

ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

IMPLEMENTACIÓN DE UN SISTEMA PARA IMPRESORAS PARA AGREGAR FUNCIONALIDADES DE IMPRESIÓN EN RED

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN REDES Y TELECOMUNICACIONES

Raúl Iván Cantuña Oña
raul.cantuna@epn.edu.ec

Rommel Josué Vega Pazmiño
rommel.vega@epn.edu.ec

DIRECTOR: ING. LEANDRO ANTONIO PAZMIÑO ORTIZ
leandro.pazmino@epn.edu.ec

CODIRECTOR: ING. MONICA DE LOURDES VINUEZA RHOR
monica.vinueza@epn.edu.ec

Quito, septiembre 2021

CERTIFICACIÓN

Certificamos que el presente trabajo fue desarrollado por el Sr. Cantuña Oña Raúl Iván y el Sr. Vega Pazmiño Rommel Josué como requerimiento parcial a la obtención del título de TECNÓLOGO SUPERIOR EN REDES Y TELECOMUNICACIONES, bajo nuestra supervisión:

**MSc. Leandro Antonio Pazmiño
Ortiz**

DIRECTOR DEL PROYECTO

**MSc. Mónica de Lourdes Vinueza
Rhor**

CODIRECTORA DEL PROYECTO

DECLARACIÓN

Nosotros Cantuña Oña Raúl Iván con CI: 1721854253 y Vega Pazmiño Rommel Josué con CI: 1723247530 declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que hemos consultado las referencias bibliográficas que se incluyen en este documento.

Sin perjuicio de los derechos reconocidos en el primer párrafo del artículo 144 del Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación – COESC-, somos titulares de la obra en mención y otorgamos una licencia gratuita, intransferible y no exclusiva de uso con fines académicos a la Escuela Politécnica Nacional.

Entregamos toda la información técnica pertinente, en caso de que hubiese una explotación comercial de la obra por parte de la EPN, se negociará los porcentajes de los beneficios conforme lo establece la normativa nacional vigente.

Raúl Iván Cantuña Oña

Rommel Josué Vega Pazmiño

DEDICATORIA

Yo Raúl Iván Cantuña Oña estudiante de la carrera de Tecnología Superior en Redes y Telecomunicaciones, dedico a este proyecto a mis padres, amigos, profesores y compañeros, quienes siguieron mi camino en el estudio para poder llevar a cabo la culminación de este.

Raúl Iván Cantuña Oña

AGRADECIMIENTO

Yo Raúl Iván Cantuña Oña estudiante de la carrera de Tecnología Superior en Redes y Telecomunicaciones, agradezco a mis padres quienes me apoyaron en cada situación tanto económica como emocional durante el transcurso de mi carrera, a mis maestros quienes me impartieron el conocimiento necesario y suficiente para poder desarrollarme tanto como persona y trabajador en el ámbito laboral, a mis compañeros y amigos quienes a pesar de conflictos siempre hubo un apoyo emocional de su parte, por ultimo me agradezco a mí mismo por no decaer en ninguna etapa de la carrera y a pesar de los problemas siempre se pudo encontrar una basta solución.

Raúl Iván Cantuña Oña

DEDICATORIA

Esta obra la dedico con todo el amor, sacrificio, honra y todo el esfuerzo que ha sido plasmado en el documento a mi madre, quien ha sido la única persona que ha estado en los momentos más difíciles a lo largo de mi carrera, sin ella, todo esto no hubiese sido posible. Su bendición siempre la traigo conmigo sabiendo que me conducirá por buenos caminos a lo largo de esta nueva etapa en mi vida. Razones me faltan para ofrecerle este trabajo en ofrenda de su sabiduría, amor y paciencia.

Rommel Josué Vega Pazmiño

AGRADECIMIENTO

El desarrollo de este documento no se lo clasifíco como una tarea fácil, pero lo que sí puedo, es mencionar que aprendí y aproveché cada momento, cada proceso, cada investigación realizada, me hizo reflexionar sobre todo el tiempo que ha transcurrido para llegar a este inigualable momento, todo este conjunto de emociones que se siente no estaría completo sin los agradecimientos plenos a las personas que hicieron, con su ayuda, consejos, paciencia y sabiduría que esto llegase a ser posible, es por eso que agradezco a mis maestros que, desde el primer día de clases estuvieron dispuestos a compartir su conocimiento para que pueda desarrollarme como un profesional más, quiero agradecer a mis amigos, que no solo compartieron el aula de clases sino que con su amistad y dedicación me mostraron que en la vida se puede confiar en personas que no comparten un vínculo sanguíneo, su sincera amistad me permitió disfrutar de cada paso que di en mi etapa universitaria.

Rommel Josué Vega Pazmiño

ÍNDICE DE CONTENIDOS

1. INTRODUCCIÓN.....	15
1.1 Objetivo general	15
1.2 Objetivos específicos.....	15
1.3 Fundamentos.....	15
Sistemas Operativos.....	15
<i>Linux</i>	15
<i>Raspberry Pi OS (Raspbian)</i>	16
<i>Ubuntu</i>	16
<i>Ubuntu Core</i>	17
Sistema de transferencia del sistema operativo	17
<i>BalenaEtcher</i>	17
<i>Rufus</i>	18
Placas de desarrollo	18
<i>Raspberry Pi</i>	18
<i>Raspberry Pi 3 Model B</i>	18
<i>Raspberry Pi 3 Model B+</i>	19
Lenguajes de programación.....	19
<i>Python</i>	19
Redes	20
Servidor de impresión	20
Aplicación <i>CUPS (Common Unix Printing System)</i>	21
Red inalámbrica <i>WLAN (Wireless Local Area Network)</i>	23
Estándar <i>IEEE 802.11</i>	23
2. METODOLOGÍA.....	24
2.1 Descripción de la metodología usada	24
Objetivo 1: Analizar los requerimientos.....	25
Objetivo 2: Seleccionar el hardware.....	25

Objetivo 3: Seleccionar el software requerido	25
Objetivo 4: Implementar el prototipo	25
Objetivo 5: Realizar pruebas de funcionamiento.....	26
Objetivo 6: Optimizar el dispositivo	26
3. RESULTADOS Y DISCUSIÓN	26
3.1 Análisis de los requerimientos	26
3.2 Análisis del hardware.....	27
Selección de la placa de desarrollo.....	27
Dispositivo de impresión	29
3.3 Análisis del software.....	29
Selección del sistema operativo.....	29
Selección del lenguaje de programación	32
Selección de la aplicación del servidor de impresión	34
Análisis del sistema de refrigeración.....	36
Análisis de los códigos implementados.....	37
Diagrama de bloques del proceso de cola de impresión	37
Diagrama de bloques del servicio de correo.....	38
Diagrama de bloques de la página <i>web</i>	39
Diagrama de bloques del medidor de temperatura.....	40
Diagrama de bloques del control de temperatura	41
Códigos implementados.....	42
Estructuración de la base de datos	46
3.4 Implementación del prototipo.....	47
Instalación de la aplicación <i>CUPS</i>	47
Instalación del <i>hotspot</i>	48
Configuración del correo electrónico emisor	50
Ambiente virtual.....	51
Servicio de correo.....	51
Propiedades del servicio de correo	52

Servicio <i>web</i>	53
Módulo <i>index.html</i>	54
Módulo <i>success.html</i>	55
Módulo <i>list_mail.html</i>	56
Puertos de la placa <i>Raspberry Pi</i>	57
3.5 Pruebas y Análisis de Resultados	58
Etapa de inicio automatizado	58
Etapa de registro de usuario	59
Etapa de registro de impresora	59
Etapa de impresión	62
3.6 Optimización del dispositivo	63
Diseño de la cubierta	63
Configuración del autodescubrimiento para la <i>Raspberry Pi</i>	68
Costo del prototipo	69
Manual de uso y mantenimiento	69
4. CONCLUSIONES Y RECOMENDACIONES	71
4.1 Conclusiones	71
4.2 Recomendaciones	73
5. REFERENCIAS BIBLIOGRÁFICAS	74
ANEXOS	77
Anexo 1: Certificado de Funcionamiento	i
Anexo 2: Modelo 3D de la <i>Raspberry Pi</i> y del adaptador lpt1	i
Anexo 3: Modelo 3D de la etapa 1 de la cubierta	ii
Anexo 4: Modelo 3D de la etapa 2 del diseño de la cubierta	iii

ÍNDICE DE FIGURAS

Figura 1.1 Placa Raspberry Pi 3 Model B+ [9]	19
Figura 1.2 Funcionamiento del Protocolo HTTP/1.1 [15].....	22
Figura 1.3 Diagrama de bloques del diseño de CUPS [17]	22
Figura 1.4 Diagrama WLAN con Raspberry Pi [19]	23
Figura 3.1 Diagrama de bloques del Proceso de Cola de Impresión	38
Figura 3.2 Diagrama de bloques del servicio de correo.....	39
Figura 3.3 Diagrama de bloques de la Página Web	40
Figura 3.4 Diagrama de bloques del medidor de temperatura.....	41
Figura 3.5 Diagrama de bloques del control del ventilador	42
Figura 3.6 Código del servicio de correo	44
Figura 3.7 Código de las propiedades del servicio de correo	44
Figura 3.8 Código de la página web.....	45
Figura 3.9 Código del medidor de temperatura	45
Figura 3.10 Código del control de temperatura	46
Figura 3.11 Estructura de la base de datos.....	46
Figura 3.12 Ingreso a la configuración del Hotspot	49
Figura 3.13 Configuración del SSID	49
Figura 3.14 Configuración de la contraseña del Hotspot.....	50
Figura 3.15 Configuración de acceso de aplicaciones.....	51
Figura 3.16 Carpeta del ambiente virtual.....	51
Figura 3.17 Identificador del archivo de impresión	52
Figura 3.18 Parámetros del archivo de propiedades	53
Figura 3.19 Llamada de los parámetros	53
Figura 3.20 Pantalla de inicio del servicio web.....	54
Figura 3.21 Diagrama de flujo del módulo index.html.....	54
Figura 3.22 Confirmación de registro del usuario	55
Figura 3.23 Diagrama de flujo del módulo success.html	55
Figura 3.24 Listado de usuarios ingresados.....	56
Figura 3.25 Diagrama de flujo del módulo list_mail	56
Figura 3.26 Puertos de la Raspberry Pi [28].....	57
Figura 3.27 Cable adaptador USB a LPT1 [29]	57
Figura 3.28 Código implementado para el inicio automatizado	58
Figura 3.29 Inicio automático del servicio de correo.....	58

Figura 3.30	Inicio automático del servicio web	59
Figura 3.31	Formulario de registro	59
Figura 3.32	Credenciales de verificación.....	60
Figura 3.33	Lista de impresoras disponibles	60
Figura 3.34	Configuración de la impresora.....	61
Figura 3.35	Drivers de modelos de impresoras.....	61
Figura 3.36	Impresoras disponibles dentro del servicio de impresión.....	62
Figura 3.37	Adición de impresora en el editor de texto.....	62
Figura 3.38	Parámetros del mensaje de verificación	63
Figura 3.39	Medidas de la Raspberry Pi	63
Figura 3.40	Primer modelo de la cubierta.....	64
Figura 3.41	Diseño de las piezas de la cubierta.....	67
Figura 3.42	Diseño de la cubierta armada digitalmente.....	67
Figura 3.43	Diseño de la cubierta armada físicamente.....	68
Figura 3.44	Ping realizado a google.com	69
Figura 3.45	Enlace al video del manual de uso.....	70
Figura 3.46	Enlace al video del manual de mantenimiento.....	70

ÍNDICE DE TABLAS

Tabla 3.1 Especificaciones de Raspberry Pi 3 y Arduino Mega 2560	27
Tabla 3.2 Especificaciones Raspberry Pi 3 B y B+	28
Tabla 3.3 Especificaciones de Ubuntu Core y raspberry Pi OS	30
Tabla 3.4 Comparación entre Ubuntu Core y Raspberry Pi OS	30
Tabla 3.5 Comparación entre Java y Python	32
Tabla 3.6 Tabla comparativa entre Samba y CUPS.....	34
Tabla 3.7 Resultados de la medición de temperatura	36
Tabla 3.8 Control de la temperatura durante 8 horas.....	65
Tabla 3.9 Control de temperatura de los dos modelos durante 8 horas.....	65
Tabla 3.10 Comparación entre posibles materiales de fabricación de la cubierta.....	66
Tabla 3.11 Tabla de costos del prototipo	69

RESUMEN

El siguiente proyecto consta de una introducción donde se justifica el planteamiento del proyecto, además, en base a la información recogida en los objetivos se mostrará los fundamentos teóricos sobre los cuales se cimentan los conceptos a utilizar en el proyecto, como placas de desarrollo, sistemas operativos y servicios.

