet,» 20 03 2019. [En línea]. Available: <https://es.mailjet.com/blog/news/que-puerto-smtp-mailjet/>. [Último acceso: 02 07 2020].
- [15] Adobe, «Adobe Illustrator,» 2020. [En línea]. Available: https://www.adobe.com/la/products/illustrator.html?gclid=CjwKCAjw57b3BRBIEiwA1lmytgMqF263CfeQGgWBm1lr8nvu9DLWwDtiB5an9tUe4iacAwTHXMNIMRoCzEYQAvD_BwE&sdid=KQPQJ&mv=search&ef_id=CjwKCAjw57b3BRBIEiwA1lmytgMqF263CfeQGgWBm1lr8nvu9DLWwDtiB5an9tUe4iacAwTHXMNIMRo. [Último acceso: 20 06 2020].
- [16] Adobe, «Adobe After Effects,» 2020. [En línea]. Available: https://www.adobe.com/la/products/aftereffects.html?gclid=CjwKCAjw57b3BRBIEiwA1lmytlGzcDy1cltznqAjlqwoHYrTAFKZwUzYOWfhf1MFkRRe0fRJ57_YcRoCfO4QAvD_BwE&sdid=KQPRU&mv=search&ef_id=CjwKCAjw57b3BRBIEiwA1lmytlGzcDy1cltznqAjlqwoHYrTAFKZwUzYOWfhf1MFkRRe0fRJ57_YcR. [Último acceso: 20 06 2020].
- [17] Adobe, «Adobe Photoshop,» 2020. [En línea]. Available: <https://www.adobe.com/es/products/photoshop/free-trial-download.html#>. [Último acceso: 20 06 2020].
- [18] Arduino, «Arduino Nano,» [En línea]. Available: <https://store.arduino.cc/usa/arduino-nano>. [Último acceso: 08 Junio 2020].
- [19] Patagonia Tec, «Arduino Nano,» Diciembre 2014. [En línea]. Available: <https://saber.patagoniatec.com/2014/12/arduino-nano-328-arduino-atmega-clon-compatible-arduino-argentina-ptec/>. [Último acceso: 08 Junio 2020].

- [20] L. d. V. Hernandez, «NodeMCU tutorial paso a paso desde cero,» Programar Facil, [En línea]. Available: <https://programarfacil.com/podcast/nodemcu-tutorial-paso-a-paso/>. [Último acceso: 09 Junio 2020].
- [21] G. Martín, «Cómo programar NodeMCU con el IDE de Arduino,» programar facil , [En línea]. Available: <https://programarfacil.com/esp8266/como-programar-nodemcu-ide-arduino/>. [Último acceso: 09 Junio 2020].
- [22] J. Rodrigo, «ESP8266 y NodeMCU: la nueva generación de sistemas embebidos,» Panama Hitek , 25 Octubre 2016. [En línea]. Available: <http://panamahitek.com/esp8266-y-nodemcu-la-nueva-generacion/>. [Último acceso: 09 Junio 2020].
- [23] Nextio Tech, «Nextion NX8048T070,» [En línea]. Available: <https://nextion.tech/datasheets/nx8048t070/>. [Último acceso: 09 Junio 2020].
- [24] Maker Electronico, «Relay 1 CH 5vdc/220vac optoacoplado,» [En línea]. Available: <https://www.makerelectronico.com/producto/relay-1-canal-5vdc-220vac-con-optoacoplador/>. [Último acceso: 10 Junio 2020].
- [25] A. G. González, «¿Qué es y cómo funciona un servomotor?,» Panama Hitek , 02 Diciembre 2016. [En línea]. Available: <http://panamahitek.com/que-es-y-como-funciona-un-servomotor/>. [Último acceso: 10 Junio 2020].
- [26] Components 101, «Servomotor SG90,» 18 Septiembre 2017. [En línea]. Available: <https://components101.com/servo-motor-basics-pinout-datasheet>. [Último acceso: 10 Junio 2020].
- [27] Elecbreaks, «Ultrasonic Ranging Module HC - SR04,» Electro Schematics , [En línea]. Available: <https://www.electroschematics.com/wp-content/uploads/2013/07/HCSR04-datasheet-version-1.pdf>. [Último acceso: 06 Junio 10].
- [28] L. Llamas, «luisllamas,» 16 06 2015. [En línea]. Available: <https://www.luisllamas.es/medir-distancia-con-arduino-y-sensor-de-ultrasonidos-hc-sr04/>. [Último acceso: 21 06 2020].
- [29] Geek Factory, «Módulo Sensor Fotorresistencia LDR,» [En línea]. Available: <https://www.geekfactory.mx/tienda/sensores/modulo-sensor-fotorresistencia->

línea]. Available: <https://www.pecs-spain.com/el-sistema-de-comunicacion-por-el-intercambio-de-imagenes-pecs/>. [Último acceso: 21 06 2020].

- [39] C. S. A. Marco, «EL MÉTODO TEACCH,» quadernsdigitals, [En línea]. Available: http://www.quadernsdigitals.net/datos_web/hemeroteca/r_1/nr_796/a_10730/10730.html. [Último acceso: 21 06 2020].

ANEXOS

ANEXO A: MANUAL DE USUARIO

El sistema consta de dos luces piloto, una tapa que se abre verticalmente hacia arriba y una pantalla de 7", como se observa en la Figura A1.

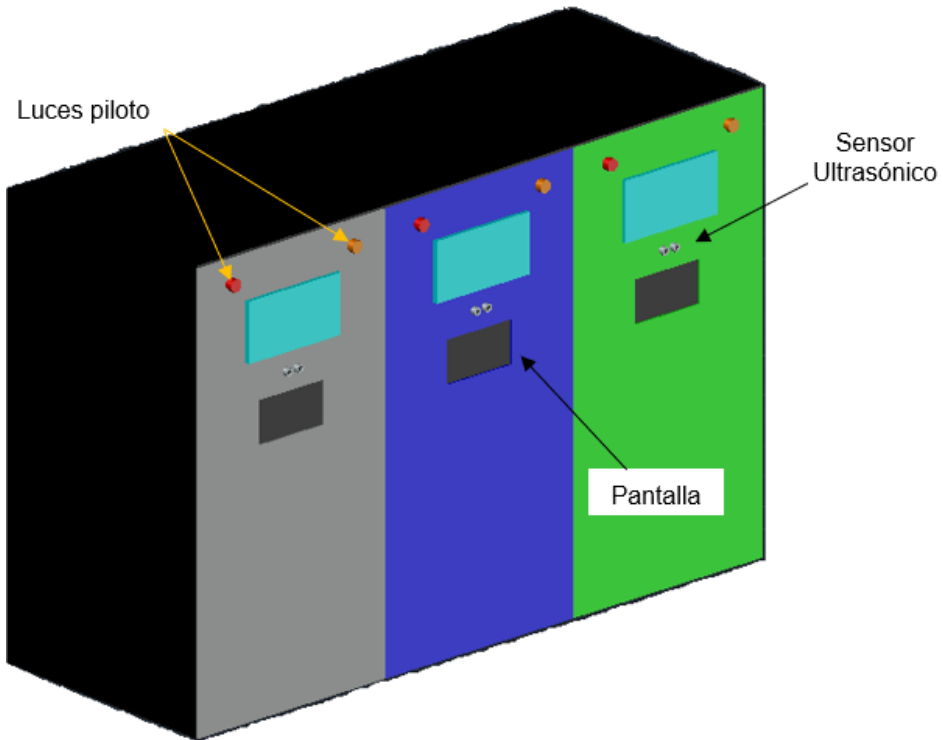


Figura A1: Ubicación de los elementos del sistema

Mientras la luz verde se encuentre encendida y la pantalla se encuentre en negro, como se muestra en la Figura A2, el sistema está en estado de reposo. Para activarlo, se debe estar a una distancia no mayor de 60 cm.

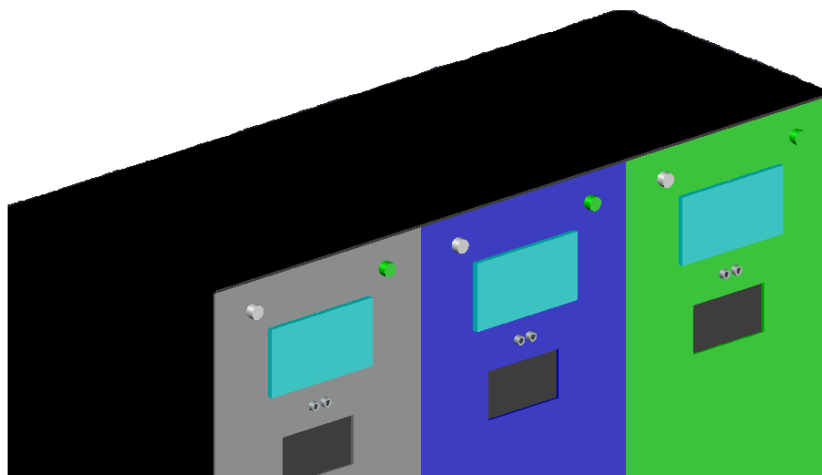


Figura A2: Sistema en reposo

En la etapa de activación se mantendrá la tapa abierta, encendida la luz verde y se muestra en la pantalla el tipo de basura que puede depositarse (Figura A3); cuando la tapa esté cerca a cerrarse, la luz cambia a rojo y poco tiempo después ésta se cierra (Figura A4). Luego de ello, la luz roja permanece encendida y no se podrá utilizar el sistema durante 5 segundos (Figura A5); pasado ese tiempo, se vuelve a encender la luz verde, la pantalla se pone en negro y el sistema está listo para ser utilizado.

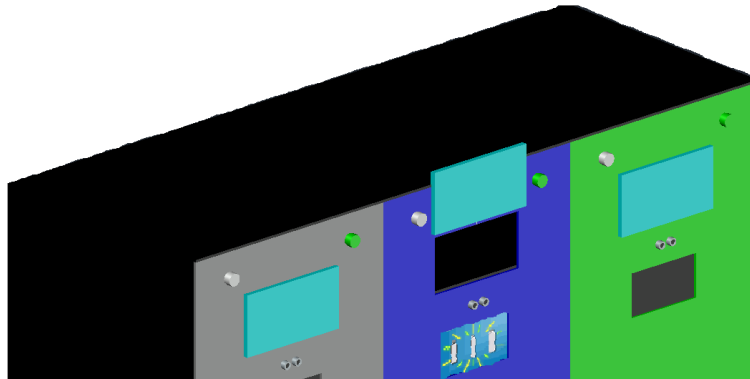


Figura A3: Inicio de la etapa de activación

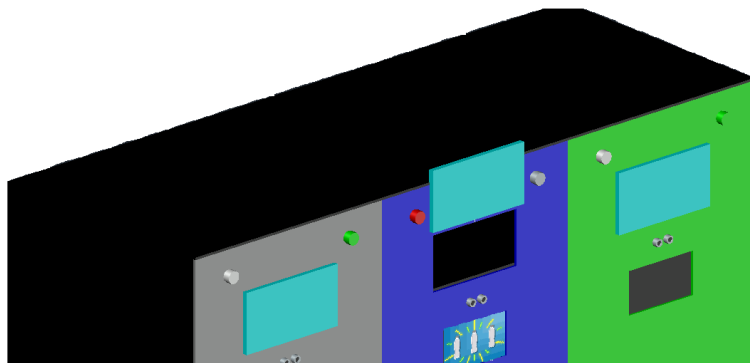


Figura A4: Intermedio de la etapa de activación

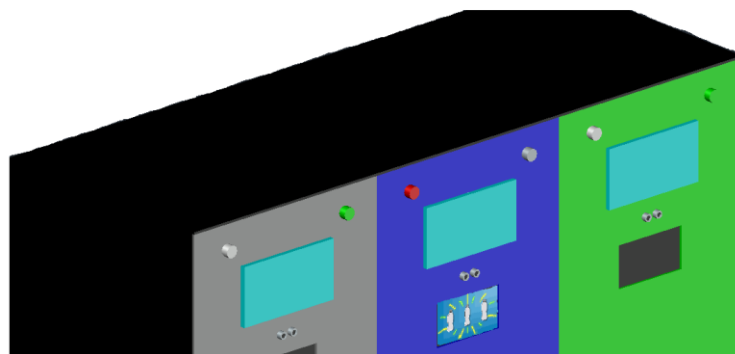


Figura A5: Fin de la etapa de activación

Cuando el contenedor interno alcanza su máxima capacidad, se enciende la luz roja y se muestra una pantalla de bloqueo indicando que no se puede depositar más basura, tal como se muestra en la Figura A6. En este estado, incluso si el usuario se encontrara dentro del perímetro de activación, la tapa no se abrirá hasta que el contenedor interno sea vaciado. El administrador recibe un correo indicando que debe vaciar el contenedor interno.

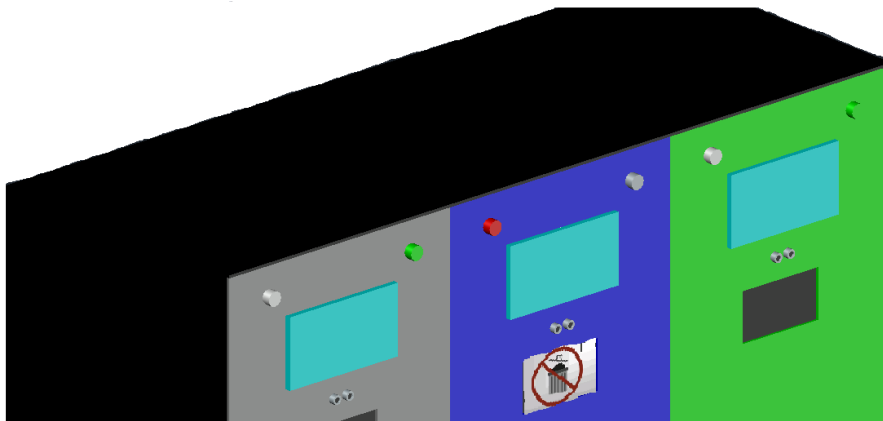


Figura A6: Etapa de bloqueo

Para vaciar el contenedor interno, se debe abrir la tapa posterior de la estructura, tal como se muestra en la Figura A7, y retirar el contenedor. Una vez vaciado el contenedor interno y vuelto a colocar en su lugar, el sistema vuelve al estado de reposo.

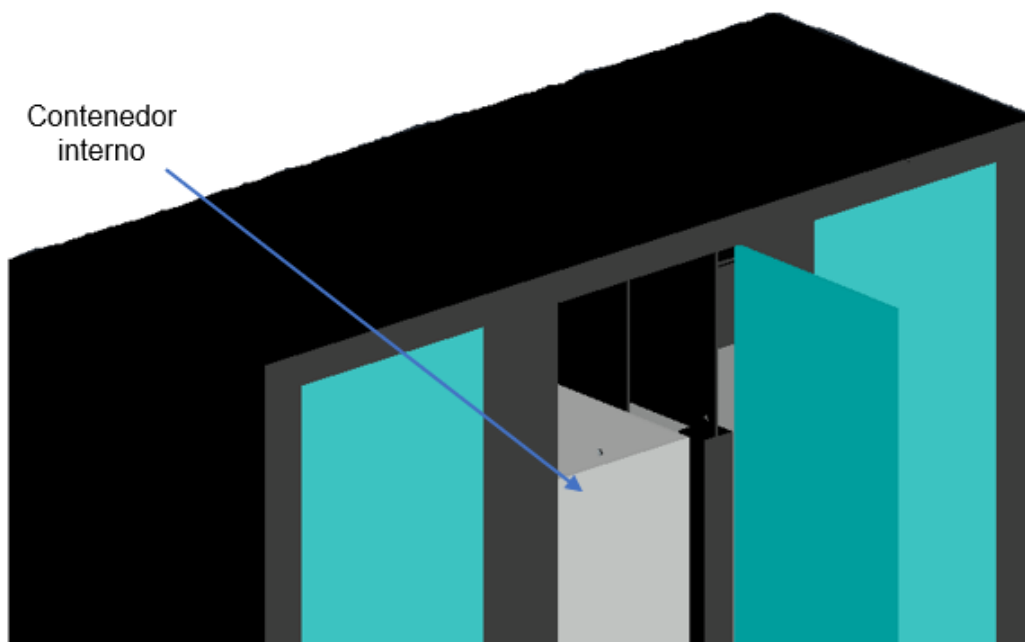


Figura A7: Vaciado del contenedor interno

ANEXO B: MANUAL DE ADMINISTRADOR

Se puede realizar cambios en la programación para solventar futuros requerimientos, tales como: cambiar el tiempo de duración del estado de activación, las imágenes que se muestran en pantalla y el mensaje enviado al administrador. También, se puede cambiar el remitente y el destinatario.

Cambio de imágenes en pantalla

Se debe descargar el editor de *Nextion* para poder cargar un nuevo proyecto a la pantalla. El programa se lo encuentra en el siguiente enlace: <https://nextion.tech/nextion-editor/>.

La Figura B1 indica la manera de descargar el programa una vez se ha abierto el enlace de descarga.

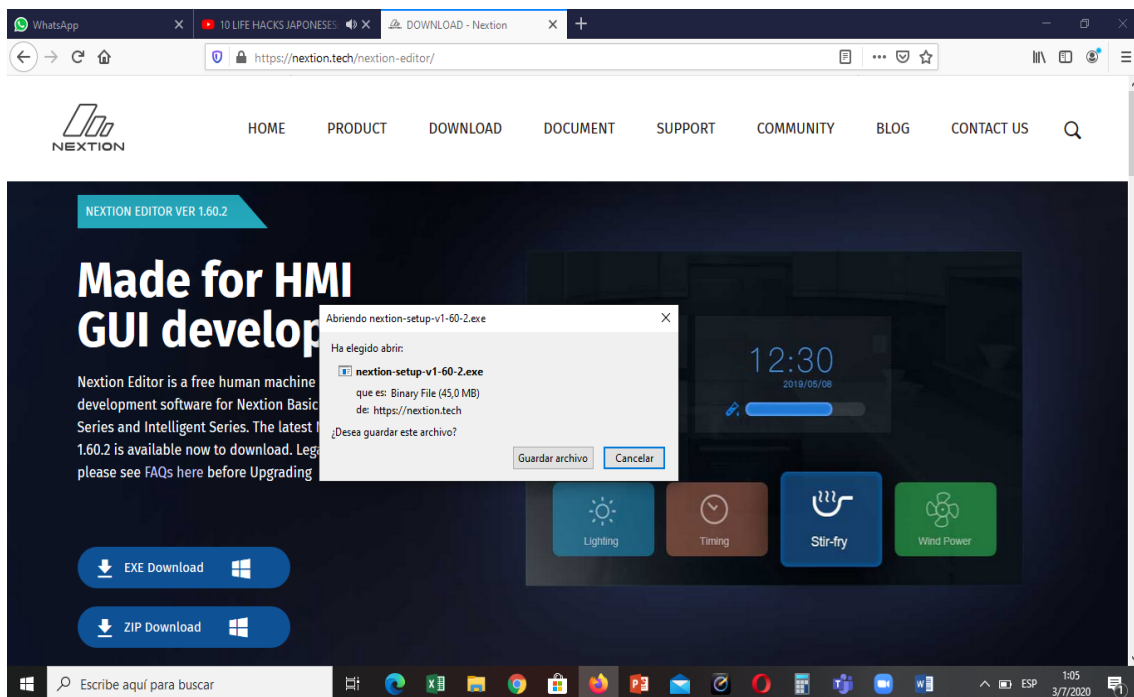


Figura B1: Descarga del editor de *Nextion*

Cuando termine la descarga, se debe ejecutar como administrador el instalador del programa, como se indica en la Figura B2.

