Piñeiro, Juan Ignacio

Diseño de un laboratorio remoto para la enseñanza de programación de sistemas embebidos

2022

Instituto: Ingeniería y Agronomía

Carrera: Ingeniería en Informática

Universidad Nacional Arturo Jauretche

Instituto de Ingeniería y Agronomía

PRÁCTICA PROFESIONAL SUPERVISADA

Informe final

Diseño de un laboratorio remoto para la enseñanza de programación de sistemas embebidos

Estudiante:

Juan Ignacio Piñeiro

Fecha:

13-12-2022

ESTUDIANTE

Nombre y Apellido: Piñeiro, Juan Ignacio

Correo electrónico: juanignacio.ing7@gmail.com

ORGANIZACIÓN DONDE SE REALIZA LA PRÁCTICA PROFESIONAL SUPERVISADA

Nombre de la Institución: Universidad Nacional Arturo Jauretche

Dirección: Av. Calchaquí 6200, Florencio Varela, (1888) Buenos Aires, Argentina

Teléfono: +54 11 4275 6100

Sector: Programa Tecnologías de la Información y la Comunicación (TIC) en aplicaciones de interés social, Instituto de Ingeniería y Agronomía

TUTOR ORGANIZACIONAL

Nombre y apellido: Ing. Osio, Jorge.

Correo electrónico: josio@unaj.edu.ar

DOCENTE SUPERVISOR

Nombre y apellido: Ing. Osio, Jorge.

Correo electrónico: josio@unaj.edu.ar

Nombre y apellido: Ing. Ayala María Florencia.

Correo electrónico: mayala@unaj.edu.ar

DOCENTE TUTOR DEL TALLER DE APOYO PARA LA PRODUCCIÓN DE TEXTOS ACADÉMICOS

Nombre y apellido: Prof. Lía Lavigna

Correo electrónico: llavigna@unaj.edu.ar

COORDINADOR DE LA CARRERA DE INGENIERÍA EN INFORMÁTICA

Nombre y apellido: Dr. Ing. Morales, Martín

Correo electrónico: martin.morales@unaj.edu.ar

1. Resumen

La presente Práctica Profesional Supervisada (PPS) fue realizada con motivo de favorecer al alumno, a través de un laboratorio en un entorno virtual, la enseñanza y el aprendizaje en la programación de aplicaciones en sistemas embebidos. Además, que el alumno pudiera acceder de manera simple y remota al laboratorio desde cualquier dispositivo y lugar donde se encuentre.

El propósito de la práctica fue llevar adelante una investigación para crear un sistema. Por un lado, investigar y aprender todo lo que rodea a un laboratorio remoto: la educación a distancia, los sistemas embebidos, los microcontroladores y la plataforma Arduino. Por otro lado, crear un sistema que fuera simple para el usuario, donde pudiera utilizar el laboratorio remoto con sus determinadas funcionalidades.

La metodología que se tomó en el trabajo fue de tipo incremental. Esto condujo que al finalizar cada etapa (por tiempo) se fueron agregando nuevas funcionalidades y así sucesivamente hasta el desarrollo completo.

Los resultados que se obtuvieron fueron satisfactorios y se relacionaron con los programas subidos (desde la web) a la placa de Arduino, y que interactuaron con módulos como los LEDS y LCD. Por otro lado, el streaming y el Sistema de Gestión de Turnos también funcionaron correctamente.

Al finalizar con el proceso de investigación y el desarrollo de la aplicación, los objetivos generales y específicos fueron alcanzados creando un sistema que cumpliera con la función esperada.

Palabras clave: Sistemas embebidos - Microcontroladores - Arduino - LED - LCD

1.1 Abstract

This Supervised Professional Practice (PPS) was carried out with the purpose of favoring the student, through a laboratory in a virtual environment, teaching and learning in the programming of applications in embedded systems. In addition, that the student could easily and remotely access the laboratory from any device and wherever they are.

The purpose of the practice was to conduct research to create a system. On the one hand, research and to learn everything that surrounds a remote laboratory: distance education, embedded systems, microcontrollers and the Arduino platform. On the other hand, creating a system that was simple for the user, where he could use the remote laboratory with certain functionalities.

The methodology used in the work was incremental. This led to the fact that at the end of each stage (by time) new functionalities were added and so on until complete development.

The results obtained were satisfactory and were related to the programs uploaded (from the web) to the Arduino board, and which interacted with modules such as LEDS and LCD. On the other hand, streaming and the Shift Management System also worked correctly.

At the end of the research process and the development of the application, the general and specific objectives were achieved by creating a system that fulfilled the expected function.

Keywords: Embedded systems - Microcontrollers - Arduino - LEDS - LCD

2. Dedicatorias y agradecimientos

Agradezco a la Universidad Nacional Arturo Jauretche, así como también a los compañeros y docentes que tuve en cada materia cursada. Además, a mis tutores Ing. Jorge Osio e Ing. María Florencia Ayala que siempre estuvieron con la mejor predisposición para ayudarme con el desarrollo de la Práctica Profesional Supervisada (PPS), al Coordinador de la carrera Ingeniería en Informática Dr. Ing. Martín Morales y a la tutora TAPTA Prof. Lía Lavigna.

Dedico la PPS a mi familia, en especial a mi madre que siempre estuvo acompañándome a lo largo de todo el trayecto de la carrera brindándome su apoyo.

Índice

Índice de figuras

3. Introducción

La presente Práctica Profesional Supervisada (PPS) consiste en un trabajo de investigación e implementación de un Laboratorio Remoto en un Entorno Virtual de Enseñanza y Aprendizaje que favorezca la experiencia del alumno respecto al tema “Programación de aplicaciones en sistemas embebidos”, mediante el control de la placa “Arduino uno” por medio de una interfaz de usuario de acceso remoto. Además, el entorno deberá proveer turnos para evitar superposiciones en el uso del Laboratorio Remoto.

La PPS se realizará en la Universidad Nacional Arturo Jauretche (UNAJ) y bajo la tutoría de los ingenieros de la UNAJ Jorge Osio y María Florencia Ayala.

Un Laboratorio Remoto consiste en un recurso de acceso remoto de naturaleza real. Este representa el acceso a un laboratorio real, es decir, a un laboratorio con equipos reales, pero de manera no presencial, el mismo se efectiviza a través de un medio de comunicación como, por ejemplo, Internet. En estos laboratorios el usuario opera y controla en forma remota el laboratorio real ubicado en los laboratorios de informática de la UNAJ, a través de un interfaz de experimentación, desarrollada e implementada desde la máquina servidor que se configura en la UNAJ

La investigación consiste en estudiar la herramienta de gestión de aprendizaje Moodle, con un conjunto de plugins específicos (para el manejo de laboratorios remotos y de turnos) y evaluar otras posibles alternativas con características similares. También se analizará incorporar una interfaz web para el usuario, donde estarán disponibles tres ventanas: una donde se muestre mensajes en la consola serie de la placa Arduino, otra en donde se muestre la cámara y la tercera con una opción de cargar el código binario que se programara en la placa. De manera opcional, estará la ventana de control de teclado.

La parte de implementación consiste en desarrollar el código y las configuraciones necesarias para poder manejar el hardware, el streaming/cámara y los equipos (placa sobre Arduino) que se encuentran disponibles en el laboratorio de forma remota.

4. Objetivos

4.1 Objetivos generales

El objetivo principal de la PPS es desarrollar una plataforma de control y acceso al Laboratorio físico de la UNAJ, conformado por el Microcontrolador Atmega328p (de Arduino) y una placa de periféricos, desde una máquina remota. Específicamente se implementará una plataforma de acceso para los alumnos donde puedan interactuar con el Laboratorio físico y de manera remota. El usuario compilará el código remotamente y el código binario resultante se cargará en la máquina de escritorio situada en el laboratorio de informática de la UNAJ, específicamente en el kit Arduino uno, en donde se ejecutará automáticamente realizando las tareas programadas.

En la siguiente figura (figura 1), se puede observar cómo es la arquitectura general del proyecto:

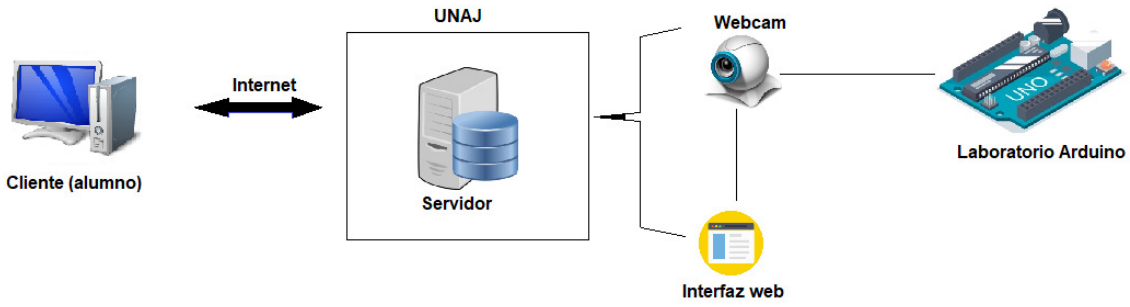


Figura 1. Diagrama del proyecto general.

Fuente: producción propia.

4.2 Objetivos específicos

Los objetivos específicos son:

- Investigar y conocer la plataforma Moodle y sus plugins
- Configurar la herramienta para reconocer el Hardware específico utilizado en el Laboratorio Real.

- Implementar un servidor web, configurar en el Moodle de la UNAJ un link que ejecute la plataforma del laboratorio, que atienda los accesos de los usuarios.
- Programar y configurar un servidor web en donde se visualizará la plataforma, con sus datos, el código binario del programa y parámetros ingresados por los alumnos.
- Resolver las solicitudes para la asignación de turnos y autenticación de los usuarios.
- Desarrollar la interfaz web que use la API REST y muestre los datos de ejecución de la aplicación cargada en la placa

5. Marco teórico

A continuación, se detallan los tópicos de estudio abordados para la realización del laboratorio remoto y su comprensión como herramienta de aprendizaje en el contexto académico.

5.1 Educación a distancia

La educación a distancia es una forma de enseñanza en la que los estudiantes no requieren asistir físicamente al lugar de estudios. En este sistema de enseñanza, el alumno recibe el material de estudio (personalmente, por correo postal, correo electrónico u otras posibilidades que ofrece Internet), permitiendo que, en el acto educativo, se empleen nuevas técnicas y estrategias de aprendizaje centradas en el propio estudiante, fomentando así el autodidactismo y la autogestión. Es decir, se trata de una forma de educación flexible, autónoma y auto-dirigida, cuyas principales herramientas son las tecnologías de la comunicación y la información. Al aprendizaje desarrollado con las nuevas tecnologías de la comunicación se le llama “aprendizaje electrónico” ("*E-learning*", en inglés) y la plataforma más utilizada, actualmente, para esta modalidad es Moodle.

5.1.1 Características de la educación a distancia

La educación a distancia se caracteriza por la flexibilidad de sus horarios, pues el mismo estudiante organiza su tiempo de estudio, lo cual requiere cierto grado de autodisciplina. Esta

flexibilidad de horarios a veces está limitada en ciertos cursos que exigen participación en línea en horarios o espacios específicos.

Otra característica de la educación a distancia es el uso de las Tecnologías de la Información y la Comunicación (TIC) para formar comunidades o redes de estudio donde los individuos pueden interactuar, fomentando el uso educativo de las redes sociales, foros de discusión y plataformas virtuales, para discutir sobre diversos temas y , a la vez, adquirir conocimientos y modernas herramientas de trabajo. También es imprescindible tener una nueva visión de los roles que desempeñan los maestros y los estudiantes en esta modalidad de estudio, el maestro deja de ser el protagonista, convirtiéndose en un facilitador del proceso educativo y le cede el paso al estudiante, el cual debe tener un compromiso firme con su propio proceso de formación.

5.1.2 Beneficios de la educación a distancia

A continuación, se enumeran los beneficios que ofrece la educación a distancia:

- La educación a distancia satisface las necesidades de los estudiantes que, de otro modo, no podrían asistir a clases presenciales, debido a las restricciones de distancia o de tiempo. Uno de los mayores beneficios de la educación a distancia es pues la flexibilidad.
- Los Programas de Educación a distancia permiten un mayor acceso al aprendizaje y lo fomentan de forma permanente. Además, permite a los estudiantes elegir entre un conjunto más amplio de instituciones académicas para su aprendizaje continuo.
- Otro beneficio de la educación a distancia para los estudiantes es la capacidad de hacer el trabajo en equipo en grupos interactivos. Los estudiantes tienen la oportunidad de comunicarse con otras personas de diferentes orígenes y escuchar a una gran variedad de expertos de todo el mundo.
- Elimina las barreras geográficas, debido a que la población puede acceder a este tipo de educación independientemente del lugar donde resida.
- Reduce costos al evitar gastos de traslados o residencia en un lugar diferente.
- Incorpora herramientas tecnológicas para el manejo de la información, las cuales son necesarias para desempeñarse profesionalmente en la sociedad en constante cambio, tales como las plataformas virtuales.

- El alumno desarrolla una alta capacidad para autorregular su propio aprendizaje favoreciendo así sus actitudes y valores de responsabilidad, disciplina y compromiso para lograr ser autónomo.
- El rol del estudiante es activo pues desarrolla estrategias intelectuales importantes para realizar tareas colaborativas, comunicarse efectivamente, ser creativo e innovador.
- El asesor lleva un seguimiento riguroso del estudiante empleando diversos instrumentos para evaluarlo respetando el ritmo de trabajo del alumno.
- Facilita a las personas, con capacidades diferentes, el acceso a cursar una carrera.
- Al término de la carrera los títulos poseen la misma validez, que aquel que cursa de manera presencial.
- Al término de la carrera los títulos poseen la misma validez, que aquel que cursa de manera presencial.
- Debido a su comodidad, el alumno tiene la capacidad de manejar el tiempo dedicado a cada actividad de acuerdo a sus otras actividades o a la rapidez con la que avance permitiendo ser flexible el tiempo de término de la carrera, dotándole de más independencia al construir su conocimiento.

5.1.3 Generaciones de la educación a distancia

Para entender mejor como ha sido la evolución de la educación a distancia, esta puede ser dividida en etapas o generaciones, ya que se trata de un conjunto de eventos cronológicos que se acompañan, se correlacionan y se diferencian de manera óptima, debido a los cambios y desarrollos de nuevas tecnologías.

El siguiente diagrama (Figura 2) presenta las cuatro generaciones de la educación a distancia, de acuerdo con su progresión temporal, presentando, asimismo, los elementos que caracterizan el tipo de comunicación a distancia que cada una de ellas utilizó centralmente:

Figura 2. Evolución de la educación a distancia.
Fuente: https://tecnologicoedupraxis.edu.ec/wp-content/uploads/2021/03/Evolucion-de-la-educacion-a-distancia.png

A continuación, se explicará la conformación de cada una de ellas, para dar cuenta de manera fehaciente los objetivos de la presente investigación.

5.1.3.1 El modelo de correspondencia 1950 – 1960

Se caracterizó por el desarrollo y predominio de materiales impresos, textos y manuales que eran distribuidos a través del correo postal. En esta generación la educación a distancia era un sistema básicamente cerrado y unidireccional centrado en el material didáctico. Los cursos se caracterizan por el uso predominante de una sola tecnología (material escrito vía postal), por lo tanto, no se tomaba en cuenta ninguna forma de apoyo al estudiante más allá del material impreso.

5.1.3.2 El modelo multimedia 1960 – 1985

Esta generación se caracteriza porque comienzan a utilizarse recursos audiovisuales (audiocasetes, diapositivas, videocasetes, etc.). El teléfono se incorpora a las acciones en este ámbito, para conectar al tutor con los estudiantes. Por primera vez, los materiales impresos comienzan a ser elaborados en forma de módulos de aprendizaje y comienza a hacerse importante la autoevaluación por parte de los estudiantes.

5.1.3.3 El modelo de tele-aprendizaje 1985 - 1995

Esta generación también recibe el nombre de digital, pues en ella se integran los recursos audiovisuales, las tecnologías de la informática y las comunicaciones (telemática). Se caracteriza porque los medios empleados permiten la interactividad del estudiante con los materiales didácticos y los profesores.

5.1.3.4 La enseñanza de aprendizaje flexible mediante E-Learning 1995 – 2005

La cuarta generación cuenta con redes de comunicación y las estaciones multimedia, que pueden enviar y recibir señales electrónicamente. El uso de la Internet ha cambiado el ritmo de las actividades en educación a distancia al permitir intercambios de información en muy corto tiempo, favorece una mayor oportunidad de interacción que sobrepasa la mera adquisición de información, con la promoción de estudiantes más activos y participativos.

Por otra parte, a mediados del año 2000 nace el concepto de B-learning que es la abreviatura de Blended Learning, término en inglés que se traduce como “Formación combinada” o “Enseñanza Mixta”. Se trata de una modalidad semi-presencial de estudios, que incluye tanto formación no presencial (cursos on-line, conocidos genéricamente como e-learning) como formación presencial.

5.1.3.4 Modelo de aprendizaje flexible inteligente 2005 – Actualidad

Este modelo no se encuentra en el diagrama porque aún está en desarrollo, pero la quinta generación se ha caracterizado por la aplicación de sistemas inteligentes de respuesta, que permiten hacer más efectivos los sistemas de tutoría y favorecer economías de escala y costo de efectividad asociados a ellas en grupos numerosos de estudiantes. Esta generación se encuentra en pleno desarrollo.

Por otra parte, en el año 2006 nace el concepto de M-Learning, que es la abreviatura de Mobile E-learning, término en inglés que en la enseñanza virtual se traduce como “Aprendizaje Electrónico Móvil”. También, es una evolución de E-learning, que proviene de la formación a distancia y está asociada, además, al uso de tecnologías móviles en la educación. Al mismo tiempo, es una metodología de enseñanza y aprendizaje que se basa en el uso pedagógico a través de pequeños dispositivos móviles como laptops, teléfonos móviles, tablets y todo dispositivo de mano que tenga alguna forma de conectividad inalámbrica.

5.2 Laboratorios remotos

Los Laboratorios Remotos (LR), son herramientas tecnológicas compuestas por software y hardware que les permite a los estudiantes, de manera remota, realizar sus prácticas como si estuvieran en un laboratorio tradicional, generalmente el acceso se realiza a través de Internet o mediante una red académica.

Los LR son normalmente utilizados cuando se requiere que los estudiantes pongan en práctica de manera autónoma lo aprendido las veces que lo requieran y esto les permite confrontar sus conocimientos previos con los nuevos conocimientos adquiridos hasta llegar a construir uno nuevo. Se podría decir, entonces, que en este nuevo escenario prevalece el modelo constructivista que les permite a los estudiantes adquirir aprendizajes significativos para la vida. Los LR son producto del desarrollo de las Tecnologías de la información y la comunicación (TIC) que contribuyen con el mejoramiento de los procesos educativos. Además, son considerados como sistemas que no están basados en prácticas simuladas, si no en acciones que permiten al estudiante realizar actividades de laboratorio con dispositivos o instrumentación real, transfiriendo información entre el procedimiento práctico y el estudiante. De este modo, el alumno teleopera y controla los recursos disponibles en el laboratorio.

El diseño y desarrollo de los LR está basado en un enfoque en la web, buscando utilizar Internet como un marco genérico de tecnología estándar ampliamente empleado a nivel mundial, para que el desarrollo sea adaptable a diversas situaciones. Dicho enfoque puede ilustrarse en la Figura 3, donde se muestran diferentes elementos involucrados en los LR.

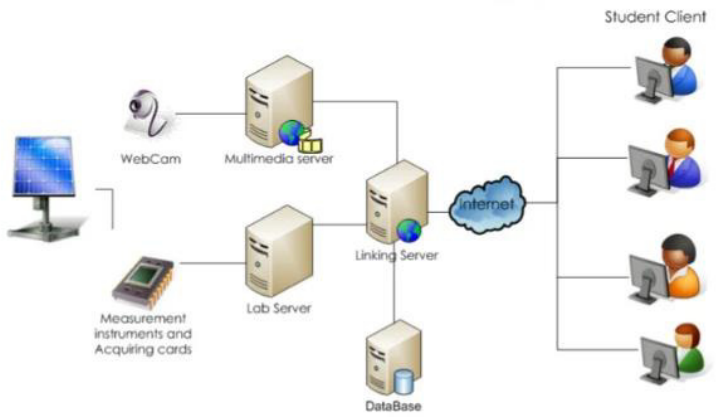


Figura 3. Elementos de los LR.

Fuente: https://es.slideshare.net/emadridnet/emadrid-2015-20-02-uned-rafael-pastor-vargas-desarrolloexplotacin-de-entornos-experimentales-basados-en-laboratorios-remotos-y-virtuales-related-44979219

En la Figura 3 se muestran los elementos de un laboratorio remoto:

- Alumno, cliente o usuario: representa al individuo que va a efectuar el laboratorio remoto mediante una computadora. La computadora está conectada a internet y mediante un navegador web establecerá una conexión con el servidor que gestiona el servicio de LR.
- Servidor de enlace: es el que gestiona y administra las conexiones de los usuarios que quieren efectuar el LR.
- Servidor multimedia: es el que provee al servidor de enlace las imágenes y el audio en tiempo real del ambiente experimental.
- Servidor de laboratorio: es el que da soporte a una aplicación en particular, en este caso, una placa de adquisición e instrumentación y está conectado al servidor de enlace.
- Objetivo experimental: objeto que está bajo el control del servidor de laboratorio y que permite adquirir o efectuar accionamientos, que se pueden medir, observar o escuchar gracias al servidor multimedia.

5.3 Microcontrolador

Un microcontrolador es un Circuito Integrado (CI) programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales que cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora:

- Un microprocesador, también denominado unidad central de procesos o CPU
- Memoria de programa y memoria de datos
- Unidades de entrada/salida

El microcontrolador resulta equivalente a un pequeño computador, que con muy poca circuitería adicional se puede usar para controlar equipos electrónicos, el diseño de sistemas de comunicación, monitoreo y adquisición de señales físicas, procesamiento y administración de señales analógicas y digitales.

En la Figura 4 se observa un esquema en bloques representativo de las funciones y componentes integrados en el chip de un microcontrolador.

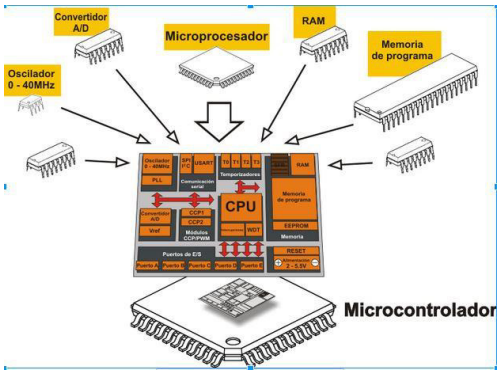


Figura 4. Componentes de un microcontrolador.

Fuente: https://aprendiendoarduino.files.wordpress.com/2014/11/microcontrolador.jpg?w=625

5.4 Sistemas embebidos

Los sistemas embebidos, también conocidos como computadoras integradas, son computadoras de factor de forma pequeño que impulsan tareas específicas. Estos pueden funcionar como dispositivos independientes o como parte de sistemas más grandes, de ahí el término "incrustado" y, a menudo, se usan en aplicaciones con restricciones de tamaño, peso, energía y costo.

Como la mayoría de las computadoras, los sistemas embebidos son una combinación de hardware y software y por lo general contienen:

- Microprocesadores o microcontroladores
- Unidades de procesamiento de gráficos (GPU)
- Memoria volátil y no volátil
- Puertos e interfaces de comunicación de entrada/salida
- Código del sistema y de la aplicación
- Fuentes de alimentación

5.4.1 Tipos de sistemas embebidos

Los sistemas embebidos se clasifican según el rendimiento y los requisitos funcionales, así como el rendimiento de los microcontroladores. Estas clasificaciones se pueden dividir en categorías y subcategorías

1) Sistemas embebidos en tiempo real:

   Los sistemas embebidos en tiempo real deben proporcionar resultados o salidas con prontitud. Se asigna prioridad a la velocidad de generación de salida, ya que estos sistemas a menudo se usan en sectores de misión crítica, como defensa y aeroespacial, que necesitan datos importantes.

   Los sistemas embebidos en tiempo real se dividen, además, en sistemas embebidos suaves en tiempo real y sistemas embebidos duros en tiempo real. Por un lado, los sistemas embebidos suaves en tiempo real tienen plazos o plazos de salida indulgentes. Por otro lado, los sistemas embebidos duros en tiempo real son la oposición de los sistemas embebidos flexibles en tiempo real. Estos sistemas deben cumplir constantemente con los plazos de salida asignados, ya que de no hacerlos se considera una falla del sistema o de la aplicación que, en muchos casos, podría tener resultados catastróficos debido a la implementación típica del sistema embebido en tiempo real en programas y aplicaciones de misión crítica.

2) Sistemas embebidos autónomos:

   Los sistemas embebidos independientes no requieren una computadora host para funcionar. Estos pueden producir salidas de forma independiente y algunos ejemplos son los siguientes: cámaras digitales, relojes de pulsera digitales, reproductores de mp3, electrodomésticos, como refrigeradores, lavadoras y hornos de microondas, sistemas de medición de temperatura y calculadoras.

3) Sistemas embebidos en red:

   Los sistemas embebidos en red se basan en redes cableadas o inalámbricas y en la comunicación con servidores web para la generación de resultados. Como ejemplo, a continuación, se citan: sistemas de seguridad para el hogar y la oficina, cajeros automáticos y sistemas de punto de venta.

4) Sistemas embebidos móviles:

Los sistemas embebidos móviles se refieren específicamente a dispositivos embebidos pequeños y portátiles como, por ejemplo: teléfonos móviles, ordenadores portátiles y calculadoras.

5.4.2 ¿Cómo funcionan los sistemas embebidos?

Los sistemas embebidos comprenden hardware y software que trabajan juntos para realizar tareas específicas. Estos se basan en microprocesadores, microcontroladores, memoria, interfaces de comunicación de entrada/salida y una fuente de alimentación para funcionar. Al igual que con prácticamente todas las computadoras, un sistema embebido emplea una placa de circuito impreso (PCB) programada con un software, que le dice a su hardware cómo operar y administrar los datos utilizando interfaces de comunicación de entrada/salida y memoria, que finalmente produce salidas valiosas para el usuario. Por lo tanto, los sistemas embebidos no son fundamentalmente diferentes de los servidores y estaciones de trabajo estándar de montaje en rack.

5.5 Plataforma Arduino

Arduino es una plataforma de código abierto utilizada para la construcción de proyectos de electrónica. Esta plataforma consta de una placa de circuito programable física (a menudo denominada microcontrolador) y una pieza de software, o IDE (Entorno de Desarrollo Integrado) que se ejecuta en su computadora y se utiliza para escribir y cargar código de computadora en la placa física.

5.5.1 Hardware Arduino

El hardware de Arduino usa microcontroladores generalmente Atmel AVR. Los más usados en las plataformas Arduino son el Atmega168, Atmega328, Atmega1280, ATmega8 por su sencillez, pero se está ampliando a microcontroladores Atmel con arquitectura ARM como el Atmel SAMD21 o los ST STM32 y también Intel. En la Figura 4 se observan las características de un microcontrolador y en la Figura 5 se presentan los distintos tipos de placas más usadas de Arduino.

Figura 5. Tipos de placa de Arduino.

Fuente: https://i.blogs.es/218ccc/formas-arduino/1366_2000.jpg

5.5.2 Software Arduino

Después de comprender el hardware de Arduino, es necesario entender el software y programación para que el Arduino cobre vida y pueda interactuar con varios módulos.

5.5.2.1 Arduino IDE

El IDE de Arduino es una aplicación multiplataforma (para Windows, macOS, Linux) que está escrita en el lenguaje de programación Java. Se utiliza para escribir y cargar programas en placas compatibles con Arduino, pero también, con la ayuda de núcleos de terceros, se puede usar con placas de desarrollo de otros proveedores.

Este programa utiliza una versión simplificada de C ++ con resaltado de sintaxis y otras características, lo que facilita el aprendizaje de la programación, por lo que es perfecto para principiantes que desean comenzar a aprender programación y codificación. En la Figura 6 se observa el entorno de desarrollo Arduino IDE.

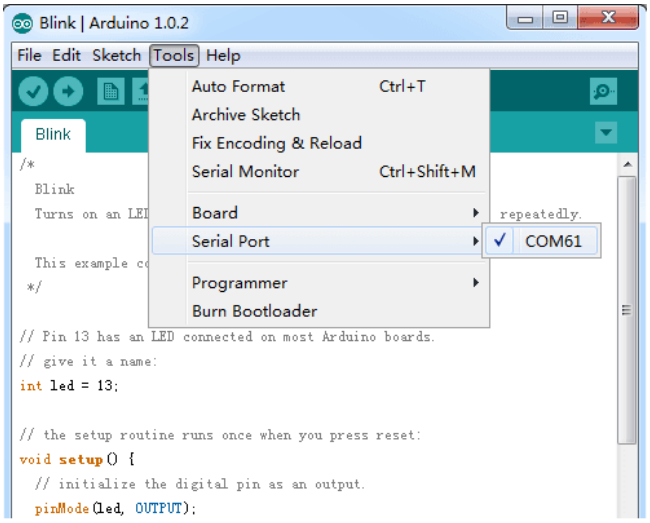


Figura 6. Entorno de desarrollo Arduino IDE.