Posteriormente en la segunda sección, se analiza el proceso de ejecución del proyecto además de los pasos necesarios para realizarlo. En esta sección se revisa la metodología aplicada durante el desarrollo del proyecto.

En la tercera sección, se detallan los requerimientos para la implementación del proyecto tanto como *hardware* y *software*, además de configuraciones y diseño final del prototipo, esta sección culmina con las respectivas pruebas de funcionamiento para verificar la fidelidad de uso que puede proporcionar el prototipo. Además, se desarrollan tanto el manual de uso como el manual de funcionamiento donde se explicará a detalle el uso y funcionamiento del prototipo además del correcto mantenimiento que se le debe dar para su óptimo funcionamiento.

En la cuarta sección, se analizan las conclusiones recogidas por cada uno de los objetivos a aplicar durante el desarrollo del proyecto, en adición, se proporcionarán recomendaciones que fueron tomadas en cuenta para el posterior uso y aplicación del prototipo.

En la quinta y última sección, se incluye la bibliografía analizada durante el desarrollo del proyecto y anexos referentes al prototipo.

PALABRAS CLAVE: Linux, servidor, impresoras, aplicación, inalámbrico.

Comentado [R01]: cuarta-quita

Comentado [MV2]: Debe ser cuarta sección ¡!!!

ABSTRACT

The following project consists of an introduction where the project approach is justified, in addition, based on the information gathered in the objectives, the theoretical foundations on which the concepts to be used in the project, such as development boards, operating systems and services, are based, will be shown.

Subsequently, in the second section, the project execution process is analyzed as well as the necessary steps to carry it out. This section reviews the methodology applied during the development of the project.

In the third section, the requirements for the implementation of the project are detailed as hardware and software, as well as configurations and final design of the prototype, this section culminates with the respective performance tests to verify the fidelity of use that the prototype can provide. In addition, both the user's manual and the operation manual are developed, where the use and operation of the prototype will be explained in detail, as well as the correct maintenance that should be given to it for its optimal operation.

In the fourth section, the conclusions collected for each of the objectives to be applied during the development of the project are analyzed, in addition, recommendations that were considered for the subsequent use and application of the prototype will be provided.

The fifth and last section includes the bibliography analyzed during the development of the project and annexes related to the prototype.

KEYWORDS: *Linux, server, printer, application, wireless.*

1. INTRODUCCIÓN

En la actualidad existen muchas impresoras que no disponen de acceso a la red por lo cual realizar una impresión desde un teléfono celular o desde otro dispositivo que no se encuentre conectado directamente a la impresora, es imposible. Muchas personas tienen en sus hogares una impresora sin acceso a la red, actualmente para facilitar un trabajo de impresión se lo hace desde un dispositivo portátil o desde un dispositivo que no esté vinculado a la impresora de forma cableada, es decir, que tenga una conexión inalámbrica con algún punto de acceso compartiendo la misma red a la cual está conectada la impresora.

Debido a que todas las personas cuentan con un capital para sustituir su impresora sin acceso a la red por una de última generación, para ello se ha creado un servidor de impresión mediante el uso de un dispositivo el cual contendrá un sistema operativo y la aplicación que administrará tanto los dispositivos conectados de forma remota como el registro de la impresora a la cual se envía la tarea de impresión.

1.1 Objetivo general

Implementar un sistema para impresoras para agregar funcionalidades de impresión en red.

1.2 Objetivos específicos

- Analizar los requerimientos.
- Seleccionar el Hardware.
- Seleccionar el software requerido.
- Implementar el prototipo.
- Realizar pruebas de funcionamiento.
- Optimizar el dispositivo.

1.3 Fundamentos

Sistemas Operativos

Linux

A simple vista, *Linux* es un Sistema Operativo en sí, pero en realidad es la primera implementación de libre distribución de *UNIX* que se caracterizaba por ser un sistema operativo portable, multitarea y multiusuario. *Linux* como sistema operativo es muy

eficiente y cuenta con un excelente diseño, tiene la capacidad de ser multitarea, multiusuario y multiprocesador. Además, limita la memoria para que ningún programa no pueda detener el proceso de arranque del sistema, cargando únicamente las partes que se van a utilizar del programa, la memoria que queda libre lo usa para el caché permitiendo utilizar bibliotecas enlazadas estática y dinámicamente [1].

Más allá de las sobresalientes especificaciones con las que cuenta Linux, lo que lo hace especial es el modo en que se lo desarrolla, llegando a ser un *shareware* (modalidad en la que un usuario evalúa de forma gratuita un producto pero en un tiempo limitado), dando como resultado, que el código fuente de Linux sea público y disponible para cualquier desarrollador que lo requiera, tiene pocas limitaciones que están reglamentadas en la Licencia Pública GNU (GPL) que se encarga de asegurar la disponibilidad de *Linux* [1].

Las nuevas versiones del *kernel* de *Linux* son realizadas por programadores y desarrolladores de todo el mundo que se unen a través de la red para comprobar su estabilidad. Las versiones salientes se numeran con las letras *x*, *y*, *z* donde, las versiones numeradas con *y* son clasificadas como estables. Si alguna de las versiones se encuentra con fallas, inmediatamente miles de personas alrededor del mundo corrigen el error en pocas horas, convirtiéndose en un sistema de alta calidad tecnológica [1].

Raspberry Pi OS (Raspbian)

Raspbian es el sistema operativo propio o por defecto de una *Raspberry Pi*, pero no es necesariamente obligatorio ya que también puede utilizar otros sistemas operativos como *Windows*. Debido a que *Raspbian* es un sistema se puede descargar de la página oficial de *Raspberry*, su uso es más frecuente, además de que cuenta con una interfaz gráfica amigable ya que se basa en una distribución de GNU/*Linux* llamada *Debian*. Este sistema cuenta con dos versiones, una en la cual su entorno gráfico está completo por lo que cuenta con un escritorio, menús, ventanas, fondos e iconos, a esta versión se le conoce como *Raspbian Pixel*. Mientras que su otra versión se basa en una consola sin gráficos, a esta versión se la conoce como *Raspbian Lite* [2].

Ubuntu

Ubuntu es un sistema operativo basado en GNU/*Linux* usado en dispositivos *Raspberry*, debido a que es una distribución de *Debian*. Este sistema operativo cuenta con una interfaz similar a la de *Windows* que permite realizar tareas como la navegación por la

web, crear documentos, editarlos y enviarlos, editar imágenes e instalar aplicaciones de entretenimiento o de ayuda [3].

Desde el nacimiento de *Ubuntu*, las actualizaciones se han lanzado cada seis meses durante abril y octubre, siendo la versión *Ubuntu 12.10* o también conocida como *Quantal Quetzal* la última publicada. *Ubuntu* ha puesto énfasis especial en desarrollarse para las nuevas tecnologías como en netbooks, informática móvil y sobre todo en la arquitectura de la nube como infraestructura de datos [3].

Ubuntu Core

Ubuntu Core es una versión de Ubuntu optimizada para sistemas integrados nativos de IoT. Si bien Ubuntu ofrece lo último y lo mejor en software de código abierto para la informática de propósito general, Ubuntu Core solo lleva paquetes binarios a los dispositivos de un solo propósito elegidos. [4]

Ubuntu Core expone un REST incorporado API para la gestión segura de dispositivos. Los clientes autenticados pueden realizar gestión de software y tareas de configuración en Ubuntu a dispositivos centrales de forma remota. [4]

Ubuntu Core es un sistema operativo de contenedor construido sobre *snaps*. Con *snaps*, los sistemas integrados se benefician de la seguridad, la inmutabilidad, así como modularidad y compatibilidad. [5]

Sistema de transferencia del sistema operativo

BalenaEtcher

BalenaEtcher o conocido solamente como *Etcher*, es un software utilizado para la escritura de archivos de formato *.iso* y *.img*, además de carpetas comprimidas para dispositivos de almacenamiento como unidades *flash* USB y tarjetas de memoria. *BalenaEtcher* fue desarrollada por *Balena* anteriormente llamada *Resin.io* y bajo la licencia de *Apache License 2.0*. La interfaz de *Etcher* permite al usuario elegir la imagen para luego grabarla al dispositivo USB, además de esta característica permite:

- Proteger los archivos del disco duro externo.
- Preparar un dispositivo USB con mayor capacidad que un DVD.
- Permitir grabar e instalar *Raspbian* en una *Raspberry* mediante una memoria *microSD* [6].

Algunas de las características que *Etcher* planea en el futuro, son para que exista una compatibilidad de almacenamiento en memorias *Live SD* y compatibilidad para flashear varias particiones de arranque en un solo dispositivo [6].

Rufus

Rufus es un *software* utilitario que ayuda en formatear y crear un soporte USB para utilizar como archivo de arranque de un sistema operativo mediante un dispositivo USB o tarjeta de memoria. Es muy útil para casos en los que se necesite crear un medio de instalación en un dispositivo *USB* a partir de archivos *.iso* arrancables como: *Windows*, *Linux*, *UEFI*, entre otros; también es muy utilizado en equipos que no tengan un sistema operativo instalado o que necesiten actualizar su *firmware* o *BIOS*. *Rufus* en comparación a otros *softwares*, es rápido y eficiente en la creación de instaladores USB desde una ISO y en la creación de instaladores USB para *Linux* [7].

Placas de desarrollo

Raspberry Pi

Raspberry Pi es una pequeña y compacta computadora de bajo costo que permite al usuario conocer más sobre computación explorando nuevos lenguajes de programación como lo es *Python*. Esta computadora es capaz de realizar la mayoría de las actividades que realizan los computadores de sobremesa, como la manipulación de documentos, navegación en Internet hasta reproducir videos en calidad alta de resolución [8].

Raspberry tiene una alta capacidad de interacción con varios dispositivos, ya que puede ser utilizada en una amplia gama de proyectos en el tema digital como, por ejemplo: detectores, reproductores de audio y video o incluso en estaciones para medir el clima, entre otros. La popularidad de esta computadora creció principalmente por su versatilidad al momento de trabajar en proyectos donde se ejecute el sistema operativo *Linux*, el cual es muy apetecido por desarrolladores que se desenvuelven en un entorno libre [8].

Raspberry Pi 3 Model B

El modelo B de *Raspberry Pi 3* supuso un rediseño completo de la placa, donde se mantuvo el tamaño y la distribución de los elementos de su antecesora, pero modificó significativamente el procesador por uno más potente el cual trabaja a 1.2 (GHz), otra de las funciones nuevas que trajo, fue la eliminación del cuello de botella durante la conexión, esto se solucionó agregando una conexión *Bluetooth 4.2* y *Wi-Fi* a banda

doble (2.4 (GHz) y 5 (GHz)), adicionalmente, se incluyó una tarjeta de red Gigabit Ethernet que es capaz de alcanzar los 300 (Mbps) [9].

Raspberry Pi 3 Model B+

La *Raspberry Pi* hace uso de un GPIO (Pin de Propósito General Entrada/Salida o *General Purpose In/Out* – en inglés) como se muestra en la Figura 1.1 **Figura 1.1** Placa Raspberry Pi 3 Model B+ de 40 pines que actúan como tanto como actuadores como sensores, es importante conocer que el GPIO trabaja con un voltaje de 3.3 (V) ya que si es necesario conectar diferentes sensores que trabajen a un voltaje superior, se necesitará de un convertor de niveles lógico. El procesador de la *Raspberry Pi 3 B+* alcanza los 1.4 (GHz), pero no cuenta con un convertor analógico-digital por lo que se requiere un convertor externo. La *Raspberry Pi* cuenta con las conexiones tradicionales que es posible encontrar en un computador común como lo son: puertos *USB*, conector *Ethernet*, puerto de 3.5 (mm), puerto HDMI, puertos *microSD* y *micro USB* [9].

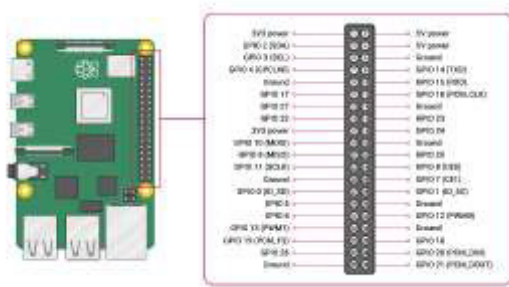


Figura 1.1 Placa Raspberry Pi 3 Model B+ [9]

Lenguajes de programación

Python

Python es un lenguaje de programación orientado a objetos fundamentado en el lenguaje ABC que es un lenguaje de propósito general y de alto nivel. *Python* como un lenguaje de alto nivel está compuesto por estructuras de datos implícitos como conjuntos, listas y diccionarios que permiten realizar tareas complejas de una manera más sencilla, resumiéndose en pocas líneas de código [10].