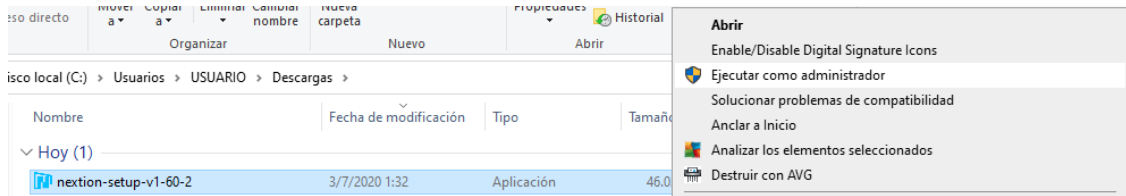


Figura B2: Instalación del editor de *Nextion*

El proceso de instalación es parecido al de cualquier otro programa, como se indica en las Figuras B3 a B7, se debe dar clic en siguiente en cada ventana que aparece a medida que avanza el proceso. Una vez finalizada la instalación, aparecerá la ventana principal del editor de *Nextion*.

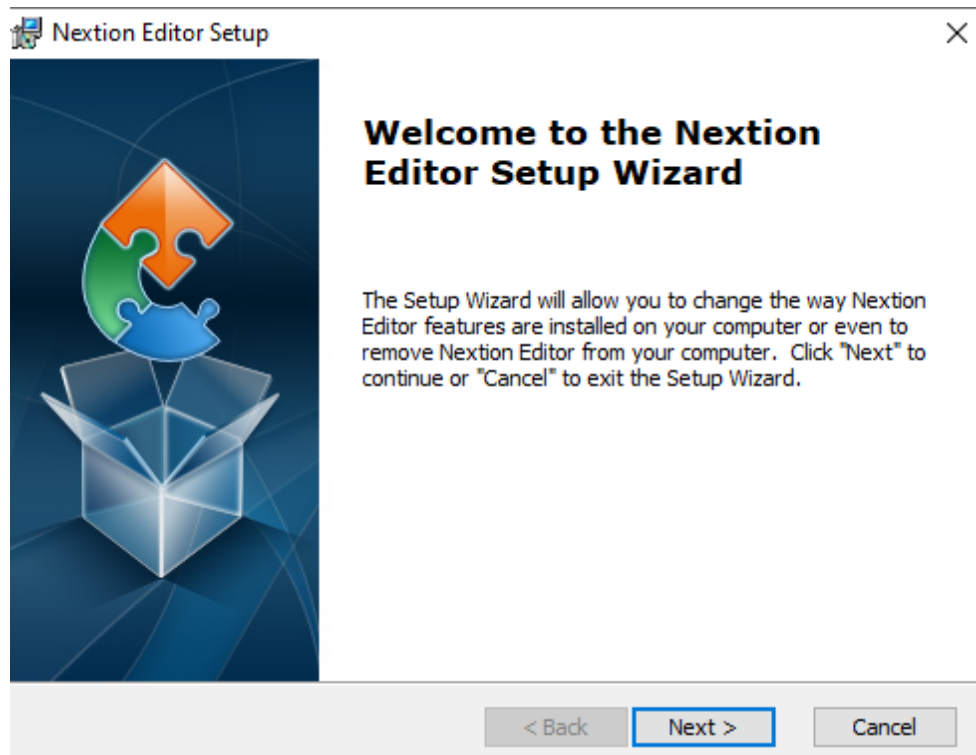


Figura B3: *Setup Wizard de Nextion Editor*

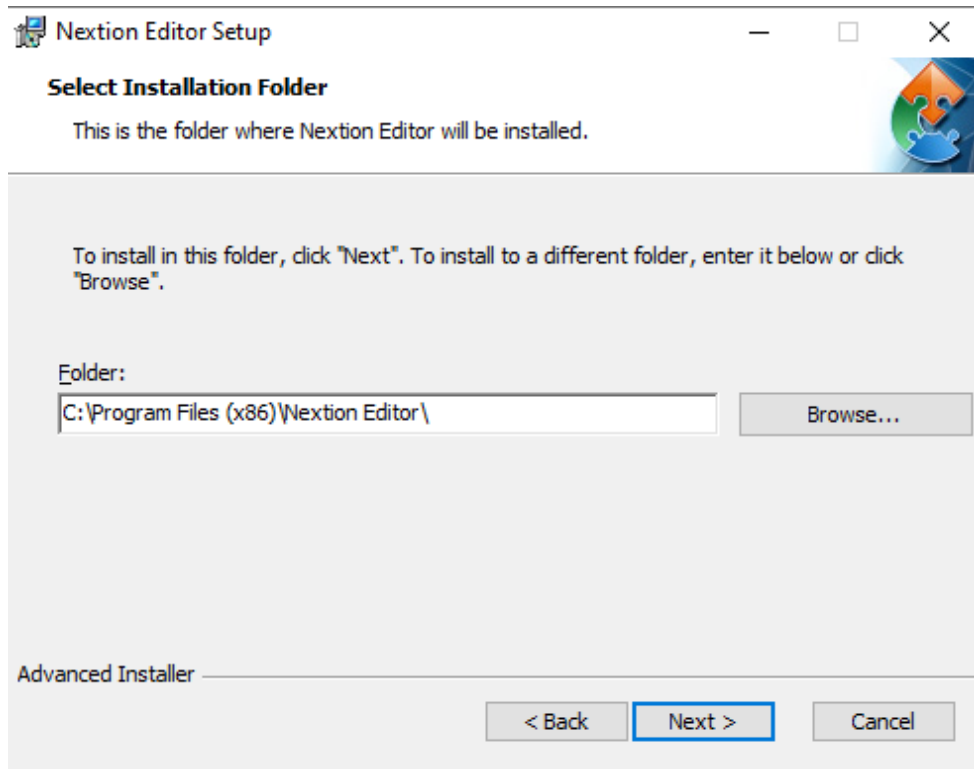


Figura B4: Selección de ubicación para *Nextion Editor*

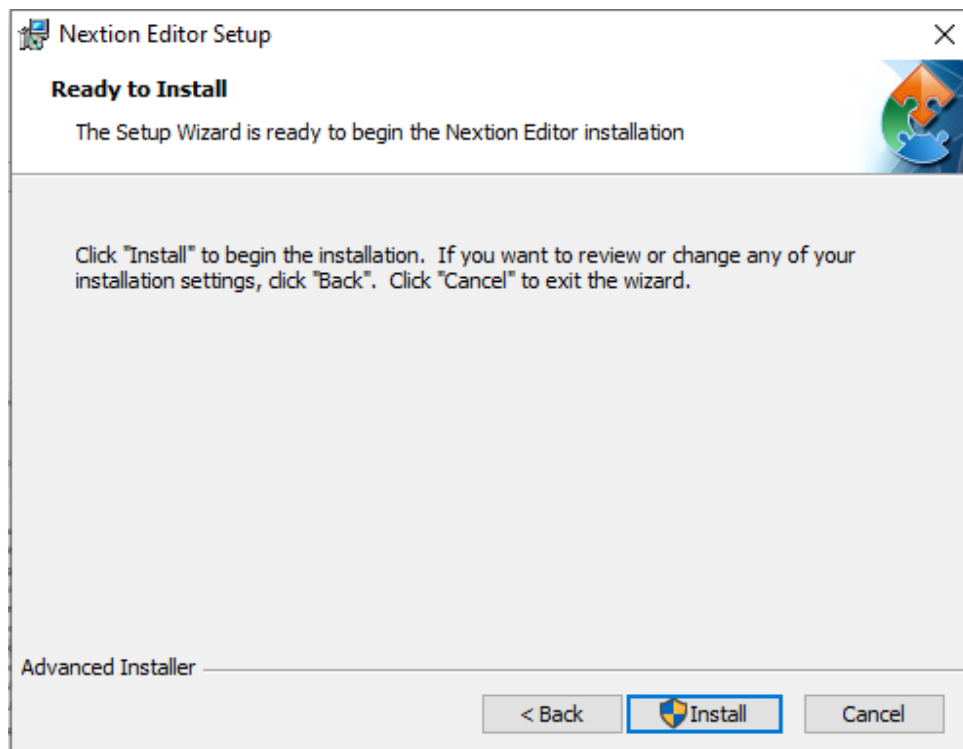


Figura B5: Inicio de instalación de *Nextion Editor*

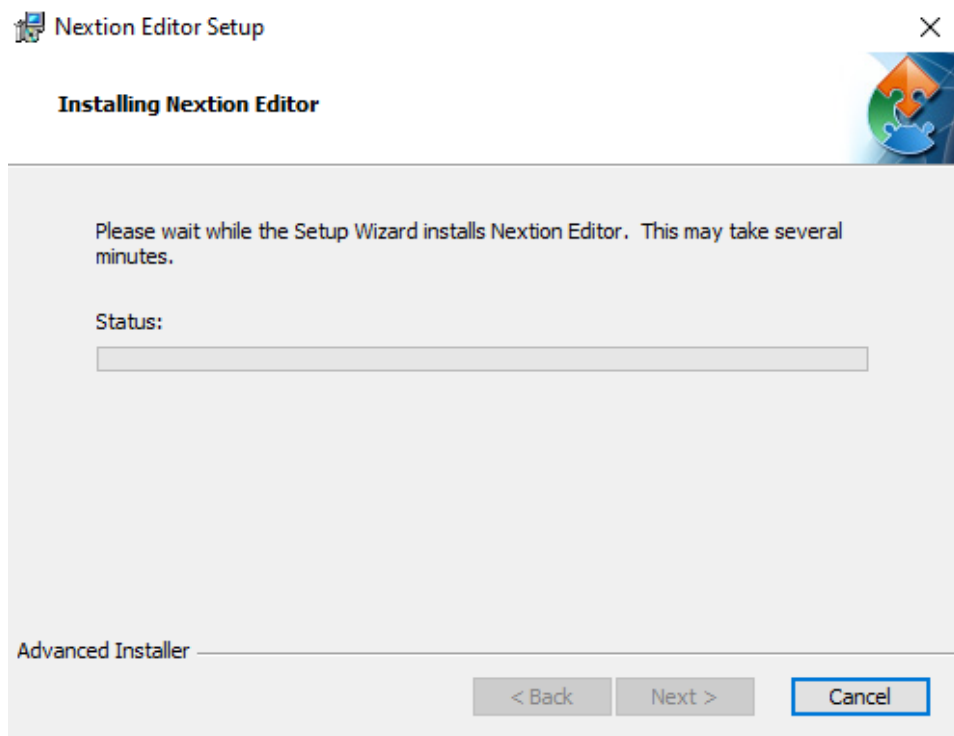


Figura B6: Instalación de *Nextion Editor* en proceso



Figura B7: Instalación de *Nextion Editor* culminada

Cuando se vaya a crear un nuevo proyecto en el editor de *Nextion*, se debe dar nombre al archivo y guardarlo en un lugar seguro donde pueda ser encontrado con facilidad. Las Figuras B8 y B9 muestran el procedimiento a seguir.

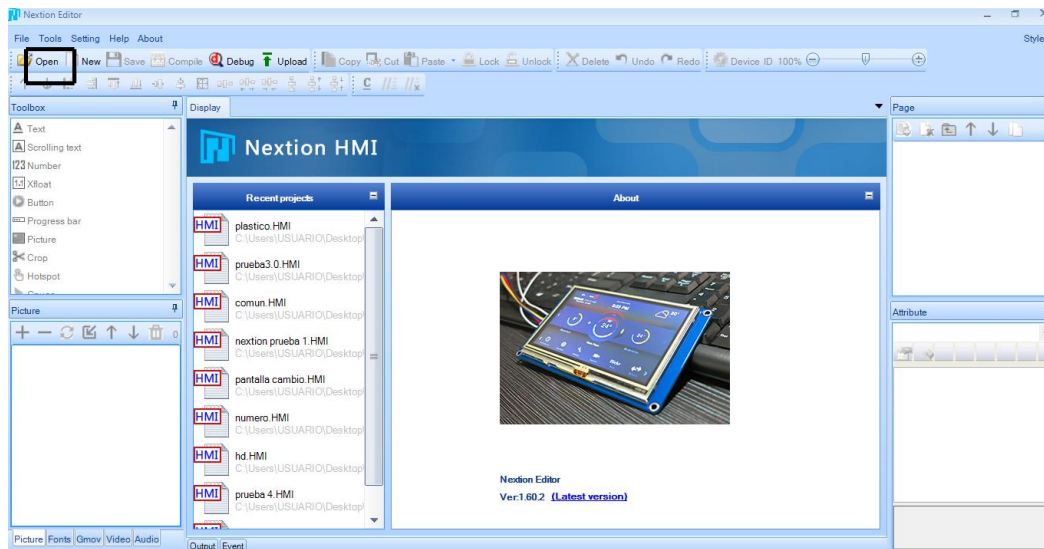


Figura B8: Creación de nuevo proyecto en *Nextion Editor*

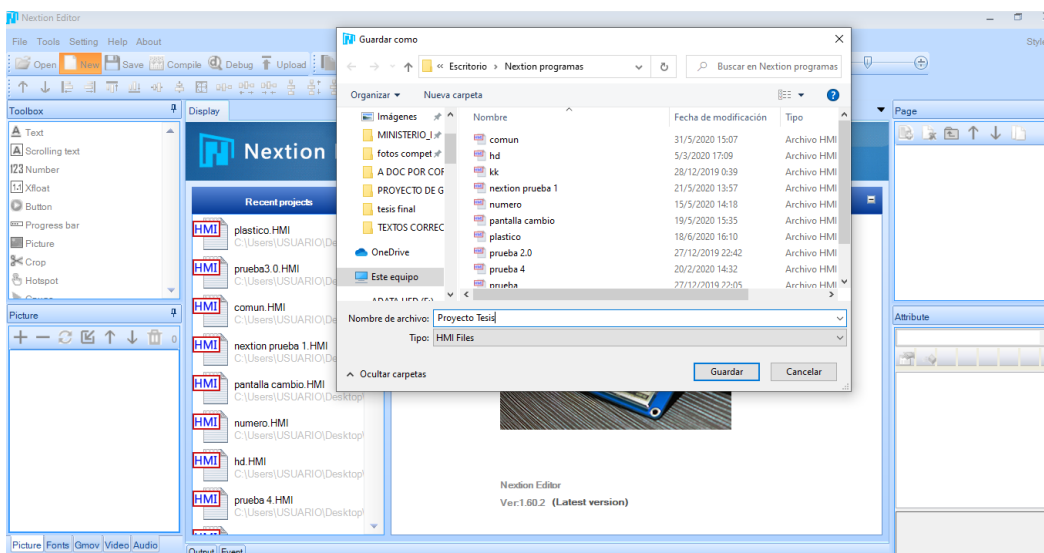


Figura B9: Guardado de proyecto en *Nextion Editor*

Quando se haya creado el archivo, aparece una ventana como la que se muestra en la Figura B10. Inmediatamente, se debe seleccionar el tipo de pantalla, su resolución y la orientación de la hoja donde se trabaja.

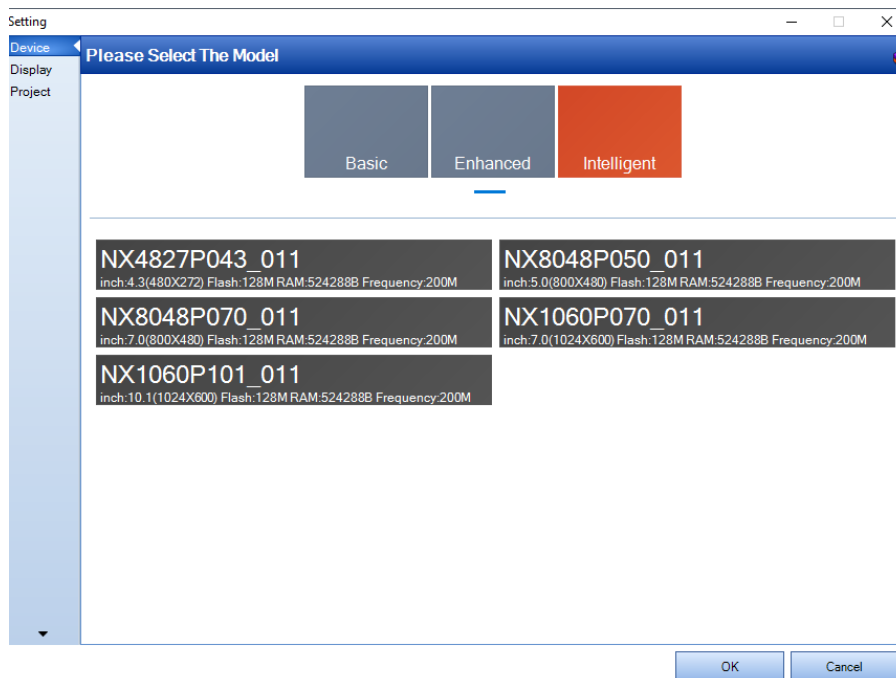


Figura B10: Selección del tipo de pantalla en *Nextion Editor*

El sitio *web* oficial de *Nextion*, posee una lista detallada con los diferentes tipos de pantalla que se ofrecen en el mercado. Para el diseño del sistema automatizado se ha elegido una versión básica de 7 pulgadas. En las Figuras B11 y B12 se puede apreciar la configuración que se utiliza para el desarrollo del proyecto.