Fuente: https://raw.githubusercontent.com/SeeedDocument/Getting_Started_with_Seeeduino/master/img/Getting_Started3.png

6. Herramientas

Este apartado tiene la finalidad de detallar cada herramienta elegida para el desarrollo del trabajo.

6.1 Hardware

Las herramientas de Hardware utilizadas fueron otorgadas al inicio del proyecto con el fin del desarrollo del mismo. Se empleó un Microcontrolador Atmega328p (de Arduino), una placa de periféricos y cámara web.

6.1.1 Kit ATmega328P (de Arduino UNO)

A continuación, se describirán las características del kit de periféricos para el ATmega328p compatible con las placas de Arduino uno y similares. En la figura 7 se presenta el kit de Arduino con los pines I/O disponibles en los conectores laterales con las distintas funcionalidades que se le pueden configurar a cada pin. En este kit los pines A0-A5 se pueden configurar como entradas analógicas o entradas/salidas digitales, según se requiera. Los pines precedidos con ~ se pueden configurar como salida PWM o entradas/salidas digitales (ejemplo

el pin ~6). En primer lugar, el resto de los pines se pueden configurar con entradas/salidas digitales. Los pines A4 y A5 (SDA y SCL respectivamente) se pueden configurar para implementar el protocolo I2C. En segundo lugar, los pines 11, 12 y 13 (MOSI, MISO y SCK respectivamente) se pueden configurar para implementar el protocolo SPI. Por último, los pines 0 y 1 (RXD y TXD respectivamente) se pueden configurar para implementar el protocolo UART (serie asincrónico).

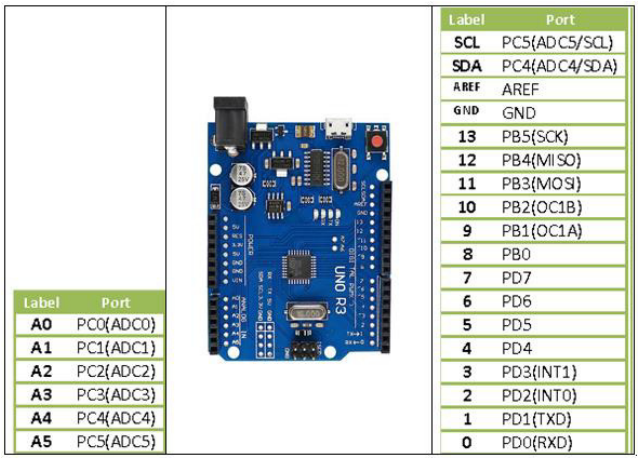


Figura 7. Pinout del Arduino UNO

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

6.1.1.1 Descripción del kit de periféricos del ATmega328p

El kit de periféricos ATmega328p fue diseñado con la idea de disponer de un kit genérico con suficientes periféricos para realizar las actividades prácticas en las diferentes materias del área de digitales de la carrera.

En la figura 8 se muestra el kit de periféricos visto de arriba, en donde se pueden apreciar una serie de conectores y componentes que conforman los periféricos. Además, se dispone de 9 jumpers que se incluyeron para compartir algunos pines I/O entre los periféricos.

Entre los periféricos principales contemplados se tiene un display LCD (puede ser de 2x16 mediante el conector LCD_2x16 o de 2x8 mediante el conector LCD_2x8), un teclado matricial de 4x4, un sensor de iluminación LDR, un potenciómetro, un led RGB ánodo común, un sensor de temperatura y humedad DHT11/DHT22 y un parlante.

Respecto a los periféricos secundarios, se provee conexión para un acelerómetro, un reloj de tiempo real, un módulo bluetooth y se dejó disponible un conector UART para conectar

cualquier dispositivo con interfaz UART (por ejemplo, módulo GSM/GPRS, módulo Wifi, módulo inalámbrico RF, etc.).

Se debe aclarar que se compartieron pines entre periféricos que tienen similares funcionalidades, principalmente de interfaz de usuario. Por ejemplo, el usuario puede interactuar con el kit por medio del teclado matricial y display LCD, por medio de leds indicadores o por medio del módulo bluetooth.

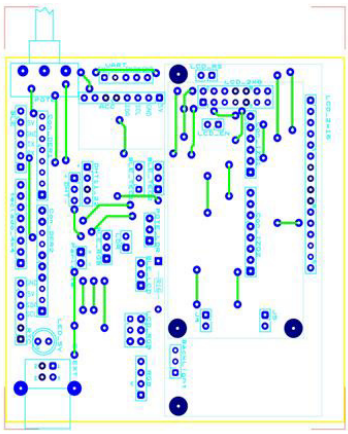


Figura 8. Vista de arriba del kit de los periféricos.
Fuente: Figura extraída del Manual de Usuario kit ATmega328P

6.1.1.2 Conexión de los Jumpers

Los jumpers simplemente permiten seleccionar si el pin correspondiente se conectará a un periférico o a otro. La conexión de estos es muy sencilla, observando la placa desde arriba, si se desea utilizar un periférico se debe conectar cada uno de los jumpers para el lado del periférico deseado. Por ejemplo, para seleccionar el display LCD se deben conectar los jumpers del conector LCD_RGB para el lado LCD (lado izquierdo), los del conector BLE_LCD para el lado LCD (lado derecho) y deben estar conectados los jumpers LCD_EN y LCD_RS.

El jumper backlight permite seleccionar si la alimentación de este se obtendrá del kit Arduino uno o del USB de la placa de periféricos (VCC_EXT.)

6.1.1.3 Esquemático kit

En esta sección se muestra a qué pines del kit Arduino está conectado cada uno de los periféricos. Se debe aclarar que en el esquemático las líneas que tengan el mismo nombre están físicamente conectadas.

En la figura 9 se pueden observar los jumpers y a qué pin del kit Arduino se conecta cada uno. Además, se muestran los conectores derechos e izquierdos del kit Arduino para entender mejor las conexiones.

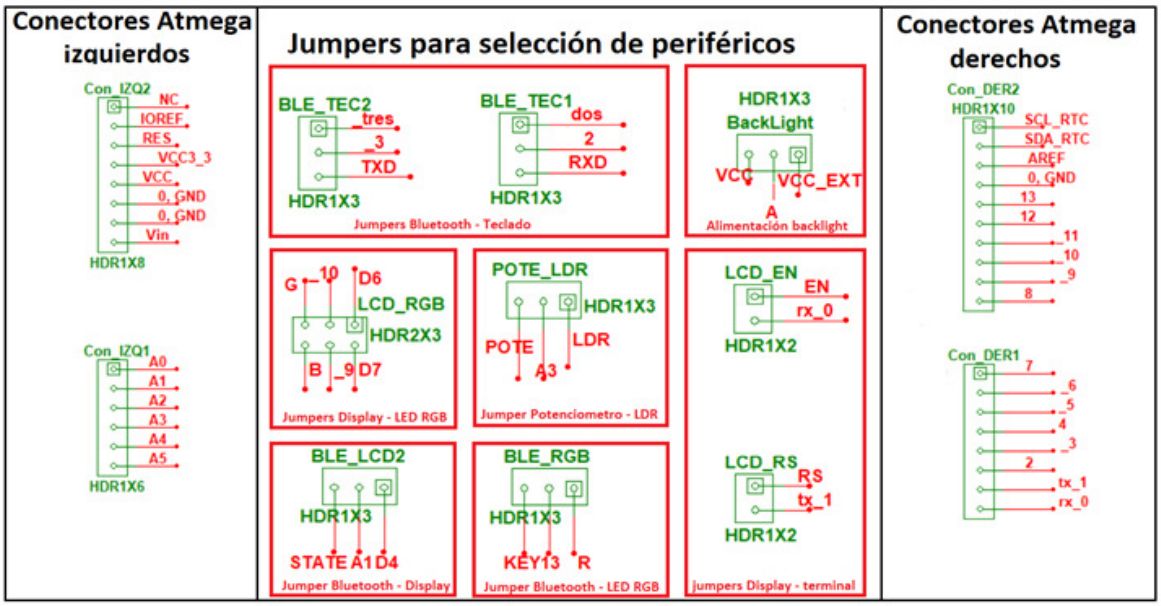


Figura 9. Esquemático con los jumpers y los conectores del kit Arduino.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

En la figura 10 se muestran los periféricos principales, en donde se puede observar que:

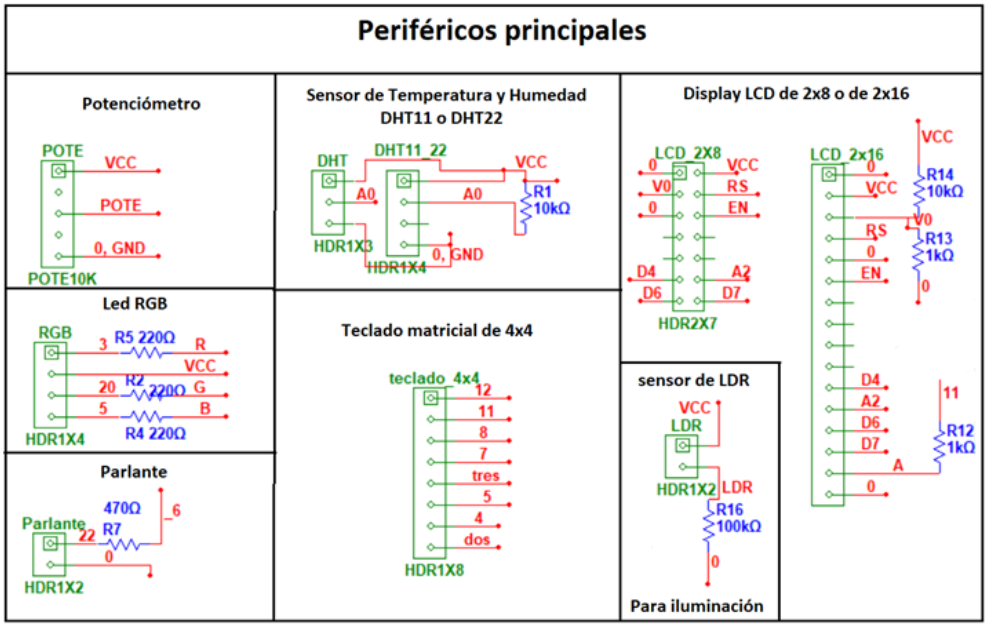


Figura 10. Esquemático con los periféricos principales.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

- Tanto el potenciómetro como el sensor LDR se conectan al pin A3 (PC3) por medio del jumper POTE_LDR (ver figura 11).

Figura 11. Dispositivos de entrada analógica A3.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

- El teclado matricial de la figura 12 está conectado a los pines 12 (PB4), 11(PB3), 8 (PB0), 7(PD7), 3(PD3), 5(PD5), 4(PD4), 2(PD2). El pin 3 se comparte mediante el jumper BLE_TEC2 y el pin 2 mediante BLE_TEC1.

Figura 12. Teclado de membrana.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

- Sensor de Temperatura y humedad (DHT11/22) al pin A0 (PC0), como se observa en la figura 13

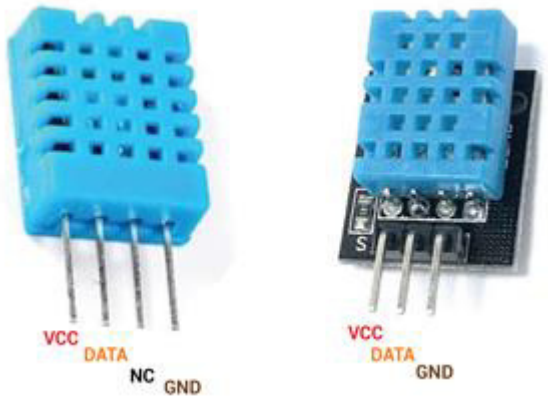


Figura 13. Sensor de temperatura y humedad DHT11.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

- El parlante o buzzer de la figura 14, al pin 6 (PD6). Se debe aclarar que es importante que el buzzer sea de 5V, ya que los de computadora son de 12V y prácticamente no emiten sonido a 5V.

Figura 14. Buzzer de 5V.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

- El Led RGB a los pines 13 (PB5), 10 (PB2) y 9 (PB1) colores rojo, verde y azul respectivamente. Los pines 9 y 10 se comparten por medio del jumper LCD_RGB y el pin 13 mediante el BLE_RGB. Se debe aclarar que al ser ánodo común cada color se enciende cuando el pin correspondiente se pone en nivel bajo (0 lógico), como se observa en la figura 15.

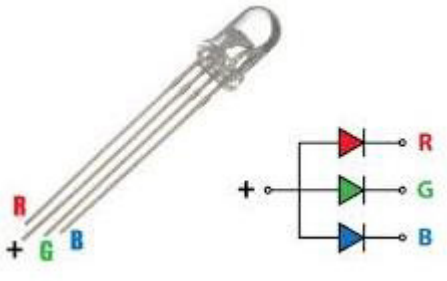


Figura 15. Led ánodo común.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

- El display LCD se conecta a los pines 0(PD0) el EN, 1(PD1) el RS, A1(PC1) el D4, A2(PC2) el D5, 10(PB2) el D6, 9(PB1) el D7. Se debe aclarar que las conexiones del

display están hechas para manejarlo con 4 líneas de datos y 2 de control, ya que el pin RW se conecta a tierra (solo escritura). Ver figura 16.

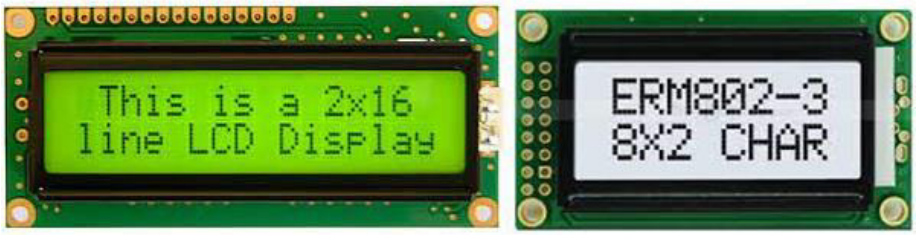


Figura 16. Display de 2x16 y de 2x8.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

En la figura 17 se muestran los periféricos secundarios, en esta se puede observar que:

- El RTC (figura 17 (a)) comparte los pines de datos con el acelerómetro (figura 17(b)) mediante el protocolo I2C. El pin SDA está conectado al A4(PC4) y el SCL al A5(PC5). La diferencia entre estos periféricos es la dirección que tienen asignada, ya que en el bus I2C se pueden conectar múltiples dispositivos siempre y cuando tengan asignadas direcciones distintas. Cabe aclarar que el RTC tiene la dirección fija de 0x68 y no se puede modificar y el acelerómetro permite seleccionar entre dos direcciones 0x68(por defecto) y 0x69 (poniendo el pin AD0 en alto). En el kit, el acelerómetro tiene asignada la 0x69 para no tener conflictos en el bus I2C con el RTC.
- El módulo bluetooth (ver figura 18) tiene el pin STATE conectado al pin A1(PC1), RXD-bluetooth al pin 2(PD2) TX-ATmega, TXD-bluetooth al pin 3(PD3) RX-ATmega, pin KEY al 13(PB15)
- El conector UART contiene el pin 0 (PD0/RXD) y 1(PD1/TXD) del ATmega

En síntesis, tanto la alimentación del módulo bluetooth como del conector UART provienen del conector USB disponible en el kit de periféricos (VCC_EXT). Teniendo en cuenta que los módulos inalámbricos tienen picos de consumo que superan la potencia entregada por el kit Arduino, se optó por obtener la alimentación de estos directamente de la PC o de cualquier power_bank USB.

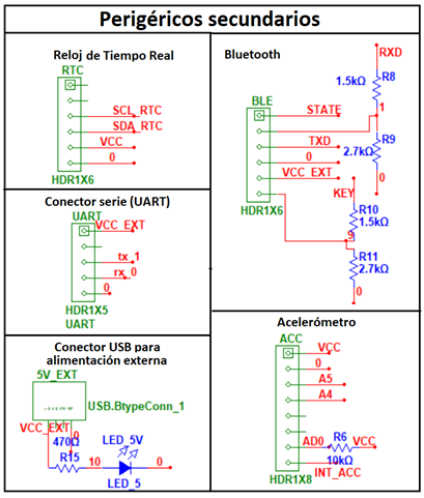
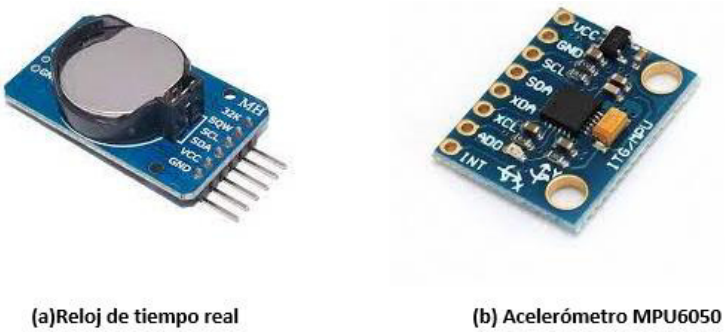


Figura 17. Esquemático con los periféricos secundarios.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P



(a)Reloj de tiempo real **(b) Acelerómetro MPU6050**

Figura 18. Dispositivos conectador al bus I2C.

Fuente: Figura extraída del Manual de Usuario kit ATmega328P

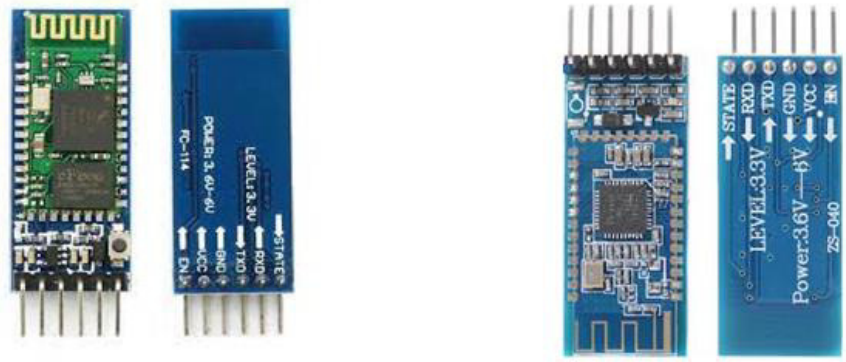


Figura 19. Módulos Bluetooth compatibles.
Fuente: Figura extraída del Manual de Usuario kit ATmega328P

6.1.2 Cámara web

Una cámara web es un dispositivo de entrada que captura imágenes digitales. Estas se transfieren a la computadora, que las traslada a un servidor y, desde allí, se pueden transmitir a la página de alojamiento. En la figura 20, se muestra la cámara web (con conexión USB) que será utilizada para el trabajo.

Figura 20. Cámara web.
Fuente: https://intercompras.com/images/product/GENIUS_32200161101.jpg

6.2 Software

Al momento de elegir el software que se iba a utilizar, se tuvo en cuenta que este fuera de Código Abierto (*Open Source*) y que contara con documentación de respaldo o foros de debate para posibles dificultades al momento de la implementación.

6.2.1 Ubuntu

Ubuntu es una distribución Linux basada en Debian GNU/Linux, que incluye principalmente software libre y de código abierto.

Puede utilizarse en ordenadores y servidores. Está orientado al usuario promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia del usuario. Está compuesto de múltiple software normalmente distribuido bajo una licencia libre o de código abierto.

Figura 21. Logo de Ubuntu.

Fuente: https://assets.ubuntu.com/v1/29985a98-ubuntu-logo32.png

6.2.2 Transmisión de video en vivo

La transmisión en vivo es un tipo de streaming en el que se transmite audio o video en vivo a través de Internet. Los medios se transmiten mientras se graban, lo que permite a los espectadores verlos o escucharlos en tiempo real.

6.2.2.1 MJPG-Streamer

MJPG-Streamer es una aplicación para la distribución web de video desde cámaras compatibles con Linux. Esta aplicación genera datos JPEG directamente desde la cámara y emite una transmisión MJPG de alta velocidad en tiempo real.

Los videos producidos por MJPG-Streamer se pueden ver fácilmente con navegadores como Chrome, Safari, Firefox, VLC y aplicaciones mplayer a través de la red.

La aplicación se creó originalmente para dispositivos integrados con pocos recursos de CPU y memoria.

6.2.3 Visual Studio Code

Visual Studio Code es un editor de código fuente gratuito, ligero pero potente que se ejecuta en su escritorio y en la web y está disponible para Windows, macOS, Linux y Raspberry Pi OS. Viene con soporte integrado para JavaScript , TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes de programación (como C++, C#, Java, Python, PHP y Go), tiempos de ejecución (como .NET y Unity), entornos (como Docker y Kubernetes) y nubes (como Amazon Web Services, Microsoft Azure y Google Cloud Platform).

Figura 22. Logo de Visual Studio Code.

Fuente: https://cdn.freebiesupply.com/logos/thumbs/2x/visual-studio-code-logo.png

6.2.3.1 PlatformIO IDE

PlatformIO es una herramienta profesional multiplataforma, multiarquitectura y marco múltiple para ingenieros de sistemas embebidos y para desarrolladores de software que escriben aplicaciones para productos integrados.

Visual Studio Code otorga una extensión llamada PlatformIO IDE.

En relación con su implementación interna, PlatformIO cuenta con los siguientes puntos clave que describen su funcionamiento:

- La plataforma de desarrollo, así como la placa física empleada son configuraciones del proyecto y como tal van en el fichero “platformio.ini.”
- En dicho fichero pueden existir diferentes configuraciones para diferentes entornos, todos ellas compartiendo el mismo código fuente.

- Relacionado con lo anterior, el código fuente es independiente de la plataforma de desarrollo empleada y va localizado bajo la carpeta “src”.
- Basándose en el fichero plaformio.ini, el sistema se encarga de descargar e instalar todos los requerimientos necesarios.
- La carpeta “lib” almacena las diferentes librerías requeridas por la aplicación que se esté desarrollando.
- El usuario dispone en una única herramienta y con una API sencilla de toda la funcionalidad que requiere para crear, compilar y subir código.

Figura 23. Logo de PlatformIO.

Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb/c/cd/PlatformIO_logo.svg/2500px-PlatformIO_logo.svg.png

6.2.3.1.1 PlatformIO Core (CLI)

PlatformIO Core (herramienta CLI) es el corazón de todo el ecosistema PlatformIO y consta de:

- Sistema de construcción multiplataforma
- Administrador de paquetes unificados
- Gestión de bibliotecas
- Buscador de dependencias de biblioteca (LDF)
- Monitoreo de puerto serie
- Componentes de integración (IDEs de escritorio y nube e integración continua).

PlatformIO Core proporciona una interfaz de línea de comandos (CLI) rica y documentada.

Se requiere aclarar que no se necesita instalar Platformio Core si se va a utilizar PlatformIO IDE, ya que Platformio Core está integrado a PlatformIO IDE y se podrá utilizar dentro de PlatformIO IDE Terminal.

6.2.4 Apache

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh, entre otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual, según la normativa RFC 2616.

El servidor Apache es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la Apache Software Foundation, dentro del proyecto HTTP Server (httpd).

Apache presenta, entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

Apache tiene amplia aceptación en la red desde 1996 y es el servidor HTTP más usado. Jugó un papel fundamental en el desarrollo de la World Wide Web y alcanzó su máxima cuota de mercado en 2005, siendo el servidor empleado en el 70% de los sitios web en el mundo. Sin embargo, ha sufrido un descenso en su cuota de mercado en los últimos años. En 2009, se convirtió en el primer servidor web que alojó más de 100 millones de sitios web.

Entre sus principales ventajas se pueden mencionar las siguientes:

- Modular
- Código abierto
- Multi-plataforma
- Extensible
- Popular (fácil conseguir ayuda/soporte)

Figura 24. Logo del Servidor HTTP Apache.

Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb/1/10/Apache_HTTP_server_logo_%282019-present%29.svg/2560px-Apache_HTTP_server_logo_%282019-present%29.svg.png

6.2.5 phpMyAdmin

phpMyAdmin es una herramienta de software libre escrita en PHP, destinada a manejar la administración de MySQL en la Web. Además, admite una amplia gama de operaciones en MySQL y MariaDB. Las operaciones de uso frecuente (administración de bases de datos, tablas, columnas, relaciones, índices, usuarios, permisos, etc.) se pueden realizar a través de la interfaz de usuario, mientras aún tiene la capacidad de ejecutar directamente cualquier instrucción SQL.

Figura 25. Logo de phpMyAdmin.

Fuente: https://www.coriaweb.hosting/wp-content/uploads/2016/06/logo-og.png

6.3 Tecnologías web

Las tecnologías web se refieren a las diversas herramientas y técnicas que se utilizan en el proceso de comunicación entre diferentes tipos de dispositivos a través de Internet. Se maneja un navegador web para acceder a las páginas web y estos navegadores web se pueden definir como programas que muestran texto, datos, imágenes, animaciones y videos en Internet. Además, del acceso a los recursos con hipervínculos en la World Wide Web utilizando las interfaces de software proporcionadas por los navegadores web.

Las tecnologías web se pueden clasificar en las siguientes secciones:

- World Wide Web (WWW): la World Wide Web se basa en varias tecnologías diferentes, entre ellos, navegadores web, lenguaje de marcado de hipertexto (HTML) y protocolo de transferencia de hipertexto (HTTP).

- Navegador web: el navegador web es un software de aplicación para explorar www (World Wide Web). Este proporciona una interfaz entre el servidor y el cliente y solicita al servidor documentos y servicios web.

- Servidor web: el servidor web es un programa que procesa las solicitudes de red de los usuarios y les sirve con archivos que crean páginas web. Este intercambio se realiza mediante el Protocolo de transferencia de hipertexto (HTTP).

- Páginas web: una página web es un documento digital que está vinculado a la World Wide Web y puede ser visto por cualquier persona conectada a Internet que tenga un navegador web.

- Desarrollo web: el desarrollo web se refiere a la construcción, creación y mantenimiento de sitios web. Incluye aspectos como el diseño web, la publicación web, la programación web y la gestión de bases de datos. También, remite a la creación de una aplicación que funciona a través de Internet, es decir, sitios web.

El desarrollo web se puede clasificar en dos formas:

- **Desarrollo Frontend**: la parte de un sitio web con la que el usuario interactúa directamente se denomina front-end. También se lo conoce como “lado del cliente” de la aplicación.
- **Desarrollo de backend**: backend es el lado del servidor de un sitio web. Es la parte del sitio web que los usuarios no pueden ver e interactuar. Además, se utiliza para almacenar y organizar datos.

6.3.1 HTML

HTML significa lenguaje de marcado de hipertexto. Se utiliza para diseñar la parte frontal de las páginas web utilizando un lenguaje de marcado. HTML es la combinación de hipertexto y lenguaje de marcado. El hipertexto define el vínculo entre las páginas web. El lenguaje de marcado se utiliza para definir la documentación de texto dentro de la etiqueta que define la estructura de las páginas web.

Figura 26. Logo de HTML.

Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb/6/61/HTML5_logo_and_wordmark.svg/1200px-HTML5_logo_and_wordmark.svg.png

6.3.2 JavaScript

JavaScript (JS) es un lenguaje de programación de computadora dinámico. Es liviano y se usa más comúnmente como parte de las páginas web, cuyas implementaciones permiten que el script del lado del cliente interactúe con el usuario y cree páginas dinámicas. También, es un lenguaje de programación interpretado con capacidades orientadas a objetos.

Figura 27. Logo de JavaScript.

Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb/9/99/Unofficial_JavaScript_logo_2.svg/1200px-Unofficial_JavaScript_logo_2.svg.png

6.3.2.1 Jquery

jQuery es una biblioteca de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

Figura 28. Logo de jQuery.

Fuente: https://blog.artegrafico.net/page-loader-con-jquery-y-css/jquery-logo

6.3.2.2 AJAX

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones web asíncronas. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible interactuar con el servidor sin necesidad de recargar la página web, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

Ajax es una tecnología asíncrona, en el sentido de que los datos adicionales se solicitan al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página, aunque existe la posibilidad de configurar las peticiones como síncronas de tal forma que la interactividad de la página se detiene hasta la espera de la respuesta por parte del servidor.

6.3.3 CSS

CSS (siglas en inglés de **C**ascading **S**tyle **S**heets), en español, «Hojas de estilo en cascada», es un lenguaje de diseño simple destinado a simplificar el proceso de hacer que las páginas web sean presentables.

CSS maneja la apariencia de una página web. Con CSS, se puede controlar el color del texto, el estilo de las fuentes, el espacio entre párrafos, el tamaño y la distribución de las columnas, qué imágenes o colores de fondo se utilizan, diseños de diseño, variaciones en la

visualización para diferentes dispositivos y tamaños de pantalla, así como una variedad de otros efectos.