El desarrollo de *Python* ha contribuido a la creación de comunidades de desarrolladores que pueden utilizar este lenguaje de programación donde tienen la posibilidad de escribir códigos dentro de sus propias reglas, formas o métodos. Existen varios principios de diseño que se deben seguir al momento de escribir un código en *Python* como: un código

legible, plano, explícito y fácil de explicar según el sitio web oficial que se resume en la frase “*Siempre hay maneras más fáciles de realizar nuestro trabajo*” [10].

Sintaxis

Python cuenta con una sintaxis sencilla que en ocasiones se asimila a un pseudocódigo, si se lo compara con otro lenguaje de alto nivel como *C++*. Se puede resaltar que *Python* es más simple que *C++* y esto es debido a que *Python* busca ofrecer claridad y sencillez para que cualquier programador pueda utilizarlo sea cual sea su nivel [11].

Librerías

La fortaleza de *Python* se encuentra en sus librerías estándar, con sus decenas de módulos que cubren la mayoría de las necesidades elementales que tiene un programador de forma intuitiva [11].

Rendimiento

El rendimiento de *Python* ha sido uno de los temas de los que más se habla, ya que al ser un lenguaje interpretado tiende a ser más lento que un lenguaje compilado, se puede pensar que tiene un rendimiento pobre, pero esto no está del todo correcto ya que sus librerías estándar han sido implementadas en lenguaje C lo que hace que las funciones primitivas sean lo suficientemente eficientes. Además, el código puede compilarse a *bytecodes* similar a lo que se usa en *Java* y *.NET*, todo este proceso optimiza todo el proceso de interpretación [11].

Redes

Servidor de impresión

Es una herramienta que permite el uso de una impresora de forma remota. Este servidor se conecta a la red permitiendo gestionar los archivos o documentos que se desea imprimir de manera más sencilla, debido a que puede estar disponible independientemente de que los equipos PC se encuentren encendidos, además, puede controlarse desde cualquier otro dispositivo, siempre y cuando se mantenga la conexión a red y el registro de la impresora a utilizar [12].

El servidor de impresión o *Print Server* es un dispositivo que permite la conexión o accesibilidad a una impresora que no cuente con características inalámbricas mediante la interconexión hacia un puerto del modem o *Router* en particular [12].

El comportamiento del servidor siempre será el de una interfaz con la que se establece una interacción de los usuarios que estén conectados a la red que necesiten hacer uso del servicio de impresión. El servidor cuenta con una administración a la que se ingresa para realizar diferentes modificaciones para permitir el acceso o negar el acceso a los usuarios conectados, otra de sus características es la de derivar usuarios a otros accesos de impresión si es que el servicio se encuentra congestionado [12].

Existen muchos beneficios de tener un servidor de impresión en un entorno doméstico tal y como se lo haría en un entorno empresarial, el ahorro del cableado es superior si en el hogar se cuenta con más de un computador porque únicamente se configura el dispositivo con la red del hogar teniendo como resultado un centro de impresión accesible y seguro para todos los integrantes del hogar [13].

Aplicación CUPS (*Common Unix Printing System*)

CUPS es el sistema de impresión que se utiliza como base para gestionar impresoras y todo lo implicado en un servicio de impresión como solicitudes, revocación de permisos y colas de impresión. Este sistema es de código abierto por lo que permite la búsqueda de impresoras dentro de la red proporcionando una interfaz de impresión igual para la red local. El sistema utiliza IPP o *Internet Printing Protocol*, el cual se utiliza para la gestión de un centro de impresión [14].

IPP es el protocolo estándar para imprimir en una red por lo que es usado localmente a través de la Internet para comunicarse de manera remota lo que permite configurar, gestionar y verificar el estado del centro de impresión a través de un navegador, esto debido a que se ubica niveles arriba del protocolo de transferencia de hipertexto HTTP que es la base para los servidores web, además de permitir el control de acceso, la autenticación y el cifrado, CUPS cuenta con una interfaz a nivel de aplicación que incluye interfaces necesarias para procesar el formato de imagen [14].

CUPS está diseñado para ser una aplicación de servidor HTTP/1.1 (versión de HTTP donde las respuestas a las peticiones se la realizan sobre texto plano como se ve en la Figura 1.2) y IPP/2.1 (versión de IPP donde es dirigido más a usuarios y dispositivos con mayor velocidad, compatibles con la ubicación y el mantenimiento del dispositivo, aquí las impresoras suelen tener más funciones de post procesamiento) [15] [16].

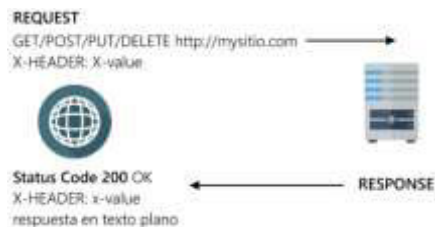


Figura 1.2 Funcionamiento del Protocolo HTTP/1.1 [15]

El protocolo HTTP es utilizado para los servicios web mediante un navegador, así como para los mensajes provenientes de IPP que son enviados a través de solicitudes de respuesta de HTTP *POST* que tiene el formato de contenido *application/ipp*. El servicio está diseñado como un servidor de un solo subproceso que ejecuta procesos externos de largo plazo como lo es la impresión, notificación y la enumeración de dispositivos o controladores además de la acción remota del dispositivo o impresora como se detalla en la Figura 1.3 **Figura 1.3** Diagrama de bloques del diseño de CUPS [17].

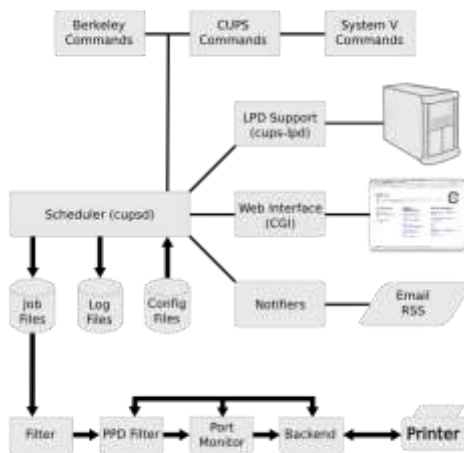


Figura 1.3 Diagrama de bloques del diseño de CUPS [17]

Las ventajas que se pueden obtener del uso de CUPS son:

1. Compatibilidad con protocolos de impresión de siguiente generación.
2. Detección automática de impresoras dentro de la red.
3. Configuración y mantenimiento a través de la interfaz *web*.

4. Capacidad de ampliar el número de impresoras conectadas.
5. Compatibilidad de descripción de impresoras PPD para ficheros.

Red inalámbrica WLAN (*Wireless Local Area Network*)

Una red inalámbrica hace referencia a la conexión a la red donde se intercambia información sin la necesidad de conexión cableada, para ello la información viaja por el aire en forma ondas. WLAN es la abreviatura de *Wireless Local Area Network* que significa una Red Inalámbrica de Área Local. Este tipo de redes se utiliza en oficinas, hogares, edificios, instituciones, entre otras, debido a que este tipo de redes no tienen un coste elevado y brindan los beneficios necesarios para la satisfacción de conexión a la red [18].

Normalmente, los dispositivos que solicitan el ingreso a la red son agregados mediante DHCP. La principal diferencia entre una LAN y una WLAN es la manera en la que se transmiten los datos, en una LAN se lo hace mediante un cable físico en cambio, en una red WLAN la comunicación se da de forma inalámbrica mediante el aire, en este proceso es posible gracias al uso de protocolos diseñados para la transmisión de información. El proceso que se cumple es el mismo si se adiciona otro dispositivo o terminal intermediario como lo es la *Raspberry Pi Model 3 B* como se muestra en la Figura 1.4

Figura 1.4 Diagrama WLAN con Raspberry Pi [19].

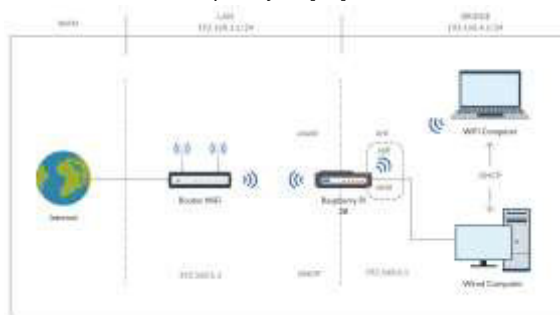


Figura 1.4 Diagrama WLAN con Raspberry Pi [19]

Estándar IEEE 802.11

Este es el estándar donde se encuentran las especificaciones físicas y de nivel de la capa MAC que se debe tener en cuenta al momento de implementar una red de área

local inalámbrica WLAN. La norma 802.11 cuenta con varias extensiones que modifican o mejoran su funcionamiento acorde a diferentes especificaciones:

802.11: norma especificada para 1 a 2 (Mbps) en la banda de 2.4 (GHz).

802.11b: extensión de la norma que proporciona 11 (Mbps).

Wi-Fi (Wireless Fidelity): certifica productos inalámbricos 802.11 b que pueden interoperar con varios fabricantes.

802.11: extensión de la norma que proporciona 54 (Mbps).

802.11g: extensión de la norma que proporciona de 20 a 54 (Mbps), esta extensión es compatible con sus antecesoras [20].

2. METODOLOGÍA

2.1 Descripción de la metodología usada

El proyecto se basa en la creación de un servidor de impresión, el cual permite la implementación de funcionalidades inalámbricas a una impresora que no disponga de este servicio. Para ello se tendrá que analizar el tipo de impresora con la que dispone el usuario, ya que en base a esto se deberá comprar los dispositivos para la conexión a Internet.

Por consiguiente, se selecciona el tipo de hardware en base a los requerimientos del usuario, en este caso se utilizó un Raspberry Pi, al cual se le tuvo que implementar el sistema operativo, Raspberry provee de un sistema oficial el cual cuenta con los requerimientos y funciones necesarias para implementar un servidor de impresión. Una vez realizada la configuración del Raspberry se tuvo que seleccionar una aplicación mediante la cual se realizó el control del servidor de impresión, CUPS es la aplicación que cuenta con la configuración necesaria para controlar tanto de manera estática como remota el servidor de impresión.

En esta aplicación se gestionó los permisos para consiguiente vincular el dispositivo de impresión y realizar las debidas pruebas mediante dispositivos remotos, en el cual, al momento de realizar una impresión de documentos u archivos, este envía un correo de verificación sobre la acción requerida.

Comentado [MV3]: Después de cada punto a parte debe ingresar un espacio adicional. Esto implca que los párrafos se van a bajar en las páginas. Revisar nuevamente la generación del índice .

Esta página esta con separación de párrafos

Comentado [RO4]: tuvo-tendrá

Comentado [RO5]: Se cambia tuvo-tendrá

Comentado [MV6]: ¡NARRAR EN PASADO!!!! Usted explica todo lo que hizo, Que metodología usaron, que resultados obtuvieron, etc ¡Esto a lo largo de todo el documento!!!!

Comentado [MV7]: Aquí debe decir: En esta aplicación se gestionó los permisos.....

Comentado [RO8]: Gestionó-gestionará

Comentado [MV9]: Debe decir envía por el tiempo en el que se narra la metodología

Comentado [RO10]: envía-enviará

Una vez realizadas las pruebas de funcionamiento del sistema, se inició la fase de comercialización del dispositivo, donde se colocó una cubierta protectora al dispositivo *Raspberry Pi*, adicional se añadió un instructivo de instalación donde el cliente puede vincular su cuenta e impresora. Además, se realizó videotutoriales del funcionamiento y mantenimiento del dispositivo para que el usuario pueda optimizar de mejor manera el uso del sistema.

Comentado [RO11]: se inició-inicia

Comentado [RO12]: colocó-colocará

Comentado [RO13]: realizó-realizará

Objetivo 1: Analizar los requerimientos

Se analizó el dispositivo de impresión, ya que este debía contar con una entrada USB tipo B el cual se conectó con la entrada de dispositivo del acceso a Internet. Esta impresora debía contar únicamente con los servicios básicos, es decir, impresión, fotocopia y escáner de documentos o archivos.

Comentado [MV14]: Este párrafo ya está escrito en pasao como debería narrarse algo que ya se ejecutó....

Comentado [MV15]: Los objetivos deben ser redactados en pasado, lo que se hizo.....

Comentado [RO16]: analizó-analiza, debía-deberá.