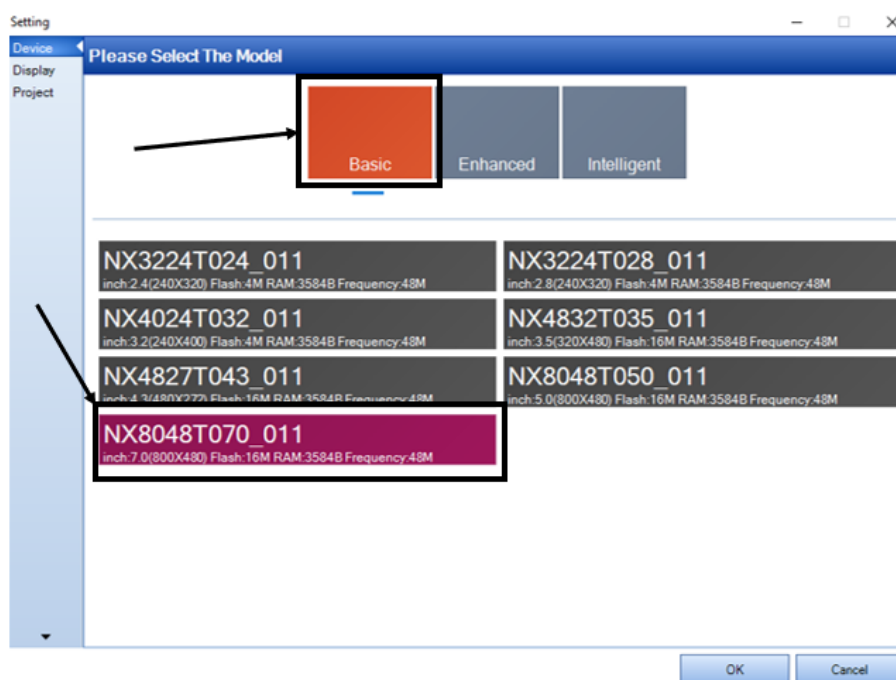


Figura 11: Selección de la pantalla de 7"

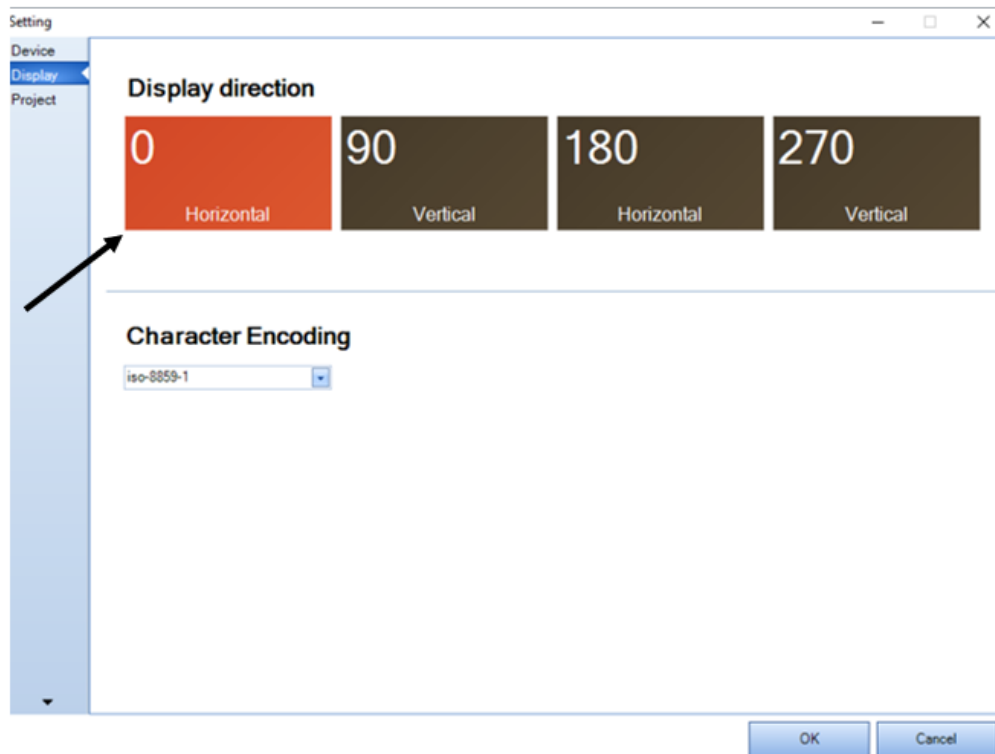


Figura 12: Configuración de la orientación de la pantalla

Finalizada la configuración, se presenta la siguiente ventana de trabajo (Figura B13).

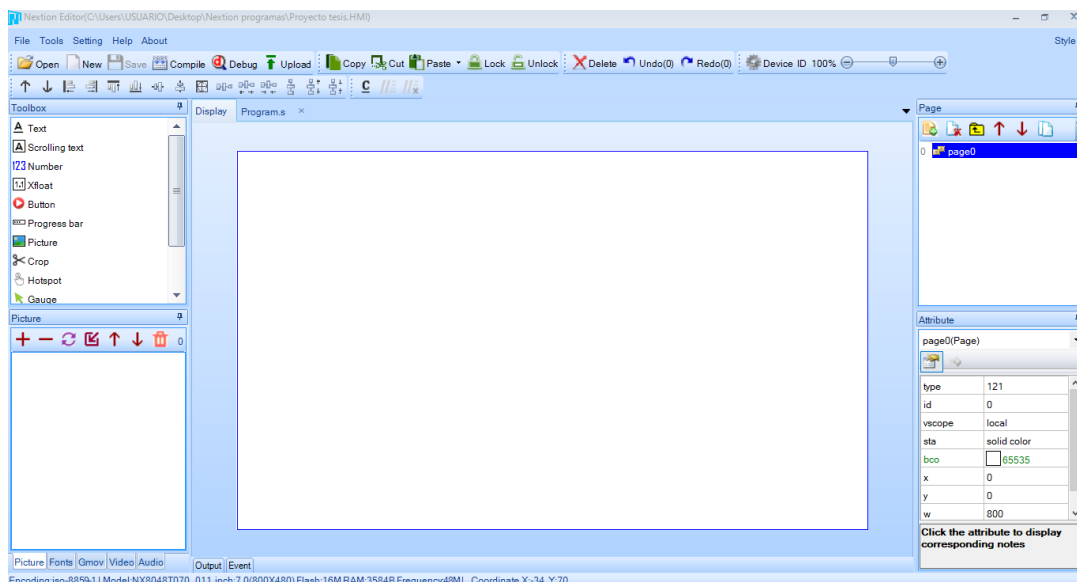


Figura B13: Lugar de trabajo en *Nextion Editor*

Se debe agregar páginas en las que se insertan imágenes, según sea el requerimiento. En la Figura B14 se muestra el procedimiento a seguir.



Figura B14: Añadir páginas en *Nextion Editor*

Una vez agregadas las páginas, se debe añadir las imágenes que van a ser utilizadas. Como se muestra en la Figura B15, se debe dar clic en el ícono “más” para poder agregar imágenes al programa.

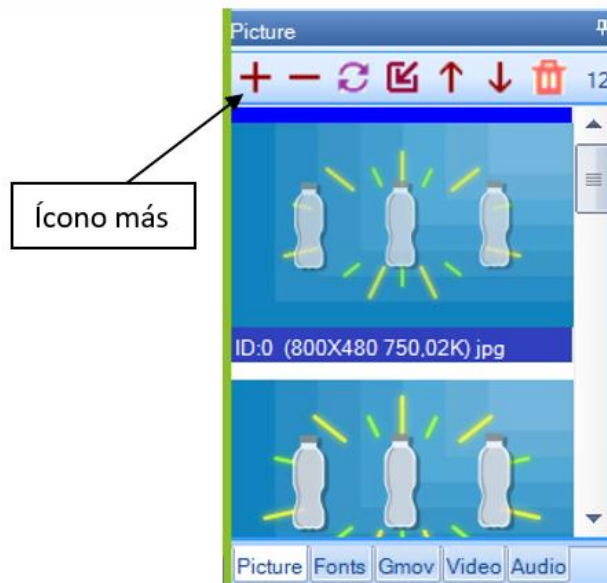


Figura B15: Adición de imágenes en *Nextion Editor*

En la parte inferior derecha del programa se encuentra la sección “*Attribute*”, como se muestra en la Figura B16; en donde, se escoge la página para proceder a editarla. Cada página tiene tres opciones para seleccionar el fondo de pantalla, para insertar una imagen, como se muestra en la Figura 17, se debe seleccionar la opción “*image*”.

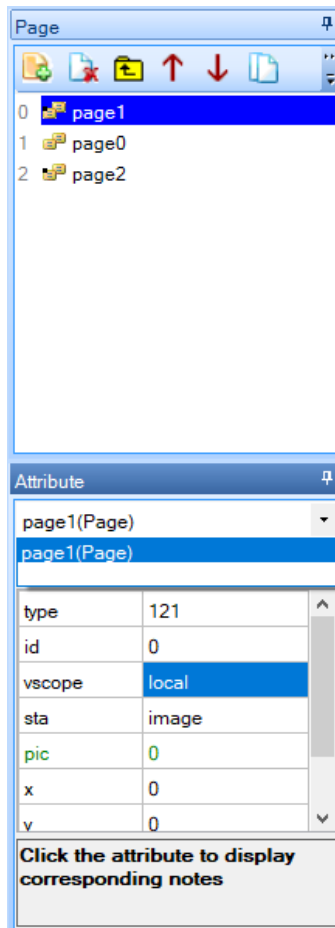


Figura B16: Elección de página para edición en *Nextion Editor*

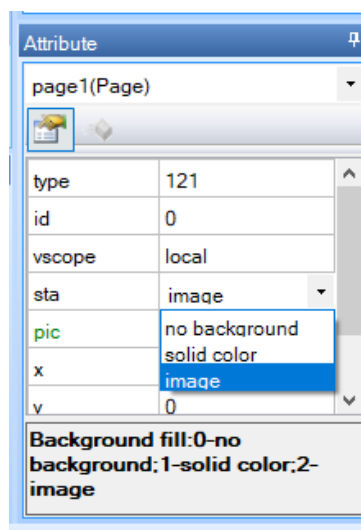


Figura B17: Adición de imagen de fondo en *Nextion Editor*

Como se muestra en la Figura B18, se debe dar clic en “*browse*” para buscar la imagen de fondo deseada entre las que fueron añadidas. Se debe seleccionar la primera imagen si se desea crear una animación.

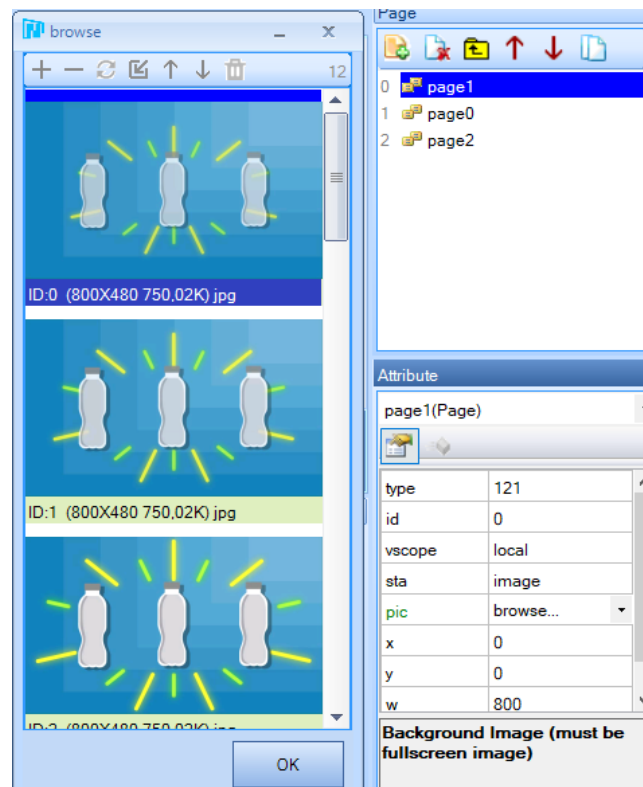


Figura B18: Selección de imagen de fondo en *Nextion Editor*

La Figura B19 muestra el uso del comando “*dim*”, el cual permite configurar el brillo de la pantalla siendo cero el mínimo y cien el máximo. Para añadir esta línea de código se debe dar clic en “*page*” para abrir la ventana de configuración.

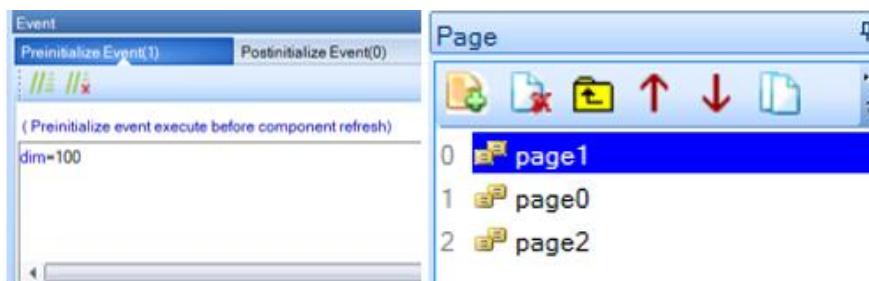


Figura B19: Establecimiento del brillo de la pantalla en *Nextion Editor*

Para poder llevar a cabo una animación, es necesario añadir el elemento *timer*, como se indica en la Figura B20. Este permite realizar la transición de varias imágenes con efectos de animación.

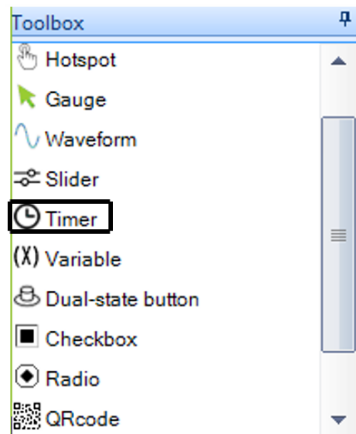


Figura B20: Adición de *timer* en *Nextion Editor*

Como se muestra en la Figura B21, una vez agregado el elemento *timer*, se despliega un ícono con el nombre *tm0*. Se debe dar clic en él para poder desplegar la ventana de configuración que desarrolla el código del *timer*.

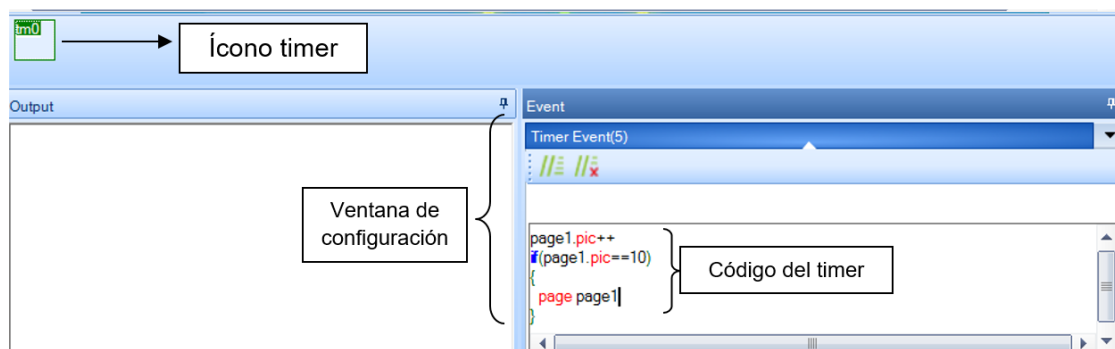


Figura B21: Configuración del *timer* en *Nextion Editor*

El código del *timer* se muestra a continuación, donde el número “10” representa el número de imágenes que conformarán la animación.

```
Page1.pic++
if(page.1pic==10)
{
  page page1
}
```

La velocidad de la transición de las imágenes se puede configurar en la ventana *Attribute*, como se muestra en la Figura B22. El sistema utiliza un tiempo de 100 milisegundos que permite obtener una animación óptima al momento de ser presentada en la pantalla. Se recomienda mantener este tiempo de transición.

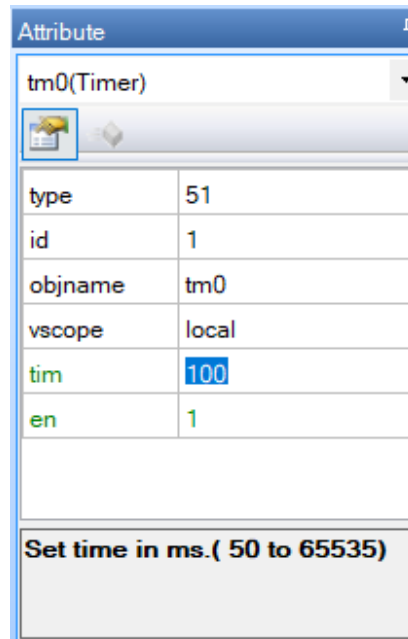


Figura B22: Configuración del tiempo de transición de imágenes

Si se ha finalizado con el proyecto, se debe guardar dando clic en save (Figura B23).

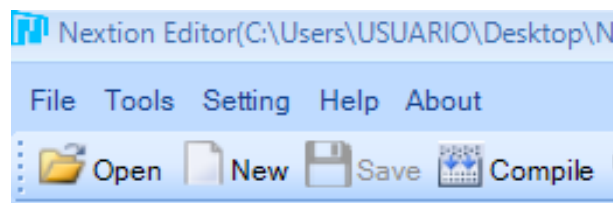


Figura B23: Save

Finalmente, se tiene dos opciones para cargar el programa en la pantalla, estas son:

- Usando un adaptador USB-TTL
- Usando una tarjeta de memoria micro SD.

Usando adaptador USB-TTL

El adaptador USB-TTL (Figura B24) permite cargar el proyecto a la pantalla *Nextion* usando uno de los puertos USB del computador.



Figura B24: Adaptador USB-TTL

En la parte posterior de la pantalla se encuentra el conector como se muestra en la Figura B25, este permite conectar a la pantalla *Nextion* con el adaptador para poder cargar el programa.

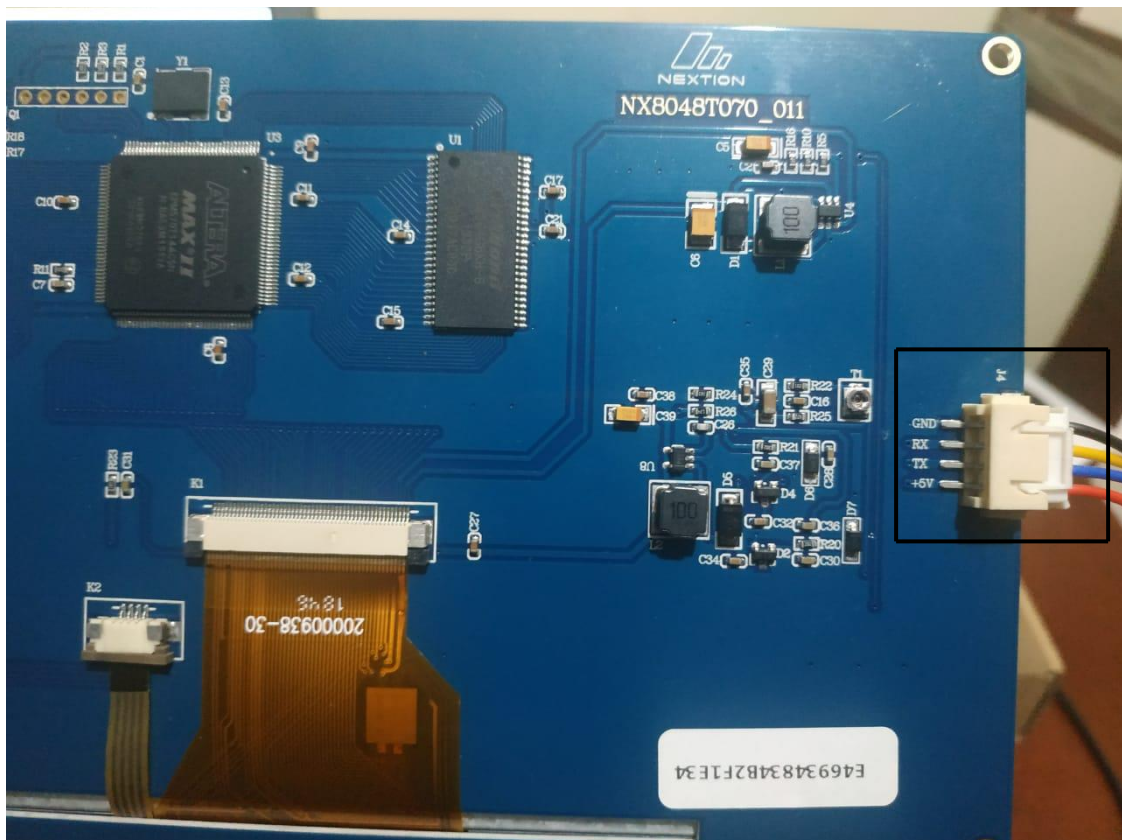


Figura B25: Conector de la pantalla *Nextion*

La Figura B26 muestra la correcta conexión entre los cables del conector de la pantalla con los *pin*s del adaptador USB-TTL. La conexión está dada de la siguiente manera:

- Cable negro (GND) con el pin GND del adaptador.
- Cable amarillo (RX) con el pin TXD del adaptador.
- Cable azul (TX) con el pin RXD del adaptador.
- Cable rojo (+5v) con el pin 5.0V del adaptador.