CSS es fácil de aprender y comprender, pero brinda un poderoso control sobre la presentación de un documento HTML. De manera usual, CSS se combina con los lenguajes de marcado HTML o XHTML.

Figura 29. Logo de CSS.

Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb/d/d5/CSS3_logo_and_wordmark.svg/1452px-CSS3_logo_and_wordmark.svg.png

6.3.3.1 Bootstrap

Bootstrap es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales. A diferencia de muchos frameworks web, solo se ocupa del desarrollo front-end.

Figura 30. Logo de Bootstrap.

Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb/b/b2/Bootstrap_logo.svg/250px-Bootstrap_logo.svg.png

6.3.4 PHP

PHP, acrónimo recursivo en inglés de *Personal Hypertext processor* (preprocesador de hipertexto) fue creado inicialmente por el programador danés-canadiense Rasmus Lerdorf en 1994. Es un lenguaje de programación del lado servidor de código abierto que se puede usar para crear sitios web, aplicaciones, sistemas de gestión de relaciones con los clientes y más. Es un lenguaje de propósito general ampliamente utilizado, que se puede incrustar en HTML. Esta funcionalidad con HTML significa que el lenguaje PHP sigue siendo popular entre los desarrolladores, ya que ayuda a simplificar el código HTML.

Figura 31. Logo de PHP.

Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb/2/27/PHP-logo.svg/200px-PHP-logo.svg.png

6.4 Motor de base de datos

Un motor de base de datos (o motor de almacenamiento) es el componente de software subyacente, que un sistema de administración de la base de datos (SGBD) utiliza para crear, leer, actualizar y eliminar (CRUD) datos de una base de datos. La mayoría de sistemas de administración de la base de datos incluyen su interfaz de programación de aplicación propia (API), que permite al usuario interaccionar con su motor subyacente sin pasar por la interfaz de usuario del SGBD.

6.4.1 MYSQL

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual, Licencia pública general/Licencia comercial, por Oracle Corporation y está considerada como la base de datos de código abierto más popular del mundo y una de las más populares en general junto a Oracle y Microsoft SQL Server, todo para entornos de desarrollo web.

MySQL fue inicialmente desarrollado por MySQL AB (empresa fundada por David Axmark, Allan Larsson y Michael Widenius). MySQL AB fue adquirida por Sun Microsystems en 2008 y esta, a su vez, fue comprada por Oracle Corporation en 2010, que ya era dueña desde 2005 de Innobase Oy, empresa finlandesa desarrolladora del motor InnoDB para MySQL.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de doble licenciamiento anteriormente mencionado. La base de datos se distribuye en varias versiones, una Community, distribuida bajo la Licencia pública general de GNU, versión 2, y varias versiones Enterprise, para aquellas empresas que quieran incorporarlo en productos privativos. Las versiones Enterprise incluyen productos o servicios adicionales tales como herramientas de monitorización y asistencia técnica oficial. En 2009 se creó un fork denominado MariaDB por algunos desarrolladores (incluido algunos desarrolladores originales de MySQL) descontentos con el modelo de desarrollo y el hecho de que una misma empresa controle a la vez los productos MySQL y Oracle Database.

Este sistema está desarrollado en su mayor parte en ANSI C y C++.4 y, tradicionalmente, se considera uno de los cuatro componentes de la pila de desarrollo LAMP y WAMP.

Figura 32. Logo de MySQL.

Fuente: https://d1.awsstatic.com/asset-repository/products/amazon-rds/1024px-MySQL.ff87215b43fd7292af172e2a5d9b844217262571.png

6.5 Servicio DNS

El sistema de nombres de dominio Domain Name System (DNS, por sus siglas en inglés) es un sistema de nomenclatura jerárquico descentralizado para dispositivos conectados a redes IP como Internet o una red privada. Este sistema asocia información variada con nombres de dominio asignados a cada uno de los participantes. Su función más importante es «traducir» nombres inteligibles para las personas en identificadores binarios asociados con los equipos

conectados a la red, esto con el propósito de poder localizar y direccionar estos equipos mundialmente.

El servidor DNS utiliza una base de datos distribuida y jerárquica que almacena información asociada a nombres de dominio en redes como Internet. Aunque como base de datos el DNS es capaz de asociar diferentes tipos de información a cada nombre, los usos más comunes son la asignación de nombres de dominio a direcciones IP y la localización de los servidores de correo electrónico de cada dominio.

La asignación de nombres a direcciones IP es ciertamente la función más conocida de los protocolos DNS. Por ejemplo, si la dirección IP del sitio Google es 216.58.210.163, la mayoría de la gente llega a este equipo especificando www.google.com y no la dirección IP.

6.5.1 NO-IP

El No-IP es un servicio DNS dinámico que toma una dirección IP dinámica y la hace actuar como si fuera estática al señalarse un nombre de host estático y verificar cada 5 minutos los cambios en la dirección IP.

La mayoría de las personas tienen direcciones IP dinámicas si tienen una conexión a Internet de banda ancha en el hogar, lo que dificulta el acceso a su red, sitio web o dispositivos cotidianos, como un servidor de archivos doméstico, un sistema de automatización del hogar o una cámara de seguridad.

Para resolver este problema, un Servicio de nombres de dominio dinámico (DDNS) puede proporcionar una dirección actualizada continuamente, lo que permite al usuario escribir un nombre constante, estático y fácil de recordar que garantiza que se conectará.

Si la dirección IP cambia, Dynamic Update Client (DUC) actualiza el nombre de host con la dirección IP actual. Esto significa que se puede ejecutar un servidor desde un lugar específico y acceder a la computadora o cámara IP de forma remota de dicho lugar. El servicio DNS administrado No-IP hace que el sitio web sea confiable y súper redundante al manejar el DNS del sitio web.

Figura 33. Logo de no-ip.

Fuente: https://dmej8g5cpdyqd.cloudfront.net/blog/wp-content/uploads/2015/12/mynoip.gif

7. Implementación del sistema

En este apartado, se detallarán todas las tareas realizadas al desarrollar la Práctica Profesional Supervisada. Cabe aclarar que el proyecto se elaboró en una distribución de Linux, la cual es Ubuntu 20.10; pero, a través de configuraciones personalizadas, se pudo lograr migrar el proyecto a la distribución más actualizada que existe y se trata de Ubuntu 22.04.1 LTS.

7.1 Coordinación del equipo de trabajo de UNAJ sobre las tareas

En esta sección se procederá a explicar brevemente la coordinación con el equipo de trabajo de la UNAJ sobre las tareas. Esta coordinación se dio a través de videollamadas y reuniones presenciales.

Inicialmente se dio una reunión presencial en la UNAJ donde se me fue entregado los materiales para el desarrollo del laboratorio. Luego, se realizó otra reunión presencial, para hallar similitudes entre mi laboratorio y otros de índole remoto; además, de la posibilidad de poder crear un sistema que gestione los turnos hasta de evaluar la utilización de la gestión de turnos desde la plataforma Moodle.

Por otro lado, se dieron a su vez, las reuniones virtuales, donde se discutieron temas de importancia como es el monitor serie, parte de los objetivos a lograr en la PPS, y otro fue evaluar la posibilidad de programar un “teclado matricial” en el laboratorio donde este mismo envíe señales (mensajes) a un microcontrolador; además, de mostrar periódicamente los avances respectivos a la PPS.

Finalmente, las reuniones finales que se realizaron fueron para ultimar detalles sobre la PPS y hablar de mis avances en la instalación del proyecto en una máquina de la UNAJ.

7.2 Investigación de la plataforma y herramientas

En esta sección, se describe cómo fueron utilizadas las plataformas mencionadas anteriormente, junto con las herramientas.

7.2.1 Configuración de entorno de trabajo

7.2.1.1 Instalación Visual Studio Code

Para instalar Visual Studio Code (VSC) en Ubuntu se tuvo en cuenta dos opciones: por línea de comandos y por Ubuntu Software. Por simplicidad, se optó por el método desde Ubuntu Software. En la figura 34 se puede ver como se instaló VSC.

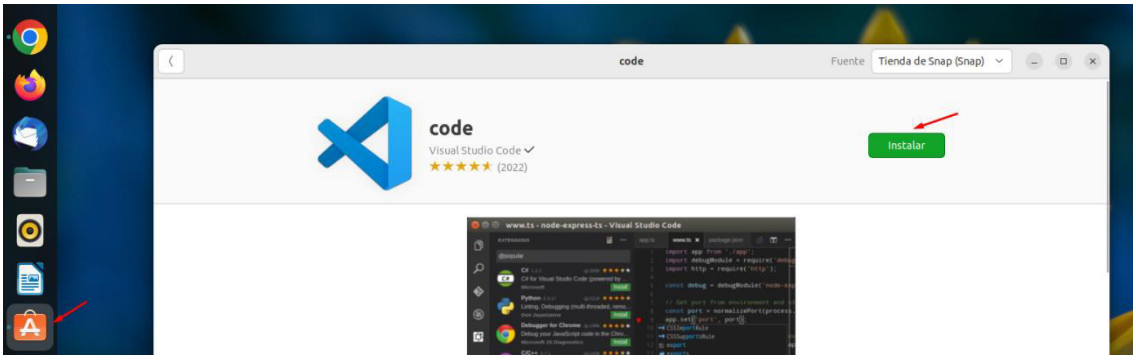


Figura 34. Instalación de VSC

Fuente: Producción propia

7.2.1.2 Extensión PlatformIO IDE

Antes de instalar la extensión de PlatformIO IDE dentro de Visual Studio Code, fue importante instalar las dependencias de Python necesarias.

En primer lugar, se actualizaron lista de repositorios mediante el siguiente comando:

```
sudo apt-get update
```

En segundo lugar, se actualizan los paquetes mediante el siguiente comando:

```
sudo apt-get upgrade
```

En tercer lugar, se instala python3 con el siguiente comando:

```
sudo apt install python3-pip
```

Finalmente, se instaló la extensión PlatformIO IDE. Para ello, se buscó la extensión en el buscador de VSC y se instaló como muestra la figura 35.

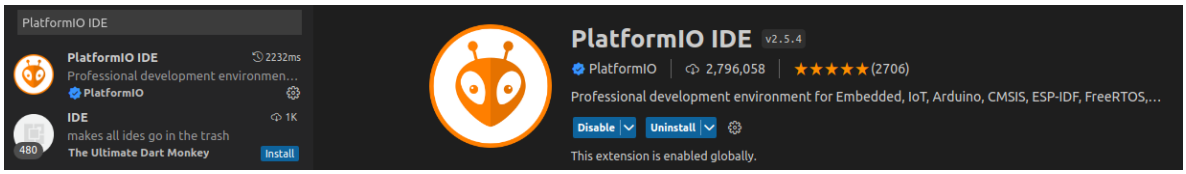


Figura 35. Instalación de PlatformIO IDE.

Fuente: Producción propia

Primera aproximación

En la extensión de PlatformoIO IDE se logró crear un proyecto para Arduino a través de los siguientes pasos:

1) Se entra a la plataforma a través de la extensión instalada
2) Se hace click en New Project.
3) Se crea el proyecto (Ver figura 36).
4) Se llenan los siguientes campos:
   - Name: Nombre de proyecto.
   - Board: La placa. En este caso, como se tiene Arduino Uno, esta fue la elegida.
   - Framework: Elegimos Arduino en este caso.
5) Se hace click en el botón “Finish” y crea el primer proyecto.

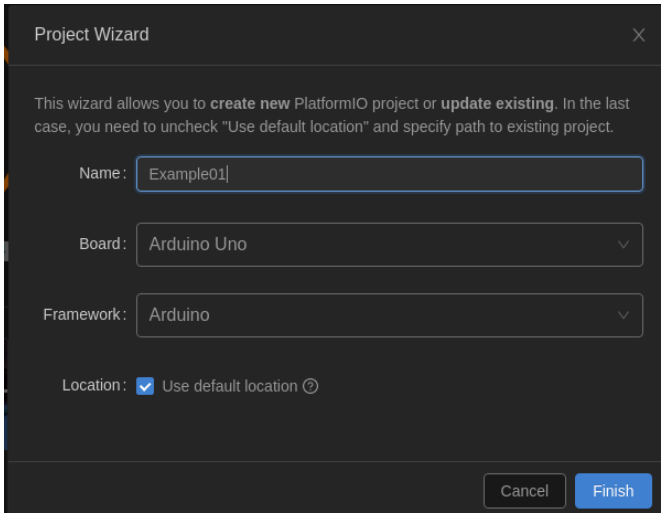


Figura 36. Creación de proyecto PlatformIO IDE.

Fuente: Producción propia

Directorio de proyectos PlatformIO

A continuación, en la figura 37 se presenta cómo se organiza un proyecto creado a través de la extensión de PlatformIO IDE.

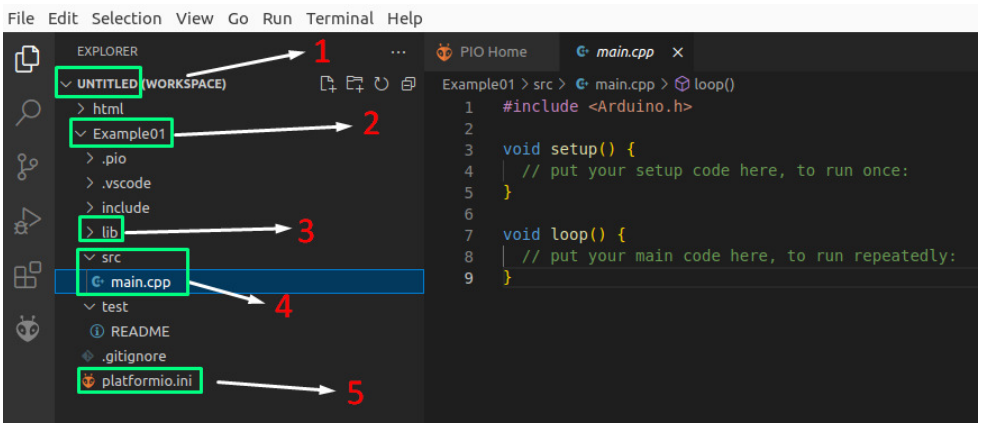


Figura 37. Organización de proyecto en PlatformIO IDE.

Fuente: Producción propia

1) Espacio de trabajo: «workspace» contiene el proyecto en el que se esté trabajando actualmente, se puede agregar más de uno y eliminarlos sin problemas.

2) Directorio del proyecto: es la carpeta principal que contiene los archivos y subcarpetas que PlatformIO crea automáticamente. Puede haber más de uno, según la cantidad de proyectos que se estén gestionando.
3) Librerías: en esta carpeta el gestor de librerías de PlatformIO incluye las librerías privadas de cada proyecto. Las librerías son muy importantes, cuando se compila un código y se ejecuta en la placa, se debe tener en cuenta qué funcionalidades tiene el código para ver si es necesario instalar una librería externa.
4) Código principal: la carpeta SCR (simboliza la abreviatura de “Source”, fuente en inglés) aquí se incluye el código principal o código fuente del proyecto. Por otro lado, PlatformIO crea un fichero con el nombre “main.cpp”. Este es el equivalente al archivo compilado por el IDE Arduino llamado: “codigo.ino”.
5) platformio.ini: es un archivo de configuración que dota de funcionalidades extras al entorno. Es capaz de agilizar el trabajo cuando se trata de grandes proyectos.

Instalación de una librería

En el Apéndice 13.1, se puede encontrar la descripción detallada de cómo se instala la librería y un ejemplo de uso.

7.2.1.3 PlatformIO por línea de comando

Como se mencionó en reiteradas ocasiones, no hace falta instalar PlatformIO Core si ya se instaló PlatformIO IDE, puesto que PlatformIO Core está integrado en PlatformIO IDE. Como se utilizará la terminal, solo se requiere instalar los comandos de Shell necesarios para usar PlatformIO Core fuera de PlatformIO IDE.

En sistemas Unix y similares a Unix, se pueden *crear enlaces simbólicos* en el directorio: $HOME/.local/bin/ a los ejecutables PlatformIO necesarios.

Esto permitirá ejecutar comandos de PlatformIO desde cualquier emulador de terminal.

Teniendo en cuenta lo mencionado anteriormente, los pasos a seguir son los siguientes:

En primer lugar, se debe exportar el directorio $HOME/.local/bin/ a la variable ambiental PATH con el siguiente comando:

```
export PATH=$PATH:$HOME/.local/bin
```

Por último lugar, ahora que ya está hecho, o si $HOME/.local/bin/ ya se exportó a la variable de entorno PATH, se crean los enlaces simbólicos con los siguientes comandos:

```
cd /
sudo ln -s ~/.platformio/penv/bin/platformio usr/local/bin/platformio
sudo ln -s ~/.platformio/penv/bin/pio usr/local/bin/pio
sudo ln -s ~/.platformio/penv/bin/piodebuggdb usr/local/bin/piodebuggdb
```

Realización del primer proyecto desde línea de comando

Para inicializar un proyecto mediante comandos en la terminal se hizo lo siguiente:

En primer lugar, se creó una carpeta para alojar el primer proyecto mediante los siguientes comandos:

```
mkdir proyecto1
cd proyecto1
```

En segundo lugar, se inicializó PlatformIO con el siguiente comando:

```
platformio init --board=uno
```

En tercer lugar, luego de ejecutar el comando `ls`, se puede observar que la estructura del proyecto en la figura 38 es similar a la estructura generada en la anterior sección cuando se creó un proyecto desde la extensión de PlatformIO IDE de VSC.

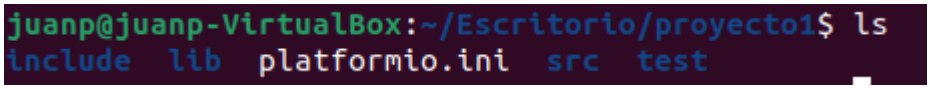


Figura 38. Estructura del proyecto de PlatformIO generada con comandos.

Fuente: Producción propia

En cuarto lugar, se accedió a la carpeta src para alojar el programa mediante el siguiente comando:

```
# cd src
```

En quinto lugar, se creó y abrió un archivo con extensión. ino para realizar el programa deseado mediante el siguiente comando:

```
# sudo nano hola.ino
```

En sexto lugar, se debe retroceder a la carpeta anterior para ejecutar comandos de PlatformIO, por lo tanto se ejecutó el siguiente comando:

```
# cd ..
```

En séptimo lugar, se ejecutó el siguiente comando para compilar el programa creado:

```
# platformio run
```

Finalmente, se subió el programa a la placa de Arduino UNO mediante el siguiente comando:

```
# platformio run --target upload
```

7.2.1.4 Instalación y configuración MJPG-Streamer

En esta sección, se procederá a detallar la instalación y configuración de MJPG-Streamer.

En primer lugar, se comprobó si estaba instalado **git** en el sistema operativo. Para ello, primero se actualizó el sistema.

```
sudo apt update
sudo apt upgrade -y
```

En segundo lugar, se comprobó si git está instalado en el sistema con el siguiente comando:

```
git –version
```

Como no salió nada, se instaló ejecutando el comando:

```
sudo apt install git
```

En tercer lugar, se instaló cmake, libjpeg, g++ porque son dependencias necesarias antes de instalar mjpg-streamer.

```
sudo apt-get install cmake libjpeg8-dev
sudo apt-get install gcc g++
```

En cuarto lugar, fueron utilizados los siguientes comandos para clonar el repositorio de la aplicación mjpg-streamer:

```
cd ~/
git clone https://github.com/jacksonliam/mjpg-streamer.git
```

Por último, se procedió a instalar mjpg-streamer mediante los siguientes comandos:

```
cd ~/mjpg-streamer/mjpg-streamer-experimental
make
sudo make install
```

Configuración e inicialización

MJPG-Streamer no tiene un archivo de configuración, pero todos los elementos de configuración deben pasarse al programa con argumento de comando. El siguiente comando sirvió para inicializar el programa y configurarlo al mismo tiempo.

```
export LD_LIBRARY_PATH=.
./mjpg_streamer -o “output_http.so -w ./www -p 8080” -i “input_uvc.so -d /dev/video2 -r 1280x720 -fps 30 -q 10”
```

Complemento de salida (output_http)

[w- | -- www]…: Carpeta que contiene páginas web en jerarquía plana (sin subcarpetas]

[-p | --port]…: Puerto TCP para el servidor HTTP. En este caso fue el 8080.

Complemento de entrada (input_uvc)

[-d | --device]…: Dispositivo de video para abrir. **Adicional:** Se ejecutó el siguiente comando para ver si el sistema operativo reconocía la cámara conectada.

```
lsusb
```

Se obtuvo la siguiente salida (donde el sistema reconoce la cámara utilizada):

```
Bus 001 Device 007: ID 80ee:0030 VirtualBox VirtualBox Webcam - eFace 1300
```

[-r | -- resolution]...: Es la resolución del dispositivo de video que se esté empleando. En este caso, el valor correspondiente fue 1280x720.

[-f| --fps]…: Fotogramas por segundo. El valor elegido fue 30.

En la figura 39 se puede ver el funcionamiento de la webcam a través de la dirección: ruta:puerto (localhost:8080).

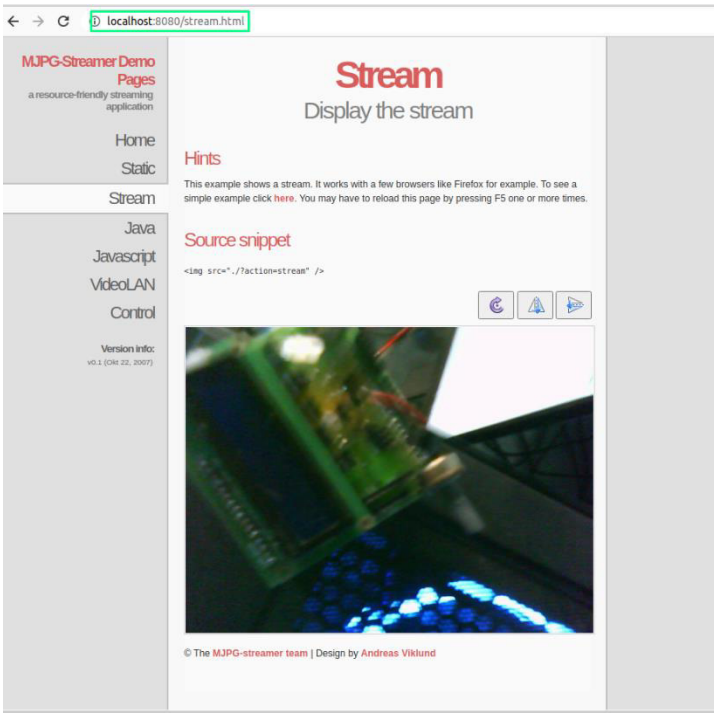


Figura 39. MJPG-Streamer Demo.

Fuente: Producción propia

Cómo lograr que MJPG-Streamer se ejecute en Shell

Después de inicializar MJPG-Streamer manualmente con un comando, se planteó la siguiente duda: ¿Qué sucede sí se reinicia el dispositivo donde está alojado el servidor? Como MJPG-Streamer no es un servicio, por lo tanto, no hay manera de que inicie de manera automática. No obstante, se realizaron unas configuraciones para crear un servicio de forma manual.

En primer lugar, se creó un script de Bash en la ruta cd ~/ como se ve en la figura 40.

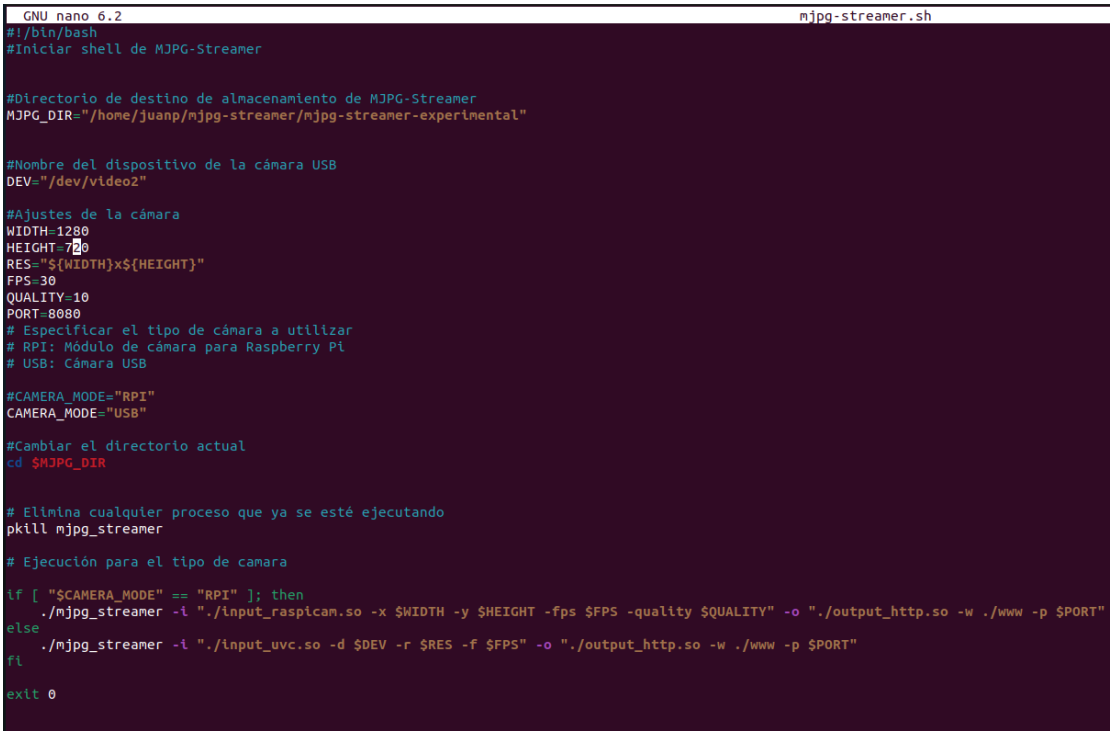


Figura 40. MJPG-Streamer creación de script de Bash

Fuente: Producción propia

En segundo lugar, se otorgó el permiso de ejecución al archivo shell creado mediante el siguiente comando:

```
chmod 755 mjpg-streamer.sh
```

En tercer lugar, se registró a MJPG-Streamer como un servicio mediante los siguientes comandos:

```
cd /etc/systemd/system
sudo nano mjpg-streamer.service
```

En la figura 41 se puede observar el contenido del archivo que corresponde al servicio que fue creado.

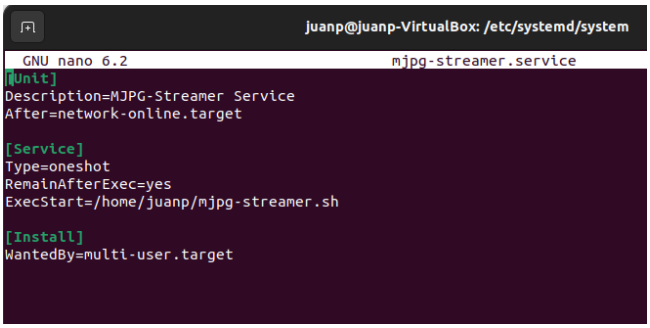


Figura 41. Creación del servicio para MJPG-Streamer.

Fuente: Producción propia

Por último, se cargó el archivo del servicio creado anteriormente en el sistema con el siguiente comando:

```
sudo systemctl deamon-reload
```

A continuación, se listan los comandos que fueron usados para iniciar, detener, iniciar automáticamente y detener el inicio automático del servicio creado:

Inicio del servicio: `sudo systemctl start mjpg-streamer`

Detener servicio: `sudo systemctl stop mjpg-streamer`

Inicio automático: `sudo systemctl enable mjpg-streamer`

Detener servicio automático: `sudo systemctl disable mjpg-streamer`

Nota: Cuando se inició el servicio fue necesario presionar ctrl + c para continuar con el funcionamiento de la terminal. Esto no afecta negativamente al funcionamiento correcto del servicio.

En este caso, se optó por la opción de iniciarlo automáticamente, ya que es vital que siempre esté en funcionamiento este servicio para el laboratorio remoto próximo a implementar.

7.2.1.5 Instalación LAMP

LAMP es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- Linux, el sistema operativo.
- Apache, el servidor web.
- MySQL/MariaDB, el gestor de bases de datos. En este caso se utilizará MySQL.
- PHP, el lenguaje de programación.

Instalación de Apache

Disponer de un servidor web como Apache permitirá alojar páginas y aplicaciones web de forma que den servicio tanto desde localhost como desde Internet. Estos serán la base que facilitará el acceso a la información por parte de los usuarios.

Para la instalación de Apache en Linux se procedió a ejecutar el siguiente comando:

```
sudo apt install apache2
```

Ahora en la ruta /var/www/html se encuentran los archivos contenidos para empezar a crear el proyecto. Además, trae un index.html por defecto. En caso de probar si todo está en orden,

se debe ir al navegador y poner la dirección localhost:80 (80 es el puerto de apache2). Una vez que se ingresa a la página, debería aparecer el index.html por defecto dicho anteriormente.