Objetivo 2: Seleccionar el hardware

Para la selección de los elementos que integraron el prototipo, se utilizó como discriminantes las necesidades determinadas de manera previa. Esta especificación de dispositivos permitió seleccionar de la manera más adecuada los módulos o elementos a fin de armar el sistema del prototipo.

Comentado [RO17]: integraon-integran, utilizó-utiliza, permitió-permita.

Objetivo 3: Seleccionar el software requerido

El *software* instalado fue de una distribución de Linux, ya que se consideró que Raspberry cuenta con su sistema operativo propio denominado Raspbian, el cual cuenta con una interfaz gráfica sencilla y amigable para el usuario, con configuraciones básicas y necesarias para implementar un servidor de impresión. Además, se tuvo que determinar y configurar las aplicaciones y programas extra requeridos para levantar el servidor de impresión.

Comentado [RO18]: software instalado-a instalar, ya que se consideró-pero hay que considerar, tuvo que-deben.

Objetivo 4: Implementar el prototipo

Una vez instalado el sistema operativo, se ingresó al terminal del sistema operativo para realizar la configuración de comandos, en la cual se creó el servidor de impresión en base a código predeterminado. Así mismo se instaló la aplicación de control del servidor de impresión, en este caso *CUPS*, en el cual se agregó el dispositivo de impresión. Por último, se gestionó los permisos para el servidor de correo ya que por defecto este vendrá con normas de seguridad, para ello se tuvo que permitir el acceso de fuentes desconocidas.

Comentado [RO19]: creó-crea, instaló-instala, agregó-agrega, gestionó-gestiona, tuvo-deberá.

Objetivo 5: Realizar pruebas de funcionamiento

Se realizó una conexión remota mediante el un dispositivo externo, el cual tuvo que contar únicamente con la dirección IP que se le designó en la configuración del software. Se realizó cambios en las configuraciones básicas para comprobar su efectividad. Así se realizó la conexión al servidor de impresión y se envió un archivo o documento desde un dispositivo celular para que realice la debida impresión. Se tuvo que comprobar la verificación de impresión mediante el servidor de correo el cual envió un mensaje tanto de Outlook como Gmail, comprobando así el correcto funcionamiento.

Objetivo 6: Optimizar el dispositivo

Una vez realizadas las pruebas de funcionamiento, empezó el proceso de optimización del dispositivo, para lo cual se implementó una carcasa para el alojamiento del Raspberry Pi con el objetivo de que el dispositivo pueda ser manipulado sin ningún tipo de problema y adicionalmente se proporciona protección extra para que el dispositivo no quede sin protección ante la intemperie. Finalmente, se realizaron video tutoriales para el uso y el mantenimiento correcto del dispositivo.

Comentado [RO20]: realizó-realiza, tuvo que-deberá, realizó-realiza, así-ahora, realizó-realiza, envió-envía, tuvo que-deberá, envió-enviará, de-a.

3. RESULTADOS Y DISCUSIÓN

3.1 Análisis de los requerimientos

Varias de las impresoras que se tienen en un hogar o en una empresa no cuentan con una conexión inalámbrica y únicamente son funcionales si están conectadas mediante cables USB a un computador, el problema yace que las redes inalámbricas están tomando la posta dentro de la cotidianidad. Hoy en día no es tan visto el uso de impresoras alámbricas en casas, escuelas, colegios, oficinas, entre otros, lo que provoca que impresoras que aún tiempo de vida útil queden en el olvido.

Con este proyecto se quiere recuperar ese tiempo de vida útil a esas impresoras que quedaron en el olvido, creando un servicio de impresiones que se ajuste a impresoras alámbricas, este servicio cuenta con la implementación de una aplicación de impresión denominada *CUPS* dentro de la placa de desarrollo *Raspberry Pi*, esta aplicación será el enlace que tenga el usuario para registrar las impresoras que serán conectadas a la placa antes mencionada.

El servicio desarrollado permite que las impresoras conectadas a la placa *Raspberry Pi* sean administradas de manera inalámbrica por dispositivos dentro de la misma red, computadoras de escritorio, laptops o incluso dispositivos móviles pueden hacer uso de este servicio.

El éxito de las impresiones será notificado a través de un correo ingresado por el usuario, este servicio está implementado en base a *scripts* o programas realizados en el lenguaje de programación *Python*.

3.2 Análisis del hardware

Selección de la placa de desarrollo

Para la selección de la placa de desarrollo, se hizo un análisis de las placas que cumplieran con los requerimientos necesarios del *software*, estos requerimientos tienen conexión directa con las especificaciones que ofrecen las placas a investigar.

En la Tabla 3.1 **Tabla 3.1** Especificaciones de Raspberry Pi 3 y Arduino Mega 2560 se concentran las especificaciones de las dos primeras placas analizadas para el desarrollo del prototipo.

Tabla 3.1 Especificaciones de *Raspberry Pi 3* y *Arduino Mega 2560* [21] [22]

Especificaciones	<i>Raspberry Pi 3</i>	<i>Arduino Mega 2560</i>
Procesador	Procesador <i>Broadcom</i> BCM2837, <i>Cortex-A53</i> de 64 bits (4 núcleos)	<i>ATmega2560</i> (1 núcleo)
Frecuencia de reloj	1.2 GHz	16 MHz
Almacenamiento	Expandible a tamaño de memoria SD insertada	256 KB
RAM	1 GB	8 KB
Entradas	40 pines <i>GPIO</i> , 4 entradas <i>USB 2.0</i> , puerto de cámara, puerto <i>displayport</i> para incluir un <i>display</i> lcd, puerto <i>microSD</i>	54 pines <i>GPIO</i> (15 pines para salida PWM y 16 pines para entrada analógica)
Sistemas Operativos compatibles	<i>Raspbian (Raspberry Pi OS)</i> , distribuciones de <i>Linux</i>	No soporta ningún sistema operativo

Especificaciones	<i>Raspberry Pi 3</i>	<i>Arduino Mega 2560</i>
Adaptación con lenguajes de programación	C++, <i>Python</i> , <i>Java</i>	C++
Precio	\$ 40 - \$46	\$42

Se concluyó que, la placa *Raspberry Pi Model 3* es la opción que más se acomoda a las necesidades del proyecto. La compatibilidad con sistemas operativos de libre distribución, la adaptación a varios lenguajes de programación además de la capacidad del procesador, han sido las características con más peso que se tomaron en cuenta al momento de elegir la placa de desarrollo.

Luego del crucial evento de elección de la placa, se tomó en cuenta que la placa *Raspberry Pi Model 3* cuenta con dos modelos (B y B+). A continuación, se presentan las características de cada modelo para la elección definitiva de la placa de desarrollo.

Tabla 3.2 Especificaciones *Raspberry Pi 3 B y B+* [22] [23]

Especificaciones	<i>Raspberry Pi 3 B</i>	<i>Raspberry Pi 3 B+</i>
Procesador	Procesador <i>Broadcom</i> BCM2837, <i>Cortex-A53</i> de 64 bits (4 núcleos)	Procesador <i>Broadcom</i> BCM2837, <i>Cortex-A53</i> de 64 bits (4 núcleos)
Frecuencia de reloj	1.2 GHz	1.4 GHz
RAM	1 GB	1 GB
Conectividad	<i>WiFi</i> doble banda (2.4 GHz y 5 GHz) AB de <i>Ethernet</i> : 100 Mbps <i>Bluetooth</i> 4.1	<i>WiFi</i> doble banda (2.4 GHz y 5 GHz) AB de <i>Ethernet</i> : 300 Mbps <i>Bluetooth</i> 4.2
Entradas	40 pines GPIO, 4 entradas USB 2.0, puerto de cámara, puerto <i>displayport</i> para incluir un <i>display</i> lcd, puerto microSD	40 pines GPIO, 4 entradas USB 2.0, puerto de cámara, puerto <i>displayport</i> para incluir un <i>display</i> lcd, puerto microSD
Precio	\$ 40	\$46.90

A simple vista, los modelos de *Raspberry Pi Model 3* no se diferencian si se analizan características físicas como se observa en la Tabla 3.2

Raspberry Pi 3 B y B+, las diferencias más grandes se encuentran en la frecuencia máxima que pueden alcanzar los núcleos del procesador que, junto con el *upgrade* de la conectividad son las características (se puede trabajar en dos bandas) que hicieron de la *Raspberry Pi Model 3 B+* la placa escogida del silo de *Raspberry Pi 3* para el desarrollo del proyecto. Se puede mencionar que en el precio no existe una gran diferencia entre un modelo u otro así que, no se contempló esta característica como crítica.

Dispositivo de impresión

La impresora o dispositivo de impresión a utilizar durante el desarrollo del proyecto deberá contar con un puerto USB tipo B, el cual es el conector predominante que se ha utilizado para interconectar ordenadores y periféricos como: escáneres e impresoras, pero con la llegada de conectores más pequeños, su uso ha ido decreciendo a lo largo de los últimos años, es por esta razón que se ha decantado por el uso de este puerto ya que la gran mayoría de las impresoras de antigua generación hacen uso del mismo.

Existen dos tipos de conectores: uno convencional o que utiliza el estándar USB 1.0 o USB 2.0 y el otro un poco más moderno que utiliza el estándar USB 3.0 que se diferencia por contar con una pestaña azul en el interior.

El dispositivo que se utilizará durante el desarrollo del proyecto será un dispositivo multifunción, es decir que puede operar como un periférico de entrada/salida a un ordenador de manera autónoma y sin necesidad de que el ordenador esté encendido, las funciones básicas con las que debe contar son: impresión, fotocopiado, y escaneado de documentos o archivos.

3.3 Análisis del software

Selección del sistema operativo

Para la selección del sistema operativo, fue necesario realizar un análisis de los sistemas operativos disponibles para ser instalados en la *Raspberry Pi Model 3B+*, la compatibilidad de la placa con sistemas operativos de libre distribución hace que este análisis contemple a *Ubuntu Core* y *Raspberry Pi Os (Raspbian)* como tal, el elegido deberá trabajar en conjunto con el *hardware* de manera eficiente, enfocándose en

características como: variedad de aplicaciones, capacidad de procesamiento, seguridad, etc.

En la Tabla 3.3 se detallan las especificaciones de los sistemas operativos.

Tabla 3.3 Especificaciones de Ubuntu Core y raspberry Pi OS

Especificaciones	Ubuntu Core	Raspberry Pi OS
Modelo de desarrollo	Código abierto	Software Libre y código abierto
Licencia	GNU General Public License entre otras	General Public License entre otras
Núcleo	Linux	Linux
Tipo de núcleo	Monolítico	Monolítico
Interfaz gráfica	CLI, GUI	PIXEL

Además de las especificaciones dispuestas en la **¡Error! No se encuentra el origen de la referencia.**, se analizarán las ventajas y desventajas con las que cuenta cada uno de los sistemas operativos como se muestra en la Tabla 3.4

Tabla 3.4 Comparación entre Ubuntu Core y Raspberry Pi OS [24] [25]

	Ubuntu Core	Raspberry Pi OS
Ventajas	<ul style="list-style-type: none"> • Permite la creación e innovación de manera segura, ya que las actualizaciones de seguridad son constantes y vienen integradas dentro del sistema operativo. • Evita la instalación de programas no autorizados. • El cifrado de disco cumple con requisitos de privacidad a nivel de usuario. 	<ul style="list-style-type: none"> • Posee miles de paquetes anteriormente compilados y estables. • El <i>kernel</i> es configurable, lo que favorece a la optimización del funcionamiento de procesador. • Es de los sistemas operativos más estables en la actualidad.

	<i>Ubuntu Core</i>	<i>Raspberry Pi OS</i>
	<ul style="list-style-type: none"> • La seguridad es un factor de vital importancia. 	<ul style="list-style-type: none"> • Es casi nula la existencia de <i>malware</i> para este sistema operativo. • Tiene compatibilidad con <i>Raspberry Pi</i>
Desventajas	<ul style="list-style-type: none"> • Una de las principales desventajas de Ubuntu Core es la limitada variedad de aplicaciones. • Una de las principales críticas a Ubuntu está relacionada a la "aparente comercialización". • Con cada actualización, Ubuntu se aleja de la identidad de código abierto. 	<ul style="list-style-type: none"> • Su instalación es compleja si no se tiene un conocimiento previo. • Las actualizaciones de software suelen retrasarse más de la cuenta.

Después de haber analizado las características y especificaciones que ofrecen cada uno de los sistemas operativos, se llegó a la conclusión de que el sistema operativo que se adhiere más a las capacidades técnicas del proyecto es *Raspberry Pi OS (Raspbian)*, ya que cuenta con la compatibilidad inherente con *Raspberry Pi*, pero cabe mencionar que esta compatibilidad no es gracias a *Raspberry* como organización, sino se debe a que la comunidad aficionada a *Raspberry Pi* que ha fundamentado esa compatibilidad en investigaciones propias y sin ánimo de lucro.