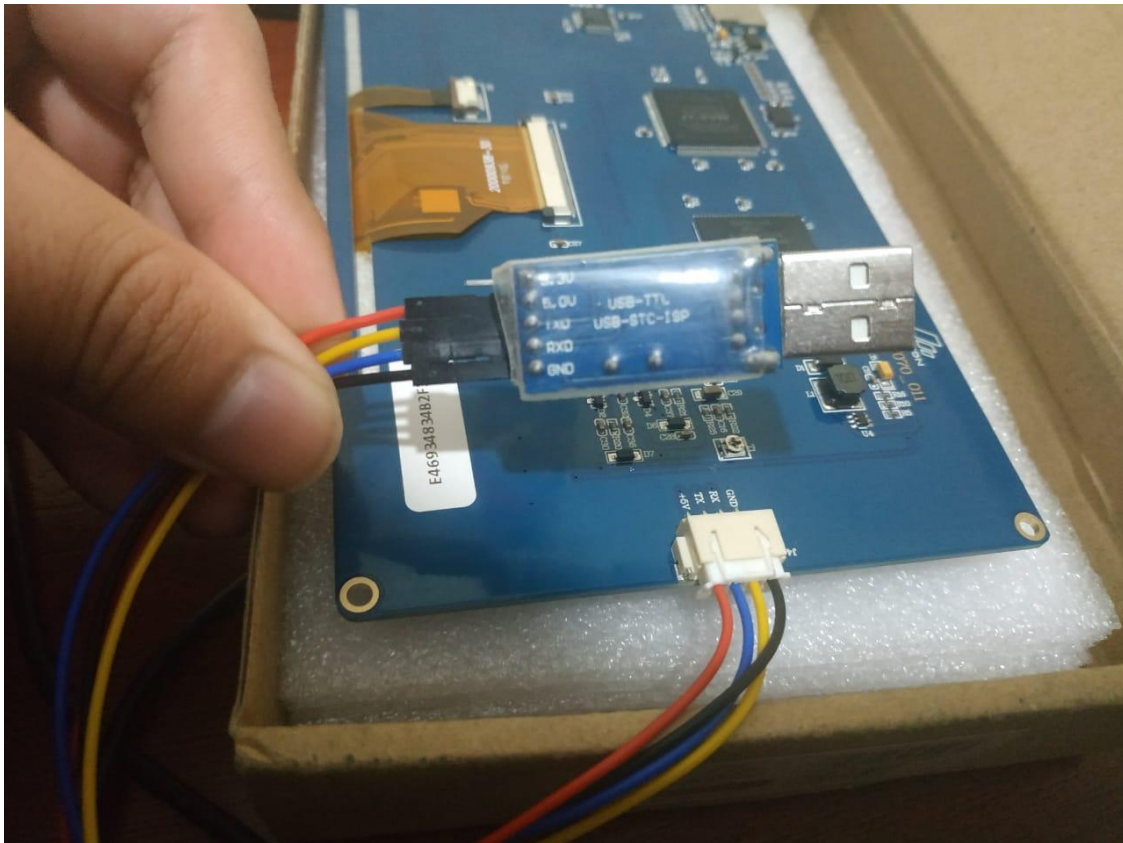


Figura B26: Conexión del adaptador USB-TTL

Cuando se haya conectado el módulo a uno de los puertos USB del computador, se debe abrir el administrador de dispositivos (Figura B27) para comprobar que se puede grabar el proyecto en la pantalla.

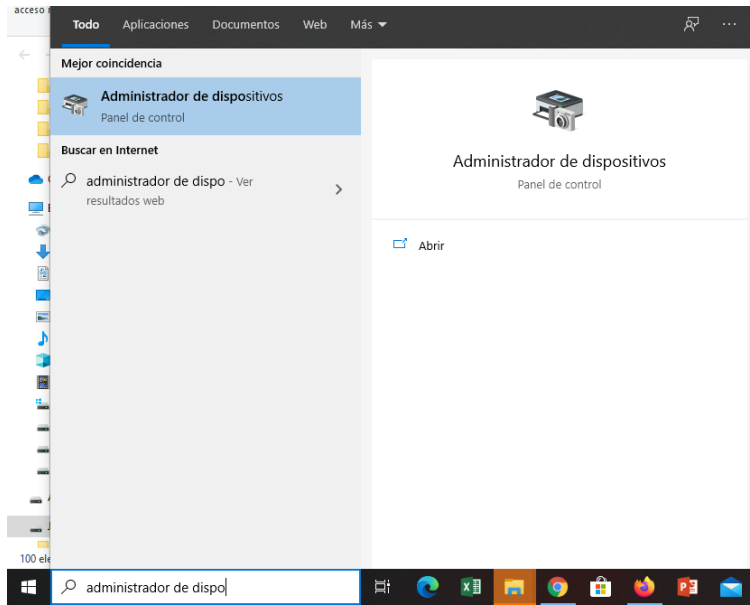


Figura B27: Ruta para abrir el administrador de dispositivos

La Figura B28 muestra un error en la sección de puertos COM y LPT, indicando que no se puede utilizar el módulo para grabar el proyecto.

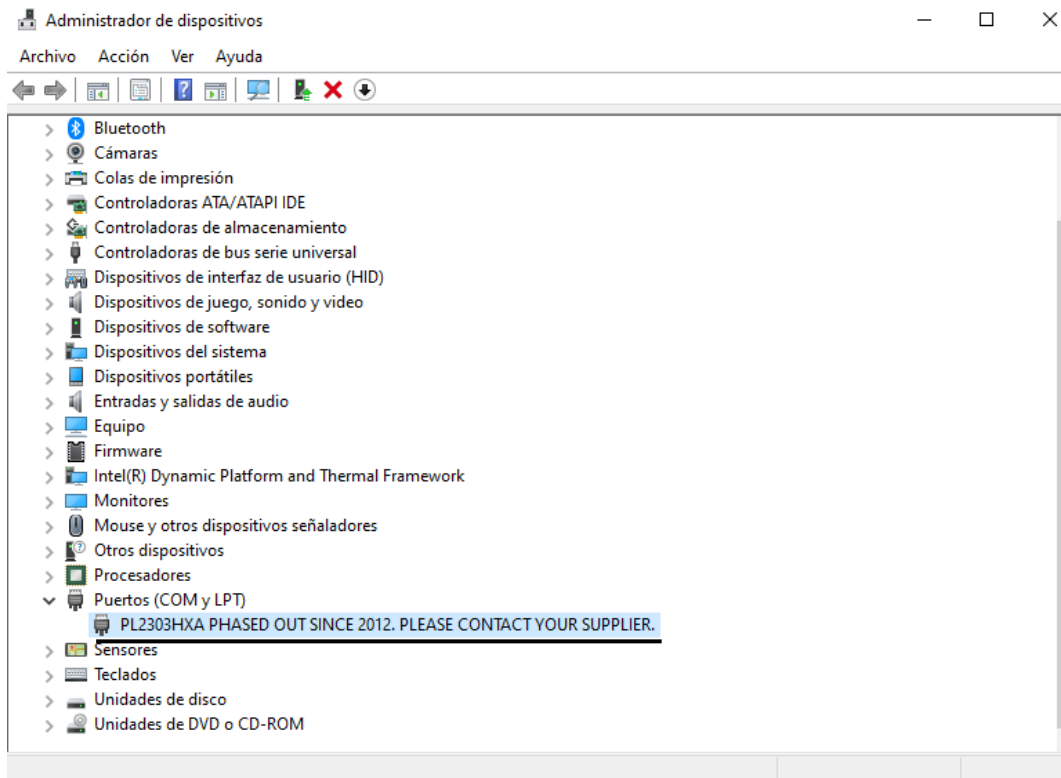


Figura B28: Error en los puertos COM y LPT

Este inconveniente se soluciona de la siguiente manera:

1. Se debe descargar el *driver* desde el siguiente enlace:

<https://drive.google.com/file/d/1DxMkpx5zM4Iq82ts015k4nkbZMmmetom/view>

La Figura B29 muestra la ventana del enlace de descarga

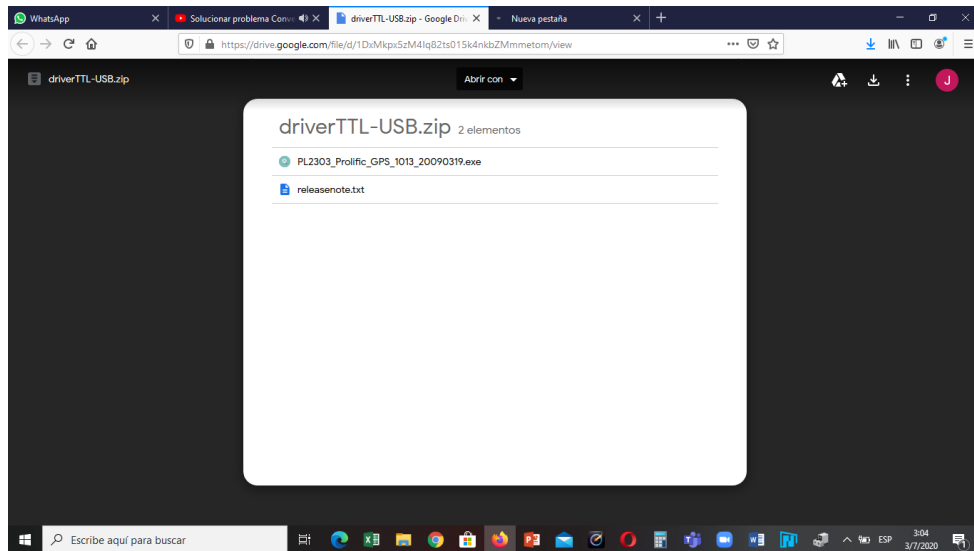


Figura B29: Ventana del enlace de descarga

2. Cuando se haya descargado, se debe extraer el archivo en una carpeta cualquiera que se haya creado. Figura B30.

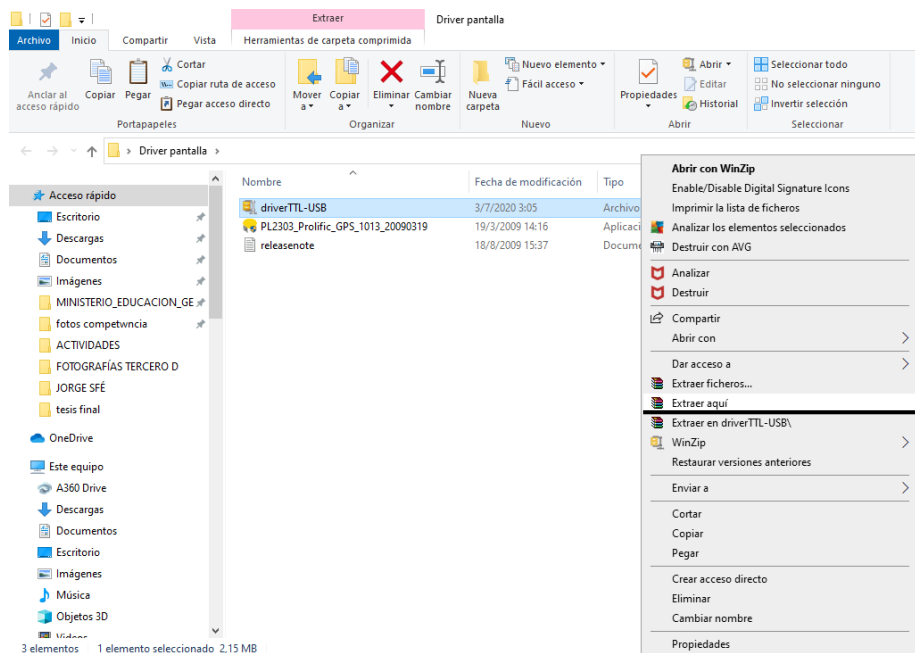


Figura B30: Extracción del archivo

3. Se debe dar clic en el ejecutable (Figura B31) y seguir los pasos descritos en las Figuras B32 a B34 para la instalación del *driver*.

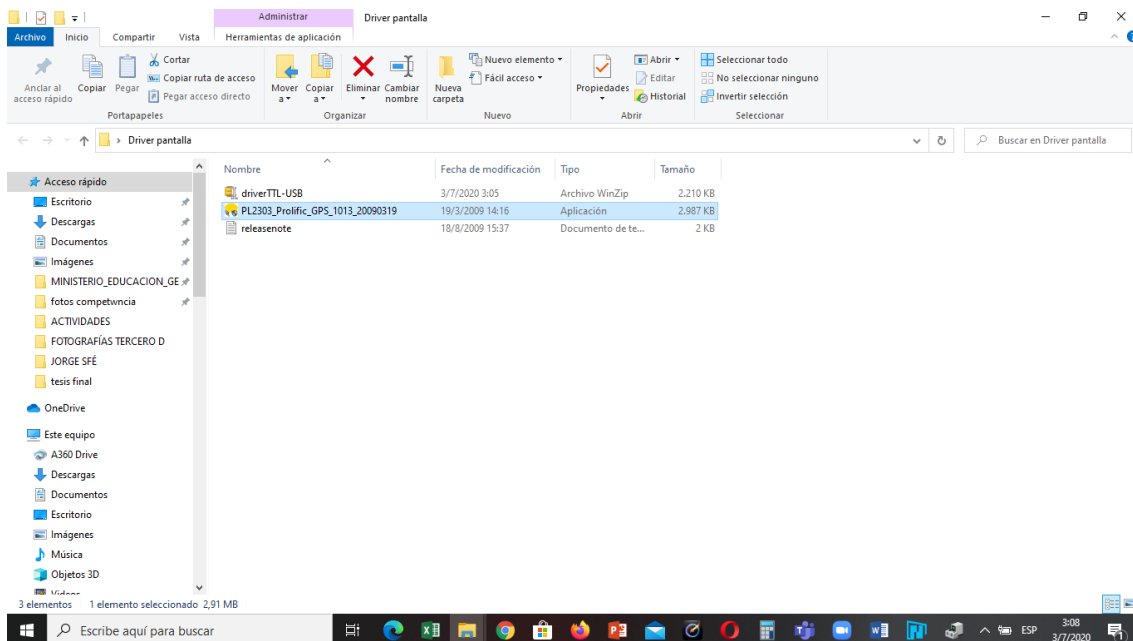


Figura B31: Ejecutable del *driver* del adaptador USB-TTL

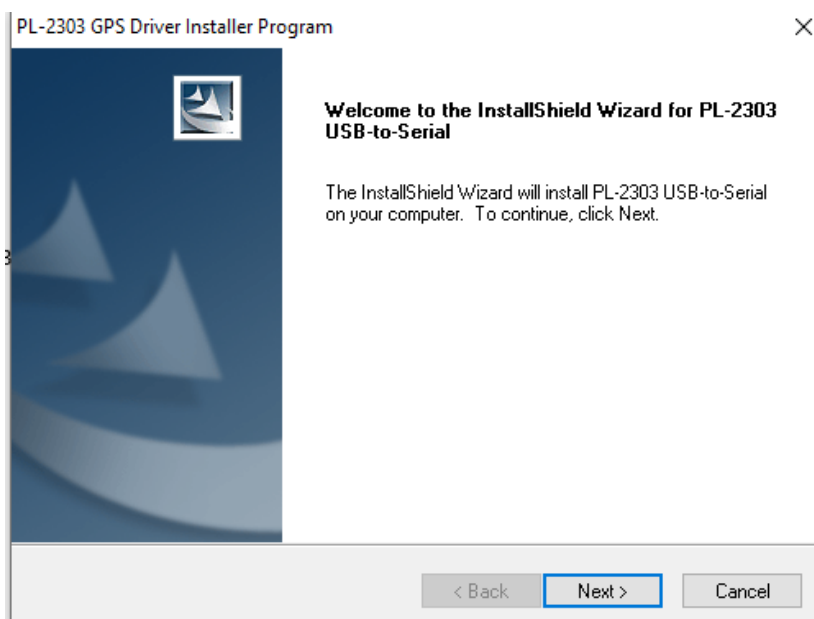


Figura B32: Instalador del *driver* del adaptador USB-TTL

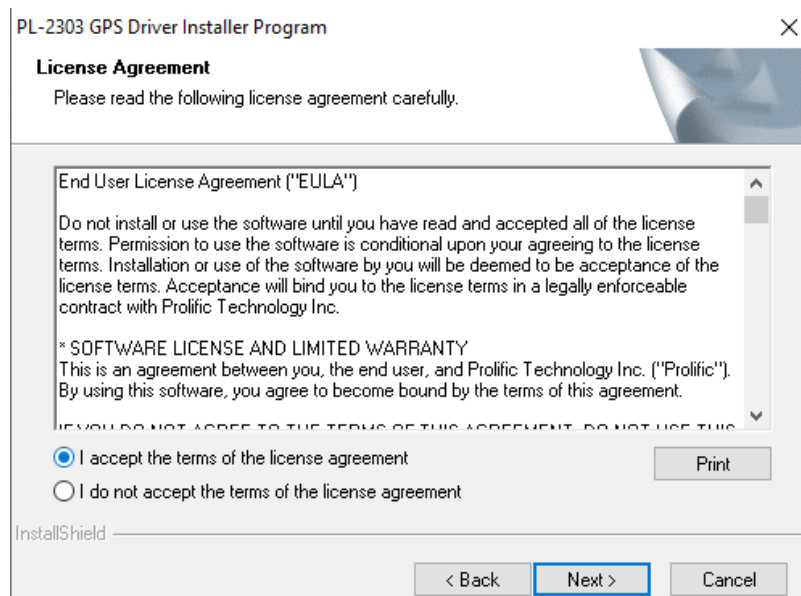


Figura B33: Contrato de licencia del *driver* del adaptador USB-TTL

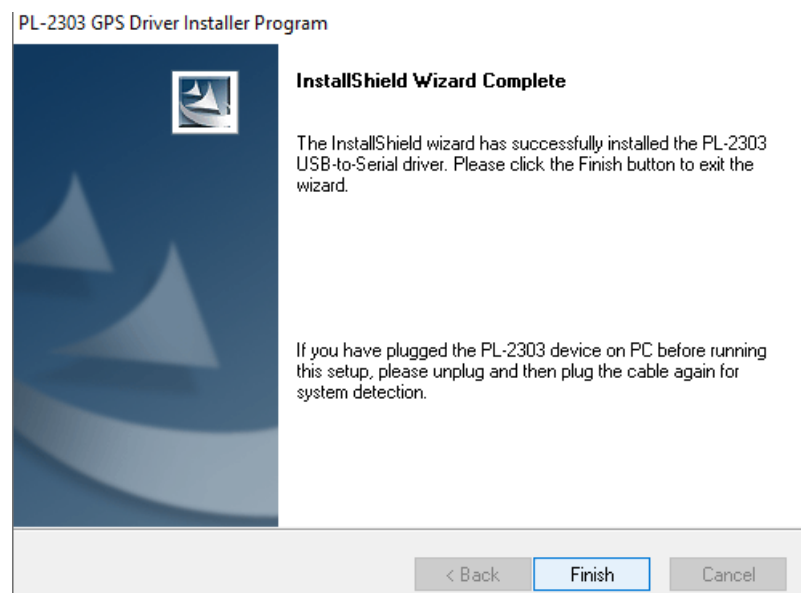


Figura B34: Instalación finalizada del *driver* del adaptador USB-TTL

4. Se abre de nuevo el administrador de dispositivos y dar clic derecho en el error que aparece (Figura B35), seleccionar actualizar controlador. En caso de que el puerto USB no reconozca al módulo USB-TTL, se debe desconectar y conectar nuevamente en el mismo puerto.