Instalación de MySQL y configuración

Para instalar MySQL se debió ejecutar el siguiente comando:

```
sudo apt-get install sql-server sql-client
```

Luego, para ingresar a MySQL a través de la terminal se hizo a través del comando:

```
mysql -u root -p
```

Para evitar problemas futuros con phpMyAdmin (administrador de base de datos) fue necesario ejecutar las siguientes sentencias:

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '1234';
```

Nota: se seleccionó ‘1234’ como contraseña por simplicidad, se puede poner la contraseña que uno desee.

Por otra parte, se actualizaron los privilegios con el siguiente comando:

```
FLUSH PRIVILEGES;
```

Finalmente, se cerró SQL con el siguiente comando:

```
quit
```

Instalación de PHP, extensiones de PHP y phpMyAdmin

La aplicación web se creó en la versión de PHP 7.4, por lo tanto, es necesario realizar una instalación en base a esa versión para que no haya futuras complicaciones.

Se ejecutó esta serie de comandos para instalar PHP 7.4 y algunas extensiones de sumo interés:

```
sudo apt update
sudo apt upgrade
sudo apt install software-properties-common apt-transport-https (
```

Instalación de software-properties-common. El paquete software-properties-common proporciona la utilidad de línea de comandos apt-add-repository, que se utilizó para añadir el repositorio de PPA (archivo de paquete personal) ondrej/php.)

```
sudo add-apt-repository ppa:ondrej/php
```

(Este comando se usó para agregar al repositorio los paquetes de PHP)

```
sudo apt update
```

```
sudo apt install php7.4
```

Además, se instalaron algunas extensiones de PHP. Las más importantes son mysql y libapache2-mod-php7. El paquete libapache2-mod-php7.4 ofrece un módulo para el servidor web Apache mientras que el módulo php7.4-mysql tiene como función conectar PHP con la base de datos MySQL. Estas extensiones se instalaron mediante los siguientes comandos:

```
sudo apt install php7.4-{cli,common,curl,zip,gd,mysql,xml,mbstring,json,intl}
sudo apt install libapache2-mod-php7.4
sudo apt-get install php7.4-dev
sudo apt install php7.4-gettext
```

Instalación y configuración de phpMyAdmin

Instalación

Se optó por instalar phpMyAdmin de manera manual. Por lo tanto, estos fueron los pasos que se siguieron.

Primero, se accede a la carpeta *tmp* mediante el siguiente comando:

```
cd /tmp
```

Segundo, se descargó el archivo .zip de la versión de phpMyAdmin elegida con el siguiente comando:

```
wget https://files.phpmyadmin.net/phpMyAdmin/4.9.5/phpMyAdmin-4.9.5-all-languages.zip
```

Tercero, se descomprime el archivo con el siguiente comando:

```
unzip phpMyAdmin-4.9.5-all-languages.zip
```

Cuarto, se cambió el nombre de carpeta por *phpmyadmin* mediante el siguiente comando:

```
mv phpMyAdmin-4.9.5-all-languages phpmyadmin
```

Por último, se movió la carpeta *phpmyadmin* a la raíz del documento con el siguiente comando:

```
sudo mv phpmyadmin /var/www/html
```

Configuración

Para configurar phpMyAdmin, en primer lugar, se fue a la ruta donde se encuentra la carpeta *phpmyadmin*, o sea: */var/www/html/phpmyadmin* y luego se actualizó la configuración de phpMyAdmin de la siguiente manera:

Primero, se ejecutó el siguiente comando para dirigirse a la carpeta donde se encuentra phpMyAdmin:

```
cd /var/www/html/phpmyadmin
```

Segundo, se abrió con el editor **nano** el archivo de configuración llamado *config.sample.inc.php*:

```
sudo nano config.sample.inc.php
```

Tercero, se actualizó la siguiente línea al host de base de datos correcto, ya que se está ejecutando MySQL en el mismo Droplet que acaba de poner localhost:

```
$cfg['Servers'][$i]['host'] = localhost
```

Finalmente, se cambió el nombre de la configuración:

```
sudo mv config.sample.inc.php config.inc.php
```

Por otra parte, se adapta mejor al flujo de trabajo si se conecta a phpMyAdmin con un usuario de MySQL dedicado en lugar del usuario raíz. Para hacer esto, se abrió el Shell de MySQL mediante el siguiente comando:

```
sudo mysql -u root -p
```

Luego, se ejecutaron las siguientes sentencias:

```
mysql> CREATE USER 'root'@'%' IDENTIFIED BY '1234';
mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION;
mysql> FLUSH PRIVILEGES;
mysql> quit
```

Nota: Se puso el usuario raíz y la misma contraseña configurada anteriormente en MySQL por simplicidad. No obstante, es recomendado, por seguridad, crear otro usuario dedicado con una contraseña segura.

Aseguración de la instancia de phpMyAdmin

Se configuró la autenticación *.htaccess* para proteger la instancia de phpMyAdmin de ataques de fuerza bruta y solo como una segunda capa de seguridad.

En primer lugar, se editó el archivo vinculado que se colocó en el directorio de configuración de Apache con el siguiente comando:

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

En segundo lugar, se agregó una *AllowOverride All* directiva dentro de la *<Directory /var/www/html/phpmyadmin>* sección del archivo de configuración, como se muestra en la figura 42.

```
<Directory /var/www/html/phpmyadmin>
    Options FollowSymLinks
    DirectoryIndex index.php
    AllowOverride All
</Directory>
```

Figura 42. Agregación de directorio a la configuración de Apache servidor.

Fuente: Producción propia

En tercer lugar, se necesitó habilitar el uso de .htaccess para la aplicación, para hacerlo, se debió crear uno para implementar algo de seguridad mediante el siguiente comando:

```
sudo nano /var/www/html/phpmyadmin/.htaccess
```

Dentro de este archivo se agregó determinada información como se ve en la figura 43.

```
  GNU nano 6.2                    /var/www/html/phpmyadmin/.htaccess
AuthType Basic
AuthName "Restricted Files"
AuthUserFile /etc/apache2/.htpasswd
Require valid-user
```

Figura 43. Archivo .htaccess.

Fuente: Producción propia

En cuarto lugar, se creó este archivo y se ingresó un usuario y contraseña mediante el siguiente comando:

```
sudo htpasswd -c /etc/apache2/.htpasswd root
```

El usuario se agregó en el mismo comando, en este caso, es root. Por otro lado, la contraseña se solicitó luego de ejecutar dicho comando.

Finalmente, se logró acceder desde el navegador al *phpmyadmin* (Ver figura 44), utilizando el usuario y contraseña elegidos en el momento de la configuración.

Después de una autenticación exitosa, se pudo acceder a la interfaz *phpMyAdmin* y usarla para administrar las bases de datos.

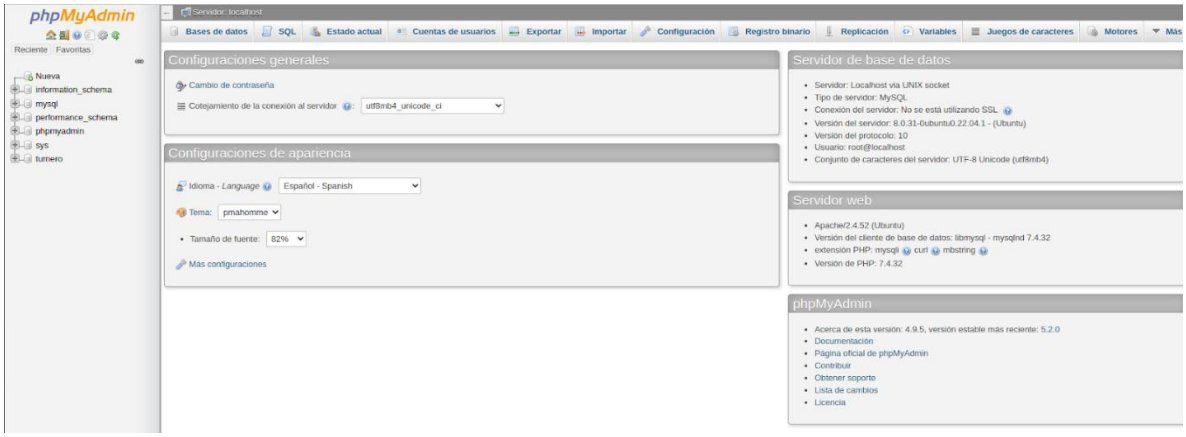


Figura 44. Inicio de phpMyAdmin.

Fuente: Producción propia

7.3 Implementación del servidor WEB

Antes de la creación del dominio en No-IP, lo primero que se hizo fue la realizar la configuración de la ip estática en el equipo. La dirección IP del equipo es preferiblemente que sea del tipo estático ya que, al tratarse de un servidor donde están conectados diversos servicios y recursos compartidos, el tener una IP dinámica (por DHCP) se puede dar lugar a que la dirección original enlazada pierda su ruta y con ello el acceso a los datos.

En primer lugar, se instaló la utilidad de red con el comando:

```
sudo apt install net-tools
```

En segundo lugar, se validó la IP actual del servidor con el comando:

```
ip -a
```

Como se está configurando desde una máquina virtual, lo que se toma en cuenta son dos parámetros: puerta de enlace y máscara de subred. Como la máquina (no virtual) es Windows 10, se utilizó un comando *"ifconfig"* desde el símbolo de sistema. Una vez que se ejecutó tal comando, se tomó en cuenta los parámetros mencionados anteriormente, por lo tanto:

- *Gateway*: 192.168.1.1
- *Mascara de subred*: 255.255.255.0

En tercer lugar, se accedió a la ruta donde está el archivo de configuración de red, mediante el siguiente comando:

```
cd /etc/netplan
```

En cuarto lugar, se editó el archivo *01-network-manager-all.yaml:*

sudo nano 01-network-manager-all.yaml

El contenido que se ingresó a el archivo se puede ver en la figura 45.

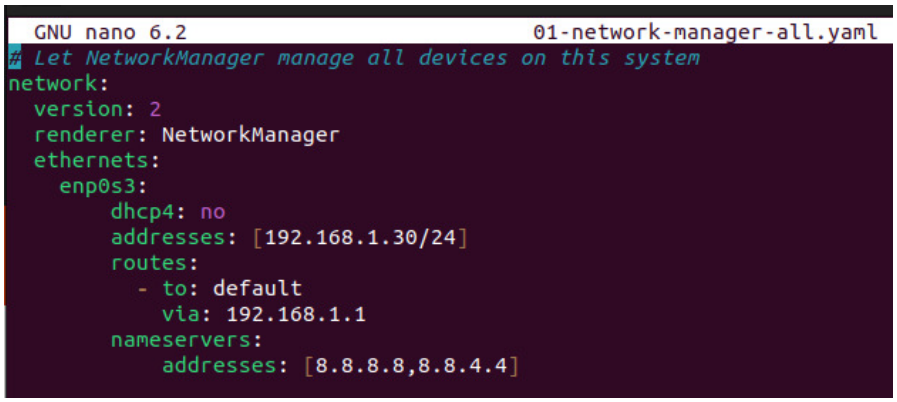


Figura 45. Contenido del archivo 01-network-manager-all.yaml.

Fuente: Producción propia

En último lugar, se aplicaron los cambios de red con el siguiente comando: sudo netplan apply.

En la figura 46 se puede observar el resultado del cambio de IP dinámica a estática en el recuadro verde.

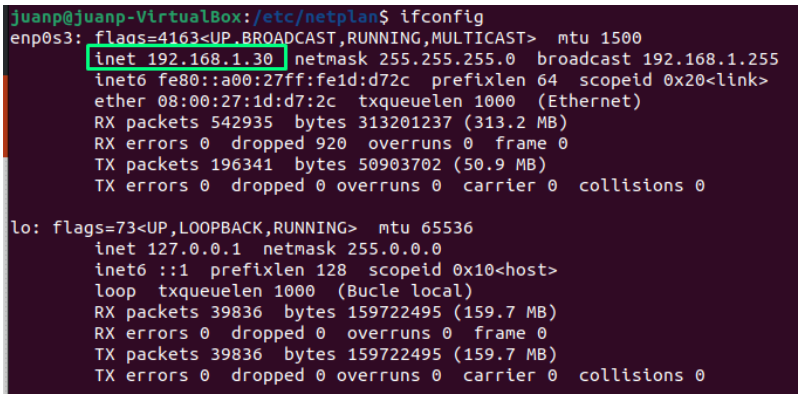


Figura 46. Cambio de IP dinámica a estática. Resultados.

Fuente: Producción propia

Creación del dominio en No-IP

Para la creación del dominio en No-IP, primero se accedió a la página de No-ip, https://www.noip.com. Una vez allí, en la sección “ingresar”, se procedió a crear una cuenta.

Para crear una cuenta, se nos solicitó usuario (mail) y contraseña. Una vez completados los datos de registro, se solicitó también la confirmación de la cuenta.

Hecha la confirmación, ya se puede crear el dominio deseado. Se puede optar por el nombre que se desee, mientras este se encuentre disponible. No-ip posee sufijos gratuitos y otros pagos; en este caso, se utilizó uno gratuito; por lo que el dominio quedó como se ve en la figura 47.

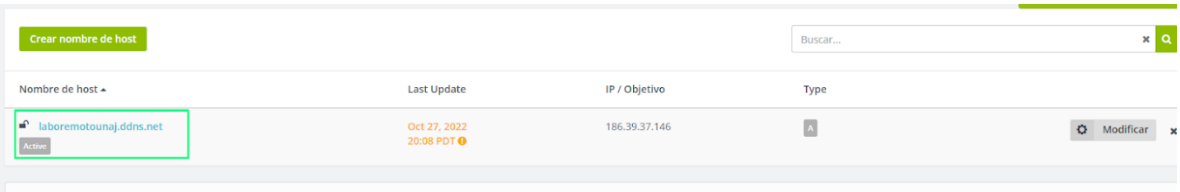


Figura 47. Creación de dominio en No-IP.

Fuente: Producción propia

7.4 Programación de funcionalidades requeridas

7.4.1 Creación del sitio web inicial

En esta etapa se fue desarrollando el sitio web en base a las necesidades demandadas para interactuar con el laboratorio remoto.

Inicialmente, se realizó una página web simple donde el usuario subía un archivo (extensión. ino, cpp) al servidor. Del lado servidor, el reto era que este archivo se compile mediante la herramienta de PlatformIO (sección 6.2.3.1.1 PlatformIO Core (CLI)). La primera aproximación en base a este reto fue lo que se puede observar en la figura 48, donde se creó un script de Bash y donde se ejecutaron los comandos de PlatformIO.

Figura 48. Script de Bash para compilar programas a través de PlatformIO.

Fuente: Producción propia

En el script se ejecutaron algunos comandos que se van a explicar por número de línea:

- (2) Este comando se sitúa en la ruta donde se encuentra la carpeta src. Esta carpeta llamada src es donde PlatformIO aloja los archivos que van a ser compilados y ejecutados. No obstante, solo puede haber un solo archivo contenido en la carpeta src, sino PlatformIO tira error por consola, ya que no sabe qué archivo de los dos compilar. En palabras más simples, sólo se puede tener un solo archivo en la carpeta src.
- (3) Se le dan permisos a la carpeta *src*, así se puede ejecutar el comando *platformio run*
- (4) Se vuelve una carpeta atrás, donde como se vio anteriormente, (sección 6.2.3.1.1 PlatformIO Core (CLI)) solo pueden ser ejecutados los comandos de PlatformIO.
- (5) Comando donde se ejecuta la compilación.

Compilación del primer archivo

El objetivo claramente fue poder compilar el archivo que provenga del usuario al servidor. Por lo tanto, la figura 49 muestra lo que fue la primera aproximación, donde se ven dos botones: uno para que el usuario suba un archivo al servidor y otro botón para compilar dicho archivo a la placa de Arduino UNO, la cual está situada en el servidor.

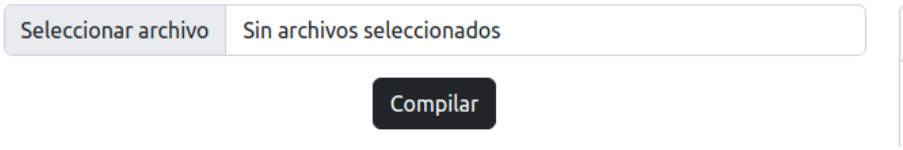


Figura 49. Botón para seleccionar un archivo (programa) y compilarlo.
Fuente: Producción propia

Recepción y compilado de archivo en el servidor

Una vez que el usuario seleccione el archivo y haga click en compilar, el formulario se enviará, ya que el botón compilar es de tipo submit (parámetro de la etiqueta HTML). En la figura 50, se observa una petición de AJAX, donde se envía el archivo a través del método POST, al archivo llamado "*compile.php*".

```
$.ajax({
    url: 'static/function/compile.php', //RUTA A LA QUE SE DIRIGE
    type: 'POST',
    enctype: 'multipart/form-data',
    data: formData, //MANDA LOS DATOS DEL FORMULARIO.
    contentType: false,
    processData: false,
    cache: false,
    success: function (response) {
```

Figura 50. Petición a través de AJAX.

Fuente: Producción propia

Lo que resta realizar es enviar el archivo a la carpeta “src”. Para esto mismo en la figura 61 se observa cómo se creó: un array con los errores frecuentes a la subida de archivos al servidor, un array con extensiones válidas, la dirección a enviar el archivo y la ruta de carga del archivo.

```
$phpFileUploadErrores = array(
    0 => 'There is no error, the file uploaded with sucess',
    1 => 'The uploaded file exceeds the upload_max filesize directive in php.ini',
    2 => 'The uplodad file exceeds the MAX_FILE_SIZE directive that was specified in the HTML form',
    3 => 'The uploaded file was only partially uploaded',
    4 => 'Archivo no seleccionado',
    6 => 'Missing a temporaly folder',
    7 => 'Failed to write file to disk',
    8 => 'A PHP extension stopped the file upload.',
);


$ext_error = false;
$extensions = array('ino','cpp'); //Extensiones validas
$dir = "/var/www/html/static/function/project/src/";  //Direccion a enviar el archivo
$ruta_carga = $dir . $_FILES['userfile']['name'];  //Ruta de carga de archivo
```

Figura 51. Visualización del archivo "compile.php".

Fuente: Producción propia

Finalmente, se envía el archivo a la carpeta “src” mediante una función de PHP llamada *“move_uploaded_file”*, esta función necesita los parámetros que se ven en la figura 52. Además, se ejecuta el script de Bash de la figura 48, mediante otra función de php *“shell_exec”* que recibe la ruta donde se encuentra el script de Bash que fue creado con anterioridad.

```
move_uploaded_file($_FILES['userfile']['tmp_name'], $ruta_carga); //movemos el archivo a la carpeta src


$comando = shell_exec('sh /var/www/html/static/bash/compile.sh'); //Compilamos el archivo.ino mediante la ejecución de un script .sh
```

Figura 52. Mover archivo y compilar.

Fuente: Producción propia

Validaciones de compilación

En la figura 52 se puede observar que la ejecución del script se guarda en la variable *"$comando"*, esto es muy importante, ya que esta variable contendrá una cadena de caracteres, los cuales son el resultado de lo que devuelve cada comando ejecutado en el script de bash. Esto sirve para saber si la validación de la compilación fue exitosa, por lo tanto, fue resuelto, simplemente con verificar si la cadena contiene "SUCESS" (Ver figura 130). En caso de ser verdadero (se ha compilado sin errores), se crea otro script de bash para eliminar el archivo contenido en la carpeta "src", ya que el objetivo es que no se acumulen los archivos en dicha carpeta. Además, se devuelve una alerta al usuario para que sepa que la compilación fue exitosa. En caso de ser falso (error en la compilación), al igual que en el caso verdadero se borra el archivo contenido en la carpeta "src", pero se devuelve una alerta al usuario para que sepa que hubo un error en la compilación.

En la figura 53 se observa cuando se selecciona el archivo (programa) y cuando se hace click en el botón "Compilar". Y (en la figura 54) se observa la alerta que indica cuál es el resultado de la compilación.

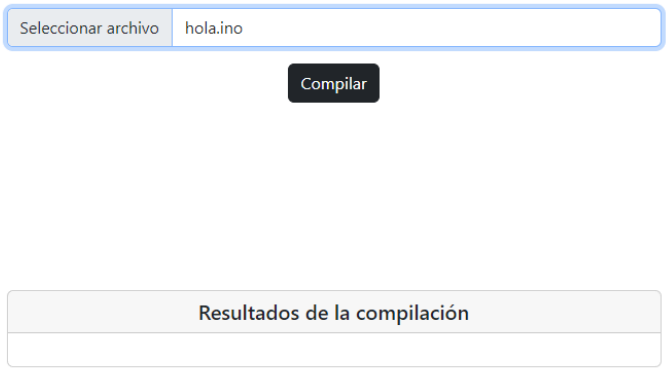


Figura 53. Selección de archivo (programa) y compilación.

Fuente: Producción propia

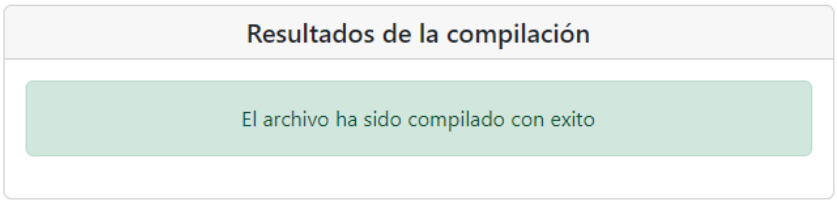


Figura 54. Resultado de la compilación del programa.

Fuente: Producción propia

Subida del primer archivo a la placa

Hasta el momento, se centró en la compilación de un programa, pero si se aplica la misma lógica que se utilizó para compilar, al caso de subir el archivo a la placa de Arduino UNO se hace más fácil, ya que solo habrá algunas pequeñas diferencias.

En primera instancia, se agregó un botón de tipo “switch” como se observa en la figura 55.

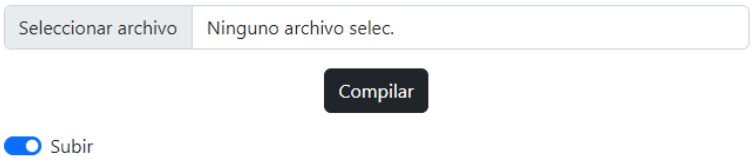


Figura 55. Botón switch.

Fuente: Producción propia

Este botón tendrá valor ‘on’ en caso de encendido y ‘off’ en caso de apagado. Por lo tanto, este valor, se agregó al formulario que se envía mediante el método POST al archivo script de PHP *(compile.php)*. Además, se creó una variable en dicho script de PHP, donde esta variable contendrá el valor del botón del switch:

$check = $_POST[‘checkupdate’]

Finalmente, solo resta agregar condicionales. En caso de que el archivo sea compilado con éxito y que el contenido de la variable *$check* sea igual a “on” se ejecuta el script de bash nuevo como se observa en la figura 56. Este script de bash es similar al compilar, lo único que

cambia es el comando de la línea (5), que es para subir el programa a la placa de Arduino UNO. En el caso de que el contenido de la variable $check sea igual a “off” significa que solamente se compilara el archivo. Si la respuesta es exitosa, se devolverá un mensaje en la ventana de resultados de compilación como se observa en la figura 57.

Nota: Antes de que haya una respuesta del servidor, por seguridad, se ejecuta un script de bash que se creó para eliminar archivos en la carpeta “src”.

```
1  #!/bin/sh
2  cd /var/www/html/static/function/project/src
3  chmod -R 777 ./
4  cd ..
5  sudo platformio run --target upload
6  rm -r src/*
7
8
```

Figura 56. Script de bash para realizar la subida del programa a través de PlatformIO.

Fuente: Producción propia

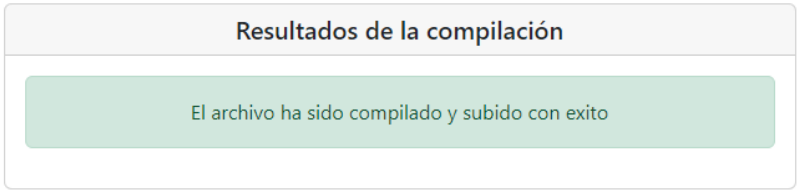


Figura 57. Resultado de la compilación y ejecución del programa.

Fuente: Producción propia

Incorporación del Streaming a la página

Hasta aquí, solo se tiene la posibilidad de que el usuario suba el archivo al servidor y este mismo se compile o suba a la placa de Arduino UNO. Pero, teniendo en cuenta que se instaló y

configuró MJPG-Streamer, la idea fue poder incrustar mediante una etiqueta simple de HTML, la dirección del Streaming.

**

En la línea anterior, se encuentra la etiqueta <img> de HTML y, en esta, hay varios atributos. El más importante es el atributo *src*, ya que este indica la URL de la imagen y, además, es obligatorio para este tipo de etiquetas.

En este caso, la URL es la dirección donde se encuentra funcionando el MJPG-Streamer.

Como se tiene funcionando el cliente No-IP, se accede a la cámara a través del dominio, sin olvidar indicar el puerto por el cual está transmitiendo la imagen:

"http://laboremotounaj.ddns.net:8080/?action=stream"
DOMINIO PUERTO RUTA STREAMING

Finalmente, se puede observar en la figura 58 el correcto funcionamiento de la cámara web transmitiendo en tiempo real.

Figura 58. Funcionamiento del Streaming.

Fuente: Producción propia

Incorporación del Monitor serie a la página

El monitor serial es el 'cable' entre el ordenador y el Arduino UNO. Este permite enviar y recibir mensajes de texto, útiles para la depuración y también control de Arduino; por ejemplo, es posible enviar comandos desde el ordenador para encender LEDs.

Cuando se empezó a pensar la idea de cómo programar el monitor serie surgieron varias opciones y se eligió la mejor para el tipo de tecnologías que utiliza el proyecto.

PHP Serial: Comunicación con un puerto serie

Investigando en la amplia comunidad de desarrolladores de PHP, se encontró una clase de PHP (https://www.phpclasses.org/) que se puede utilizar para comunicarse con un puerto serie en Linux o Windows. Esta toma la ruta (como "/dev/ttyS0" para Linux o "COM1" para Windows) del dispositivo serial y verifica si es válido antes de abrir una conexión con él. Una vez que se abre la conexión, puede enviar datos al puerto serie y leer las respuestas (la lectura solo se implementa para Linux).

La clase también puede cambiar los parámetros de conexión para el dispositivo serial dado.

Funcionamiento de la clase phpSerialPort

Para hacer uso de la clase, en primer lugar, se creó un script PHP; en segundo lugar, incluyo la clase ‘PhpSerial.php’ en dicho script; en tercer lugar, se inicializó la clase; en cuarto lugar, se especificó el dispositivo serial; en quinto lugar, se configuraron los parámetros de tasa de baudios, la paridad, la longitud, los bits de parada y el control de flujo; por último, luego de configurar todo lo comentado previamente, se abre el puerto serial, se lee el puerto o se envían mensajes al puerto y se cierra el puerto serial. El cierre del puerto es importante, ya que, si se deja trabajando, se limitarán las demás tareas que se quieran ejecutar en la placa de Arduino UNO. En la figura 59, se puede observar un programa genérico para hacer funcionar la comunicación con el puerto serial.