En realidad, estos dos sistemas operativos ofrecen una experiencia de desarrollo libre, pero características como la compatibilidad y en gran medida, el poder de modificación del *kernel* para la optimización han hecho que se decante por *Raspberry Pi OS*, también es importante mencionar que, la seguridad que ofrece *Ubuntu Core* a los dispositivos

asociados es magnífica pero la poca variedad de aplicaciones dificulta el desarrollo del proyecto.

Raspberry Pi OS cuenta con tres versiones: *Raspberry Pi OS* con escritorio y software recomendado, *Raspberry Pi OS* solo con escritorio y *Raspberry Pi OS Lite*, siendo la versión con escritorio y *software* recomendado la más completa de entre las disponibles.

La versión solamente escritorio cuenta con la experiencia que *Raspberry Pi OS* ofrece, pero de una manera más compacta, para el desarrollo del proyecto no era necesario contar con *softwares* extras los cuales puedan ocasionar un rendimiento menor al esperado, debido al consumo innecesario de recursos extra.

Selección del lenguaje de programación

En la selección del lenguaje de programación se tomaron en cuenta varias características como la compatibilidad con el sistema operativo instalado y la complejidad de su uso.

En la Tabla 3.5 se presenta una comparación entre los lenguajes de programación *Java* y *Python*.

Tabla 3.5 Comparación entre *Java* y *Python* [26] [27]

	<i>Java</i>	<i>Python</i>
Ventajas	<ul style="list-style-type: none"> • Es simple de implementar. • Ofrece funcionalidades de un lenguaje potente, eliminando características confusas que confundan al programador. • Incorpora características muy útiles. • La liberación de memoria es llevada a cabo por la aplicación reduciendo la fragmentación de la memoria. 	<ul style="list-style-type: none"> • Es un lenguaje simplificado, lo que provoca una adaptación a un modo de lenguaje de programación. • Ofrece muchas herramientas, donde a veces no es necesario declarar variables en todo momento. • Es sencillo de aprender y aplicar. • Es un lenguaje muy ordenado, cuenta

	<i>Java</i>	<i>Python</i>
	<ul style="list-style-type: none"> • Reduce errores comunes que se pueden encontrar en <i>C</i> y <i>C++</i>. • Con es un lenguaje orientado a objetos, se vuelve en un lenguaje dinámico. • Es un lenguaje interpretado, lo que significa que puede ejecutar directamente el código objeto. 	<ul style="list-style-type: none"> • con módulos bien organizados. • Ofrece una gran cantidad de funciones y librerías. • Es un lenguaje multiplataforma y orientado a objetos. • Tiene licencia de código abierto. • Es un gran lenguaje para realizar scripts.
Desventajas	<ul style="list-style-type: none"> • Disminuye en gran medida el rendimiento durante la ejecución de los programas. • Se necesita una experiencia previa, si se entra al mundo de la programación no es recomendable usar Java. • La sintaxis de Java es un poco más complicada a diferencia de Python. • Posee una limitación con distintos programas lo que hace que la experiencia se entorpezca. 	<ul style="list-style-type: none"> • La implementación de funciones web no es tan sencilla. • La configuración es un poco difícil. • Las librerías que vienen incluidas no son de completa utilidad y se opta por el uso de librerías externas. • Es un poco lento por ser un lenguaje interpretado.

Habiendo analizado las ventajas y desventajas que se tienen con el uso de los lenguajes de programación mencionados, se decidió utilizar *Python*, los antecedentes presentados son la muestra de lo que este lenguaje de programación puede ofrecer un desarrollo óptimo.

La compatibilidad con sistemas operativos de libre distribución como lo es *Raspberry Pi OS* hace que con *Python* se tenga varias posibilidades de aplicación, su uso simple y ordenado, la simplificación en la escritura de variables, su gran uso en la realización de *scripts* y el uso de librerías que vienen por defecto, hacen de este lenguaje de programación el mejor candidato.

Selección de la aplicación del servidor de impresión

En la selección de la aplicación del servidor de impresión se tomó en cuenta dos de las opciones que más resaltan para Linux y los sistemas basados en *Linux*, las opciones que se manejaron fueron *CUPS (Common Unix Printing System)* y *Samba o SMB (Server Message Block)*.

Samba, siendo básicamente una colección de aplicaciones de *Unix* que se implementa a través del protocolo *SMB* el cual es utilizado para tareas u operaciones en un modelo cliente-servidor dentro de una red, lo más importante que puede ofrecer *Samba* es la comunicación con productos de *Windows*, es decir, que con *Samba* se puede ingresar a una red *Microsoft* actuando como servidor [28].

Por otra parte, *CUPS* es un sistema de impresión que utiliza el protocolo *IPP (Internet Printing Protocol)* que es utilizado para gestionar de manera óptima una cola de impresión. El protocolo *IPP* permite que *CUPS* sea una aplicación dinámica de impresión ya que puede administrar las solicitudes de impresión, colas de impresión, además, permite explorar para buscar impresoras que se encuentren dentro de la red [17].

En la Tabla 3.6 se presenta una comparación de las principales ventajas y desventajas de las dos aplicaciones.

Tabla 3.6 Tabla comparativa entre Samba y CUPS [28] [17].

	Samba	CUPS
Ventajas	<ul style="list-style-type: none">• Permite compartir varios sistemas de archivos.	<ul style="list-style-type: none">• Compatibilidad con protocolos de impresión de siguiente generación.

	<i>Samba</i>	<i>CUPS</i>
	<ul style="list-style-type: none"> • Las impresoras serán compartidas mediante una instalación en el servidor como en el cliente. • Provee de una previsualización de los clientes en red. • Permite verificar clientes mediante un <i>login</i> en un dominio de <i>Windows</i>. • Los recursos de disco pueden ser compartidos entre <i>Linux</i> y <i>Windows</i>. 	<ul style="list-style-type: none"> • Detección automática de impresoras dentro de la red. • Configuración y mantenimiento a través de la interfaz web. • Capacidad de ampliar el número de impresoras conectadas. • Compatibilidad de descripción de impresoras <i>PPD</i> para ficheros.
Desventajas	<ul style="list-style-type: none"> • El acceso y los permisos son complicados de manejar sin tener conocimiento previo. • Es afectado por los cambios que se efectúen en <i>Windows</i>, con esto <i>Samba</i> se puede ver afectado por varios periodos de tiempo hasta que se actualicen los cambios. 	<ul style="list-style-type: none"> • El uso de protocolos como <i>LPD</i> se pueden usar a través de Internet, pero no ofrecen un servicio de autenticación. • La gestión de documentos al igual que el formato, deben ser administrados por la máquina que envía la información.

Luego de haber analizado con mayor detalle las aplicaciones que se tienen disponibles, se decidió utilizar la aplicación CUPS, el hecho de que CUPS ofrezca una capacidad de trabajar con más protocolos de impresión de siguiente generación, el tener una configuración más sencilla de implementar, y la gestión de solicitudes y colas de impresión son las características que permiten desarrollar de mejor manera el proyecto.

Realizando un inciso del tema con *Windows*, *Samba* ofrece cierta compatibilidad de *Linux* con *Windows* pero debido a la cantidad de actualizaciones que tiene, llegaría a ser uno de los principales problemas que se podría tener al momento de implementar el servidor, por lo que se planea realizar una página *web* en HTML básica, la cual será la interfaz que vea el usuario para poder registrar su nombre y dirección de correo electrónico que posteriormente se utilizará para enviar un correo de verificación de impresión.

Análisis del sistema de refrigeración

El análisis que se realiza a continuación se basa en la importancia de que el procesador trabaje de manera eficiente, a altas temperaturas, el procesador empieza a realizar los procesos de manera lenta por lo que un sistema de refrigeración es vital para el desarrollo del proyecto.

Primeramente, para comprobar la temperatura de la placa de desarrollo se realizó un programa detallado en la siguiente sección de Análisis de los códigos implementados (Figura 3.9) con el que se verifica la temperatura que tiene el procesador en ese momento, por lo que se realizó el mismo proceso agregando componentes refrigerantes como disipadores de calor o un ventilador 5 (V_{DC}) al procesador para observar los cambios de temperatura y ver cuál es la más favorable.

En la Tabla 3.7 se detallan los resultados que se obtuvieron ejecutando el *script* en tres escenarios: procesador sin ningún aditamento, procesador incorporado un disipador de calor, procesador incorporado un ventilador 5 (V_{DC}).

Tabla 3.7 Resultados de la medición de temperatura.

Mediciones	<i>Solo procesador</i>	<i>Disipador de calor</i>	<i>Ventilador DC</i>
1	60.1 °C	52.3 °C	33.2 °C
2	60.2 °C	51.9 °C	33.4 °C
3	60.8 °C	51.8 °C	33.5 °C
4	60.9 °C	52.0 °C	33.0 °C
5	60.6 °C	51.9 °C	32.9 °C
6	60.7 °C	51.7 °C	32.7 °C
7	60.4 °C	51.6 °C	32.6 °C
8	60.2 °C	51.9 °C	32.3 °C

Como se puede observar en los resultados obtenidos de la **Tabla 3.7**, hay una enorme diferencia entre el procesador sin ningún aditivo y el uso de un ventilador DC, la temperatura baja casi a la mitad, el uso de un disipador de calor de aluminio también ayuda en mitigar las altas temperaturas, pero no es tan efectivo como lo es el ventilador, pero cabe mencionar que es una buena alternativa si no se tiene un ventilador cerca.

La temperatura lograda con el ventilador DC es la temperatura óptima para trabajar con una placa como lo es una *Raspberry Pi*, los procesos se realizan de mejor manera manteniendo una baja temperatura en el procesador.

Análisis de los códigos implementados

Para analizar los códigos desarrollados para el proyecto, se hará uso de diagramas de bloque, donde se detallan los procesos a seguir de una manera simplificada de cada sección del código empleado.

Diagrama de bloques del proceso de cola de impresión

En la Figura 3.1 Diagrama de bloques del Proceso de Cola de Impresión

se detalla el proceso de extracción del archivo generado en el directorio de la aplicación de impresión *CUPS*, este directorio muestra los archivos que se generan al momento de iniciar una cola de impresión. Este archivo cuenta con un distintivo de los demás, *c0* es el prefijo con el que cuentan los archivos generados por la cola de impresión. En el momento que el programa encuentra el archivo generado con el prefijo antes mencionado, procederá a enviar el correo de verificación al usuario previamente ingresado y guardado en la base de datos.



Figura 3.1 Diagrama de bloques del Proceso de Cola de Impresión

Diagrama de bloques del servicio de correo

En la Figura 3.2 se muestra el diagrama de bloques que representa el proceso que realiza el servicio de correo, donde, recopila información la información ingresada por el usuario y guardada en la base de datos para luego, buscar el archivo generado en la ruta especificada donde se crean las colas de impresión, luego verificará la creación de este enviando un mensaje de verificación al usuario por medio de un correo electrónico.

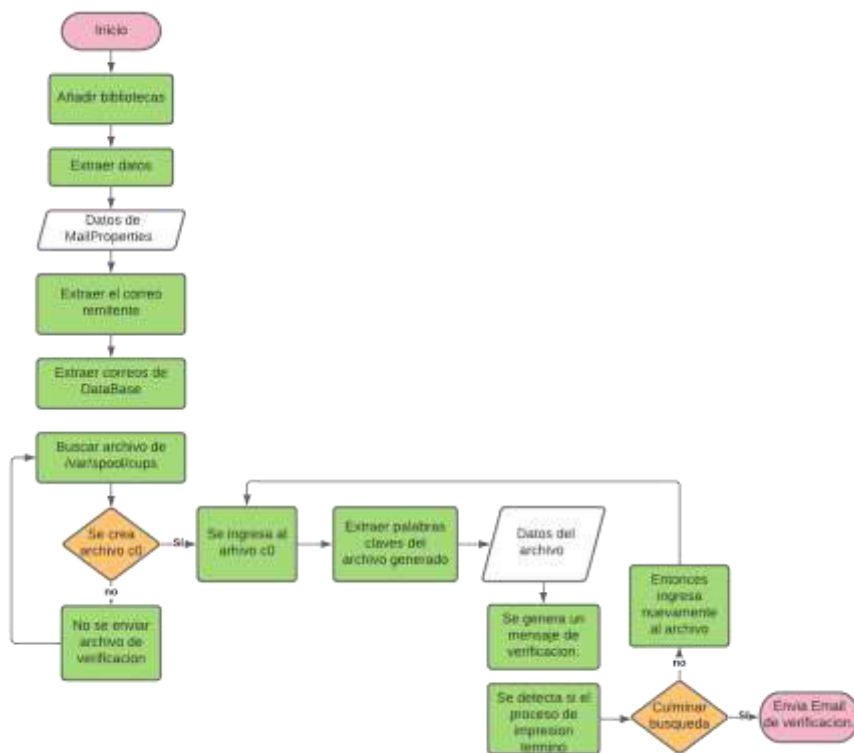


Figura 3.2 Diagrama de bloques del servicio de correo

Diagrama de bloques de la página web

La página web cuenta con una pestaña para agregar nuevos usuarios y otra donde se pueden verificar los usuarios anteriormente agregados, así como se muestra en la Figura 3.3. El proceso de introducción de un nuevo usuario es sencillo, el sistema permite ingresar una cadena de texto para identificar al usuario y un correo electrónico, este último será verificado si en realidad tiene la sintaxis de un correo común, eso quiere decir, que tenga @ y termine en .com, si el sistema verifica estos aspectos procederá a guardarlo en la base de datos y, por consiguiente, podrá ingresar a registrar su impresora si es la primera vez que ingresa.