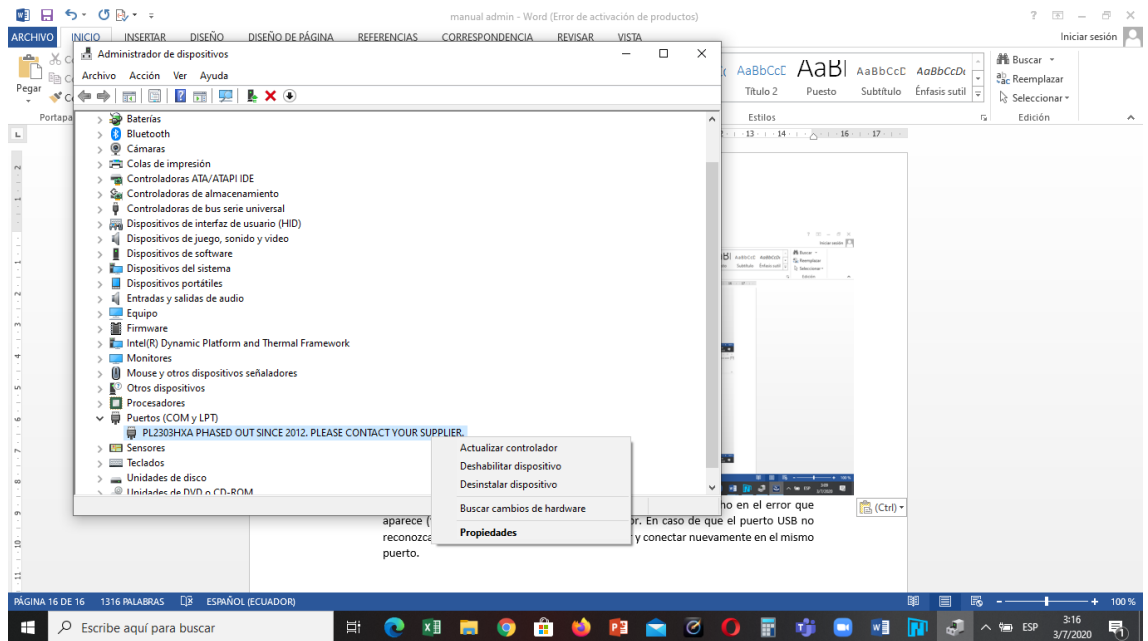


Figura B35: Administrador de dispositivos

5. La Figura B36 indica la ventana que se despliega luego de seleccionar la opción actualizar controlador. Se debe dar clic en la opción buscar *software* de controlador en el equipo.

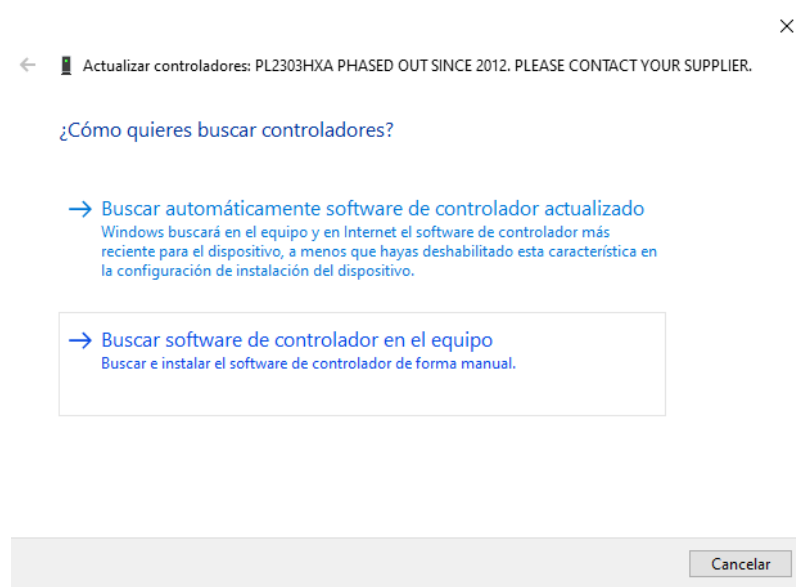


Figura B36: Formas de buscar actualizaciones para el *driver* del adaptador USB-TTL

6. Como se observa en la Figura B37 se debe seleccionar la opción “elegir una lista de controladores disponible en el equipo”.

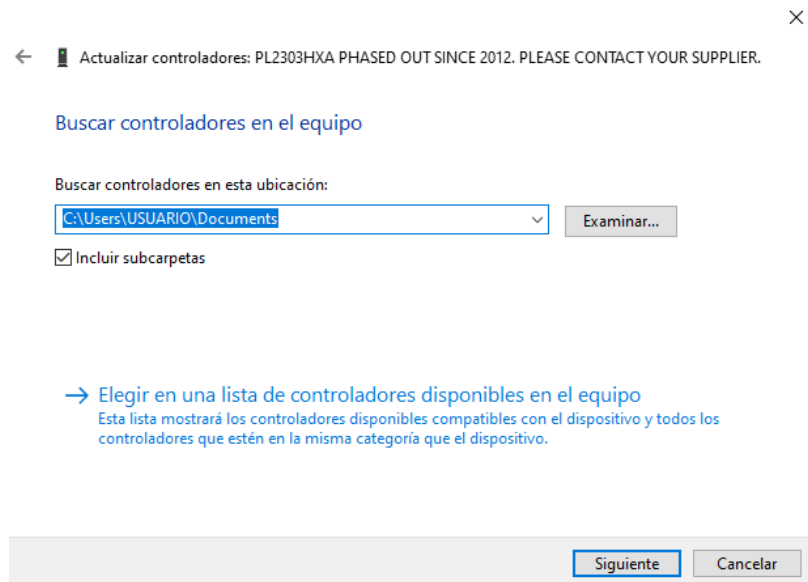


Figura B37: Carpeta para actualizar el *driver* del adaptador USB-TTL

7. Se debe seleccionar la versión más antigua para este dispositivo (Figura B38). Y dar clic en siguiente.

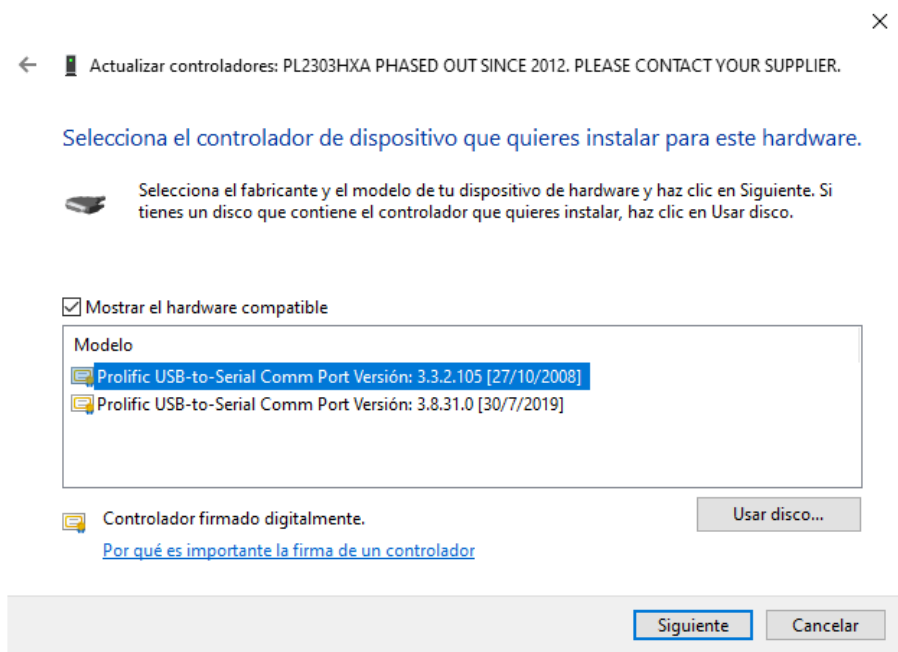


Figura B38: Seleccionar la versión del *driver* del adaptador USB-TTL

8. Como indican las Figuras B39 y B40, el adaptador se puede utilizar con normalidad.

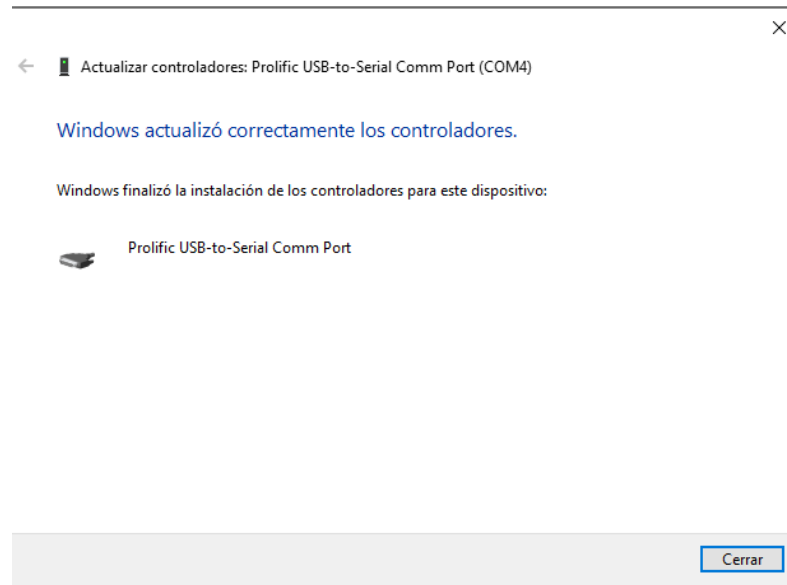


Figura B39: Actualización finalizada del *driver* del adaptador USB-TTL

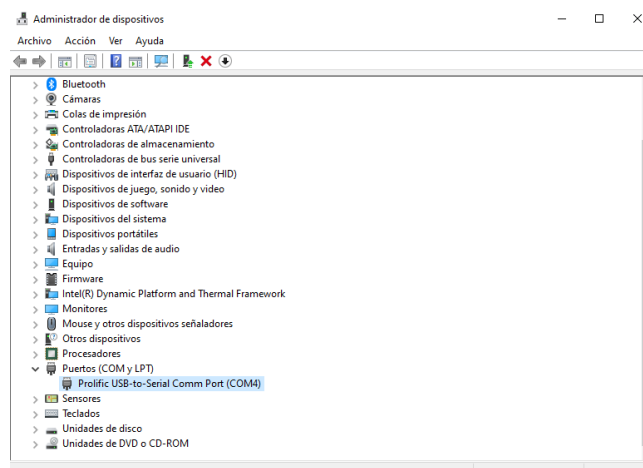


Figura B40: *Driver* del adaptador USB-TTL actualizado

La Figura B41 muestra el ícono *compile*, el cual permite revisar si el proyecto fue creado correctamente y es apto para ser grabado en la memoria de la pantalla.

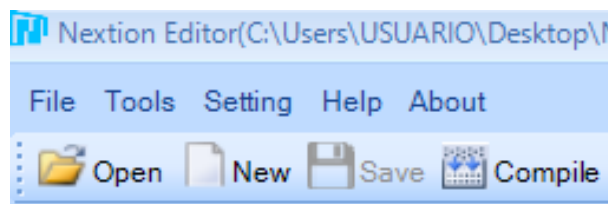


Figura B41: *Compile*

Si no se han detectado errores, se debe dar clic en el ícono *upload* (Figura B42).

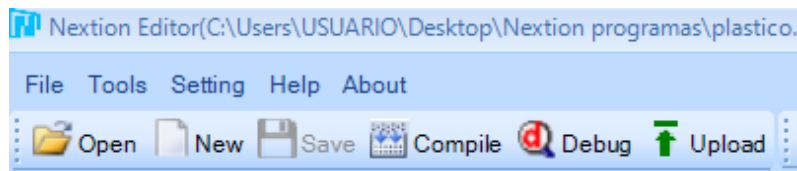


Figura B42: *Upload*

La Figura B43 indica la ventana que se despliega, se debe dar clic en el botón *go* para grabar el programa.

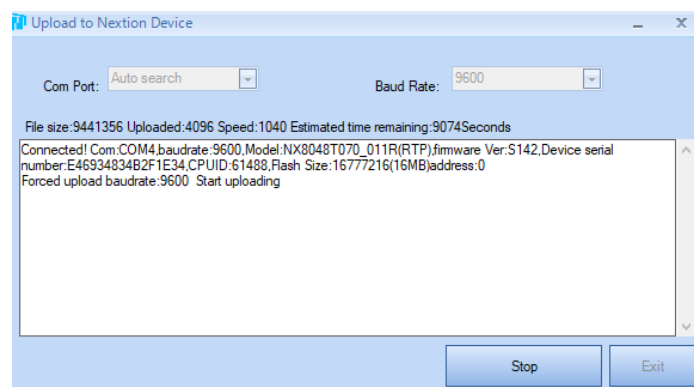


Figura B43: *Upload* del programa

La Figura B44 indica cómo el proyecto va cargándose en la pantalla.

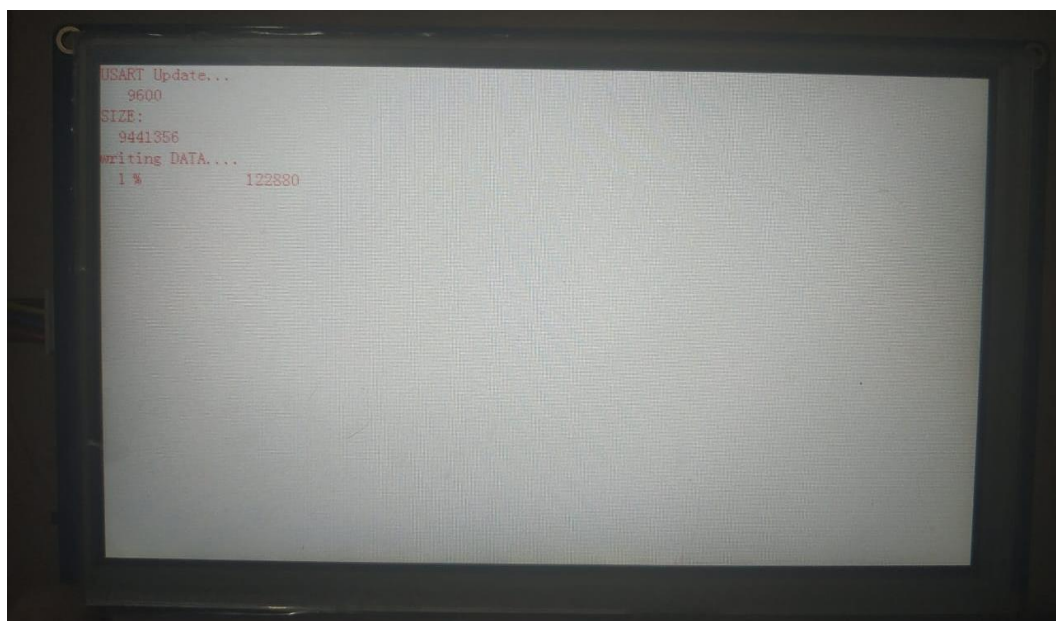


Figura B44: Carga del programa en la pantalla *Nextion*

Usando una memoria micro SD

La tarjeta micro SD que se vaya a utilizar para cargar el programa a la pantalla *Nextion*, no puede ser mayor a los 32GB.

El procedimiento para cargar el programa es más sencillo; simplemente se debe copiar el archivo tipo HMI (Figura B45) a la tarjeta micro SD, como se indica en la Figura B46.

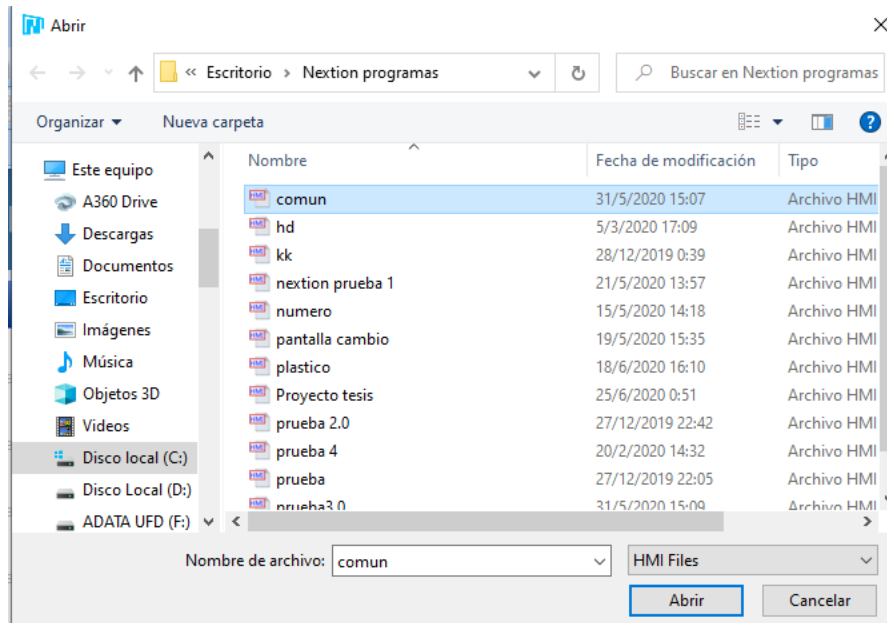


Figura B45: Archivo tipo HMI

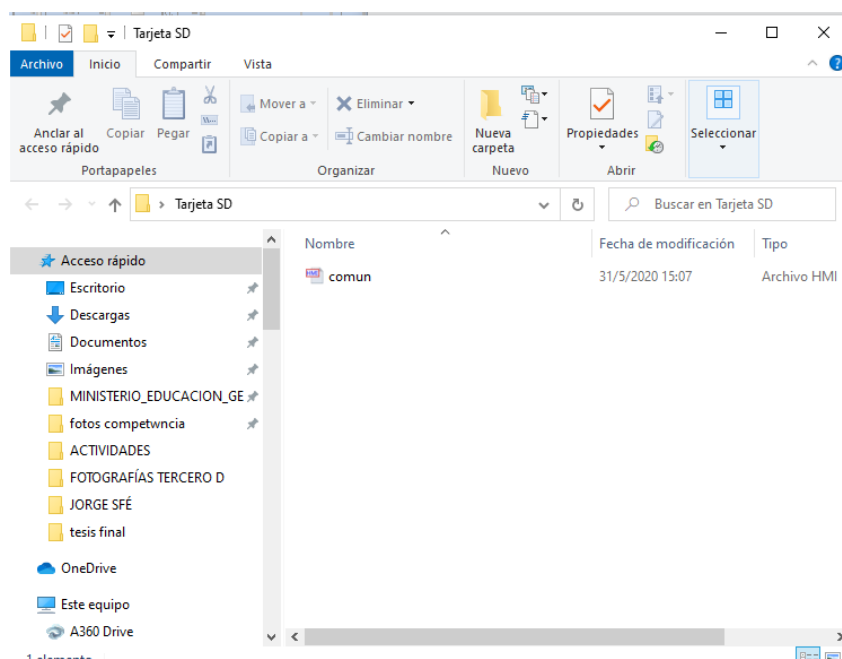


Figura B46: Archivo HMI cargado a la micro SD

La ranura en la que se debe colocar la tarjeta micro SD se encuentra en la parte posterior de la pantalla, como se muestra en la Figura B47; este procedimiento debe realizarse una vez se haya copiado el archivo a la tarjeta. Cargar el programa usando la tarjeta micro SD es más rápido que hacerlo mediante el adaptador USB-TTL.