```php
<?php
include 'PhpSerial.php';

// Empezar la clase
$serial = new PhpSerial;

// Primero debemos especificar el dispositivo. Esto funciona tanto en Linux como en Windows.
// El dispositivo serial de Linux podria ser /dev/ttyS0 para COM1, etc.)
$serial->deviceSet("COM1");

//Podemos cambiar la tasa de baudios, la paridad, la longitud, los bits de parada, el control de flujo
$serial->confBaudRate(2400);
$serial->confParity("none");
$serial->confCharacterLength(8);
$serial->confStopBits(1);
$serial->confFlowControl("none");

// Entonces tenemos que abrirlo
$serial->deviceOpen();

//Leer mensaje
$read = $serial->readPort();
//Enviar mensaje
$serial->sendMessage("Hello !");

//Si deseamos cambiar la configuración, el dispositivo debe estar cerrado.
$serial->deviceClose();

<?
```

Figura 59. Funcionamiento de la clase "PhpSerial.php".

Fuente: Producción propia

Diseño e implementación del módulo de consola serie

Teniendo en cuenta el funcionamiento de la clase *phpSerial.php* quedó por definir la creación de una ventana en la página web donde se mostrarán los mensajes en la consola serie de la placa Arduino. Para acercarse a esta idea, primero se hicieron pruebas, y a través de la realización de un debug de la clase “phpSerial”, se observó que solo se alcanzan a leer datos en una fracción de tiempo tan corta, que es imposible que se carguen los datos en la variable *“read”* (Ver figura 59). Por lo tanto, bajo estas circunstancias, se planificó este módulo tal como se ve en el esquema realizado en la figura 60 donde se tiene en la ventana del navegador, dos datos que se seleccionan. Por una parte, la tasa de baudios es un dato importante a tener en cuenta a la hora de configurar el puerto serie. Por otra parte, los segundos, cuyo dato es relevante, ya que a través de una función llamada **sleep**, propia de PHP, se puede retrasar la ejecución del programa durante un número de segundos (determinados por el usuario) para finalmente capturar la mayor cantidad de datos provenientes del puerto serie.

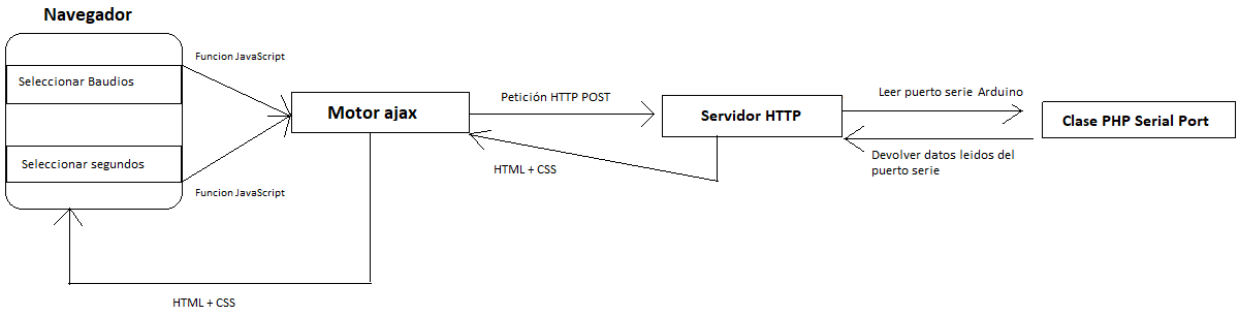


Figura 60. Esquema para hacer funcionar al monitor serie.

Fuente: Producción propia

Luego de realizar el esquema, se diseñó la ventana donde se deben seleccionar la tasa de baudios y los segundos, los cuales indicarán el tiempo de captura de datos del puerto serie (Ver figura 61).

Tasa de baudios posibles de seleccionar: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200.

Segundos posibles de seleccionar: 5, 10, 20, 40, 60, 120.

Figura 61. Ventana del monitor serie.

Fuente: Producción propia

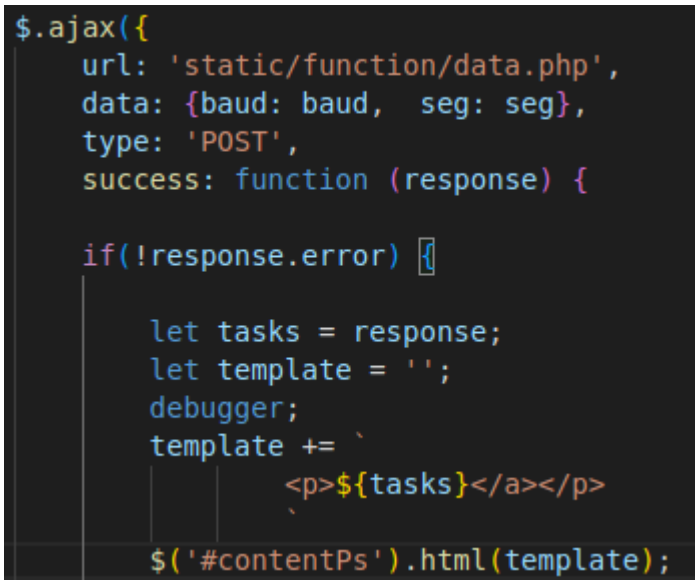
La ventana muestra dos botones: Iniciar y Limpiar. La función del botón iniciar, técnicamente, lo que hace, es activar un evento (click) de JavaScript donde se dispara una petición POST a través del Motor de Ajax al servidor (Ver Figura 62). El servidor toma en cuenta dos datos que

se enviarán a través de la petición POST, uno es la tasa de baudios y otro son los segundos. El servidor manejará estos datos de la siguiente manera:

- La tasa de baudios se pasará por parámetro a la función *confBaudRate* proveniente de la clase PhpSerial.php (Ver Figura 63).
- Los segundos se pasarán a la función sleep() de PHP. (Ver Figura 63).

En la figura 62 se puede observar la petición hecha a través de AJAX, donde se enviarán los datos al servidor y en caso de no haber un error en la respuesta del servidor, se agrega el contenido de la variable "*read"* (Ver figura 63) a un de HTML con el ID *"contentPs".*

Finalmente, en la figura 63 se puede ver el código PHP del lado servidor donde maneja, tanto la tasa de baudios, como el tiempo en segundos que se van a estar capturando datos del puerto serial.



```
$.ajax({
    url: 'static/function/data.php',
    data: {baud: baud,  seg: seg},
    type: 'POST',
    success: function (response) {

    if(!response.error) {

        let tasks = response;
        let template = '';
        debugger;
        template += `
                <p>${tasks}</a></p>
                `
        $('#contentPs').html(template);
```

Figura 62. Petición vía AJAX de datos provenientes del monitor serie.

Fuente: Producción propia

```php
<?php
include "phpSerialPort/php_serial.class.php";

$baudios =(int)$_POST['baud'];
$seg =(int)$_POST['seg'];

if(!empty($baudios)){

    $serial = new phpSerial();
    // Especificamos el dispositivo.
    $serial->deviceSet("/dev/ttyACM0");
    // Set for 9600-8-N-1 (no flow control)
    $serial->confBaudRate($baudios); //Baud rate: 9600
    $serial->confParity("none");  //Parity (this is the "N" in "8-N-1")
    $serial->confCharacterLength(8); //Character length     (this is the "8" in "8-N-1"
    $serial->confStopBits(1);  //Stop bits (this is the "1" in "8-N-1")
    $serial->confFlowControl("none");

    // Abrimos serial port.
    $serial->deviceOpen();

    sleep($seg);//delay, ya que sino se ejecuta el puerto serial vacio
    // Read data
    $read = $serial->readPort();
    echo $read;

    // Cerramos serial port.
    $serial->deviceClose();

}
```

Figura 63. Código PHP servidor de la gestión del monitor serial.

Fuente: Producción propia

Incorporación del teclado matricial a la página

De manera opcional, se agregó un teclado matricial al laboratorio remoto. Este mismo se desarrolló reutilizando la clase de "*PhpSerial*" de PHP (Ver figura 59). Así como se pueden leer datos del puerto serie, también se pueden enviar mediante la siguiente línea de código:
$serial->sendMessage();

En la anterior línea de código, entre paréntesis, se coloca el mensaje a enviar a la placa de Arduino que se tenga conectada mediante USB a la máquina. Teniendo en cuenta esto, cuando se presione una tecla del teclado matricial observado en la figura 64 se ejecutará una acción mediante JavaScript y el dato que corresponda a dicha tecla se enviará mediante una petición de AJAX al servidor.

En el lado servidor, se recibirá un dato (*$key=$_POST['key']*) en un script de PHP creado y este mismo dato se agregará entre paréntesis a la línea de código mencionada anteriormente y se tendrán conectadas mediante cables USB dos placas Arduino a la máquina servidor: Arduino uno y Arduino nano. Finalmente, el mensaje que se enviará será a la placa de Arduino Nano, por lo tanto, solo restará especificar la ruta del puerto de dicha placa (Ver figura 63) en la siguiente línea de código:

$serial->deviceSet("ruta del puerto USB de la placa de Arduino NANO")

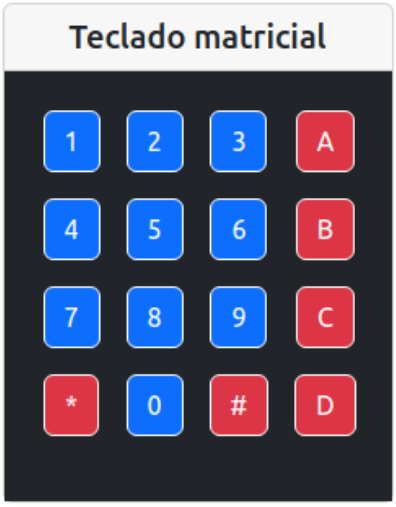


Figura 64. Teclado matricial.
Fuente: Producción propia

7.4.2 Implementación del Sistema de Gestión de reservas

Hasta el momento se implementó el laboratorio remoto. Pero, surge un problema cuando entran dos o más usuarios al mismo tiempo. Si ocurre esto, no se garantiza el funcionamiento adecuado de la placa Arduino y el Streaming, por lo que es necesario que el usuario, con el turno reservado, sea solamente quien pueda hacer peticiones a la plataforma (PlatformIO) que interactúa con el laboratorio físico. Para solucionar este problema, se desarrolló un Sistema De Reservas que bloquea los recursos durante un periodo de tiempo a todos los usuarios, menos a aquel que posea la reserva vigente.

7.4.2.1 Planificación del desarrollo del Sistema de Gestión de Reservas

A modo de planificación, el Sistema de Gestión de Reservas, deberá constar de los siguientes requisitos:

- El sistema debe tener un Login y un Registro. Esto es necesario, para segmentar el sistema por usuarios.

- Tiene que haber un formulario para agregar reserva. El formulario, constaría de un campo para agregar fecha y otro para agregar la hora de la reserva.
- Las reservas serán solo de 30 minutos.
- Los horarios para agregar reservas solo deberán ser horas que tengan minutos 00 y 30. La franja horaria debe ser de la siguiente manera: 00:30, 01:00, 01:30, 02:00, 02:30… y así sucesivamente hasta las 00:00 del otro día.
- Cuando el usuario seleccione la fecha aparecerá en una ventana los horarios reservados para ese día, incluyendo los de él como de otros usuarios.
- El sistema necesita restringir al usuario agregar una reserva que esté ocupada por el mismo o por otro usuario en esa fecha y hora.
- Cada usuario, tanto cuando inicie sesión como cuando termine de agregar la reservación, visualizara a través de una ventana, sus reservaciones.
- La ventana de reservaciones de cada usuario tiene que ser de tipo tabla, donde tendrá como encabezado (mínimamente) las siguientes columnas: Fecha (fecha del día de la reservación), Hora inicio (hora de inicio de la reservación), Hora fin (fecha fin de la reservación) y Acciones (Modificar, Borrar, Entrar).
- La acción de “Entrar” tendrá un botón que puede estar bloqueado o desbloqueado. Si está desbloqueado, cuando el usuario apriete click en el botón se le redireccionará a la URL donde se encuentra el laboratorio remoto.
- El botón para entrar a la reserva estará bloqueado y sin acceso hasta que llegue la fecha y hora de inicio de dicha reserva. Una vez desbloqueado, el usuario se encontrará habilitado para poder entrar al laboratorio remoto.
- Mientras haya una reserva en curso, a cualquier usuario externo se le restringirá el acceso al laboratorio remoto, incluso aunque acceda mediante la URL.
- El sistema debe estar restringido a los usuarios que no inicien sesión, independientemente si en el momento de tratar de acceder están en la fecha y en el rango de la hora de inicio y fin correctas de la reserva.
- La reserva tiene que estar activa hasta que cumpla su hora de finalización. Sin embargo, el usuario tendrá a disposición dentro de la web del laboratorio remoto, la posibilidad de eliminar la sesión correspondiente a la reserva.
- Dentro de la web del laboratorio remoto se contará con una ventana que tendrá la siguiente información: nombre de usuario de la sesión actual, la reserva activa (fecha, hora de inicio y hora de fin), el tiempo restante como un temporizador y un botón cuya función sea eliminar la sesión de la reserva actual.

- Un nuevo usuario puede utilizar un turno reservado en caso de que este se libere, quedando disponible el tiempo restante.

En la figura 65, se observa un esquema de cómo funciona el Sistema de Gestión de Reservas. A modo ejemplo, se supone que se tiene una reserva x la cual tiene una fecha del 30/10/2022, un horario de inicio a las 14:00 y un horario de fin a las 14:30. El usuario llamado “user1” fue el que agregó dicha reserva.

El “user1” inicia sesión a las 14:00 en el sistema. Por lo tanto, el laboratorio remoto ya está accesible únicamente para el “user1”. Por otro lado, si otro usuario, por ejemplo, “user3”, intenta ingresar al laboratorio mediante la URL, este mismo tendrá el acceso restringido. Otra suposición puede ser que el usuario “user1” ya utilizó el laboratorio remoto y decide eliminar la sesión de la reserva a las 14:15. En ese instante, inicia sesión otro usuario llamado “user3” agrega la reserva x, por lo tanto “user3” tendrá acceso al laboratorio remoto, pero el temporizador le marcara que le restan 15 minutos, hasta la hora fin de la reserva x, o sea, 14:30.

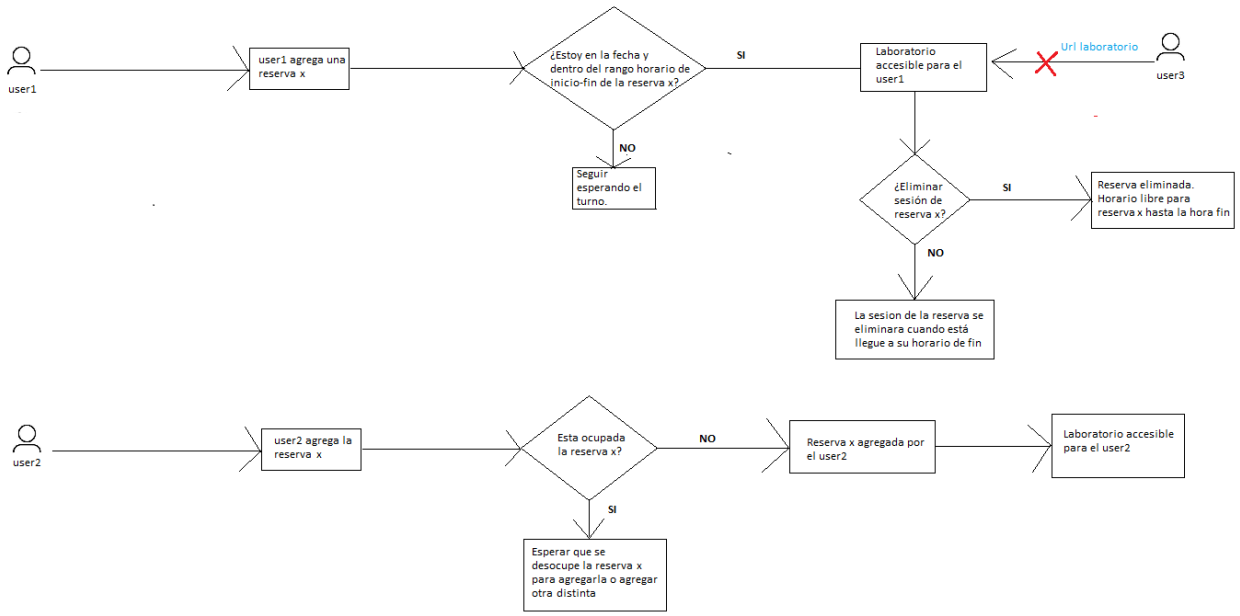


Figura 65. Esquema del funcionamiento referente al Sistema de Gestión de Reservas.

Fuente: Producción propia

7.4.2.2 Diseño de base de datos

Se creó una base de datos llamada “turnero”. La cual tiene dos tablas: “session” y “user”. La tabla “user” tiene los siguientes campos:

- **id:** int, no nulo, auto-incremental. Este campo es el identificador único de cada usuario que se va guardando.
- **username:** varchar. Este campo representa el usuario con el que se registra o inicia sesión al sistema.
- **password:** varchar. Este campo representa la contraseña con la que se registra o inicia sesión al sistema el usuario.
- **firstname:** varchar. Este campo representa el nombre del usuario
- **lastname:** varchar. Este campo representa el apellido del usuario

La tabla “session” tiene los siguientes campos:

- **id:** int, no nulo, auto-incremental. Este campo es el identificador único de cada sesión de reserva que se va guardando.
- **time_start:** time. Este campo representa la hora (formato [Horas, minutos, segundos]) en la que inicia la reserva.
- **time_end:** time. Este campo representa la hora (formato [Horas, minutos, segundos]) en la que finaliza la reserva.
- **duration**: int. Este campo es la duración de la sesión de la reserva, por lo tanto, siempre tendrá el valor de 30. Puede ser redundante, pero no afecta significativamente el rendimiento.
- **date:** date. Este campo es la fecha (formato [Año, mes, día]) en la que está registrada la reserva.
- **user**: int, clave foránea que hace referencia a la tabla “user”. Este campo representa a qué usuario le corresponde la sesión.
- **data_set:** datetime. Este campo toma por defecto el valor de fecha y hora que tenga el servidor en ese momento.

En la figura 66, se puede ver un diagrama representativo de la base de datos creada.

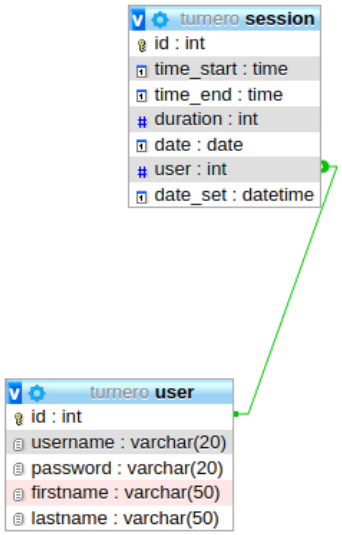


Figura 66. Diagrama de la Base de Datos "turnero".

Fuente: Producción propia

7.4.2.3 Creación del Sistema de Gestión de Reservas

Diseño de la interfaz del Login

En primer lugar, se diseñó la interfaz del Login como se ve en la figura 67.

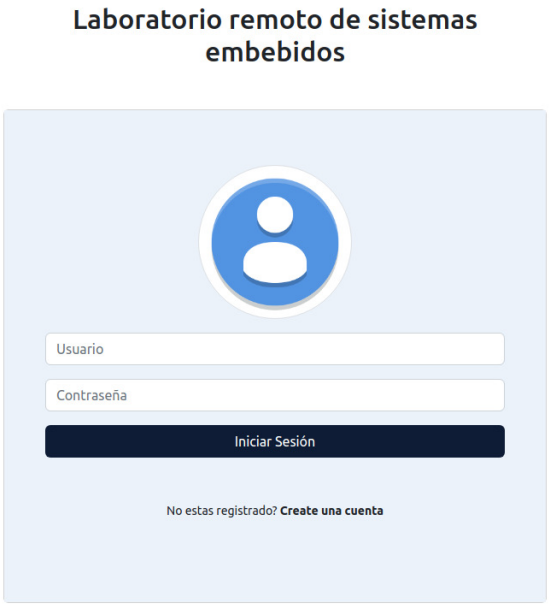


Figura 67. Interfaz del Login.

Fuente: Producción propia

Diseño de la interfaz del registro

En segundo lugar, se diseñó la interfaz del registro como se ve en la figura 68.

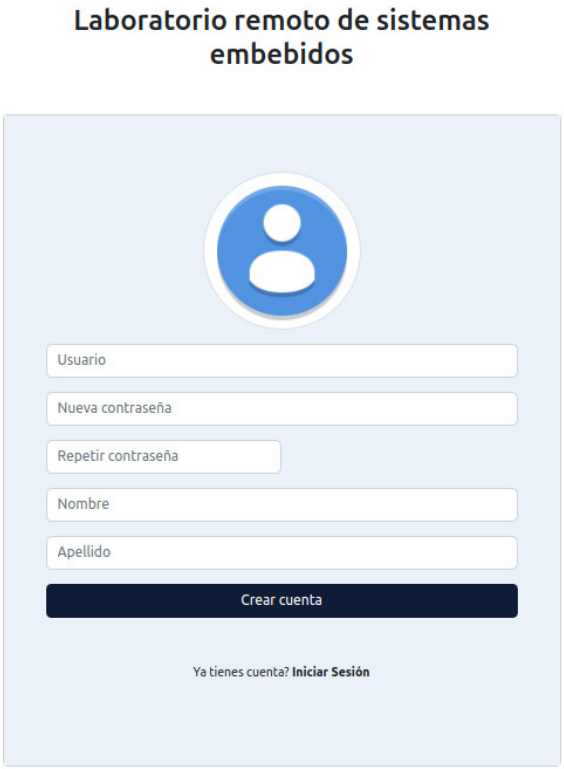


Figura 68. Interfaz del registro.

Fuente: Producción propia

Diseño de la interfaz de inicio del Sistema de Gestión de Reservas

Por último, se diseñó la interfaz del inicio del Sistema de Gestión de Reservas como se observa en la figura 69.

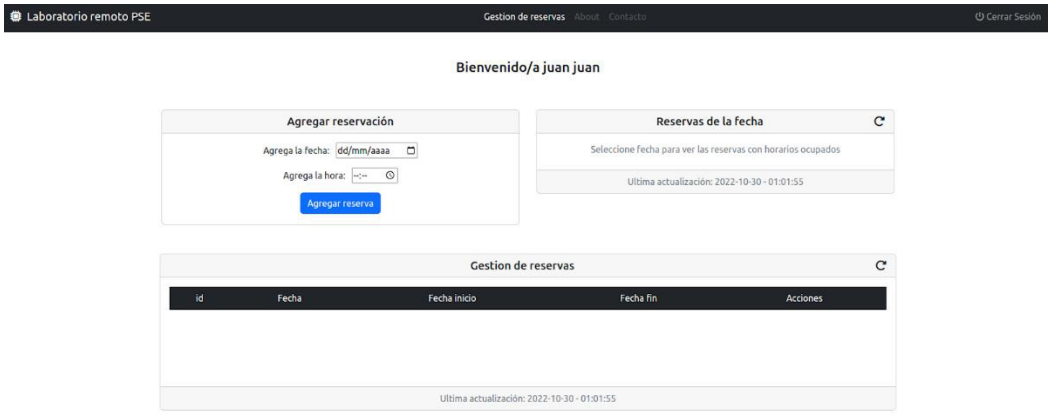


Figura 69. Interfaz del Inicio de la Gestión de Reservas.

Fuente: Producción propia

Desarrollo del registro

Como se ve en la figura 68, para registrarse se deben llenar todos los campos. Una vez que se llenen los campos, se agrega el registro a la base de datos. Todos estos campos se incluyeron dentro de la etiqueta <form> de HTML. Como se presenta en la figura 70 el método que se utilizó para llenar el formulario es POST. Cuando se clickea en “Crear cuenta” este formulario se envía a un script de PHP (el que está en el atributo “action” dentro de la etiqueta <form> de HTML). Por otro lado, las validaciones del formulario de registro fueron desarrolladas mediante JavaScript.

```
<form method='POST' id="formregistro" action='static/function/student_register.php'>
```

Figura 70. Formulario de registro HTML.

Fuente: Producción propia

En la figura 71 se muestra cómo se desarrolló este script de PHP para el registro. En primer lugar, se incluyó otro script llamado *“database.php”* que es para conectarse a la base de datos, donde este script tiene unas pocas líneas de código; las más relevantes son: *mysqli_connect(servidor, usuario, contraseña, nombre_database)* y *session_start()* que crea una sesión o reanuda la actual basada en un identificador de sesión pasado mediante una petición GET o POST, o pasado mediante una cookie. En segundo lugar, se creó una variable

llamada “exits” donde se declaró una sentencia de SQL que busca si en la tabla “users” existe algún usuario (*“username”*) igual al que se ingresó en el formulario de registro. En tercer lugar, se guardó el resultado en la variable “result”. Por último, se creó la variable “count” para verificar el número de registros encontrados (sería 1 si encontró un usuario con el mismo *“username”* ingresado el formulario y sería 0 si no encontró ninguno). En caso de no existir *(no se encontró un usuario con ese “username”)*, se inserta el registro a la tabla “users” de la base de datos y además se inicia sesión. En caso contrario, aparecerá una alerta en la web que le avisa al usuario que ya existe un usuario con el “*username”* que está intentando crear.

```php
<?php
include 'database.php';

$username = $_REQUEST['username'];
$password = $_REQUEST['pass1'];
$firstname = $_REQUEST['firstname'];
$lastname = $_REQUEST['lastname'];
$exits  = "SELECT * FROM `user` WHERE username= '$username'";

$result = mysqli_query($link, $exits);
$count = mysqli_num_rows($result);

if ($count == 0) {

    $sql = "INSERT INTO user(username, password, firstname, lastname) VALUES ('$username','$password','$firstname','$lastname')";
    if (mysqli_query($link, $sql)) {

        session_start();
        $query2    = "SELECT * FROM `user` WHERE username='$username'
            AND password='$password'";
        $result2 = mysqli_query($link, $query2) or die(mysqli_error($link));
        $user = mysqli_fetch_array($result2);
        $_SESSION["id"] = $user['id'];
        $_SESSION["user"] = $username;
        $_SESSION["pass"] = $password;
        $_SESSION['firstname'] = $user['firstname'];
        $_SESSION['lastname'] = $user['lastname'];

        echo "  <script type='text/javascript'>
        alert('Usuario registrado');
        window.location='../../index.php?';
        </script>";
    }
} else {

    echo "  <script type='text/javascript'>
        alert('El usuario ingresado ya existe');
        window.location='../../index.php?';
        </script>";
}
```

Figura 71. Script PHP para el registro de usuario.

Fuente: Producción propia

Desarrollo del login

En el caso del Login, fue similar al registro, se deben llenar dos campos: usuario y contraseña (Ver figura 67). O sea, se aplicó la misma lógica, otro formulario HTML, método POST y script PHP que se va a ejecutar cuando se haga click en Iniciar Sesión. El script de PHP para el inicio de sesión se observa en la figura 72. Básicamente, se realiza una consulta

de SQL para verificar si existe un usuario registrado con *username y password* que se ingresaron en los campos de inicio de sesión. Si no existe, le aparecerá una alerta al usuario que le avisa que las credenciales son incorrectas y en caso de existir se crea la sesión con todos los datos del usuario y mediante una función de javascript “windows.location” se redirige a main.php que es donde se encuentra finalmente el inicio del Sistema de Gestión de Reservas (Ver figura 69).

```php
<?php
    include 'database.php';
    // Cuando se envíe el formulario, verifique y cree una sesión de usuario.
    if (isset($_POST['username'])) {
        $username = $_REQUEST['username'];
        $password = $_REQUEST['password'];
        // Comprobar que el usuario existe en la base de datos
        $query    = "SELECT * FROM `user` WHERE username='$username'
                     AND password='$password'";
        $result = mysqli_query($link, $query) or die(mysqli_error($link));
        $count = mysqli_num_rows($result);
        if($count == 0){
            echo "  <script type='text/javascript'>
                        alert('Incorrect Credentials');
                        window.location='../../index.php';
                    </script>";
        }
        else {
            $user = mysqli_fetch_array($result);
            $_SESSION["id"] = $user['id'];
            $_SESSION['user'] = $username;
            $_SESSION['pass'] = $password;
            $_SESSION['firstname'] = $user['firstname'];
            $_SESSION['lastname'] = $user['lastname'];

            echo "  <script type='text/javascript'>
                        window.location='../../main.php';
                    </script>";

            }
    }
```

Figura 72. Script PHP para iniciar sesión.

Fuente: Producción propia

Medidas de seguridad

Se crearon scripts de PHP para evitar que los usuarios puedan entrar al sistema sin autenticarse, que puedan entrar al laboratorio remoto sin autenticarse y que puedan entrar al laboratorio remoto sin sacar una reserva.

Desarrollo de las funcionalidades del Sistema de Gestión de Reservas

A continuación, se procederá a describir cómo se desarrollaron las funcionalidades del Sistema de Gestión de Reservas.

- **Agregación de reserva**

En primera instancia, se agrega una reserva como se ve en la figura 73, la cual se agrega a la tabla “session” de la base de datos. Los campos de agregar fecha y agregar hora forman parte de la etiqueta <form> de HTML como se ve en la figura 74. El atributo *“step=”1800””* de la etiqueta <input> es representativa del campo de “agregar hora” y sirve para validar que se agreguen solo horarios 02:00, 02:30, 03:00 …etc. Por lo tanto, si se selecciona un horario que no sea exactamente así, aparecerá una alerta que avisará al usuario que debe seleccionar horarios con esas características.

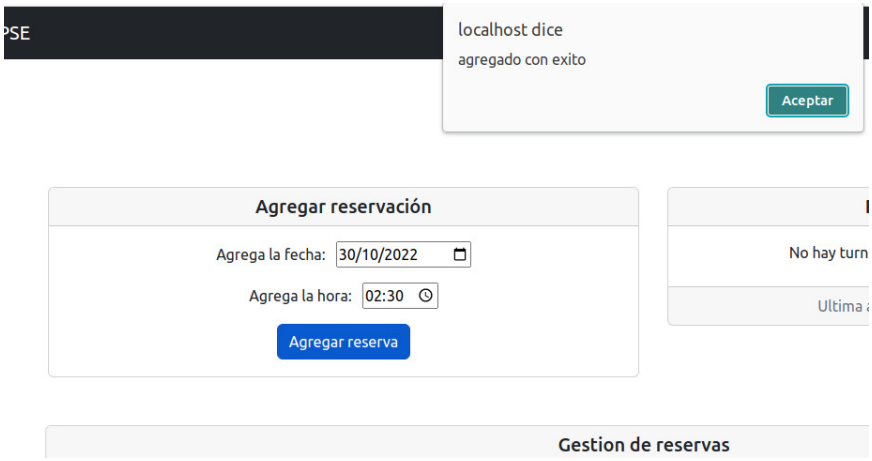


Figura 73. Agregar reserva.

Fuente: Producción propia

```
<form class="form-inline justify-content-center" id="data" method="post">


    <input type='hidden' name='user_id' value='<?php echo $_SESSION['id']; ?>'>


    <label for="date">
        <p class="card-text">Agrega la fecha:   </p>
    </label>
    <input id='date' type='date' name="date" class="sendDat" required>

    <p class="card-text mt-3">
        <label for="date">Agrega la hora:  </label>
        <input type='time' name='time_start' id='time_start' class="" step="1800" required>
    </p>
    <input type='hidden' id='duration' value='30' name='duration'>

    <input type="submit" class="btn btn-primary" value="Agregar reserva"></input>


</form>
```

Figura 74. Formulario de reserva.

Fuente: Producción propia

Cuando se envíe el formulario, se hará a través de una petición por AJAX como se presenta en la figura 75. El atributo *serialize ()* lo que hace es enviar todos los datos que se han llenado en los campos del formulario de reserva.