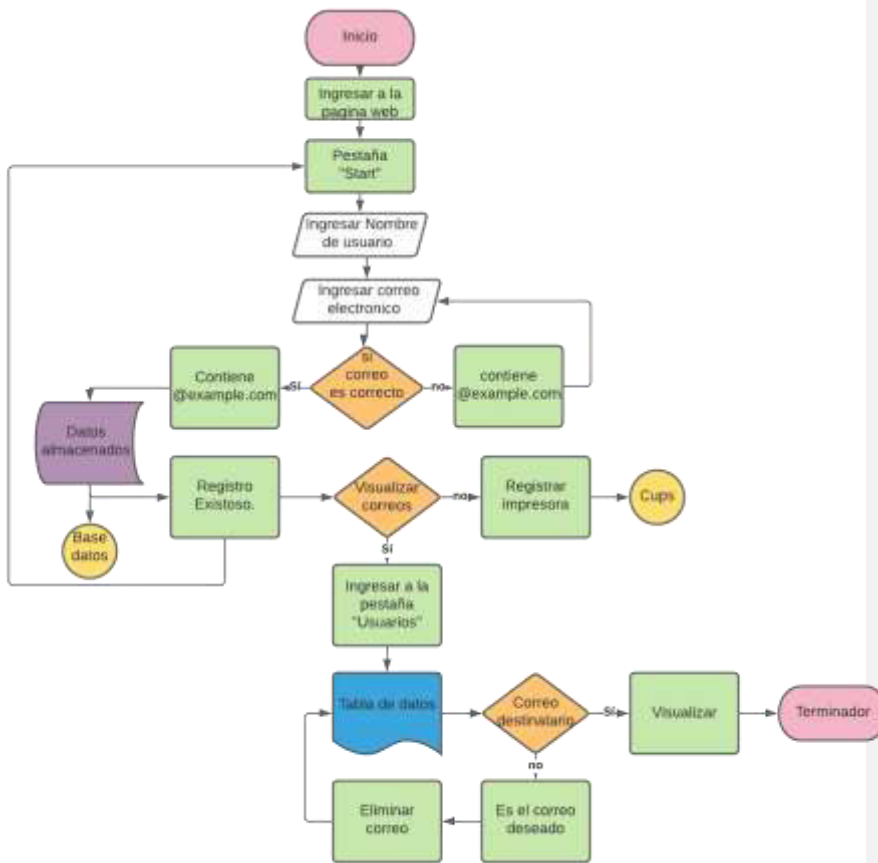


Figura 3.3 Diagrama de bloques de la Página Web

Diagrama de bloques del medidor de temperatura

En la Figura 3.4 se representa el funcionamiento del código para determinar la temperatura de la placa *Raspberry Pi*, el diagrama muestra el proceso que realizan las herramientas aplicadas como la usada en determinar los valores que arroja el GPU cada cierto tiempo como la temperatura del CPU alojada en el directorio presentado.

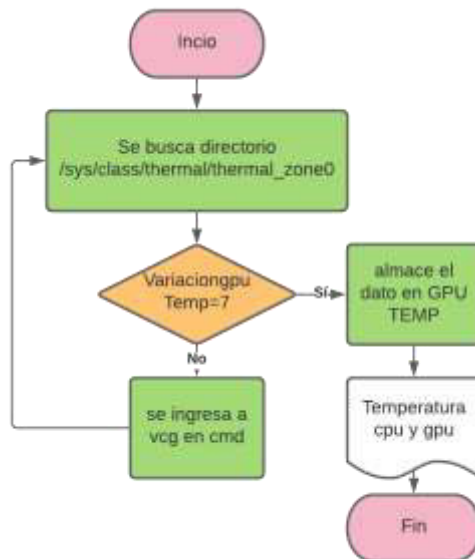


Figura 3.4 Diagrama de bloques del medidor de temperatura

Diagrama de bloques del control de temperatura

En la **¡Error! No se encuentra el origen de la referencia.** se detalla el proceso de control del ventilador, en base a las mediciones realizadas y constantes que cumplen la función de limitar la temperatura tanto máxima como mínima, esta temperatura será constantemente muestreada para poder generar un ciclo de giro en base a las necesidades ya que, el ventilador no solo estará en reposo y girando al ciento por ciento de su capacidad, sino que, pueda girar en función a un cálculo hecho utilizando las muestras de temperatura que se arroja en la verificación.

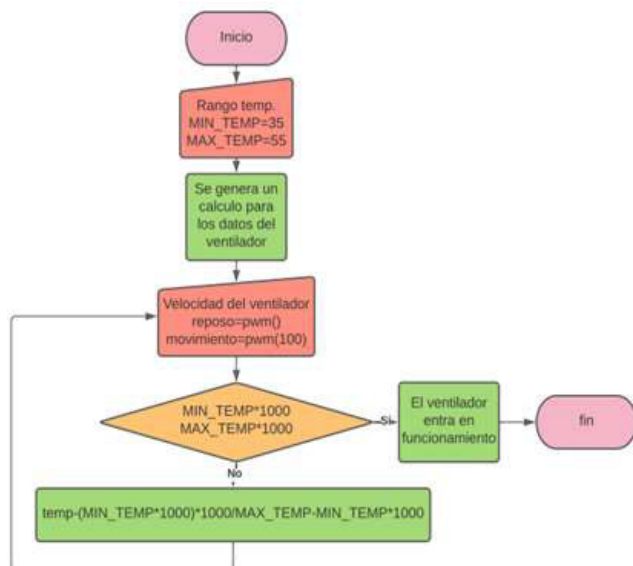


Figura 3.5 Diagrama de bloques del control del ventilador

Códigos implementados

Este proyecto cuenta con códigos complementarios que en conjunto hacen posible el desarrollo de este proyecto. Los códigos implementados se encuentran estructurados por librerías nativas de *Python*, pero otras instaladas desde un repositorio gracias a la terminal con la que cuenta el Sistema Operativo. El código principal desarrollado es el de Servicio de Correo, el cual actúa de base o núcleo que permite ir estructurando cada código en base a esta. El código implementado consiste en librerías que monitorean los eventos que ocurren en tiempo real durante la ejecución de un programa, esta librería monitorea en específico el proceso de creación de algún archivo que se genera dentro de la ruta especificada en el archivo de propiedades del servicio (**¡Error! No se encuentra el origen de la referencia.**) que es la ruta donde se guarda el archivo generado por la aplicación *CUPS* al momento de iniciar una nueva cola de impresión.

Este código cuenta con una cabecera donde se encuentran las propiedades utilizadas en el archivo de propiedades del servicio de correo y que fueron declaradas variables globales las que serán usadas en la creación del mensaje dentro del correo electrónico.

El primer punto de control es la función *CheckNewFileHandler* la cual escucha o verifica si en la ruta especificada se da origen a nuevos archivos, esta función trabaja en conjunto con la librería *watchdog* la cual fue anteriormente descrita.

Como segundo punto, el código cuenta con la función o método *read_file* donde se hace la lectura del archivo en dos puntos:

- Primero, lee el archivo de propiedades del servicio de correo.
- Segundo, lee el archivo creado y guarda toda la información en la cabecera, es decir, se guarda toda la información en las variables declaradas en la cabecera para la construcción del mensaje.

En el tercer punto se encuentra la función *print_state* que únicamente mostrará el estado de la impresión, exitoso o fallido.

El cuarto punto de control especifica la función que crea y envía el correo electrónico al usuario registrado, este proceso se lo hace mediante el uso del protocolo *SMTP* a través de un host *Gmail* para el envío del correo.

Finalmente, una función *main* que ejecuta el proceso de escuchar o verificar la creación de archivos en la ruta especificada, es decir, esta función principal ejecuta la función *CheckNewFileHandler*. Además, tiene por objetivo el mantener como un servicio al código ya que lo mantiene prendido hasta colocar el atajo por teclado *Ctrl+C* que termina o mata el proceso.

Alertas o *Flags*

Una vez extraído el archivo *c0* de la ruta especificada */var/spool/cups* creado por la creación de una nueva cola de impresión, es fundamental tomar en cuenta los datos que se desea extraer ya que en base a esto se crea el archivo de notificación que toma los parámetros indicados dentro del código del servicio de correo del archivo de propiedades, el formato del mensaje está indicado en la función que se encarga de enviar el correo electrónico al usuario registrado.

En la Figura 3.6 se detalla el código desarrollado.



Figura 3.6 Código del servicio de correo

En la Figura 3.7 se detalla el código de las propiedades del Servicio de Correo de donde se irán tomando los atributos para generar el mensaje de verificación enviado por medio de correo electrónico al usuario.



Figura 3.7 Código de las propiedades del servicio de correo

En la Figura 3.8 se muestra el código de la página *web* desarrollada que actuará de interfaz gráfica que será presentada al usuario y donde podrá agregar y suprimir usuarios, además, registrar su impresora si es un usuario nuevo.



Figura 3.8 Código de la página web

En la Figura 3.9 se muestra el código realizado para obtener mediciones de la temperatura tanto del *GPU* como del *CPU* de la placa *Raspberry Pi*.



Figura 3.9 Código del medidor de temperatura

Finalmente, en la Figura 3.10 se detalla el código utilizado para controlar el ciclo de giro del ventilador que en función de condiciones pueda girar dependiendo de la temperatura arrojada.



Figura 3.10 Código del control de temperatura

Estructuración de la base de datos

Para recopilar los datos ingresados por el usuario al momento de registrarse en el servicio web se usó *DB Browser for SQLite* para crear la base de datos que almacene los parámetros como: *ID*, nombre, *email*, fecha de creación del usuario en el servicio. En la Figura 3.11 se muestra como estará almacenada la información, teniendo registro de cada uno de los parámetros anteriormente indicados.

Name	Type	Schema
[-] Tables (2)		
[-] sqlite_sequence		CREATE TABLE sqlite_
name	TEXT	`name` TEXT
seq	TEXT	`seq` TEXT
[-] u_login_info		CREATE TABLE u_login
u_id	INTEGER	`u_id` INTEGER NOT N
u_name	VARCHAR (...	`u_name` VARCHAR (:
u_email	VARCHAR (...	`u_email` VARCHAR (:
u_created	DATE	`u_created` DATE NOT
u_last_conn	DATE	`u_last_conn` DATE NC

Figura 3.11 Estructura de la base de datos

3.4 Implementación del prototipo

Instalación de la aplicación *CUPS*

Una vez instalado el sistema operativo, se ingresa al símbolo del sistema, en el cual se instala la aplicación *CUPS*, para ello se actualizan los paquetes y se ingresa la siguiente línea de comandos:

```
sudo apt update
```

```
sudo apt upgrade
```

```
sudo apt-get install cups
```

```
sudo usermod -a -G lpadmin pi
```

Se ingresa al archivo .txt mediante:

```
sudo nano /etc/cups/cupsd.conf
```

En el archivo se realizan cambios acorde a la red:

En la sección "*Listen localhost:631*" se agrega "#" al inicio de la línea, esto para que se pueda escuchar o comunicar mediante el puerto 631. Además, debajo de esta línea se agrega "*Listen *:631*"

En la sección "*Restrict access to the server*" se agrega "*Allow 192.168.100.**" debajo de "*Order allow,deny*"

En la sección "*Restrict access to the admin pages*" se agrega "*Allow 192.168.100.**" debajo de "*Order allow,deny*"

En la sección "*Restrict access to configuration files*" se agrega "#" delante de "*Require user @SYSTEM*", debajo de esta línea se agrega "*Allow 192.168.100.**"

Una vez realizada la configuración se guarda con "*ctrl + x*".