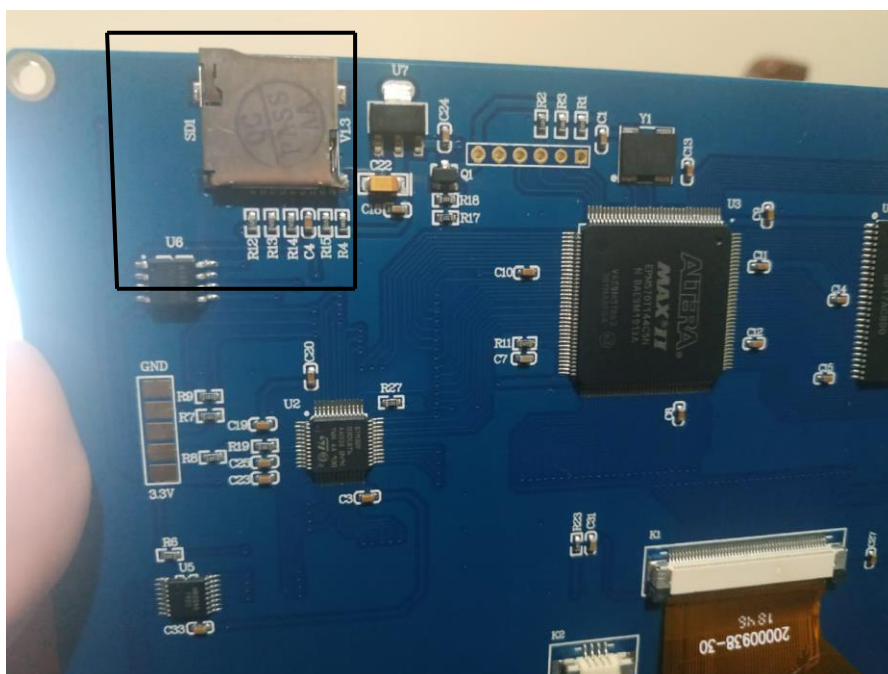


Figura B47: Ranura para la tarjeta micro SD

Cambiar el tiempo del estado de activación

Durante el estado de activación se asignó un tiempo de 10 segundos para realizar la acción de depositar la basura y un tiempo de 3 segundos para que el sistema permita realizar nuevamente la acción si en la primera oportunidad no pudo ser completado el proceso.

Es necesario contar con el programa *Arduino* IDE versión 1.8.1. Se puede descargar desde el siguiente enlace: <https://www.filehorse.com/es/descargar-arduino/28950/descargar/>

La Figura B48 muestra la ventana del enlace de descarga.

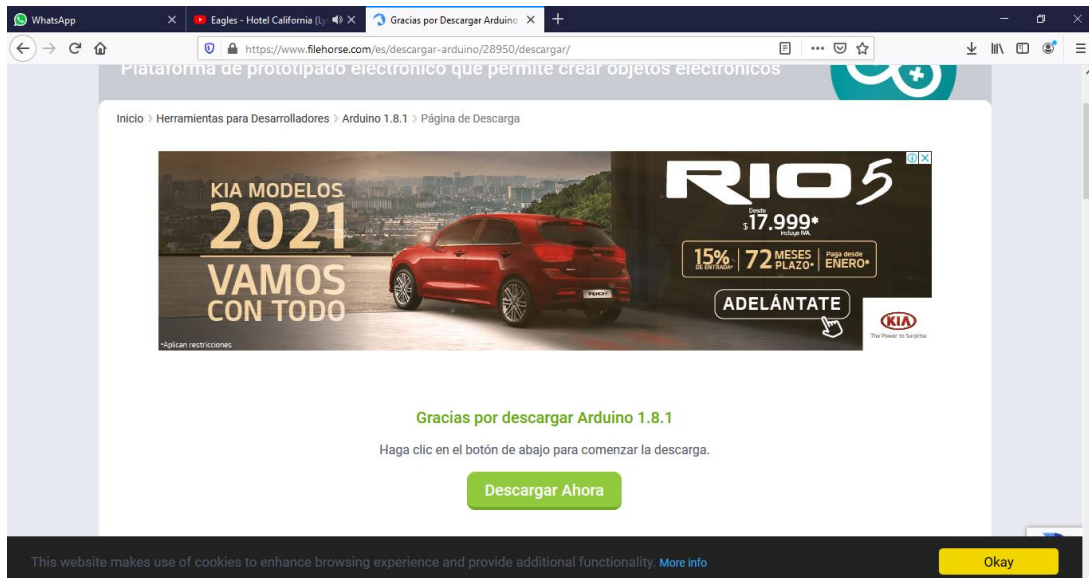


Figura B48: Ventana de descarga de *Arduino IDE*

El procedimiento de instalación es similar a los anteriores, siendo necesario únicamente dar en siguiente para completar la instalación.

En las Figuras B49 a B51 se detallan los pasos para la instalación.

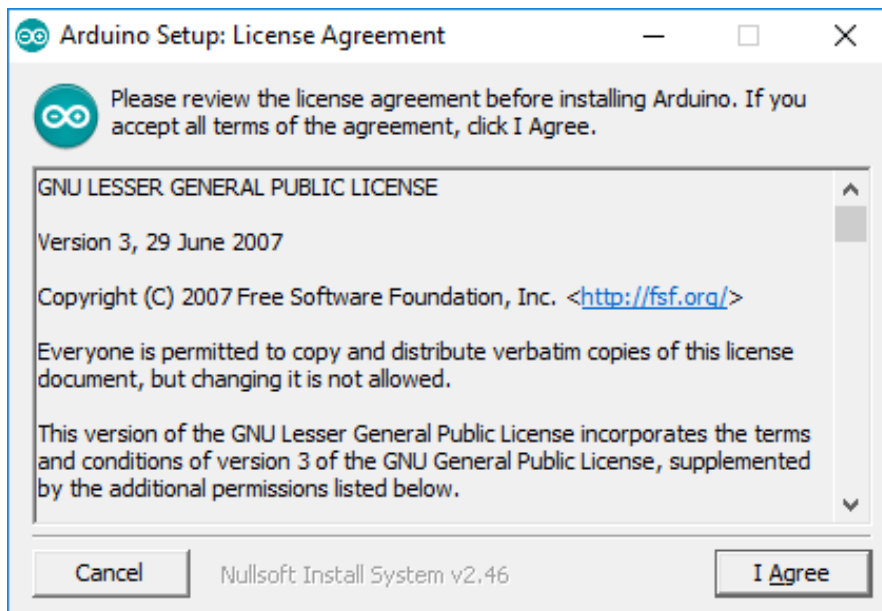


Figura B49: Acuerdo de licencia de *Arduino*

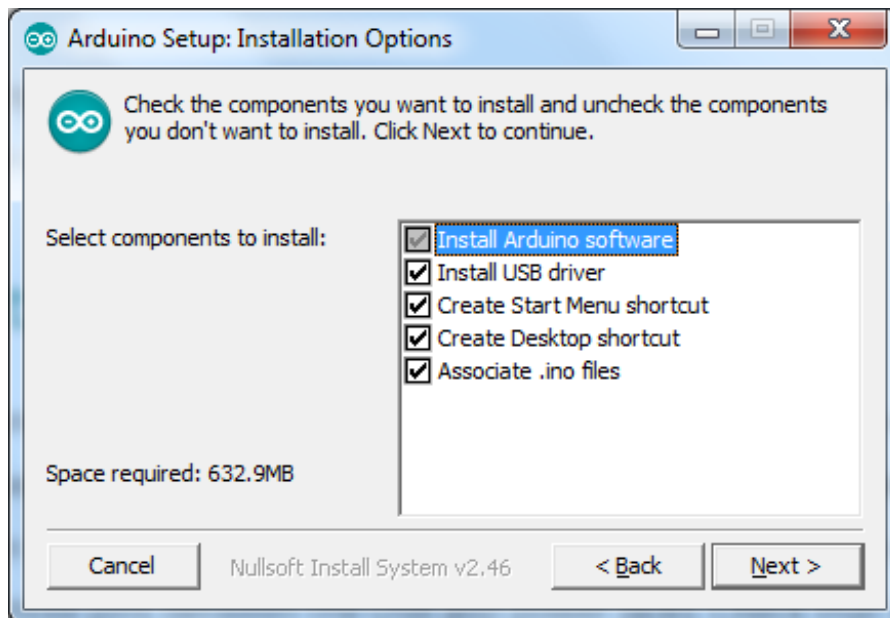


Figura B50: Opciones de instalación de *Arduino*

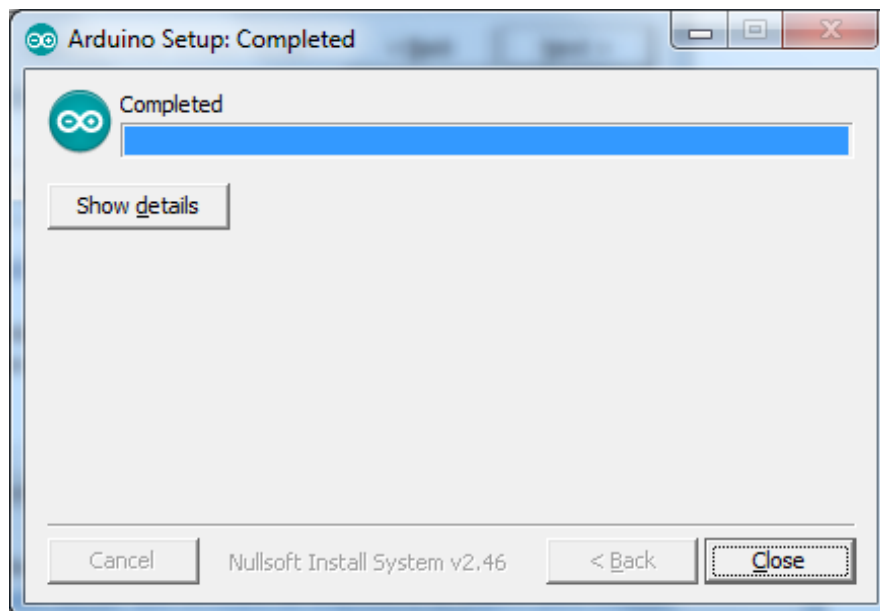


Figura B51: Instalación de *Arduino* IDE completa

La Figura B52 muestra la ventana del programa *Arduino* IDE cuando ha sido finalizada la instalación.

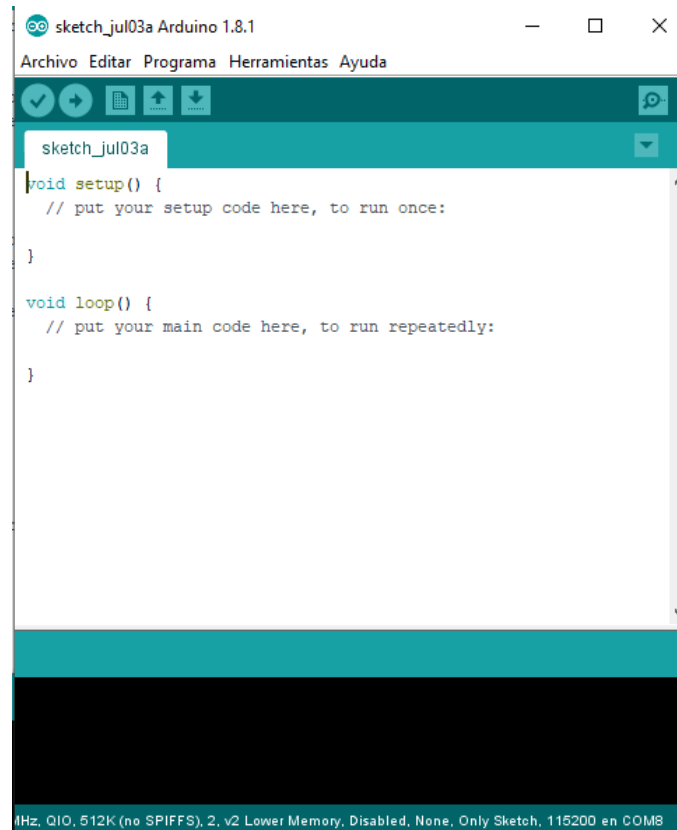


Figura B52: Sketch de Arduino IDE

Se proveerá el archivo con el programa principal, se puede apreciar en la Figura B53 la carpeta que lo contiene.

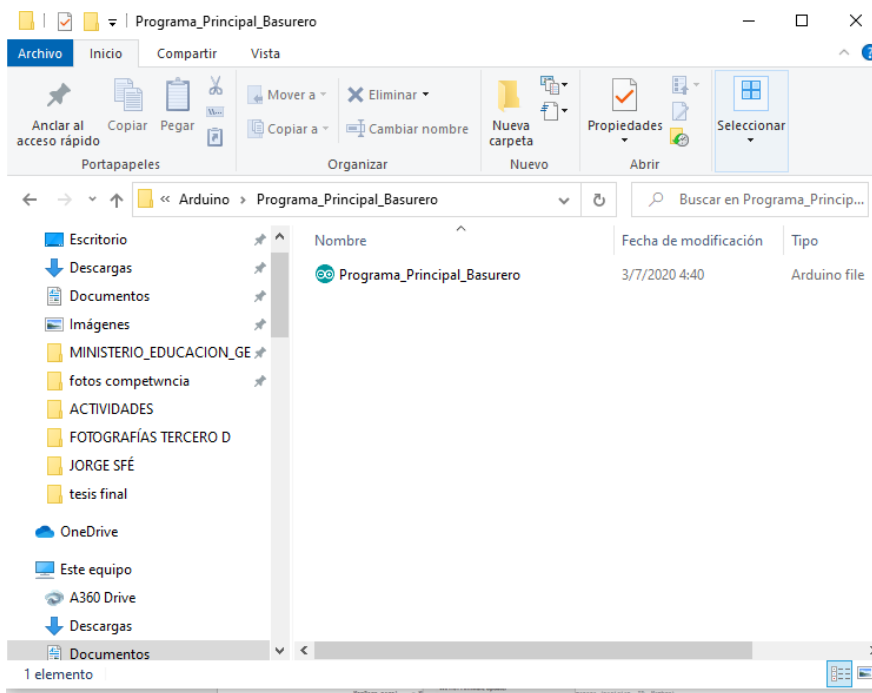


Figura B53: Carpeta del programa principal

Se debe abrir el archivo para poder proceder con la configuración para el cambio de tiempos en la etapa de activación. En la Figura B54 se muestra el programa principal.



Figura B54: Programa principal dentro de *Arduino IDE*

Para cambiar los tiempos del estado de activación se debe revisar la siguiente sintaxis del programa.

```

else { //Condicional dentro del rango de medida
  page1.show(); //Muestra pag1 de la pantalla
  digitalWrite(luces, HIGH); //Enciende luz verde
  servoMotor1.write(180); //180 grados cerrado
  servoMotor2.write(0); //0 grados cerrado
  delay(100);
  servoMotor1.write(0); //0 grados abierto
  servoMotor2.write(180); //180 grados abierto
  delay(7000);
  digitalWrite(luces, LOW); //Enciende luz roja
  delay(3000);
  servoMotor1.write(180); //180 grados cerrado
  servoMotor2.write(0); //0 grados cerrado
  delay(5000);
  dat=digitalRead(in_sensor); //Lee el sensor de nivel
}

```

El *delay* de 7000 milisegundos (7 segundos) corresponde al tiempo que se tiene para botar la basura antes de que se cierre la tapa del sistema automatizado.

El *delay* de 3000 milisegundos (3segundos) representa el tiempo que tarda en cerrarse la tapa y encenderse la luz piloto roja indicando que el tiempo ha finalizado.

Por último, el *delay* de 3000 milisegundos (3 segundos) es el tiempo que se debe esperar para tener un nuevo intento si no se realizó la acción en la primera oportunidad.

Únicamente se debe cambiar el valor entre paréntesis que está alado de los *delays* para configurar el nuevo tiempo.

Se debe dar clic en el ícono “verificar” para guardar los cambios realizados al programa principal. (Figura B55)

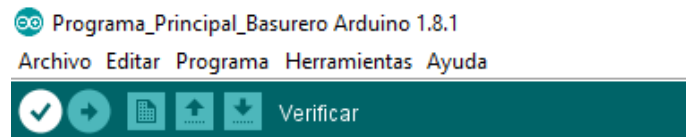


Figura B55: Guardar programa en *Arduino IDE*

Cambio de imágenes en pantalla

Si se ha creado un nuevo proyecto para la pantalla *Nextion*, es necesario añadir la librería de la pantalla al *software* de *Arduino IDE* para realizar las modificaciones necesarias.

Para incluir la librería, se debe seguir los siguientes pasos:

1. Descargar la librería en el siguiente enlace: https://nextion.tech/nextion-editor/#_section3

La Figura B56 muestra la ventana una vez abierto el enlace, se debe elegir la opción descargar zip.

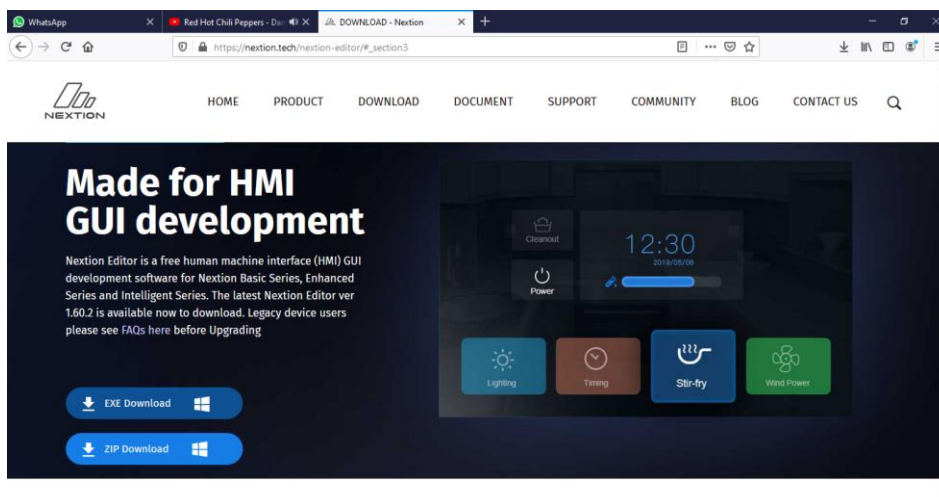


Figura B56: Ventana donde descargar librería de *Nextion*

- Una vez descargado el archivo, se debe abrir el programa principal del basurero. Como se indica en la Figura B57, se debe ir a la pestaña programa, sección incluir librería, y elegir añadir librería zip.