```
$("#data").submit(function(e) {

    e.preventDefault();

    $.ajax({
        url: 'static/function/insert_reservation.php',
        type: 'POST',
        data: $('#data').serialize(),
        processData: false,
        cache: false,
        success: function(response) {
```

Figura 75. Petición vía AJAX para agregar reserva.

Fuente: Producción propia

En la figura 76, se observa el script PHP al que se redirige la petición mediante AJAX. En primer lugar, se guardarán los datos necesarios en variables de PHP para utilizarlos en la sentencia de SQL guardada en la variable *"compDate"*. Además, la variable "*time_end"* tiene una función de PHP que va a recibir atributos que son necesarios para construir la hora de finalización de la reserva. En segundo lugar, como se muestra en la figura 77, en caso de que *"date"* (proveniente del formulario) sea mayor que la fecha actual se ejecutarán las demás líneas de código, ya que es necesario que la fecha que se elija no sea anterior a la actual. En último lugar, se verifica que no exista una reservación con los datos que se ingresaron en los

campos del formulario, y en caso de que no exista, se agrega finalmente la reservación. La petición de AJAX (Ver figura 75) tendrá una respuesta de parte del servidor y se evaluará mediante condicionales que tipo de “código” se recibió (Ver figura 77) y en base a eso se creará una alerta correspondiente para cada caso.

```php
<?php
include "database.php";
    $user =  $_REQUEST['user_id'];
    $date =  $_REQUEST['date'];
    $time_start =  $_REQUEST['time_start'];
    $duration = $_REQUEST['duration'];
    $time_end = date('H:i:s',strtotime($time_start." +$duration mins"));
    $compDate  = "SELECT * FROM `session` WHERE time_start= '$time_start' AND time_end= '$time_end' AND date = '$date'";
    $result = mysqli_query($link, $compDate) ;
    $count = mysqli_num_rows($result);
    $dateActual = date('Y-m-d');
```

Figura 76. Script PHP para agregar reserva.

Fuente: Producción propia

```php
if($date  >= $dateActual){

    if($count == 0){


        $sql = "INSERT INTO `session`(`time_start`, `time_end`, `duration`, `date`, `user`, `date_set`) VALUES ('$time_start','$time_end','$d

        //}

    if (mysqli_query($link, $sql)) {
      echo "1";
    }
    else {
      echo "Error: " . $sql . "<br>" . mysqli_error($link);
    }
    }

    else{

        echo "2";
    }

} else{

    echo "3";
}
```

Figura 77. Script PHP para agregar reserva e insertar datos.

Fuente: Producción propia

- **Tabla de la gestión de reservas**

El proceso para cargar la reserva cuando se agrega a la tabla (de la figura 78) es el siguiente. En primer lugar, si la reserva fue exitosa, es decir, que la respuesta fue “1” del lado del servidor (Ver figura 77), se hará otra petición AJAX para que se recargue la tabla, esto se observa en la figura 79.

Figura 78. Tabla de gestión de reservas.

Fuente: Producción propia

```
if (response == 1) {
    document.getElementById("data").reset();
    alert("agregado con exito");


    $.ajax({
        url: 'static/function/reload_sessions.php',
        type: 'POST',
        data: {},
        dataType: "html",
        success: function(response) {

            $('#lab-table tbody').html(response);


        }

    });
```

Figura 79. Petición vía AJAX para recargar la tabla de reservas.

Fuente: Producción propia

En segundo lugar, como se ve en la figura 80, se creó un script PHP para recargar la tabla de la gestión de reservas. Las sentencias de SQL DELETE ejecutadas son importantes porque se necesita que las reservas que hayan cumplido su ciclo se eliminen, tanto las de fechas anteriores, como las que hayan terminado el mismo día. La primera sentencia DELETE busca registros y los elimina. Deben ser registros que sean anteriores a la fecha actual o iguales y que, además, el tiempo de terminación de la reserva sea menor o igual a la hora actual. Por ejemplo, si, se tiene una reserva que comienza a las 14:00 y termina a las 14:30 y se accede al laboratorio remoto a las 14:15, la sentencia no encontrará tal registro, ya que, 14:30 no es menor o igual a 14:15. Sin embargo, si se accede a las 14:31, si se encontrara tal registro ya que 14:30 es menor a la hora actual y por lo tanto lo elimina. La segunda sentencia hace lo mismo, pero es más simple, ya que solo se evalúa en términos de fecha.

```php
<?php
include "database.php";

$student_id = $_SESSION['id'];


$date = date('Y-m-d');
$horaActual = date('H:i');



//DELETE POR SI EL USUARIO REFRESCA LA PAGINA EL MISMO DÍA
mysqli_query ($link, "DELETE FROM `session` WHERE `date` <= '$date' AND `time_end` <= '$horaActual' "); //Borramos todas las sessiones registradas anteriormente a

//DELETE POR SI EL USUARIO REFREZCA LA PAGINA OTRO DÍA
mysqli_query ($link, "DELETE FROM `session` WHERE `date` < '$date'"); //Borramos todas las sessiones registradas anteriormente a la fecha ACTUAL

$unlockedSession = mysqli_query ($link, "SELECT `id`  FROM `session` WHERE `time_start` <= '$horaActual' AND  `time_end` >=  '$horaActual' AND `date` = '$date'");

$idUnique = mysqli_fetch_array($unlockedSession);
$id_Unlocked = $idUnique['id'];


$session_selects = mysqli_query($link, "SELECT * FROM `session` WHERE user = '$student_id' ORDER BY `date`, `time_start` ");
```

Figura 80. Script PHP para recargar tabla de gestión de reservas. Parte 1.

Fuente: Producción propia

En tercer lugar, se observa una sentencia de SQL que se guarda y ejecuta en la variable *“unlockedSession”.* Esta sentencia tiene como finalidad ser usada para desbloquear el acceso del usuario al laboratorio remoto, siempre y cuando el usuario tenga una reserva activa. La sentencia busca la id de la sesión, cuya sesión, la hora de inicio sea menor o igual a la hora actual, que la hora final sea mayor o igual a la hora actual, y que la fecha sea igual a la actual. En palabras más simples, la sentencia busca un registro en el que la hora actual esté dentro del rango de la inicial y la final, si encuentra uno, significa que hay una sesión actualmente activa. La variable *id_unLocked* guarda el valor del id (PK de la tabla “sesión”) proveniente de la sentencia ejecutada en la variable mencionada anteriormente (“*unlockedSession”).* Además, se tiene la sentencia que se encuentra contenida en la variable *“session_selects”*. Esta simplemente lo que hace es traer todas las reservas del usuario actual.

En cuarto lugar, como se observa en la figura 81, una vez hecha la consulta de “*session_selects”* con un ciclo while, se empezará cargar cada registro en cada fila de la tabla de gestión de reservas.

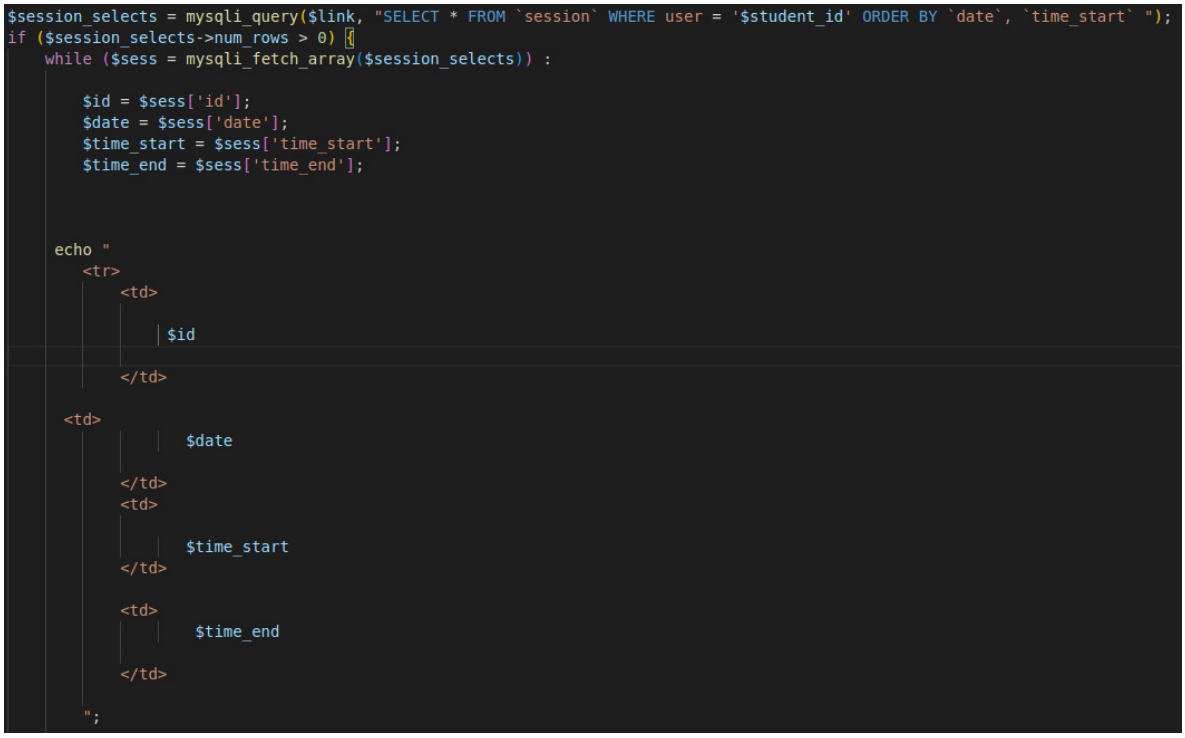


Figura 81. Script PHP para recargar la tabla de gestión de reservas. Parte 2.

Fuente: Producción propia

Por último, se observa en la figura 82, todo el bloque de código que está dentro del while (Ver Figura 81). Con cada ciclo while siempre se estará comprobando si alguna ID de la tabla "sessions" es igual a la *"id_Unlocked"*. En caso de que ambas ID coincidan, entonces existe una reserva activa. Por lo tanto, se agregará una etiqueta HTML *<form>* (como se ve remarcada en rojo en la figura 82) donde se tendrá un atributo *action* que se activará al clickear el ícono del candado desbloqueado (Ver figura 78), donde se redirige a un script de PHP llamado "lab.php". Este script PHP es donde está alojado el **laboratorio remoto**. En el caso de que las ID no coincidan simplemente no se agregará la etiqueta <form> a las acciones, sino un botón con un candado amarillo sin ningún uso más que mostrar al usuario que aún la reserva no está activa, o sea, bloqueada.

```php
if($id ==$id_Unlocked){

echo "
            <td>
            <div class='row-btn-containers'>
                <button class='btn btn-danger'>
                <i class='fa-solid fa-ban'></i></button>


                <button class='btn btn-danger'>
                <i class='fa-solid fa-ban'></i></button>

                <form class='' id='data' method='post' action='lab.php' style= 'display: inline;'>
                <input type='hidden' name='id_session' id = 'id_session' value='$id'>
                <button type = 'submit' class='btn btn-success'>
                <i class='fa-solid fa-unlock'></i></button>
                </form>

            </div>
        </td>
    </tr>


";
}else{

    echo "
    <td>
    <div class='row-btn-containers'>
        <button name = 'paid' class='btn btn-primary paid_button' value = '$id'>
        <i class='fa-solid fa-pen-to-square'></i></button>


        <button name = 'deleteRegister' class='btn btn-danger del_button' value = '$id'>
        <i class='fa-solid fa-trash'></i></button>

    <button class='btn btn-warning row-lock-btn'>
    <i class='fa-solid fa-lock'></i></button>
</div>
    </td>
</tr>

";
}
```

Figura 82. Script PHP para recargar tabla de la gestión de reservas. Parte 3.

Fuente: Producción propia

Nota: Todos los registros que trae el *while* son reservaciones del USUARIO ACTUAL, es decir, el usuario que inicia sesión. Es decir, se compara las id de todas las reservaciones del usuario en específico, con la id la reservación activa.

- **Modificación de reservas**

En el caso de modificar reservas, se debe apretar click en el ícono del lápiz azul y saldrá un modal con dos datos a modificar la fecha y el tiempo inicial, como se ve en la figura 83. En el modal hay más campos, solo que están ocultos y estos campos son los siguientes: id del usuario e id de la reserva. Cuando se le dé click en el botón de “Update” se realizará una petición a través de AJAX al script PHP observado en la figura 84, donde se ve primeramente

una sentencia SQL SELECT. Esta sentencia se utilizará para validar si ya existe una reservación a esa fecha y hora (de otro usuario). En caso de existir, se ejecutará el bloque de código contenido en el condicional *else* y se devolverá un número que representará “un código de error” que le advertirá al usuario mediante una alerta que ya existe una reservación a esa fecha y hora. Si no existe, se hará UPDATE (modificará) a la tabla de base de datos (session) utilizando los datos necesarios.

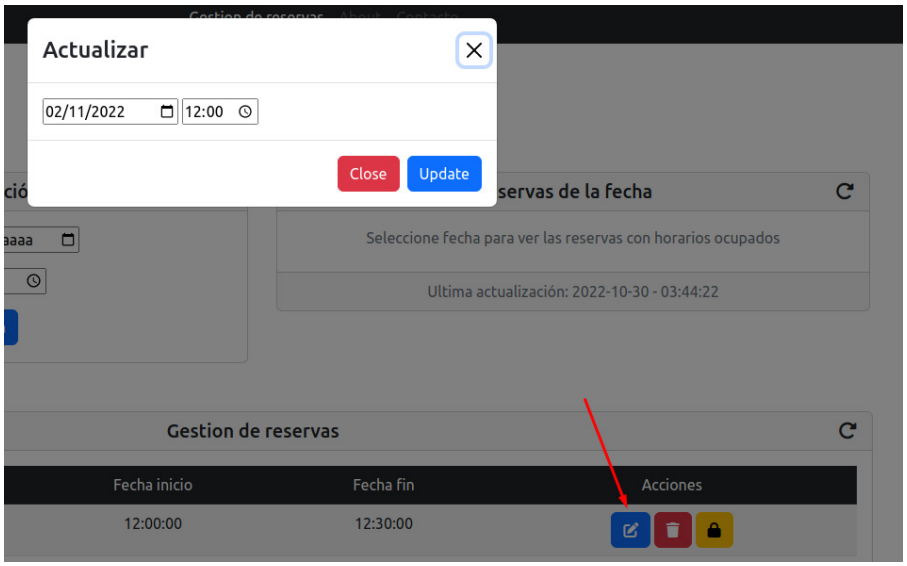


Figura 83. Modal Update reserva.

Fuente: Producción propia

```php
<?php

include 'database.php';

$id_user = $_SESSION['id'];
$id_session = $_POST['id_session'];
$date = $_POST['id_date'];
$time_start = $_POST['id_time'];
$duration = $_REQUEST['duration'];
$time_end = date('H:i:s',strtotime($time_start." +$duration mins"));


$existe= mysqli_query($link,"SELECT * FROM `session` WHERE date='$date' AND time_start='$time_start'");

$count_registro = mysqli_num_rows($existe);

if($count_registro==0){

$sql = "UPDATE `session` SET  `id` ='$id_session', `time_start`='$time_start', `time_end`='$time_end', `duration`='$duration', `date`='$date', `user` ='$id_user' WH

if(mysqli_query($link,$sql)){

    echo "1";
}

}else{

    echo "2";
}


?>
```

Figura 84. Script PHP para realizar la modificación de una reserva.

Fuente: Producción propia

- **Eliminación de reservas**

En el caso de eliminar reservas, se emplea una lógica similar a modificarlas, solo que la sentencia SQL es DELETE, como se ve en el modal en la figura 85 y el script PHP para eliminar una reserva se observa en la figura 86.

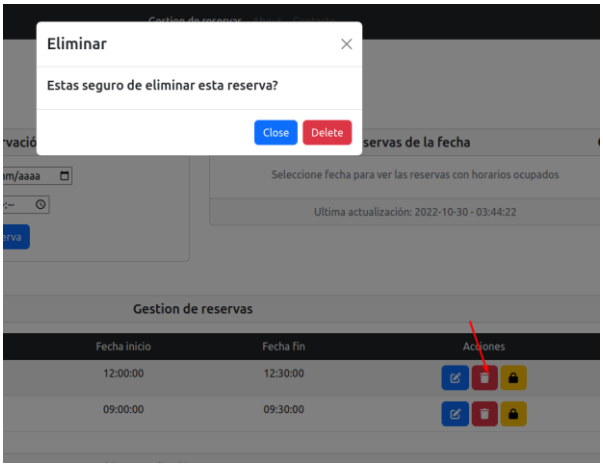


Figura 85. Modal DELETE reserva.

Fuente: Producción propia

```php
<?php
include 'database.php';

$IdDeleteSession = $_POST['date'];

$sql = "DELETE FROM `session`  WHERE `id` ='$IdDeleteSession'";

if(mysqli_query($link,$sql)){

    echo "1";
}



?>
```

Figura 86. Script PHP para realizar un eliminado de una reserva.

Fuente: Producción propia

- **Visualización de los horarios reservados al seleccionar una fecha**

Es importante para el usuario que el sistema le muestre las reservas (hechas por otros usuarios o el mismo incluso) que se han hecho en la fecha que seleccionó, ejemplo de esto se observa en la figura 87. El funcionamiento es de la siguiente manera: mediante un evento que detecta cuando el usuario cambia de fecha en el campo, se envía una petición mediante AJAX con la fecha seleccionada y el servidor devuelve una respuesta con la lista de reservas de dicha fecha y mediante una función de Jquery (biblioteca de JavaScript) se agregaron las listas al contenido de un de HTML.

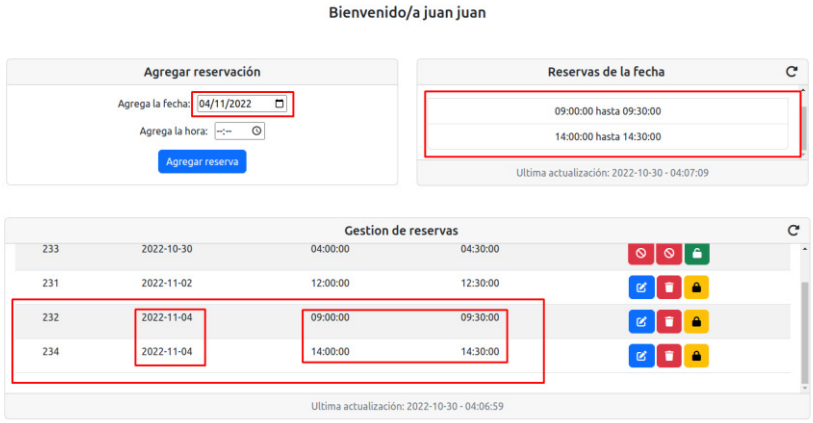


Figura 87. Reservas de la fecha.

Fuente: Producción propia

En la figura 88, se observa el script de PHP para cargar las reservas de la fecha. Existe una reutilización de este script en el caso de que se apriete el botón de actualizar en la ventana de “Reservas de la fecha” por si el usuario se ausenta y otro agrega una reserva en la misma fecha. Por lo tanto, hay dos tipos de fecha, la proveniente del botón actualizar y la del cambio de fecha del campo.

El funcionamiento del script es el siguiente: se tiene la variable *“dats”* que guarda la ejecución de una sentencia de SQL SELECT de las reservaciones que coincidan con la fecha proveniente de las variables de $_POST. Luego, se tiene un while que terminará ejecutando todos los registros en caso de que los haya, poniéndolos en una etiqueta <li> de HTML. Si no hay registros, saldrá un mensaje en la ventana de “reservas de la fecha” (Ver Figura 87) que no hay turnos reservados para esa fecha.

```php
<?php

include "database.php";
//fecha proveniente del evento key
$date = $_POST['date'];

//fecha proveniente del evento de click en actualizar

$fecha = $_POST['fecha'];

if (isset($fecha)) {

    $date = $fecha;

}


$dats = mysqli_query($link,"SELECT * FROM `session` WHERE date='$date' ORDER BY `time_start` ASC");

$count = mysqli_num_rows($dats);

if (!(empty($date))){

if(!($count == 0)){

echo "<p class='card-text'> Turnos reservados en la fecha: $date</p>";
while($dat=mysqli_fetch_array($dats)):
    $time_start = $dat['time_start'];
    $time_end = $dat['time_end'];
    // echo "$time_start'>$time_end";
    echo "<li class='list-group-item'> $time_start hasta $time_end </li>";

endwhile;

}else{

    echo "<p class='card-text'>No hay turnos reservados para la fecha: $date </p>";

}
} else{

    echo "<p class='card-text text-muted'>Seleccione fecha para ver las reservas con horarios ocupados</p>";

}
```

Figura 88. Script PHP para cargar las reservas de la fecha.

Fuente: Producción propia

- **Gestión de la sesión**

La gestión de la sesión es una ventana, como se observa en la figura 89, que se diseñó dentro del laboratorio remoto para que el usuario tenga un control de cuánto es el tiempo restante que tiene su reservación y que, además, tenga la posibilidad de eliminar la sesión para poder darle la oportunidad a otro alumno que tal vez quiera usar el laboratorio en ese momento.

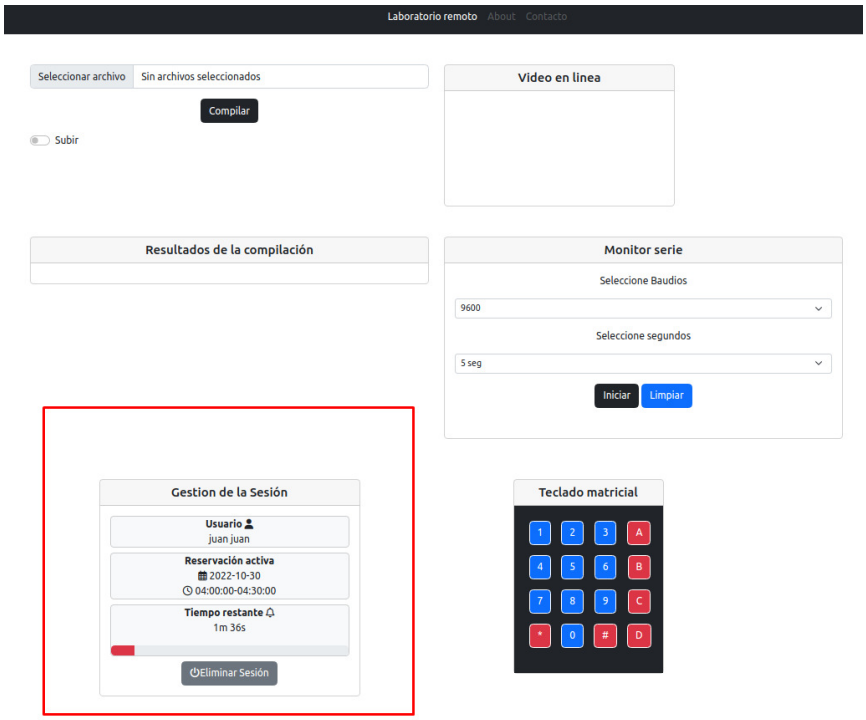


Figura 89. Gestión de la sesión.

Fuente: Producción propia

Funcionamiento de la gestión de la sesión

El funcionamiento de la cuenta regresiva está explicado en los comentarios del código que se presenta en la figura 90.

```
//Establecer la fecha en la que estamos contando hacia atrás. En este caso solo es la hora final de la reserva.
var countDownDate = new Date("<?php echo $dateAndTimeEnd ?>").getTime();

// Actualizar la cuenta regresiva cada 1 segundo
var x = setInterval(function() {

    // Obtener la fecha y hora de hoy
    var now = new Date().getTime();

    //var test = new Date().getHours();
    // Encuentra la distancia entre ahora y la fecha de la cuenta regresiva
    var distance = countDownDate - now;

    // Cálculos de tiempo para días, horas, minutos y segundos.
    var days = Math.floor(distance / (1000 * 60 * 60 * 24));
    var hours = Math.floor((distance % (1000 * 60 * 60 * 24)) / (1000 * 60 * 60));
    var minutes = Math.floor((distance % (1000 * 60 * 60)) / (1000 * 60));
    var seconds = Math.floor((distance % (1000 * 60)) / 1000);

    // Muestra el resultado en un elemento con id="demo"
    document.getElementById("wrapper-timer").innerHTML = minutes + "m " + seconds + "s ";



    // Si la cuenta regresiva ha terminado, volvemos a la gestión de reservas. Se termino la sesión
    if (distance < 0) {
        clearInterval(x);
        window.location.replace('/main.php');

    }

    if (minutes <= 30 && minutes > 25) {

        $("#progress").css("width", "100%");

    }
```

Figura 90. Cuenta regresiva. Código.

Fuente: Producción propia

Cuando se le dé click a el botón “Eliminar Sesión” (Ver figura 89) se crea un evento de JavaScript donde se realiza una petición mediante AJAX como se ve en la figura 91. En dicha petición se envían, a través del método POST, datos al servidor. Del lado del servidor, como se observa en la figura 92, se obtiene la ID de la sesión actual y con una sentencia SQL DELETE se eliminará la sesión. Por lo tanto, la respuesta de la petición será “1” y el “If” será true (Ver figura 91). Finalmente, se le redirigirá al usuario a través del método de JavaScript *“windows.location.replace”* a la url de *“main.php”* que es donde se encuentra el inicio de la gestión de reservas.

```
$(document).on("click", ".btn-secondary", function() {

    var val = $(this).val();

    $.ajax({
        type: "POST",
        url: "static/function/delete_sesion_lab.php",
        data: {
            date: val
        },

        success: function(response) {


            if (response == 1) {

                window.location.replace('/main.php');

            }

        }
    });


});
```

Figura 91. Petición mediante AJAX para eliminar sesión.

Fuente: Producción propia

```
<?php
include 'database.php';

$IdDeleteSession = $_POST['date'];

$sql = "DELETE FROM `session`  WHERE `id` ='$IdDeleteSession'";

if(mysqli_query($link,$sql)){

    echo "1";
}



?>
```

Figura 92. Script PHP para eliminar sesión.

Fuente: Producción propia

Script PHP para recargar tabla de gestión de reservas

Las demás funcionalidades dependen mucho del script PHP que se creó para recargar la tabla de gestión de reservas (Ver figura 80, 81 y 82). Este script PHP se reutiliza al hacer click en el botón actualizar de la tabla de gestión de reservas (ver Figura 78), al modificar una reserva, al eliminar una reserva, al agregar una reserva nueva, cuando se finaliza la sesión de una reserva, cuando se elimina la sesión de la reserva y cuando se recarga la página.

Cerrar sesión

Finalmente, queda explicar un pequeño script de PHP para cerrar sesión del sistema como se observa en la figura 93. Con la función *sessión_unset()* de PHP se pueden liberar todas las variables de la sesión y con la función *session_destroy* se destruye toda la información registrada de la sesión.