Se procede a reiniciar la aplicación mediante:

```
sudo service cups restart
```

Ahora para poder compartir el servicio mediante la red se instala samba, mediante:

```
sudo apt-get install samba
```

Una vez instalado se modifica el archivo nano mediante:

```
sudo nano /etc/samba/smb.conf
```


En la sección `[printers]` se modifica la línea `"Read only"` en donde se cambia la palabra `"yes"` por `"no"`.

En la sección `[print$]` se modifica la línea `"guest ok"` en donde se cambia la palabra `"no"` por `"yes"`.

Se guarda el archivo mediante `"ctrl + x"` y se procede a reiniciar el samba mediante:

```
sudo systemctl restart smb
```

Ahora se ingresa al navegador *Chromium* y en la barra de buscador se ingresa:

```
192.168.100.143:631 o localhost:631
```

Ahora en la sección de Administrador en la pestaña de Impresoras se elige la opción de añadir impresora. En esta sección se verifica que la impresora esté conectada a la *Raspberry* y encendida, por consiguiente, se pide la contraseña y el usuario de la *Raspberry* (en este caso por defecto es usuario: *pi*, contraseña: *Raspberry*) y ahora en la pestaña que se abre se escoge la marca y versión de la impresora.

Se selecciona la casilla de compartir esta impresora y siguiente.

Se elige el paquete de instalación acorde a la impresora del usuario, y se añade las configuraciones.

En la sección de impresoras se observará que el usuario tiene conectada a la *Raspberry Pi*.

Instalación del hotspot

Para realizar la instalación de *Hotspot* se realiza:

```
sudo apt update
```

```
sudo apt upgrade
```

```
curl -sL https://install.raspap.com | bash // Esta aplicación será la que ayudará a realizar un Hotspot.
```

Mientras se esté instalando los paquetes se realiza una confirmación mediante la letra `"s"`, al terminar la instalación de la aplicación se procede a reiniciar el sistema para que se acoplen los cambios.

Para configurar el punto de acceso, es necesario entrar desde un navegador colocando *Raspberrypi.local* como dirección en la barra de búsqueda, aparecerá la ventana de acceso como se aprecia en la Figura 3.12 donde se colocarán las credenciales, usuario: *admin* y contraseña: *secret*.



Figura 3.12 Ingreso a la configuración del *Hotspot*

Dentro de la página de configuraciones, en la opción "*Hotspot*" se realiza la configuración básica de una red, en este caso el canal y la *SSID*, en la Figura 3.13 se muestra el cambio realizado al *SSID* realizado.

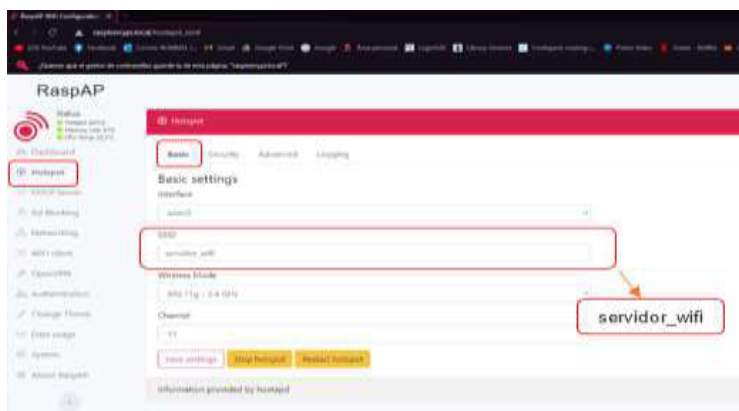


Figura 3.13 Configuración del *SSID*

En la opción de seguridad se podrá realizar la configuración de la contraseña para acceder al punto de acceso, en la Figura 3.14 se muestra la configuración de la contraseña.

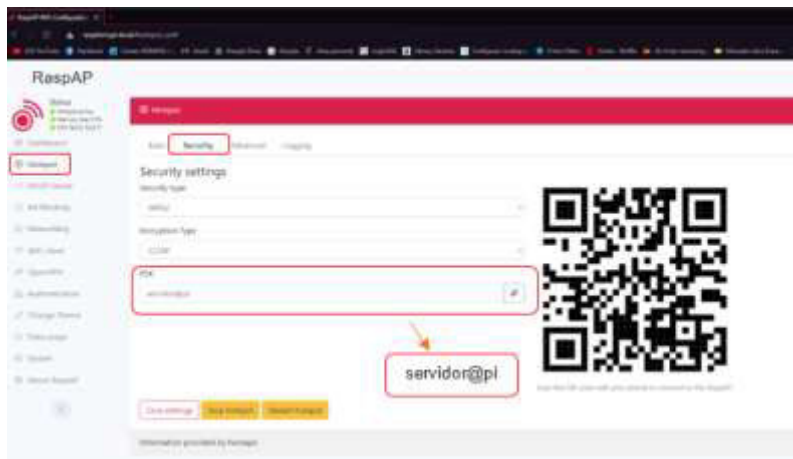


Figura 3.14 Configuración de la contraseña del *Hotspot*

Por último, en la pestaña Avanzado se puede configurar el número de usuarios a conectarse y el código de país.

Es importante guardar los cambios realizados dando clic en el botón “*save settings*” para que se ejecuten y actualicen.

Configuración del correo electrónico emisor

Para el uso del correo electrónico emisor se debe configurar los permisos en la cuenta de *Google*, es importante destacar que de preferencia el correo emisor debe ser un *Gmail*.

Lo primero es entrar al administrado de cuenta en la opción “Administrar tu cuenta de *Google*”.

Por consiguiente, se dirige a la opción de “seguridad”, en la cual se busca la opción de “Acceso de *apps* menos seguras” ubicada al final de la lista. Una vez se ingresa a esta opción se la activa (Figura 3.15), para de esta forma se podrá usar el correo para enviar los correos de verificación.



Figura 3.15 Configuración de acceso de aplicaciones

Ambiente virtual

A nivel de *Visual Studio Code* que es el editor de código fuente para *Python*, existen los ambientes virtuales o *virtual environments* que no son más que ambientes encapsulados donde ya se incluyen dependencias y librerías directamente desde una carpeta sin la necesidad de que estén instaladas a nivel del sistema operativo, en este ambiente todas las herramientas se encuentran empaquetadas y listas para correrlas evitando descargarlas de nuevo.

El objetivo de crear un ambiente virtual en la *Raspberry Pi* es tener todo el espacio que sea posible para un mejor rendimiento y distribución de los recursos para la ejecución de los servicios. En la Figura 3.16 se muestran la carpeta del ambiente virtual creado donde se encapsula las librerías utilizadas en el desarrollo del proyecto.

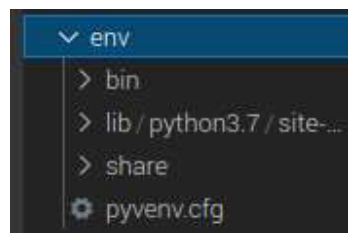


Figura 3.16 Carpeta del ambiente virtual

Servicio de correo

Para el servicio que soporta el envío de mensajes se implementó un programa en base a código desarrollado en *Python* explicado en el diagrama de flujo en la Figura 3.6 que centraliza las actividades del desarrollo del proyecto, este servicio implementa el control

necesario para el sistema de envío de mensajes a través de correo electrónico a los usuarios registrados previamente.

El sistema entra en funciones cuando el programa detecta un nuevo archivo creado en el directorio `/var/spool/cups` el cual es el lugar en donde se generan los archivos de cola de impresión almacenados en la carpeta raíz de la aplicación *CUPS*, este archivo tiene un identificador que lo diferencia de los demás y por eso es posible su detección. En la Figura 3.17 se pudo observar el identificador que tienen esta clase de archivos.

```
*****Welcome Back*****
-->Reading properties file
Got event for file /var/spool/cups/00000003
Name of file: 00000003
Got event for file /var/spool/cups/c00046.N
Name of file: c00046.N
```

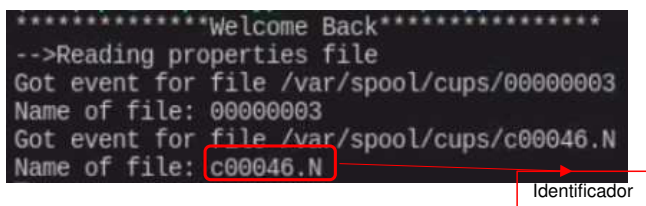


Figura 3.17 Identificador del archivo de impresión

Una vez detectado el archivo, el programa entra en un tiempo de reposo hasta que la impresión culmine, durante este tiempo se evita que el programa vuelva a ejecutarse teniendo varias lecturas y por tal enviando varios correos electrónicos.

Al finalizar la impresión, el programa devuelve un *token* para verificar que la impresión fue exitosa, a continuación, procede a enviar los correos electrónicos a los usuarios registrados.

Cuando la impresión haya finalizado, el programa recogerá la información del usuario que se almacenó en la base de datos y además recogerá parámetros encontrados en un archivo de configuraciones realizado en adición del código principal del servicio de correo, esto para mejorar el rendimiento del programa ya que este solo tendrá que recoger las propiedades del archivo y ya no del código en sí.

Propiedades del servicio de correo

Como se mencionó anteriormente, se desarrolló un archivo de propiedades para el servicio de correo con el propósito de mejorar la ejecución del programa principal. Este archivo contiene los parámetros que serán utilizados en el proceso de envío del correo electrónico como dirección *IP* del origen, dirección *IP* de la impresora a través de la aplicación *CUPS*, usuario, tipo de archivo, nombre de la impresora, estado de la impresión, el correo electrónico remitente y principalmente el directorio en donde se genera el archivo de cola de impresión de *CUPS* (Figura 3.18).

```

srcFolder=/var/spool/cups
startFileNames=c0
regex=\x00|\x01|\x02|\x03|\x04|\x05|\x06|\x07|\x08|\x09|\x0a|\x0b|\x0c|\x0d|\x0e|\x0f
printer_host_name=6ipps:
printer_name=ipp:
type_document=Office
print_stats=job-impressions-completed
origin_host_name=192
user_name= Redmi
sender=raulivancantunanna@gmail.com
password=ricol721854253
destline=rjvegapa295@gmail.com
subject=Mensaje de verificación
isB=Yes

```

Figura 3.18 Parámetros del archivo de propiedades

Los parámetros que se configuran en este archivo deben ser declarados en el código principal para poder ser llamados. En la Figura 3.19 se observa como en el programa principal llama a cada uno de los parámetros configurados en el archivo de propiedades sin la necesidad de ser desarrollados dentro del programa.

```

if name_file == "mail_properties":
    file = open(name_file, "r")
    for line in file:
        v_property = line.split("=")
        var_properties.update({v_property[0]: v_property[1].split("\n", 1)[0]})
    else:
        # Read file
        header_mail["created_date"] = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime())
        print(header_mail["created_date"])
        with open(name_file, "r", encoding="ascii", errors="replace") as input:
            file = input.readlines()
            regex = re.compile(var_properties["regex"])
            for line in file:
                for x in re.split(regex, line):
                    if x.startswith(var_properties["printer_host_name"]):
                        header_mail["printer_host_name"] = re.split(":", x)[1]
                        print(header_mail["printer_host_name"])
                    elif x.startswith(var_properties["type_document"]):
                        header_mail["type_document"] = x
                        print(header_mail["type_document"])
                    elif x.startswith(var_properties["user_name"]):
                        header_mail["user_name"] = re.split(" ", x)[0]
                        print(header_mail["user_name"])
                    elif x.startswith(var_properties["origin_host_name"]):
                        header_mail["origin_host_name"] = re.split(":", x)[1]

```

Figura 3.19 Llamada de los parámetros

Servicio web

El servicio *web* fue desarrollado en base a un marco de trabajo o plantilla para simplificar la elaboración de la aplicación *web*. Una de las aplicaciones para trabajar en conjunto con *Python* es *Flask* que permite crear aplicaciones *web* de manera sencilla, la utilidad dada a esta aplicación es que cuenta con una gran variedad de complementos o *plugins* que facilita el trabajo si se desea agregar nuevas funcionalidades durante el desarrollo.

El uso de *flask* permitió desarrollar de manera diferenciada el modelo de datos que se lograba obtener, ya que se separó la parte de información (registro que se va a tener en la aplicación que normalmente está almacenada en la base de datos), el control de datos

del usuario (formularios de gestión para las peticiones de la aplicación *web*) y por supuesto, la interfaz gráfica que el usuario podrá visualizar (página *HTML*).