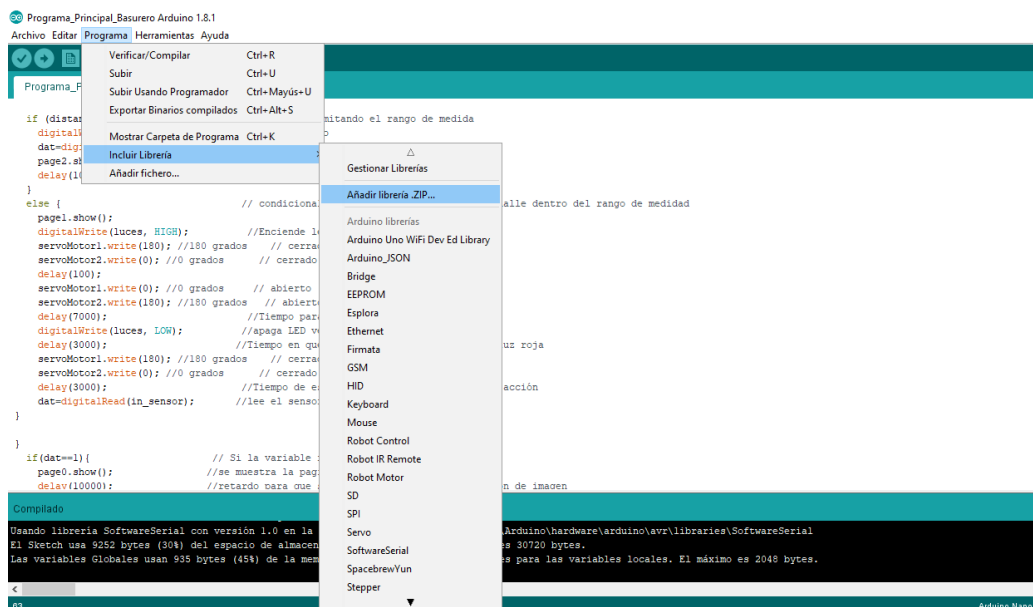


Figura B57: Añadir librería .zip en Arduino

- Se debe seleccionar el archivo en la carpeta donde se ha descargado, dar clic en abrir y la librería se instalará. (Figura B58)

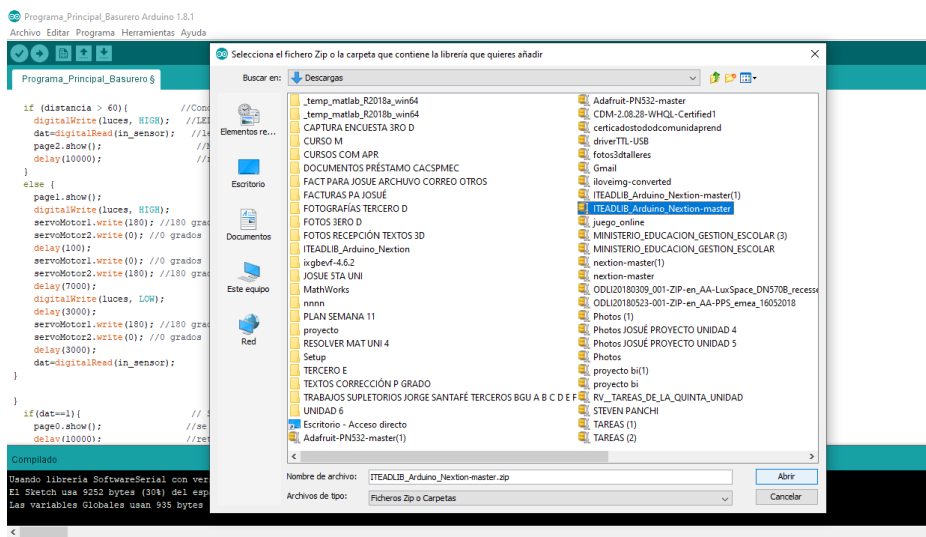


Figura B58: Selección del archivo .zip

- Una vez instalada la librería, se puede realizar los cambios para mostrar las imágenes en las etapas correspondientes del sistema automatizado.

En la siguiente sintaxis se explica cómo deben colocarse los identificadores de las páginas creadas en el proyecto de la pantalla *Nextion*.

```
NexPage page0 = NexPage(1, 0, "page0"); //Se indica las paginas a ser usadas
NexPage page1 = NexPage(0, 0, "page1"); //de la siguiente manera (posicion, ID, Nombre)
NexPage page2 = NexPage(2, 0, "page2");
```

La Figura B59 muestra la ubicación de los identificadores dentro del editor de *Nextion*, para ser copiados en el programa principal de los basureros.

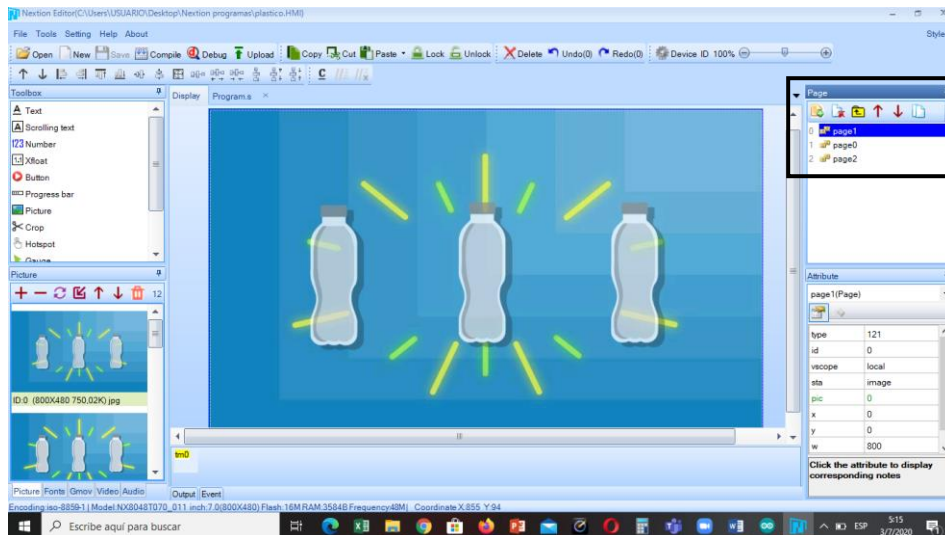


Figura B59: Identificadores de las páginas en *Nextion*

5. El sistema tiene 3 etapas: activación, interrupción y reposo. Cada página posee una imagen diferente. En el programa principal sin modificaciones las páginas están configuradas de la siguiente manera:
 - Página 0: Imagen de basurero lleno. Estado de interrupción.
 - Página 1: Animación. Estado de activación.
 - Página 2: Pantalla negra en mínimo consumo de energía. Estado de reposo.

Si en el nuevo proyecto se ha cambiado este orden, se debe modificar la línea de código *page#.show()*; siendo el símbolo de numeral el número de la página que se mostrará en pantalla.

Las sintaxis de las diferentes etapas se indican a continuación:

➤ Activación:

```
else { //Condicional dentro del rango de medida
  page1.show(); //Muestra pag1 de la pantalla
  digitalWrite(luces, HIGH); //Enciende luz verde
  servoMotor1.write(180); //180 grados cerrado
  servoMotor2.write(0); //0 grados cerrado
  delay(100);
  servoMotor1.write(0); //0 grados abierto
  servoMotor2.write(180); //180 grados abierto
  delay(7000);
  digitalWrite(luces, LOW); //Enciende luz roja
  delay(3000);
  servoMotor1.write(180); //180 grados cerrado
  servoMotor2.write(0); //0 grados cerrado
  delay(5000);
  dat=digitalRead(in_sensor); //Lee el sensor de nivel
}
```

➤ Interrupción:

```
if(dat==1){ //Si la variable recibe 1 lógico
  digitalWrite(luces, LOW); //Enciende luz roja
  page0.show(); //Muestra la pag0 de la pantalla
  delay(3000); //Retardo para vizualizar la imagen
  digitalWrite(wifi,HIGH); //Led indicador encendido wi-fi
  delay(3000);
  digitalWrite(wifi,LOW); //Led indicador apagado wi-fi
  dat=digitalRead(in_sensor); //Testea entrada sensor de nivel
  while(dat==1){ //Bucle detecta cambio de estado sensor de nivel
    delay(100);
    dat=digitalRead(in_sensor);
  }
}
```

➤ Reposo:

```
if (distancia > 60){ //Condicional if delimitando el rango de medida
  page2.show(); //Muestra pag2 de la pantalla
  delay(100); //Retardo para vizualizar la imagen
  digitalWrite(luces, HIGH); //Enciende luz verde
  dat=digitalRead(in_sensor); //Lee el pin del sensor de nivel
}
```

En cada caso, únicamente debe cambiarse el número de la página que se mostrará en pantalla.

➤ Grabar cambios en la tarjeta *Arduino Nano*

Realizados los cambios, es necesario volver a grabar el programa en la tarjeta *Arduino Nano* (Figura B60), posee un cable USB que permite conectarlo a uno de los puertos del computador para subir el programa.

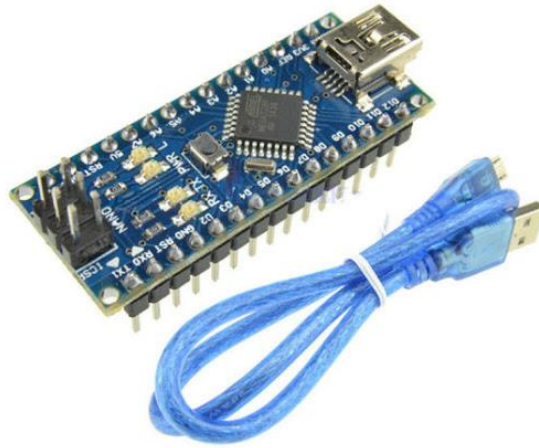


Figura B60: *Arduino Nano* con su cable de conexión

Primero, se debe conectar el *Arduino Nano* al computador mediante el cable USB, abrir el archivo con el programa principal, luego ir a la pestaña de herramientas, seleccionar la placa *Arduino Nano* y seleccionar el puerto en el que se halla conectado, como se indica en la Figura B61.

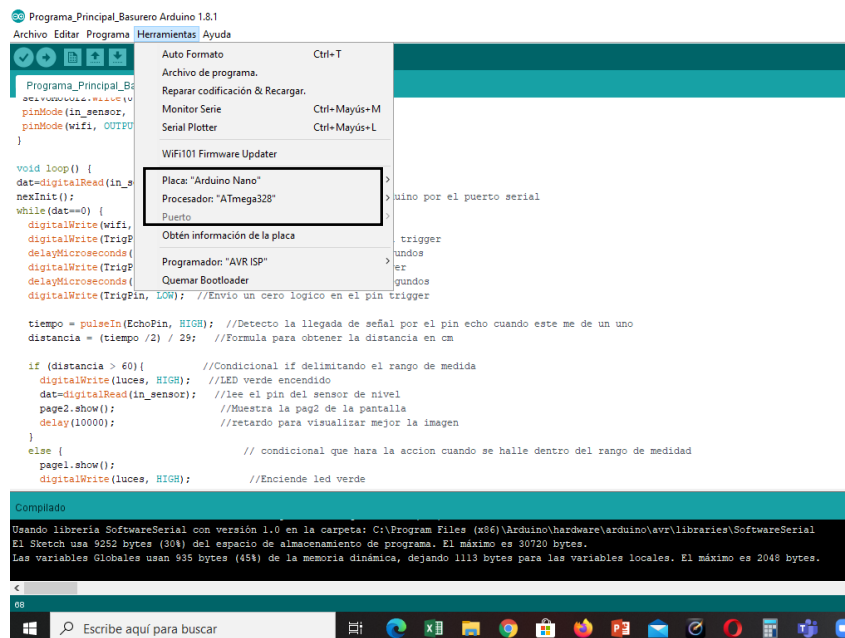


Figura B61: Selección de la placa *Arduino*

Una vez realizada la acción anterior, la tarjeta *Arduino Nano* debe estar conectada al computador. Se debe dar clic en el ícono “subir”, como se indica en la Figura B62 para grabar el programa en la tarjeta.

```

Programa_Principal_Basurero Arduino 1.8.1
Archivo Editar Programa Herramientas Ayuda
Subir
Programa_Principal_Basurero $
// CERRADO
// Definición de pines
pinMode(in_sensor, INPUT); //declaro sensor entrada
pinMode(wifi, OUTPUT); //declaro wifi salida
}

void loop() { //Inicia el programa
  dat=digitalRead(in_sensor); //Lee la entrada in_sensor
  nexInit(); //Comunica la pantalla con el arduino por el puerto serial
  while(dat==0) {
    digitalWrite(wifi, LOW); //Salida pin WiFi cero
    digitalWrite(TrigPin, LOW); //Envio un cero logico por el pin trigger
    delayMicroseconds(4); //Realizo un retraso de 4 microsegundos
    digitalWrite(TrigPin, HIGH); //Envio uno logico en el pin trigger
    delayMicroseconds(10); //Realizo un retraso de 10 microsegundos
    digitalWrite(TrigPin, LOW); //Envio un cero logico en el pin trigger

    tiempo = pulseIn(EchoPin, HIGH); //Detecto la llegada de señal por el pin echo cuando este me de un uno
    distancia = (tiempo / 2) / 29; //Formula para obtener la distancia en cm

    if (distancia > 60){ //Condicional if delimitando el rango de medida
      digitalWrite(luces, HIGH); //LED verde encendido
      dat=digitalRead(in_sensor); //lee el pin del sensor de nivel
      page2.show(); //Muestra la pag2 de la pantalla
      delay(10000); //retraso para visualizar mejor la imagen
    }
    else { // condicional que hara la accion cuando se halle dentro del rango de medidad
      page1.show();
      digitalWrite(luces, HIGH); //Enciende led verde
    }
  }
}

```

Compilado
 Usando libreria SoftwareSerial con versión 1.0 en la carpeta: C:\Program Files (x86)\Arduino\hardware\arduino\avr\libraries\SoftwareSerial
 El Sketch usa 9252 bytes (304) del espacio de almacenamiento de programa. El máximo es 30720 bytes.
 Las variables Globales usan 935 bytes (45t) de la memoria dinámica, dejando 1113 bytes para las variables locales. El máximo es 2048 bytes.

Figura B62: Programa subido en *Arduino IDE*

Cambios para el módulo *Wi-Fi*

Se provee la carpeta con los archivos para la configuración del módulo *Wi-Fi* ESP8266 (Figura B63).

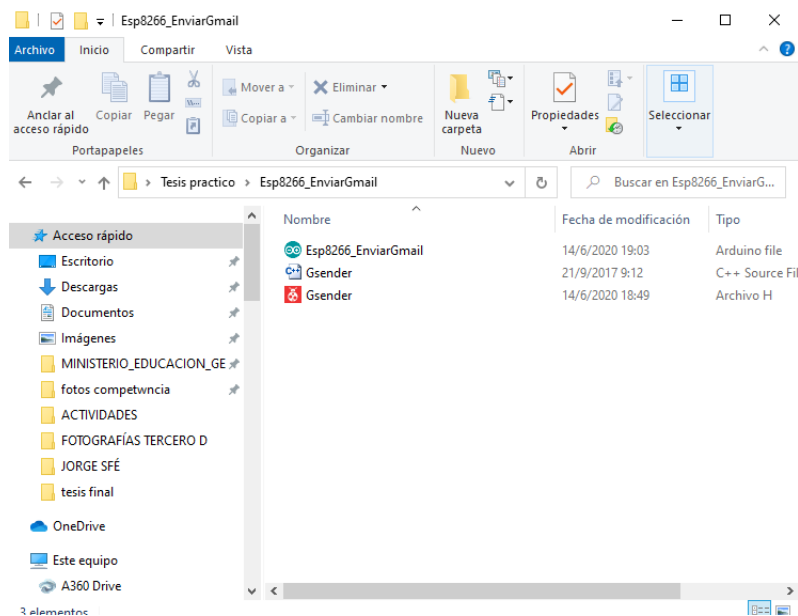


Figura B63: Archivos para la configuración del módulo *Wi-Fi*

Los tres archivos deben permanecer en la misma carpeta para que la librería *Gsender* funcione correctamente.

En el módulo *Wi-Fi ESP8266 NodeMCU V3* se pueden realizar los siguientes cambios.

- Conectarse a una nueva red de área local o actualizar las credenciales de acceso.
- Ingresar un nuevo correo remitente.
- Ingresar un nuevo correo destinatario.
- Actualizar el mensaje que le llega al administrador.

Es importante agregar la versión de la placa del módulo *Wi-Fi* para grabar el programa.

Los pasos son los siguientes:

1. Acceder al repositorio digital de *Github* mediante el siguiente enlace: <https://github.com/esp8266/arduino> .

La Figura B64 muestra la ventana una vez abierto el enlace. Se debe copiar el URL

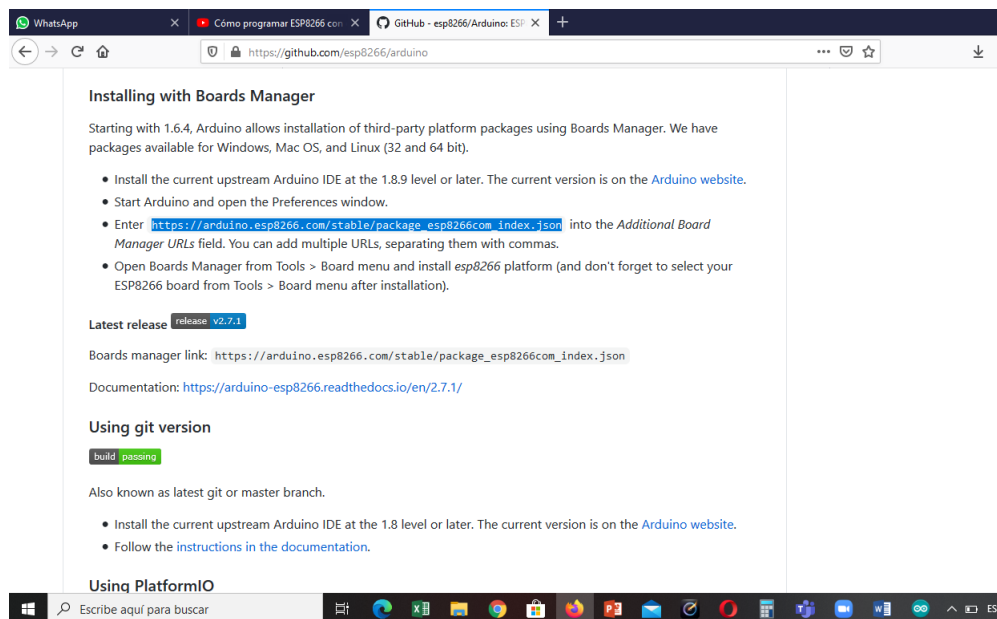


Figura B64: Repositorio *Github*

2. Una vez copiado el enlace, se debe ir al programa del módulo *Wi-Fi*. Abrir la pestaña archivo y seleccionar la opción de preferencias, como se indica en la Figura B65.