```php
<?php
    include "database.php";
    session_unset();
    session_destroy();
    echo "  <script type='text/javascript'>
            window.location.replace('/index.php');
            </script>";

?>
```

Figura 93. Script PHP para cerrar sesión.

Fuente: Producción propia

8. Prueba de funcionamiento y resultados

Funcionamiento del host-dominio

En la sección 7.3 (Implementar servidor WEB) se explicó la configuración hecha para que funcione correctamente el host-dominio. La URL, a modo de prueba, quedó como *http://laboremotounaj.ddns.net/*. Si se ingresa por el navegador, está funcionando correctamente como se observa en la figura 94.

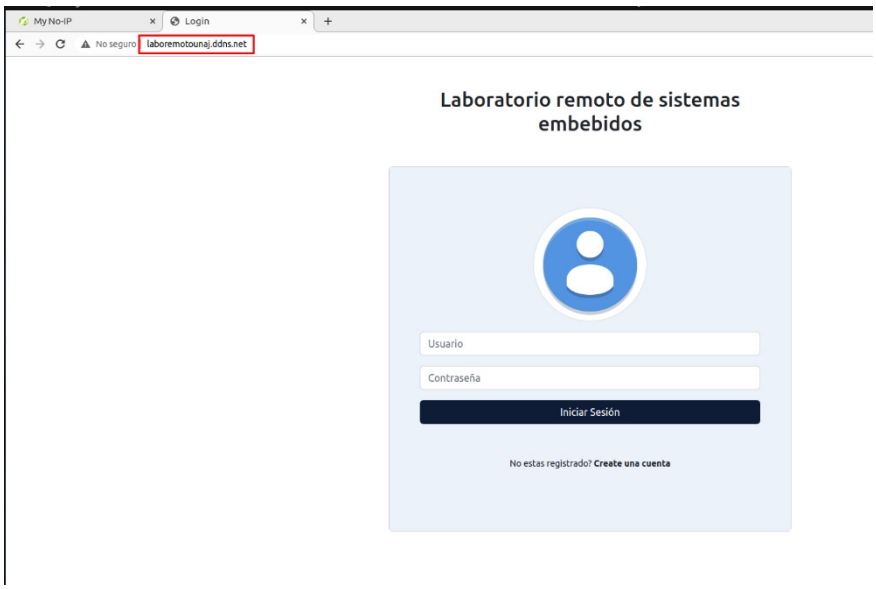


Figura 94. Funcionamiento de la página web.

Fuente: Producción propia

Funcionamiento de registro

- **Validaciones**

Validación de contraseña

Cuando se ingresa una contraseña nueva, se debe confirmar. En caso de que la confirmación sea distinta a la contraseña elegida como nueva, el sistema desactiva el botón de "Crear cuenta". (Ver figura 95)

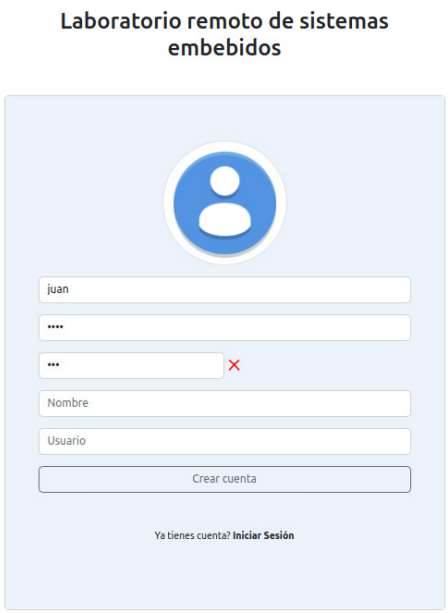


Figura 95. Validación de contraseñas.

Fuente: Producción propia

Validación de campos

Cuando se registra un usuario, pero falta completar un campo, saldrá una advertencia. (Ver figura 96).

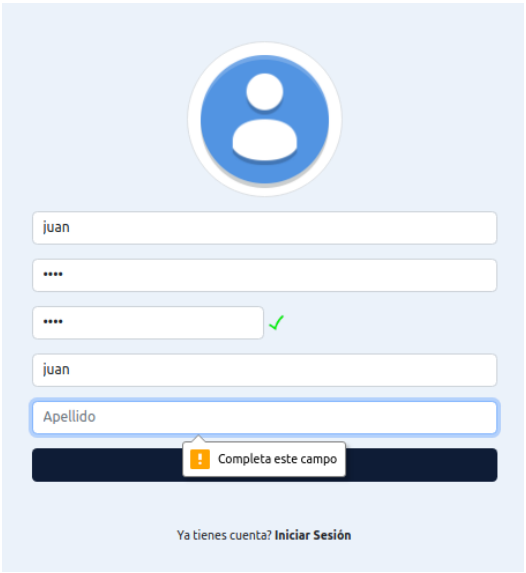


Figura 96. Validación de apellido.

Fuente: Producción propia

Validación del registro en general

En primer lugar, se registra el usuario, por ejemplo, con las credenciales que se ven en la figura 97:

- Usuario: juan
- Contraseña: 1234
- Nombre: Juan
- Apellido: Piñeiro

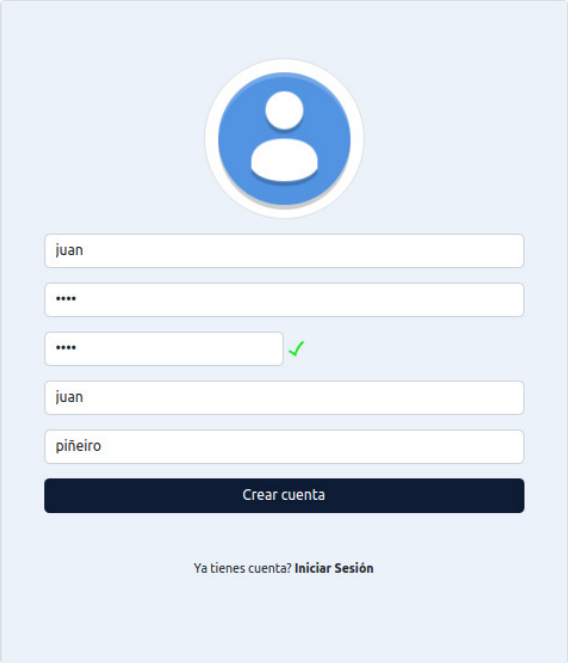


Figura 97. Registro de usuario.

Fuente: Producción propia

Por último, una vez registrado el usuario, se procederá a validar el inicio de sesión.

Funcionamiento del inicio de sesión

- **Validaciones**

Validación de autenticación

Siguiendo el ejemplo, con las credenciales registradas anteriormente, se intentó iniciar sesión, poniendo un usuario distinto y el resultado es lo que se ve en la figura 98. Lo mismo sucede si se ingresa una contraseña errónea.

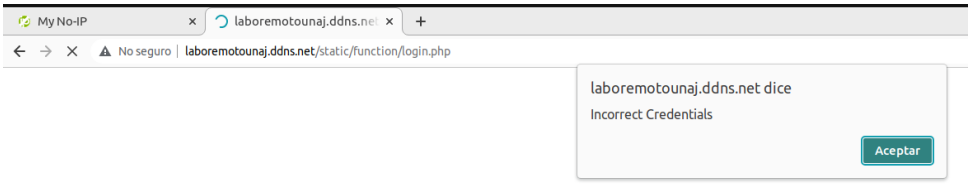


Figura 98. Intento de autenticación fallida.

Fuente: Producción propia

Validación de inicio de sesión correcto

Si se ingresan las credenciales correctas para el inicio de sesión. La pantalla de inicio sería la que se ve en la figura 99.

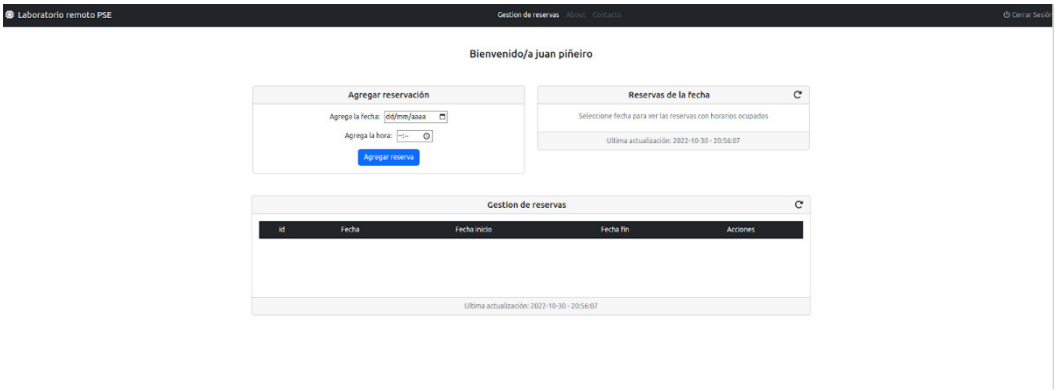


Figura 99. Pantalla de inicio del sistema.

Fuente: Producción propia

Funcionamiento del Sistema de Gestión de Reservas

Agregación de reserva

Si se agrega una reserva ingresando la fecha y la hora (como se mencionó en sección 7.4.2.2), en la figura 100 se presenta qué ocurrirá lo siguiente: la reserva se agregará, luego de presionar el botón “aceptar”, a la tabla de gestión de reservas como se ve en la figura 101. En este caso, al haber sacado la reserva en la fecha y hora actuales, se desbloqueará la web del laboratorio remoto para el usuario, que puede ingresar dando click al ícono de candado verde.

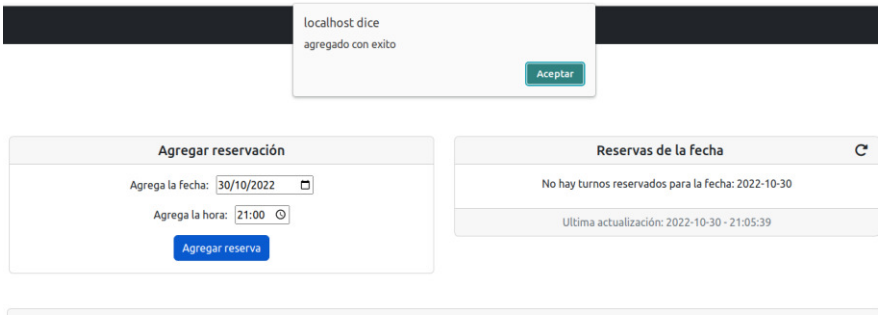


Figura 100. Agregar una reserva.

Fuente: Producción propia

Figura 101. Reserva agregada a la tabla.

Fuente: Producción propia

- **Agregación de reserva – validaciones**

Agregar una reserva a una fecha anterior a la actual

En caso de que la reserva se quiera registrar en una fecha anterior a la actual, se validará. Si el usuario la intenta hacer, saldrá una advertencia como se ve en la figura 102.

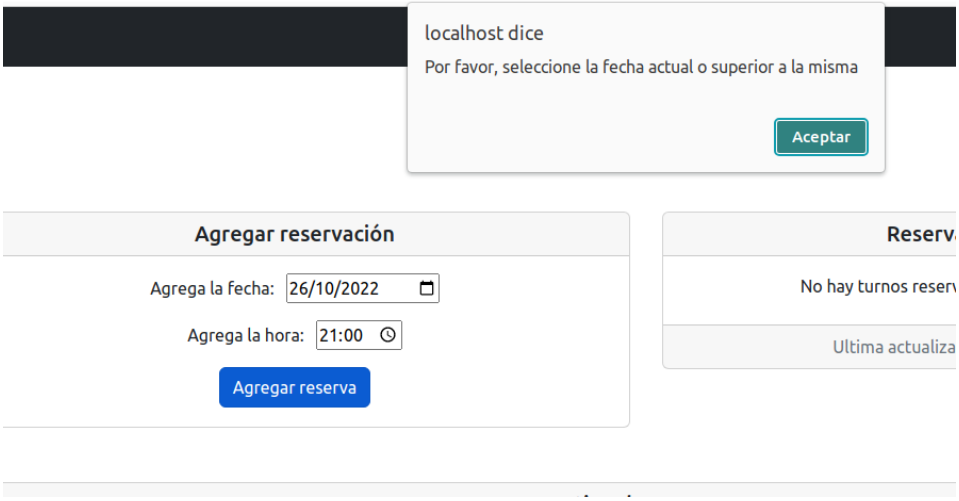


Figura 102. Validación de la fecha actual.

Fuente: Producción propia

Validación de reserva con dos cuentas

Supongamos que entra otro usuario llamado “test” como se ve en la figura 103. Este usuario agrega una reserva con la fecha 2022-11-06, con la hora de inicio 22:00 y por supuesto con la fecha de fin 22:30. Luego, entra a la cuenta el usuario “juan” (Ver figura 99) e intenta agregar la misma reserva que agregó el usuario “test”. Lo sucederá será lo siguiente:

- Cuando el usuario “juan” seleccione fecha en el campo, en la ventana de “reservas de fecha” le saldrá todos los horarios ocupados para tal fecha, incluido el horario de la reserva hecha por el usuario “test” como se observa en la figura 104.
- En caso de que el usuario “juan” intente agregar la reserva a la misma hora, le aparecerá una advertencia de que ya existe una reserva para esa fecha y hora, impidiendo agregar la reserva como se observa en la figura 105.

Figura 103. Validación de la existencia de otra reserva a la misma fecha y hora.

Fuente: Producción propia

Figura 104. Reservas de la fecha ocupadas.

Fuente: Producción propia

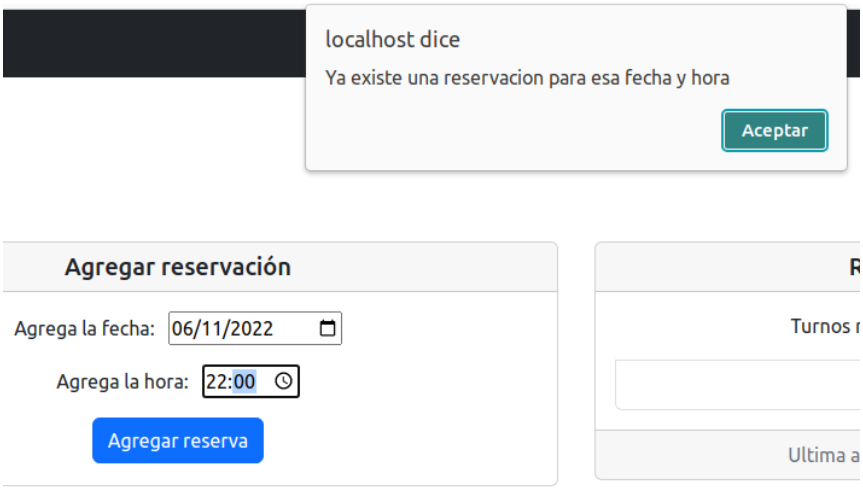


Figura 105. Validación de reserva a la misma fecha y hora.

Fuente: Producción propia

Modificación de una reserva

En caso de querer hacer una modificación de una reserva, se puede hacer simplemente clickeando el ícono del lápiz y saldrá un modal para ingresar los datos nuevos como se ve en la figura 106. Si se le da click al botón de “Update”, se modificará la reserva. (Ver figura 107).

Figura 106. UPDATE de reserva, modal.

Fuente: Producción propia

Figura 107. UPDATE realizado correctamente.

Fuente: Producción propia

- **Validaciones de modificar reserva**

En caso de intentar modificar la fecha y hora ya reservada por otro usuario, aparecerá un error como se ve en la figura 108.

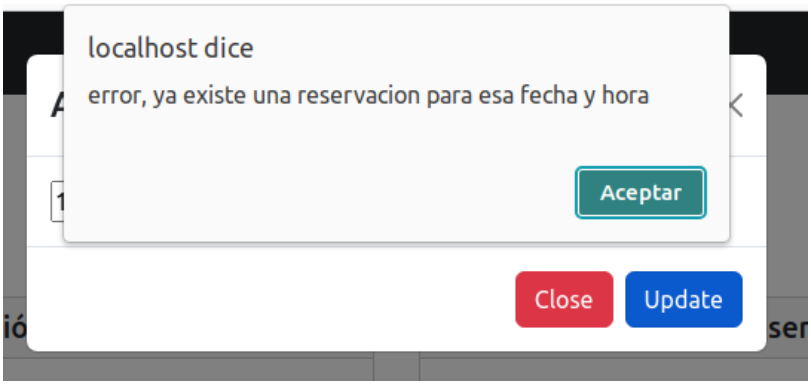


Figura 108. Error al realizar una modificación de la reserva.

Fuente: Producción propia

Eliminación de reserva

El eliminado de la reserva se realiza dándole click al ícono de basura rojo y confirmando a través de un modal si se quiere eliminar la reserva como se observa en la figura 109. En caso de que se le dé click al botón “Delete”, se eliminará correctamente la reserva como se observa en la figura 110.

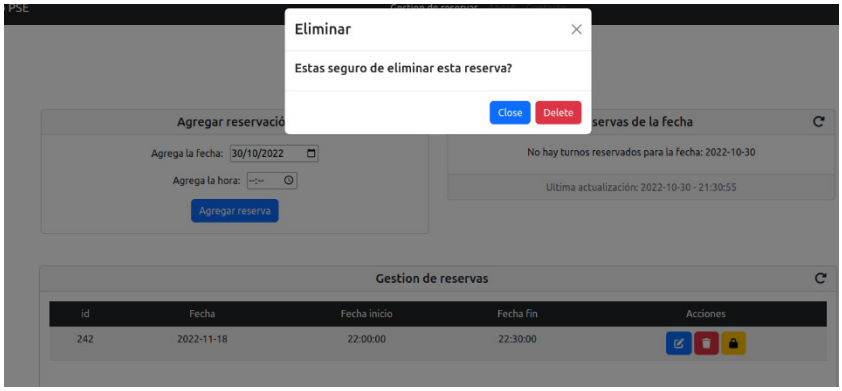


Figura 109. Eliminar reserva, modal.

Fuente: Producción propia

Figura 110. Reserva eliminada correctamente.

Fuente: Producción propia

Por otro lado, para eliminar la reserva desde la web del laboratorio remoto, eliminando la sesión activa, primero busca la ventana de la gestión de la sesión y posteriormente se le da click al botón “Eliminar Sesión” (Ver figura 111).

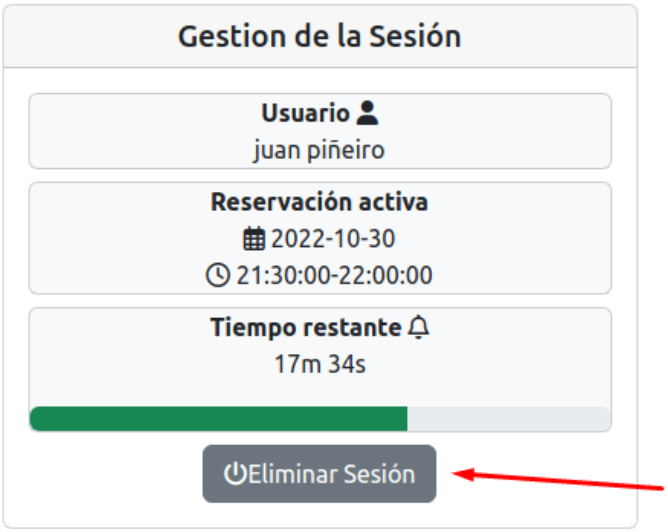


Figura 111. Eliminar sesión activa.

Fuente: Producción propia

Seguridad del sistema

- **Entrar a la web del laboratorio remoto sin iniciar sesión**

En este caso, si se ingresa la URL de la web del laboratorio remoto sin iniciar sesión (independientemente si se tiene una reserva activa) aparecerá una alerta que dirá que el acceso está restringido, tal como se observa en la figura 112. En caso de presionar el botón “Aceptar” se realiza una redirección a la página de inicio de sesión.

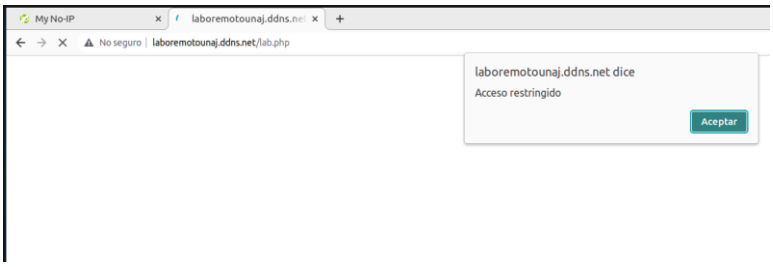


Figura 112. Acceso restringido a la web del laboratorio remoto.

Fuente: Producción propia

- **Entrar a la web del laboratorio remoto, iniciando sesión, pero sin sacar la reserva.**

En este caso, si se ingresa la URL de la web del laboratorio remoto con la sesión de usuario activa, aparecerá una alerta que dirá que se debe sacar turno (o reserva) para ingresar al laboratorio, tal como se observa en la figura 113. Si se presiona el botón “Aceptar” se realiza una redirección a la página de inicio del Sistema de Gestión de Reservas.

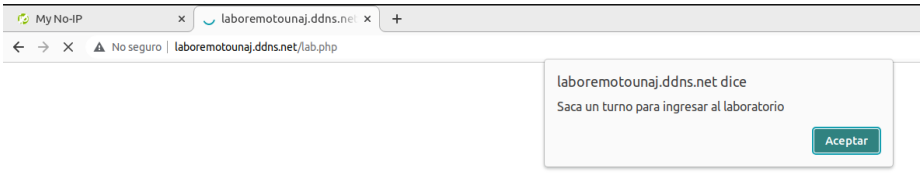


Figura 113. Acceso restringido a la web del laboratorio remoto con la sesión del usuario activa.

Fuente: Producción propia

Funcionamiento del laboratorio remoto

Ejemplo Arduino programa LCD

- **Compilación del programa**
    - **Caso de éxito**

Se presenta un programa (ver Figura 114) para subir a la placa de Arduino. Este programa lo que hará es dejar un mensaje en el LCD. En la primera fila del LCD aparecerá “END-Ing” y en la

segunda “TEST:” junto con un contador de segundos. El programa no tiene ningún error de sintaxis, por lo que debería compilar correctamente.

Se compiló correctamente en el laboratorio remoto como se ve en la figura 115.

```
#include <Arduino.h>
#include <Wire.h>


#include <LiquidCrystal.h> // Librería para que funcione el LCD

LiquidCrystal lcd(1, 0, A1, A2, 10, 9); // RS=1 , E=0, A1=D4, A2=D5, 10=D6, 9= D7 esta línea muestra en que pines debe ir conectado la pantalla LCD a nuestro Arduino

void setup() {
        lcd.begin(16, 2); // Con esta instrucción indicamos cual es la proporción de nuestro LCD
        lcd.print("END-ing-"); // Una instrucción muy importante ya que con esta imprimiremos el texto en nuestra pantalla
}

void loop() {
        lcd.setCursor(0, 1); // con esta línea damos a entender que la primera fila de nuestro LCd ya esta siendo usada por lo cual querenos usar la segunda
        lcd.print("TEST:"); // Al ser colocadar debajo de la instrucción anterior, este mensaje se mostrara en la segunda fila del LCD
        lcd.println(millis() / 1000); // Con esta instrucción realizamos un simple contador y con Println indicamos que querenos que se imprima en la segunda fila, después de la instrucción anterior.
}
```

Figura 114. Programa de Arduino para mostrar mensajes en el LCD.

Fuente: Producción propia

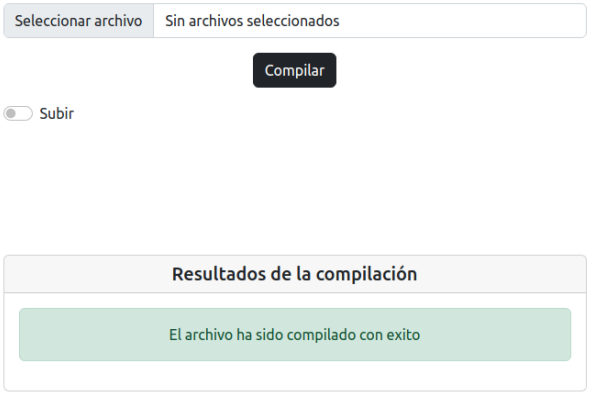


Figura 115. Compilación exitosa del programa de Arduino para mostrar mensajes en el LCD.

Fuente: Producción propia

- **Caso de error**

Supongamos que hay un error de sintaxis y falto cerrar paréntesis en el *void setup()* en la línea de código 9 (Ver figura 114). Por lo tanto, el resultado de compilación será el que se ve en la figura 116.

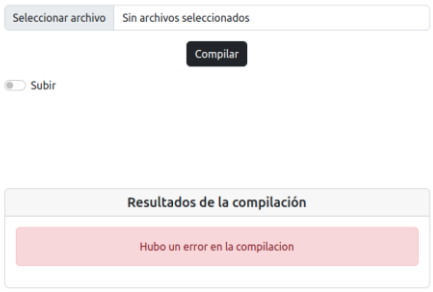


Figura 116. Compilación errónea del programa de Arduino para mostrar mensajes en el LCD.

Fuente: Producción propia

- **Subir programa**

En el caso de querer subir el archivo a la placa de Arduino UNO. En primer lugar, se selecciona el archivo.ino (ver Figura 114). En segundo lugar, se activa el botón de subir y se le da click a “Compilar”. En último lugar, se obtendrá el resultado de la compilación y la subida en la ventana “Resultados de la compilación”. Además, se puede ver el resultado contemplado en la placa de Arduino UNO mediante la ventana de “Video en línea” como se ve en la imagen 117.

Figura 117. Subir programa a Arduino para mostrar mensajes en el LCD.

Fuente: Producción propia

Ejemplo Arduino programa monitor serie

- **Subir programa**

Con el fin de darle uso a la ventana de “Monitor serie” se propone el siguiente programa como se observa en la figura 118. El programa lo que hará, básicamente, será enviar el

mensaje que se ve en la línea de código 15: *“Couldn’t find RTC”* y una fecha. Para subir el programa a la placa, se realizó el mismo proceso del ejemplo anterior.