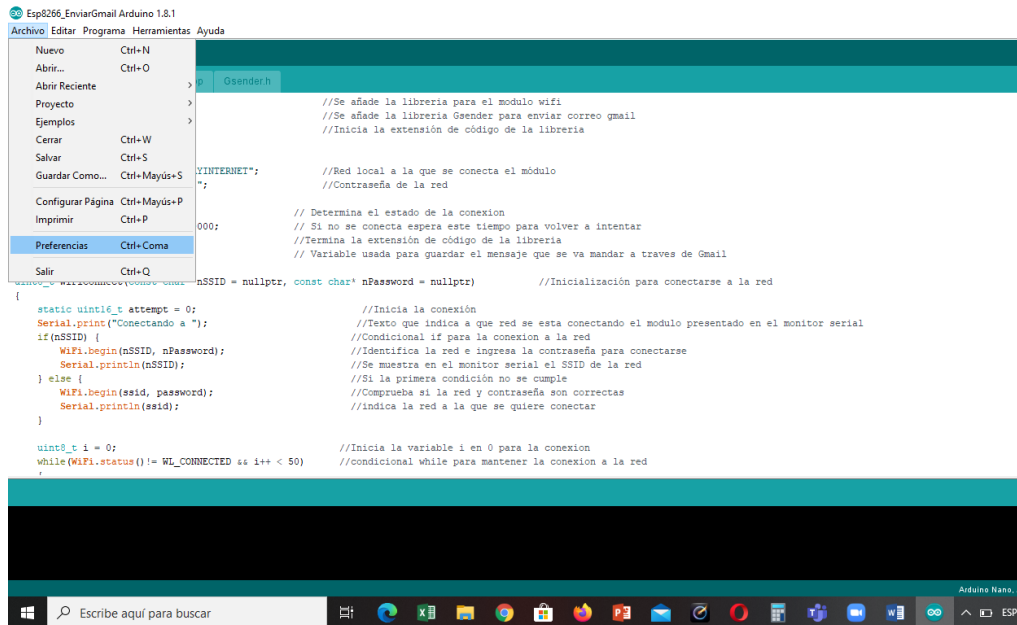


Figura B65: Añadir tarjeta ESP8266 NodeMCU V3 al Arduino IDE

3. La Figura B66 muestra la ventana que se despliega cuando se selecciona la opción preferencias. En la opción de gestor de *url* de tarjetas, se debe pegar el enlace que fue copiado en el primer paso. Dar clic en ok.

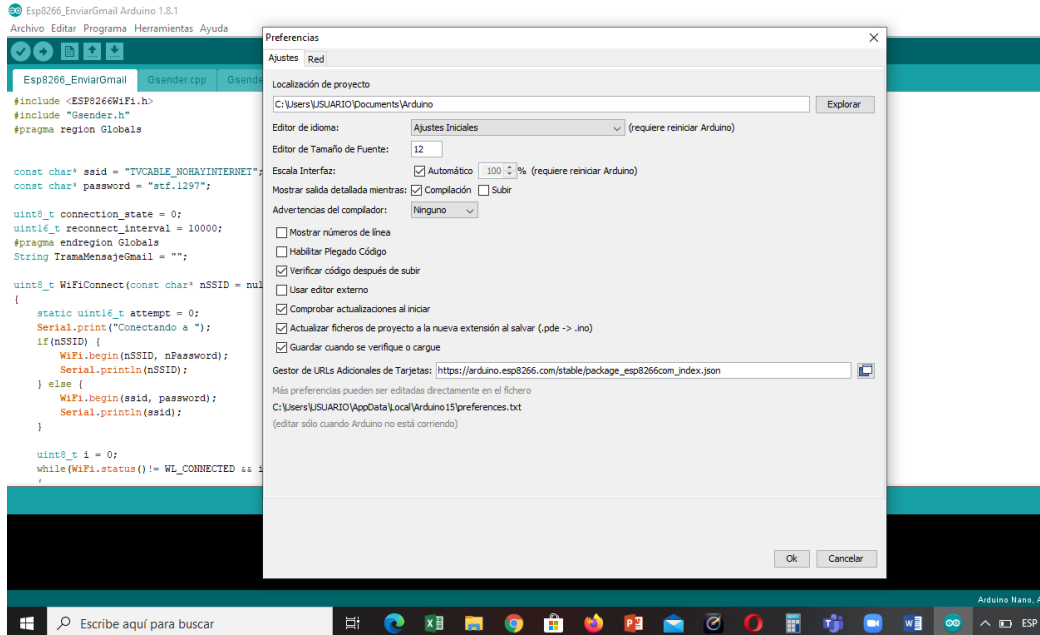


Figura B66: Gestor de preferencias Arduino IDE

- Como siguiente paso, se debe abrir la pestaña herramientas y en ella seleccionar la opción placas -> gestor de tarjetas. Figura B67.

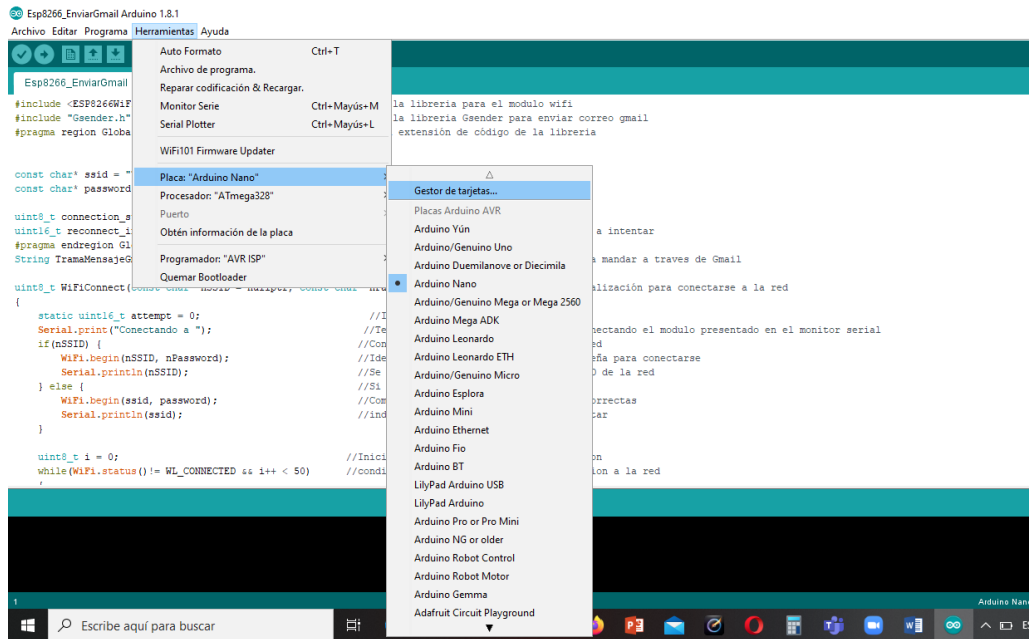


Figura B67: Gestor de tarjetas en Arduino IDE

- Como se indica en la Figura B68, la ventana del gestor de tarjetas permite añadir la placa para configurar el módulo ESP8266 *NodeMCU* v3. En el buscador, se debe colocar el nombre de la tarjeta ESP8266; una vez encontrada, se debe elegir e instalar la versión 2.4.2, debido a que es compatible con la librería *Gsender*.

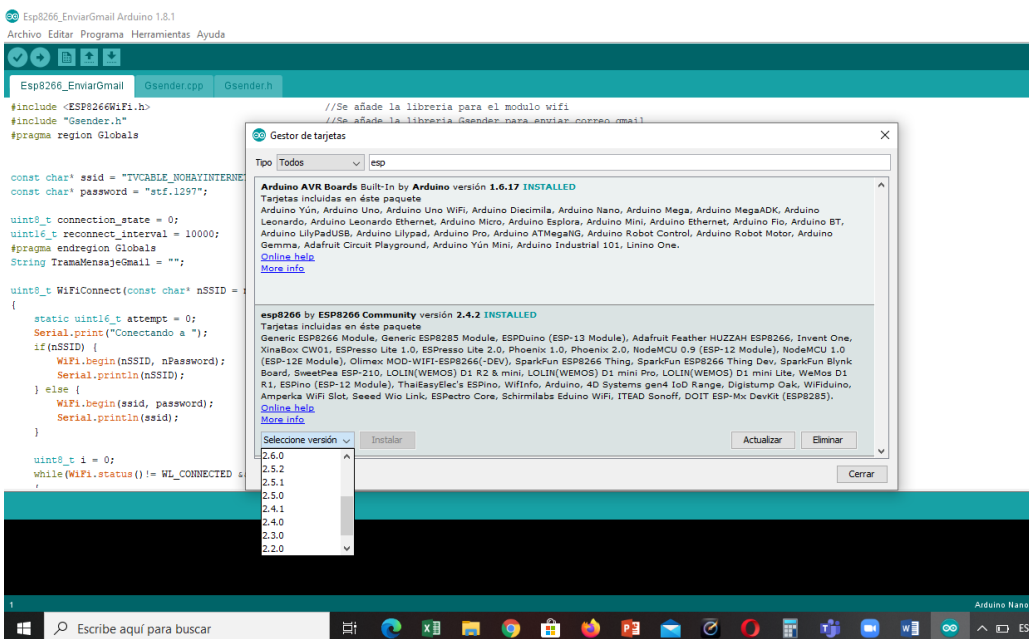


Figura B68: Instalación de librería ESP8266

- Si la placa fue instalada correctamente, se debe abrir nuevamente la pestaña herramientas y seleccionar la placa *Generic ESP8266 Module*; se debe configurar todas las opciones, como se muestra en la Figura B69.

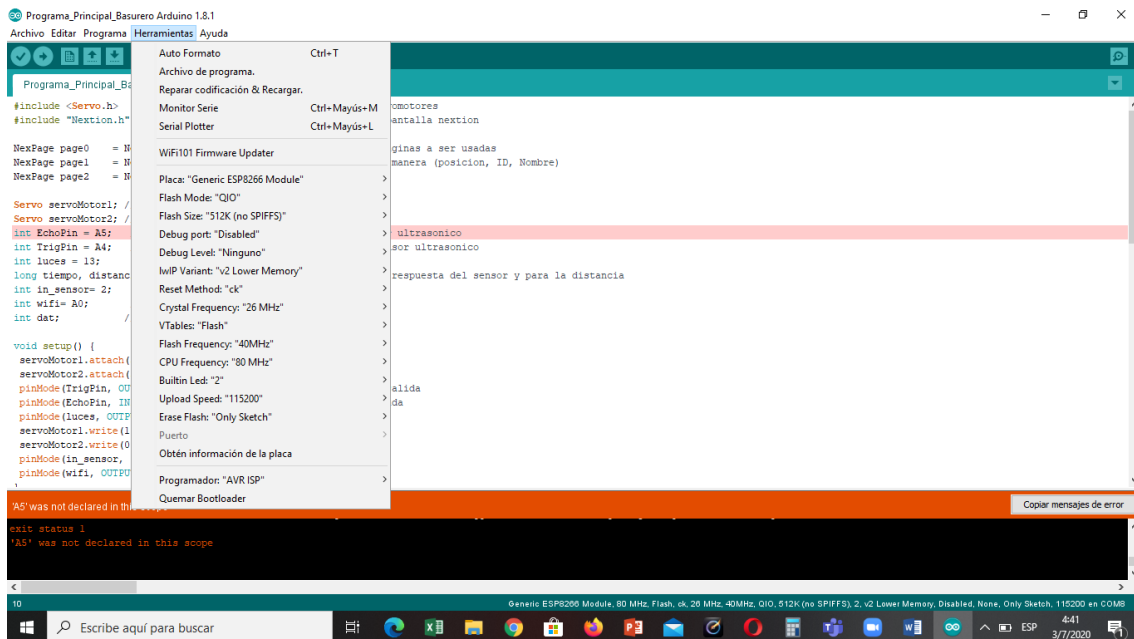


Figura B69: Configuración de la tarjeta ESP8266 en *Arduino IDE*

Una vez realizados todos los pasos, la tarjeta está lista para ser programada.

Cambiar de red local o actualizar las credenciales

En caso de que se requiera actualizar la contraseña y red local a la cual se conecta el módulo *Wi-Fi*, se debe modificar la siguiente sintaxis del programa principal:

```
#include <ESP8266WiFi.h> //Se añade la librería para el modulo wifi
#include "Gsender.h" //Se añade la librería Gsender para enviar correo gmail
#pragma region Globals //Inicia la extensión de código de la librería

const char* ssid = "T[redacted]"; //Red local a la que se conecta el módulo
const char* password = "st[redacted]"; //Contraseña de la red
```

Actualizar correo emisor

En el archivo de *Arduino* se debe cambiar a la pestaña *Gsender.h*; en ella se puede cambiar el correo electrónico al cual se accede para enviar el mensaje al administrador.

La siguiente sintaxis indica en qué líneas se debe colocar el correo y sus credenciales.

```

const int SMTP_PORT = 465; //Uso del puerto 465 de SMTP para enviar correo
const char* SMTP_SERVER = "smtp.gmail.com"; //Conexion al servidor SMTP
const char* EMAILBASE64_LOGIN = "YWxlamF [redacted] aWwuY29t"; //email en base 64 de donde se enviara el correo electronico
const char* EMAILBASE64_PASSWORD = "cm [redacted] j=="; //contraseña en base 64 del correo
const char* FROM = "a [redacted] .com"; //email escrito normal
const char* _error = nullptr; //comprobacion de acceso

```

Se debe colocar el correo y contraseña en código base 64 en las líneas que lo indican porque es un requerimiento de la librería. Para ello se puede ingresar desde cualquier navegador al siguiente enlace: <https://www.base64encode.org/>

La Figura B70 muestra el sitio *web* que permite la conversión a código base 64 de las credenciales de acceso a la cuenta de correo electrónico, se debe realizar el mismo procedimiento para la contraseña. La cuenta debe ser *Gmail*.

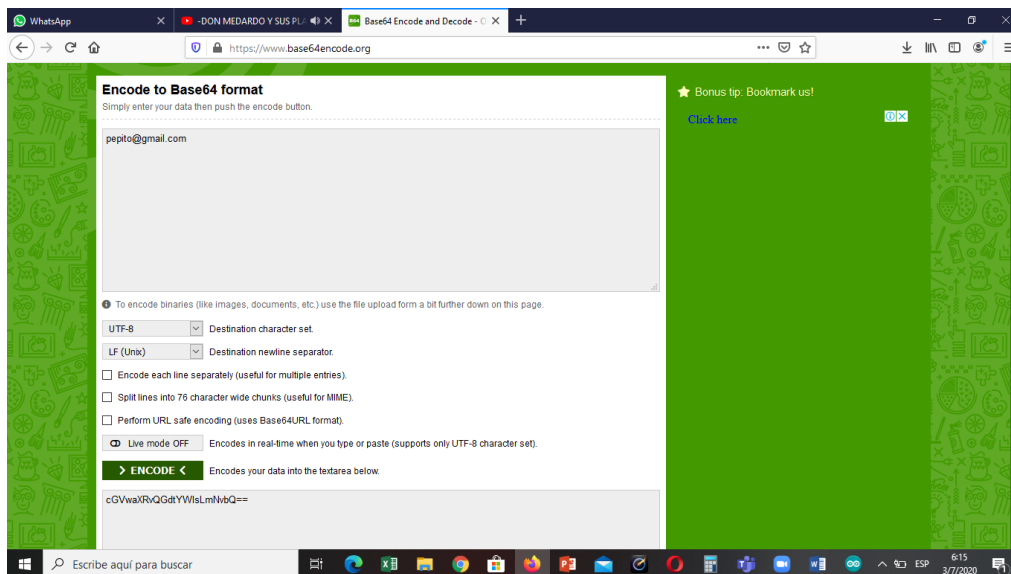


Figura B70: Página para transformar a base64

Es necesario activar el acceso de aplicaciones poco seguras para establecer la conexión con la cuenta *Gmail*, esto se encuentra en la opción de seguridad de la cuenta.

Las Figuras B71 y B72 muestran el procedimiento para realizar la acción.

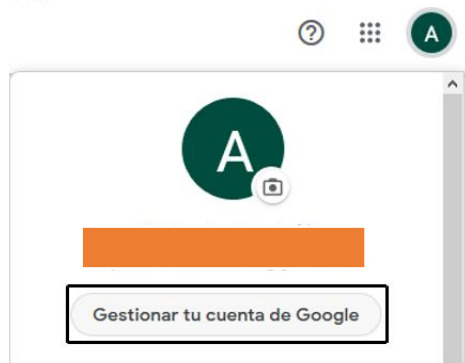


Figura B71: Configuración en la cuenta de *Gmail*

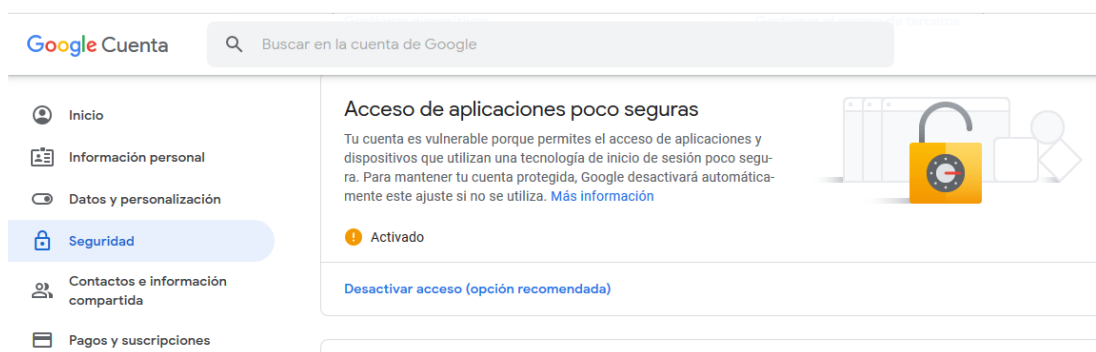


Figura B72: Acceso del *NodeMCU V3* a la cuenta de *Gmail*

Ingresar un nuevo receptor

En la pestaña `ESP8266_EnviarGmail` se encuentra la siguiente sintaxis. Únicamente se debe cambiar el correo electrónico en caso de ser necesario.

```
if(gsender->Subject(subject)->Send("[REDACTED].com", TramaMensajeGmail)) {
```

Para guardar los cambios, se debe dar clic en el botón verificar como se muestra en la Figura B73.

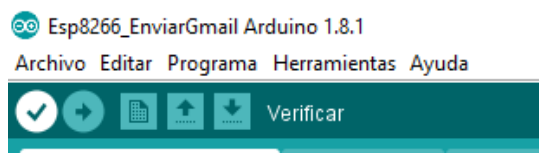


Figura B73: Guardado de cambios *Arduino IDE*

El procedimiento para cargar el programa al módulo *Wi-Fi* es similar al realizado con la placa *Arduino Nano*, se debe dar clic en subir como se muestra en la Figura B74.

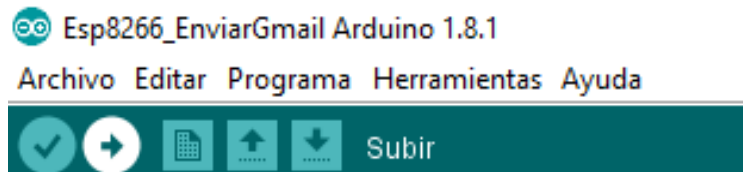


Figura B74: Cargar programa en *Arduino IDE*

Como dato importante, cuando se esté subiendo el programa, se debe presionar por 5 segundos el botón *flash* (Figura B75); esta acción permite que el programa se grabe de manera correcta

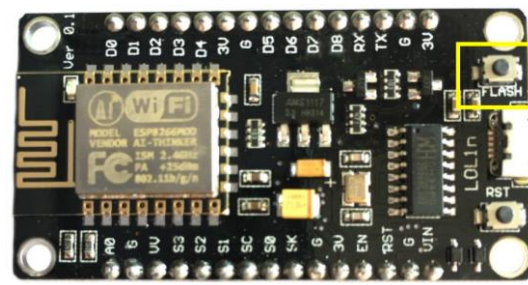


Figura B75: ESP8266 *NodeMCU V3*