```
1  #include <Wire.h>
2  #include "RTClib.h"
3
4  // RTC_DS1307 rtc;
5  RTC_DS3231 rtc;
6
7  String daysOfTheWeek[7] = { "Domingo", "Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado" };
8  String monthsNames[12] = { "Enero", "Febrero", "Marzo", "Abril", "Mayo",  "Junio", "Julio","Agosto","Septiembre","Octubre","Noviembre","Diciembre" };
9
10 void setup() {
11    Serial.begin(9600);
12    delay(1000);
13
14    if (!rtc.begin()) {
15      Serial.println(F("Couldn't find RTC"));
16    }
17
18    // Si se ha perdido la corriente, fijar fecha y hora
19    if (rtc.lostPower()) {
20       // Fijar a fecha y hora de compilacion
21       rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
22
23
24    }
25 }
26
27 void printDate(DateTime date)
28 {
29    Serial.print(date.year(), DEC);
30    Serial.print('/');     // Fijar a fecha y hora especifica. En el ejemplo, 21 de Enero de 2021 a las 03:00:00
31       // rtc.adjust(DateTime(2021, 1, 21, 3, 0, 0));
32    Serial.print(date.month(), DEC);
33    Serial.print('/');
34    Serial.print(date.day(), DEC);
35    Serial.print(" (");
36    Serial.print(daysOfTheWeek[date.dayOfTheWeek()]);
37    Serial.print(") ");
38    Serial.print(date.hour(), DEC);
39    Serial.print(':');
40    Serial.print(date.minute(), DEC);
41    Serial.print(':');
42    Serial.print(date.second(), DEC);
43    Serial.println();
44 }
45
46 void loop() {
47    // Obtener fecha actual y mostrar por Serial
48    DateTime now = rtc.now();
49    printDate(now);
50
51    delay(3000);
52 }
```

Figura 118. Mostrar mensajes en el monitor serie.

Fuente: Producción propia

- **Monitor serie – Iniciar**

Para iniciar el monitor serie se debe elegir los baudios y los segundos. Se eligió la tasa de baudios de 9600 porque fue la que se declaró en el mismo programa en la línea 11 (Ver figura 118) y 5 segundos porque es solamente un mensaje simple a imprimir. Como se puede observar en la figura 119, sale el mensaje esperado. Es importante aclarar que mientras se haga uso del monitor serie (durante los segundos que han sido seleccionados) no se puede utilizar la compilación y la subida en la placa, es por esto mismo que se desactivan todas las funcionalidades mientras está en uso el funcionamiento del monitor serie.

Figura 119. Funcionamiento del monitor serie. Ejemplo.

Fuente: Producción propia

8.2 Inconvenientes

A continuación, se mencionan los inconvenientes que surgieron a lo largo de cada actividad realizada junto con su causa y resolución (si es que la tuvo) o con la alternativa que permitió subsanar el problema.

- **Errores en la terminal al intentar ejecutar comandos de PlatformIO**

Causa:

Al optar instalar PlatformIO desde la terminal, esta instalación no marcaba los complementos faltantes que necesitaba PlatformIO para que funcione correctamente.

Solución:

Se instaló PlatformIO IDE como extensión de Visual Studio Code donde se menciona los complementos que necesita para que funcione correctamente el programa.

- **Error al integrar PlatformIO CLI por enlaces simbólicos**

Causa:

Los enlaces simbólicos estaban siendo ejecutados desde una ruta distinta a la raíz.

Solución:

Se debían ejecutar los comandos de los enlaces simbólicos desde la ruta raíz del sistema a través del comando `cd /`

- **[upload] could not open port /dev/ttyACM0: [Errno 13] Permission denied: '/dev/ttyACM0'**

Causa:

PlatformIO IDE no tiene permisos para el puerto donde está conectada la placa de Arduino.

Solución:

se otorgaron permisos a el dispositivo a través de los siguientes comandos:

```
sudo chmod 666 /dev/ttyACM0
```

```
sudo usermod -a -G dialout “user” (
```

se agregó el usuario al grupo dialout, “user” es el usuario de Ubuntu).

- **Error al ejecutar scripts de Bash desde el servidor**

Causa:

Los scripts de Bash creados no están activos para ser ejecutables.

Solución:

En primer lugar, se debe ir a la ruta donde se encuentra el script de Bash mediante el siguiente comando:

```
cd /ruta
```

Por último, lugar, se debe ejecutar el siguiente comando:

```
chmod +x archivo.sh
```

- **Error al ejecutar los comandos de los scripts de Bash desde el servidor**

Causa:

Los scripts de Bash necesitan pertenecer al grupo de usuarios www:data (que es el que usuario web de los servidores en Ubuntu) para ser ejecutados desde el servidor correctamente.

Solución:

  - En primer lugar, a cada script de bash se le asignó el usuario/grupo www:data mediante los siguientes comandos:

    ```
    sudo -i
    ```

    ```
    chown www-data:www-data archivo.sh
    ```

- En segundo lugar, como los comandos utilizados en los scripts de bash interactúan con PlatformIO se debe agregar también la carpeta “src” y archivo “platformio.ini” al usuario/grupo www:data mediante los siguientes comandos:
  chown www-data:www-data src
  chown www-data:www-data platformio.ini
- En tercer lugar, se necesita darle permisos de sudo al usuario/grupo a www:data:www-data, para esto se realiza la edición del archivo “sudoers” con el siguiente comando:
  nano /etc/sudoers
- Finalmente se agregó la siguiente línea al final del archivo (sudoers):
  www-data ALL=(ALL) NOPASSWD: ALL

- **Bugs de la aplicación web**

Causa:

Cuando se instaló la última versión de Ubuntu y los componentes necesarios de PHP para hacer pruebas de la aplicación web creada, sucede que se suele instalar componentes con la última versión de PHP, pero todo el trabajo se desarrolló en PHP 7.4; por lo tanto, algunas funcionalidades pueden ser deprecadas u obsoletas y eso es la causa principal de los bugs.

Solución:

La solución fue instalar PHP y todos sus módulos en la versión 7.4 como se explicó en la sección 7.2.1.5 (Instalación de LAMP).

- **Errores de phpMyAdmin**

Causa:

Al instalar otra versión de *phpMyAdmin* de manera manual, puede haber algunos errores como se observa en la figura 120.

- *(blowfish_secret). phpMyAdmin utiliza el secreto de blowfish para la autenticación de cookies.*
- *phpMyAdmin no puede almacenar plantillas en caché y será lento debido a esto.*
- *Error de eficiencia por no tener la base de datos de phpMyAdmin creada.*

Solución:

Se puede especificar el secreto blowfish usando un archivo de configuración de phpMyAdmin.

phpMyAdmin lo que hace primero es cargar el archivo de la ruta:

/rutaDeCarpetaphpmyadmin/phpmyadmin/libraries/config.default.php y luego anula esos valores con cualquier cosa que se encuentre en la ruta:

/rutaDeCarpetaphpmyadmin/phpmyadmin/config.inc.php.

Por lo tanto, para solucionarlo se debe:

1) Crear un archivo llamado *config.inc.php* con el siguiente comando:
   sudo nano /rutaDeCarpetaphpmyadmin/phpmyadmin/config.inc.php.
2) Finalmente, se escribe las líneas de código en el archivo *config.inc.php de* la figura 121 y se guarda.

   Error $cfg[‘TempDir’] (./tmp/) no es accesible

   Para solucionar este error basta con crear un directorio en la carpeta de *phpmyadmin* y hacerlo accesible con el siguiente comando:

   sudo mkdir /rutaDeCarpetaphpmyadmin/phpmyadmin/tmp && sudo chmod 777 /rutaDeCarpetaphpmyadmin/phpmyadmin/tmp

   Error por no tener una base de datos phpMyAdmin

   Por instalar de forma manual otra versión de phpMyAdmin, puede aparecer un error en términos de eficiencia porque no está creada la base de datos de phpMyAdmin con sus respectivas tablas. Esto es normal que suceda, ya que la base de datos siempre es creada mediante una instalación de forma automática. La solución es simple, pues se presiona en el mismo error y phpMyAdmin a través de una sugerencia y confirmación, creará la base de datos correspondiente.

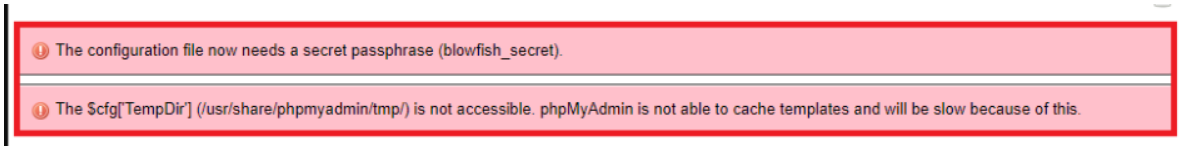


Figura 120. Errores de phpMyAdmin.

Fuente: Producción propia

```php
<?php
// use here a value of your choice 32 chars long
$cfg['blowfish_secret'] = 'PASTE__32__CHAR__BLOWFISH_SECRET';

$i=0;
$i++;
$cfg['Servers'][$i]['auth_type']     = 'cookie';
```

Figura 121. Archivo "conf.inc.php".

Fuente: Producción propia

- **Errores de código PHP**

Causa:

Los errores de PHP son comunes cuando se desarrolla una aplicación web, pueden ser errores de sintaxis, errores en la lógica, etc.

Solución:

Una herramienta que fue de ayuda para identificar los errores y solucionarlos es una extensión de PHP llamada Xdebug que proporciona la capacidad de depuración de código. Además, se puede instalar como extensión en el editor de código Visual Studio Code.

- **Errores de librerías en el archivo *platformio.ino***

Causa:

Cuando se trata de compilar o subir un archivo .ino o .cpp para la placa Arduino, suelen aparecer una serie de errores relacionados con las librerías. La causa del problema es la forma en la que se recorren las clases de las librerías y es un problema de la plataforma de PlatformIO.

Solución:

La solución va a depender de la librería en conflicto que aparezca como error al compilar el archivo. Por cada librería, hay una solución distinta y en cada solución se debió buscar en la comunidad de PlatformIO, la cual es amplia. Generalmente, se editó el archivo de *platformio.ino* acomodando la estructura de las librerías de tal manera que se recorran las clases de manera directa y no circular.

- **Error de acceso a los puertos con no-ip**

Causa:

Cada cierto tiempo, la IP privada, que es la que identifica un dispositivo conectado a la red interna, cambia. Cuando cambia, el host-dns creado desde no-ip no encuentra el dispositivo donde está alojado el servidor. Esto pasa porque cuando se realiza la apertura del puerto 80 (puerto del servidor Apache) y del puerto 8080 (puerto utilizado para MJPG-Streamer) se necesita la dirección IP privada del dispositivo en el que está alojado el servidor, pero si esta dirección cambia periódicamente, entonces los puertos quedan inaccesibles.

Solución:

La solución fue configurar la IP como estática, para que quede fija y no cambie. (Sección 7.3)

- **Error de acceso al host-dns cada cierto tiempo debido al cambio periódico de IP pública.**

Causa:

Similar a lo que ocurre con las IP privadas, las IP públicas cambian periódicamente o cada vez que se reinicia el router, y esto como consecuencia trae que se caiga la página.

Solución:

Para solucionar este problema, se encontraron dos soluciones. La primera fue entrar a la configuración del router para hallar la opción de agregar DNS dinámico (no todos los routers tienen esta opción). Cuando se agrega el DNS dinámico, como se observa en la figura 122, se solicita el proveedor (en este caso es No-IP), el hostname (por ej: *laboremotounaj.ddns.net*), el usuario de la cuenta de No-IP y la contraseña. La segunda solución es instalar el cliente de No-IP, que verifica cada 30 minutos si la IP pública del router cambio, en caso de que haya cambiado, también la cambia en el servicio No-ip.

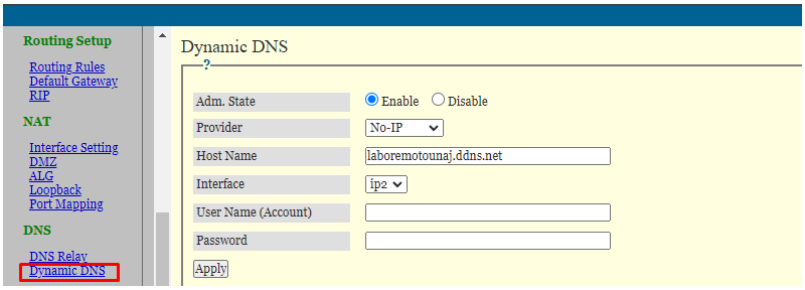


Figura 122. Configuración DNS dinámico.

Fuente: Producción propia

9. Conclusiones

En el proceso de desarrollo de la PPS se tuvo que realizar una investigación sobre las generaciones de la educación a distancia, hasta llegar a la actualidad donde se ve un fuerte impacto de la tecnología con modelos que se basan en conceptos como E-Learning o M-Learning. Fue importante tener en cuenta estos conceptos para entender la relación que se sostiene con la realización de un laboratorio remoto para la programación de sistemas embebidos.

Se vive en un mundo de constantes cambios tecnológicos, que dan como resultado nuevas formas de aprendizaje y una de estas formas es la que se adaptó principalmente el sistema desarrollado. Además, se realizó una investigación sobre sistemas embebidos, microcontroladores y la plataforma de Arduino. Siendo esto crucial para poder comprender los conceptos que se abordaron en el desarrollo de la práctica.

La elección del stack de tecnologías, se basó esencialmente en lo que se ha aprendido a lo largo de toda la carrera. Además, el lenguaje de programación del lado servidor (PHP) ofrece, en funciones simples, la capacidad de poder ejecutar scripts de bash. Estos scripts fueron fundamentales al momento de desarrollar el módulo del laboratorio remoto ya que, al poder ejecutar líneas de comandos en la terminal, también se podía hacer uso de la plataforma PlatformIO, que a su vez pudo interactuar con la placa de Arduino UNO.

Uno de los mayores retos presentados fue solucionar la cuestión de la superposición al momento que los alumnos ingresen al laboratorio remoto. Al respecto, se consideraron las siguientes alternativas:

- Realizar la gestión de los turnos desde Moodle y agregar un link de acceso al servidor del laboratorio.
- La plataforma “Weblab Deusto” desarrollada por la Universidad de Deusto, con capacidad de utilizar laboratorios preexistentes de otras instituciones e interactuar con hardware especializado.
- Plataforma VCL (laboratorio de Computación Virtual).

Luego de analizar en detalle cada una de las alternativas, ninguna fue convincente por cuestiones de complejidad o porque las plataformas utilizaban versiones de tecnologías obsoletas. Por lo tanto, se optó por diseñar e implementar un Sistema de Gestión de Reservas.

Otro reto también fue realizar el módulo del monitor serie, ya que se complejizaba el hecho de traer datos desde el puerto serie al navegador. Después de investigar de manera exhaustiva,

se encontró una solución, pero esta no se adaptaba al stack de tecnologías elegido al principio, ya que era una librería de la tecnología Node js llamada SerialPort. Por lo tanto, dadas las circunstancias, se optó por investigar alguna clase desarrollada en PHP donde se realice una comunicación con el puerto serie.

En cuanto a los resultados obtenidos, estos fueron los esperados respecto a los objetivos planteados, ya que todos los módulos en conjunto funcionaron correctamente. Se logró crear un sistema de gestión de turnos, simple e intuitivo para el usuario, subir varios ejemplos a la placa de Arduino UNO, visualizar los resultados a través de la webcam, utilizar el monitor serie que lea datos de la Placa Arduino UNO. Por otro lado, respecto al teclado matricial (para enviar datos), se implementó el código, pero falta el hardware para hacer pruebas futuras.

En conclusión, se pudo cumplir con los objetivos propuestos y afrontar los problemas que se fueron presentando para finalmente obtener resultados satisfactorios. Además, el trabajo desarrollado fue migrado a una máquina de la UNAJ que actuará como servidor.

10. Líneas futuras

A continuación, se mencionan algunas mejoras que se podrían realizar al sistema.

En primer lugar, se podría utilizar protocolo HTTPS en la página donde se aloja el sistema. Este protocolo es más seguro que HTTP, ya que encripta las comunicaciones impidiendo los ataques mediante intercepción de datos.

En segundo lugar, se podría mejorar la seguridad de datos mediante la encriptación de campos de SQL.

En tercer lugar, podría escalarse el sistema mediante una red LAN de servidores donde cada uno corresponda a un laboratorio remoto distinto. Esto mejoraría el sistema, creciendo el número de alumnos, ya que pueden usar otros laboratorios remotos, sin depender de un único laboratorio remoto.

En cuarto lugar, en vez de utilizar MJPEG-Streamer se podría utilizar un vivo de youtube. Esto mejoraría la calidad de imagen y, además, agregaría audio (por si hay un programa que se quiera subir a la Placa de Arduino que involucra el sonido).

Por último, se podría mejorar el módulo del puerto serie, donde lea el puerto serie en tiempo real. Esto se podría realizar utilizando una librería de Node.Js llamada "serialport", sin embargo, requeriría algunas configuraciones extras como agregar una web node js con nginx (proxy

inverso) a l servidor Apache o en última instancia migrar el proyecto al stack de tecnologías MERN (Mongo DB, Express Js, React JS y Node JS).

11. Reflexión sobre la Práctica Profesional Supervisada como espacio de formación

Realizar este proyecto propuesto en la presente Práctica Profesional Supervisada, fue un gran desafío para mí, ya que tenía poco conocimiento acerca de microcontroladores y temas de índole similar. Sin embargo, gracias a los conocimientos que obtuve en todas las materias cursadas de la carrera, se me hizo más flexible asimilar los conceptos que al final terminé incorporando a lo largo de la práctica.

Quisiera destacar esta práctica como fuente enriquecedora para reforzar mis conocimientos acerca del desarrollo web, tanto del backend como del frontend. Estoy satisfecho por el trabajo realizado y sé que me servirá tanto en mi formación laboral, como en mi formación personal.

Finalmente, quiero destacar la colaboración de los tutores María Florencia Ayala, Jorge Osio, y también a la tutora TAPTA Lía Lavigna, que estuvieron presentes a lo largo del proceso del trabajo realizado, respondiendo dudas y estando atentos cuando los necesité.

12. Bibliografía

- ✓ Aprendiendoarduino.wordpress.com, *Microcontrolador.* Disponible en: https://aprendiendoarduino.wordpress.com/category/microcontrolador/ [Consulta: 17/10/2022]
- ✓ Arduino.cc, Arduino. Disponible en: https://www.arduino.cc/en/about [Consulta: 20/06/2022/].
- ✓ Code.visualstudio.com, *Visual Studio Code.* Disponible en: https://code.visualstudio.com/docs [Consulta: 18/10/2022]
- ✓ Desarrolloweb.com, *Hoja de estilos en cascada.* Disponible en: https://desarrolloweb.com/home/css [Consulta: 20/06/2022/].
- ✓ Docs.platformio.org, *Comandos de instalación de Shell*. Disponible en: https://docs.platformio.org/en/latest/core/installation/shell-commands.html#piocore-install-shell-commands [Consulta: 21/10/2022]
- ✓ Docs.platformio.org, *PlatformIO*. Disponible en: https://docs.platformio.org [Consulta: 19/10/2022]
- ✓ Dronebotworkshop.com, *Primeros pasos con PlatformIO.* Disponible en: https://dronebotworkshop.com/platformio/ [Consulta: 20/10/2022]
- ✓ Elbinario.net, *Programar para arduino y otros microcontroladores desde la línea de comandos.* Disponible en: https://elbinario.net/2015/07/14/programar-para-arduino-y-otros-microcontroladores-desde-la-linea-de-comandos/ [Consulta: 18/06/2022/].
- ✓ Github.com, *MJPG-STREAMER.* Disponible en: https://github.com/jacksonliam/mjpg-streamer [Consulta: 23/10/2022]
- ✓ Help.ubuntu.com, *Preguntas comunes sobre Ubuntu.* Disponible en: https://help.ubuntu.com/community/CommonQuestions [Consulta: 18/10/2022]
- ✓ Html.com, *HTML.* Disponible en: https://html.com/ [Consulta: 19/06/2022/].
- ✓ Httpd.apache.org, *Servidor HTTP Apache.* Disponible en: https://httpd.apache.org/ABOUT_APACHE.html [Consulta: 18/10/2022]
- ✓ Javascripttutorial.net, *JavaScript.* Disponible en: https://www.javascripttutorial.net/what-is-javascript/ [Consulta: 20/06/2022/].
- ✓ Llanos Tobarra, *Laboratorios Remotos para cursos de Energías Renovables en Jordania.* Disponible en: https://es.slideshare.net/emadridnet/emadrid-2015-20-02-uned-rafael-pastor-vargas-desarrolloexplotacin-de-entornos-experimentales-basados-en-laboratorios-remotos-y-virtuales-related-44979219 [Consulta: 17/10/2022]

- ✓ Noip.com, *No-IP*. Disponible en https://www.noip.com/ [Consulta: 20/10/2022]
- ✓ Osio, Jorge Rafael (2022). *Manual de Usuario kit ATmega328P.* Universidad Nacional Arturo Jauretche.
- ✓ PHP.net, *PHP.* Disponible en: http://php.net/ [Consulta: 18/06/2022/].
- ✓ Revuelta, Miguel Ángel, *Laboratorio Remoto en un Entorno Virtual de Enseñanza Aprendizaje. Disponible en:* http://sedici.unlp.edu.ar/bitstream/handle/10915/55888/Documento_completo.pdf?sequence=3 [Consulta: 20/06/2022/].
- ✓ Sebastián Luchetti, tech.tribalyte.eu, *Sistemas embebidos y sus características.* Disponible en: https://tech.tribalyte.eu/blog-sistema-embebido-caracteristicas [Consulta: 17/10/2022]
- ✓ Sofía García Bullé, observatorio.tec.mx, *¿Qué es el m-learning? ¿Es una opción viable para la educación del siglo XXI?.* Disponible en: https://observatorio.tec.mx/edu-news/que-es-mobile-learning/ [Consulta: 15/10/2022]
- ✓ Tecnologicoedupraxis.edu.ec, *Como ha evolucionado la educación a distancia.* Disponible en: https://tecnologicoedupraxis.edu.ec/como-ha-evolucionado-la-educacion-a-distancia/ [Consulta: 15/10/2022]
- ✓ Wikipedia.com, *Educación a distancia.* Disponible en: https://es.wikipedia.org/wiki/Educaci%C3%B3n_a_distancia [Consulta: 15/10/2022]

13. Apéndice

13.1 Instalación de una librería en PlatformIO IDE.

Las librerías son aquellas dependencias que sirven para que el código funcione correctamente. En la siguiente figura (123) se pueden ver los pasos que se siguieron para instalar una librería.

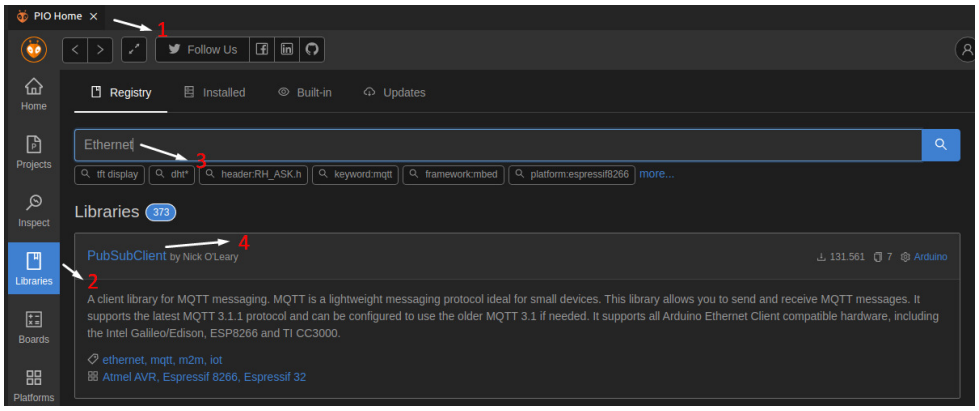


Figura 123. Instalación de una librería en PlatformIO IDE.

Fuente: Producción propia

1) Se debe ir a PIO Home, lo que es el inicio de la extensión de VSC.
2) Elegir la opción “Libraries”.
3) Se busca la librería deseada
4) Click en la librería

Para finalizar la instalación en la otra ventana se debió hacer click en “Add to Project” y elegir el proyecto en el cual se quiere incluir la librería como se ve observa en la figura 124.

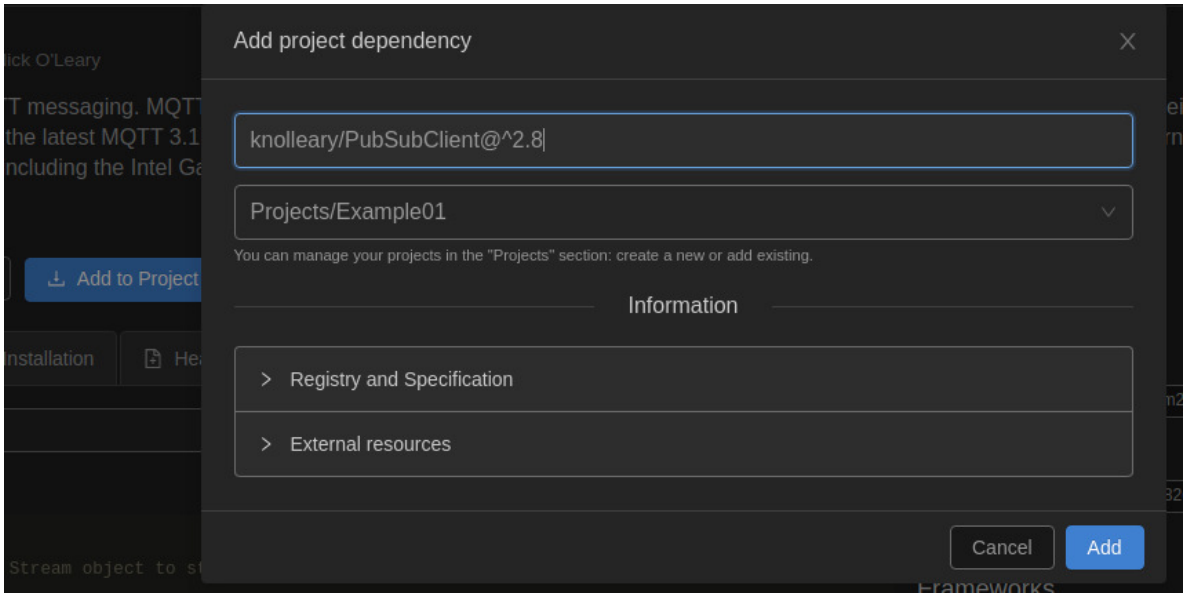


Figura 124. Final de instalación de la librería.

Fuente: Producción propia

Platformio.ini

Cada vez que se instala una librería, esta misma se declara en el archivo mencionado anteriormente llamado “Platformio.ini”. En la figura 125 se puede ver cómo se agregó la librería en la sección “libs_deps”. La próxima vez que se compile y ejecute un programa, se instalará la librería correspondiente.

PIO Home platformio.ini ×

Example01 > platformio.ini

```
; PlatformIO Project Configuration File
;
;   Build options: build flags, source filter
;   Upload options: custom upload port, speed and extra flags
;   Library options: dependencies, extra library storages
;   Advanced options: extra scripting
;
; Please visit documentation for the other options and examples
; https://docs.platformio.org/page/projectconf.html

[env:uno]
platform = atmelavr
board = uno
framework = arduino
lib_deps = knolleary/PubSubClient@^2.8

```

Figura 125. Agregar librería a PlatformIO IDE en el archivo platformio.ini

Fuente: Producción propia

13.1.1 Ejemplo del uso de la librería.

Se demostrará, a través de un ejemplo, el funcionamiento básico de PlatformIO IDE. Se carga a la placa un programa simple que muestra en el periférico del LCD un mensaje, con un contador cada un segundo. En la figura 126 se muestra el código del programa.

Example01 > src > main.cpp > loop()

```
#include <Arduino.h>

#include <Arduino.h>
#include <Wire.h>


#include <LiquidCrystal.h> // Libreria p

LiquidCrystal lcd(1, 0, A1, A2, 10, 9);

void setup() {
lcd.begin(16, 2); // Con esta instrucció
lcd.print("LLS-Ing-"); // Una instrucció
}

void loop() {
lcd.setCursor(0, 1); // con esta linea d
lcd.print("UNLP:"); // Al ser colocadar
lcd.println(millis() / 1000); // Con est
}
```

Figura 126. Programa para cargar en la placa de Arduino UNO.

Fuente: Producción propia

Importante: Sale una advertencia en la línea de código (7) “#include <LiquidCrystal.h>” esto es porque no está instalada tal librería. Pero una vez instalada, como se explicó anteriormente, no se presentarán inconvenientes (Ver figura 42).

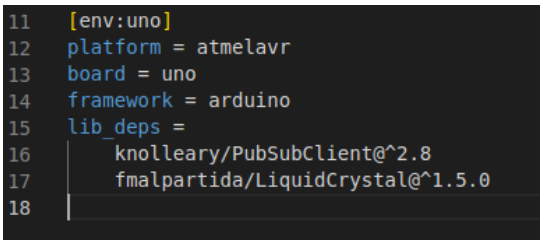


Figura 127. Contenido del archivo platformio.ini.

Fuente: Producción propia

Antes de compilar y ejecutar fue necesario verificar los permisos (Ver figura 128) para que PlatformIO IDE interactúe con el puerto USB donde estaba conectada la placa de Arduino. Fue verificado a través del siguiente comando:

ls -l /dev/ttyACM0

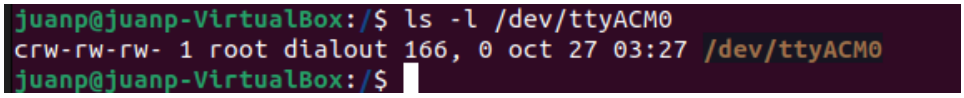


Figura 128. Verificar permisos de dispositivos USB.

Fuente: Producción propia

Solo queda compilar y ejecutar (Ver figura 129):

- Click en el ícono ✓ para compilar
- Click en el ícono -> para subir.

Figura 129. Compilación y ejecución del programa.

Fuente: Producción propia

Salida

Cuando se le da click a compilar, PlatformIO IDE, ejecuta un comando: *platformio run*. Como vimos en otra sección, PlatformIO Core viene incorporado dentro de PlatformiIO IDE y

proporciona una interfaz de línea de comandos (CLI). Esto dará un indicio a futuro. El resultado de la compilación fue satisfactorio como se ve en la figura 130.

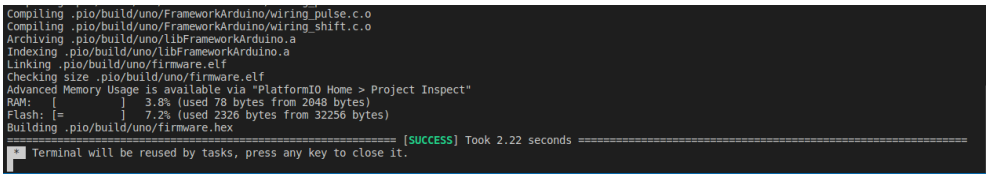


Figura 130. Resultados de la compilación del programa.

Fuente: Producción propia

Finalmente, solo falta hacer un “upload”, esto es, subir el programa a la placa Arduino UNO. En este caso, se ejecutó el siguiente comando: *platformio run –target upload* (Ver figura 131)

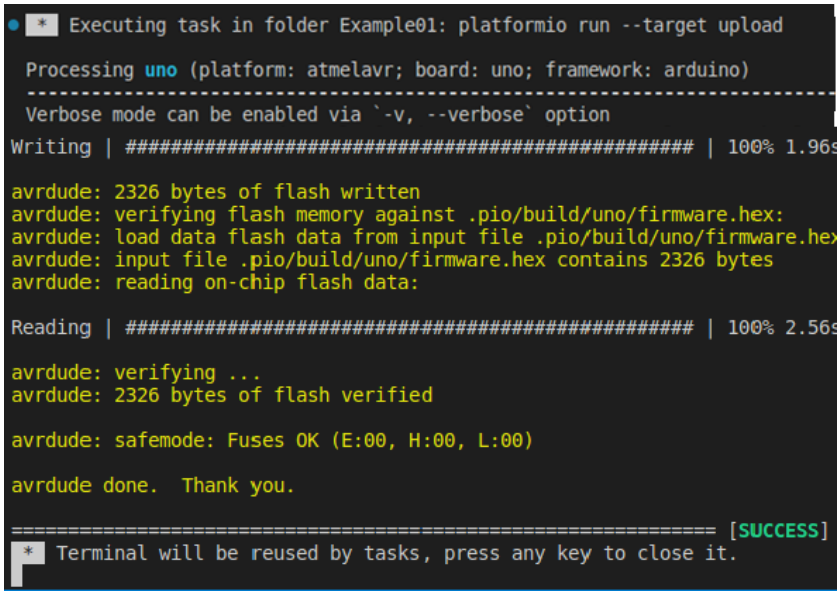


Figura 131. Resultados de la subida del programa a la placa de Arduino UNO.

Fuente: Producción propia

Comprobación real

Se comprobó en la placa si ocurre lo programado y efectivamente así es. Esto se observa en la figura 132.

Figura 132. Comprobación real.

Fuente: Producción propia