El servicio *web* está estructurado con tres módulos: *index.html*, *success.html* y *list_mail.html* que pertenecen a cada una de las pantallas que el usuario podrá visualizar; a cada uno de los módulos se les asocia una ruta o *path* que gestiona cada una de sus funcionalidades.

Módulo *index.html*

El usuario podrá visualizar este módulo como la pantalla de inicio al momento de iniciar el servicio *web* en el navegador, como se observa en la Figura 3.20 el usuario podrá asignar un nombre de usuario y colocar una dirección de correo electrónico con el que podrá recibir el mensaje de éxito de impresión por parte del servicio de correo.



Figura 3.20 Pantalla de inicio del servicio web

El funcionamiento de este módulo se muestra en el diagrama de flujo de la Figura 3.21.

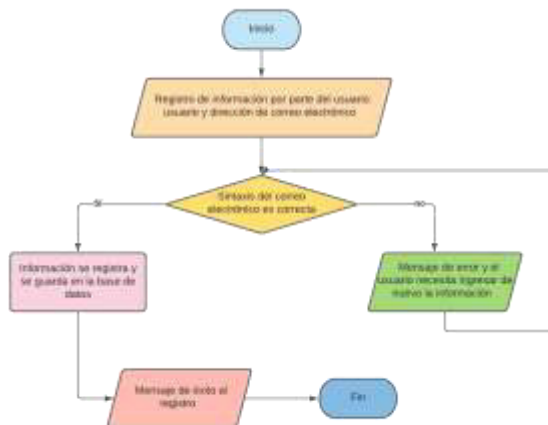


Figura 3.21 Diagrama de flujo del módulo *index.html*

Módulo *success.html*

Este módulo es básicamente el registro exitoso de la información del usuario tanto el nombre de usuario como la dirección de correo electrónico, al llenar los campos solicitados se desplegará la pantalla de confirmación de registro del usuario como se observa en la Figura 3.22 dentro del sistema. Para más detalle, en la Figura 3.23 se muestra el diagrama de flujo del funcionamiento de este módulo.

Dentro del módulo está asociada la dirección *IP* correspondiente a la aplicación *CUPS* para que el usuario pueda ingresar a registrar su impresora si es la primera vez que va a utilizar el servicio o si desea agregar una nueva impresora al servidor.

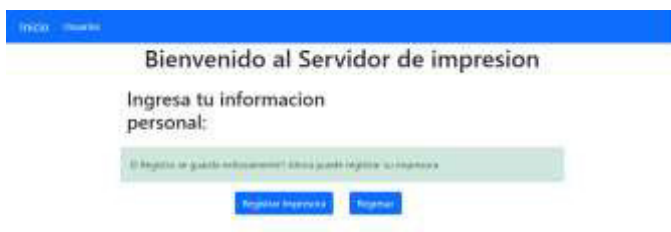


Figura 3.22 Confirmación de registro del usuario

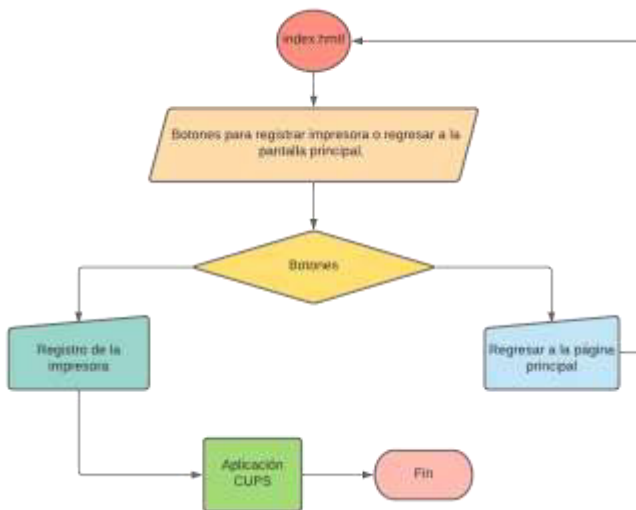


Figura 3.23 Diagrama de flujo del módulo *success.html*

Módulo *list_mail.html*

El módulo *list_mail.html* es el listado de usuarios que han ingresado al servicio *web*, por lo que se encontrarán listados, en adición, cada uno de los usuarios contará con la fecha de creación y la fecha de ingreso (Figura 3.24). Para el ingreso erróneo de cualquier usuario se colocó un botón de operación con el que se podrá eliminar cualquier usuario ingresado incorrectamente. En la Figura 3.25 se muestra el diagrama de flujo del funcionamiento de este módulo.

ID	Nombre	Correo	Fecha Creación	Fecha Login	Operaciones
28	issac	lorgapic29@gmail.com	2021-08-22	2021-08-22	[Botón de borrado]
29	Marcos	ymmelorga@proton.me	2021-08-22	2021-08-22	[Botón de borrado]
30	issac	lorgapic29@gmail.com	2021-08-22	2021-08-22	[Botón de borrado]

Figura 3.24 Listado de usuarios ingresados

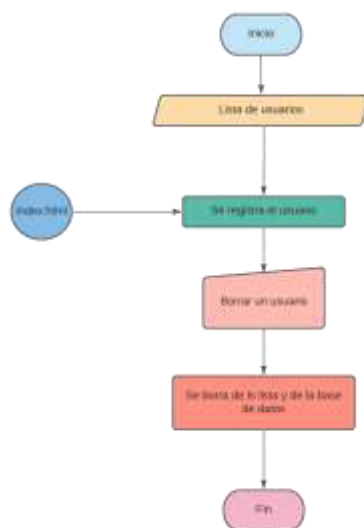


Figura 3.25 Diagrama de flujo del módulo *list_mail*

Puertos de la placa *Raspberry Pi*

Para la implementación del proyecto, fue necesario adaptar uno de los puertos *USB* de la placa a un puerto paralelo *LPT1* o *Line Print Terminal* por sus siglas en inglés, este puerto de 25 pines servía a las impresoras que realizaban el trabajo de impresión de manera línea a línea. El paso del tiempo y con la llegada de la impresión a láser, ha provocado que este tipo de impresoras hayan sido descontinuadas.

En la Figura 3.26 se observa los 2 pares de puertos *USB 2.0* que tiene la placa y el puerto que será adaptado a un puerto *LPT1*.

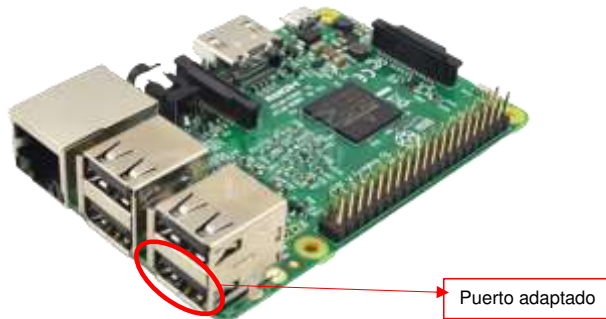


Figura 3.26 Puertos de la Raspberry Pi [28]

La adaptación se logrará mediante el uso de un cable adaptador *USB* a *LPT1* (Figura 3.27) que será colocado en la parte baja de la cubierta pensada para la placa.



Figura 3.27 Cable adaptador *USB* a *LPT1* [29]

3.5 Pruebas y Análisis de Resultados

Etapa de inicio automatizado

Al ofrecer un servicio en donde no se requiere intervención del usuario con el prototipo, es necesario contar con que los servicios inicien de forma automática, por tal motivo, se realizó un arranque automatizado en base a comandos adjuntando el directorio de donde se encuentra alojado el servicio.

Los servicios iniciarán de manera automática a través de la carpeta `/etc/systemd/system`, en donde se crearán dos archivos, uno para el servicio *web* y otro para el servicio de correo, en la Figura 3.28 se muestra el código implementado para el inicio automatizado.

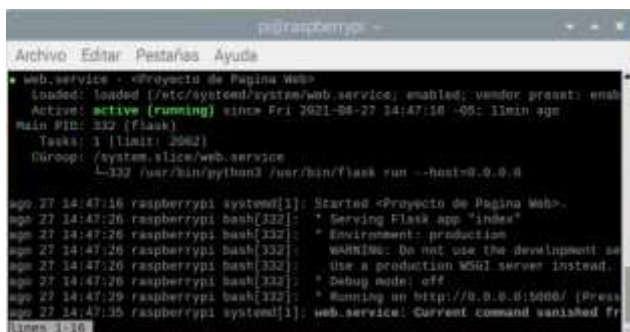


Figura 3.28 Código implementado para el inicio automatizado

En la Figura 3.29 se muestra la ejecución del servicio de correo y en la Figura 3.30 se muestra la ejecución del servicio *web*.

```
pi@raspberrypi ~$ systemctl status mail.service
mail.service - Proyecto de Mail
Loaded: loaded (/etc/systemd/system/mail.service; enabled; vendor preset: ena
Active: active (running) since Fri 2021-08-27 14:47:16 -05; 18min ago
Main PID: 234 (sudo)
Tasks: 5 (limit: 2048)
Group: /system.slice/mail.service
       └─ 234 sudo python3 mailService.py
          └─ 422 python3 mailService.py
ago 27 14:47:16 raspberrypi systemd[1]: Started Proyecto de Mail.
```

Figura 3.29 Inicio automático del servicio de correo



```
pi@raspberrypi ~$ systemctl status web.service
web.service - «Proyecto de Pagina Web»
Loaded: loaded (/etc/systemd/system/web.service; enabled; vendor preset: enab
Active: active (running) since Fri 2021-04-27 14:47:16 -05; 11min ago
Main PID: 332 (flask)
Tasks: 3 (limit: 9962)
Group: /system.slice/web.service
CGroup: /system.slice/web.service
└─332 /usr/bin/python3 /usr/bin/flask run --host=0.0.0.0

ago 27 14:47:16 raspberrypi systemd[1]: Started «Proyecto de Pagina Web».
ago 27 14:47:26 raspberrypi bash[332]: * Serving Flask app "index"
ago 27 14:47:26 raspberrypi bash[332]: * Environment: production
ago 27 14:47:26 raspberrypi bash[332]: WARNING: Do not use the development se
ago 27 14:47:26 raspberrypi bash[332]: * Use a production WSGI server instead.
ago 27 14:47:26 raspberrypi bash[332]: * Debug mode: off
ago 27 14:47:29 raspberrypi bash[332]: * Running on http://0.0.0.0:5000/ [Press
ago 27 14:47:35 raspberrypi systemd[1]: web.service: Current command vanished fr
lines 1-10
```

Figura 3.30 Inicio automático del servicio web

Etapas de registro de usuario

Una vez iniciados los servicios automáticamente, el proceso continúa con el registro del usuario en la página web. Para acceder a la página se debe digitar en la barra de búsqueda del navegador la siguiente dirección: *Raspberrypi.local:5000/start*, esta dirección corresponde al lugar donde se aloja el servicio web y por el puerto donde se encontrará el proceso de intercambio de información. El usuario podrá registrarse con cualquier identificador y usando cualquier dirección de correo, es necesario mencionar que, se debe ingresar una dirección de correo válida y con la sintaxis correcta de la misma de lo contrario, el usuario no podrá registrarse. En la Figura 3.31 se observa el registro de un nuevo usuario en la plataforma desde un navegador.

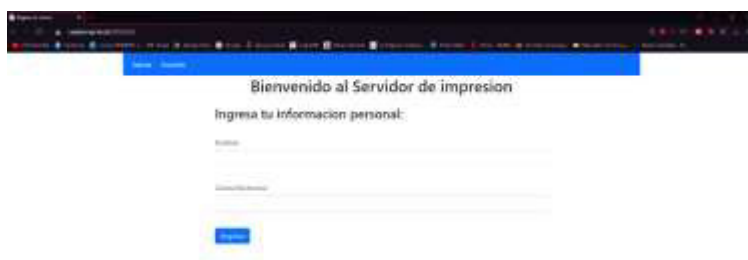


Figura 3.31 Formulario de registro

Luego de registrado el usuario, se desplegará un mensaje de registro exitoso por parte del servicio web, si el usuario desea registrar una impresora

Etapas de registro de impresora

Dentro de la aplicación CUPS en la sección "Administración", se debe asegurar que las casillas "Compartir impresoras conectadas a este sistema" y "Permitir administración

remota” se encuentren marcadas, si ese no es el caso, marcarlas y guardar los cambios realizados, sino las impresoras registradas no podrán ser compartidas a través de la red.

Al añadir una impresora la aplicación pedirá una validación con respecto al dispositivo se refiere, en el cuadro desplegado como se observa en la Figura 3.32, se pedirán las credenciales del servidor, en este caso serán las mismas que vinieron por defecto en la *Raspberry Pi*, usuario: *pi*, contraseña: *raspberrypi*

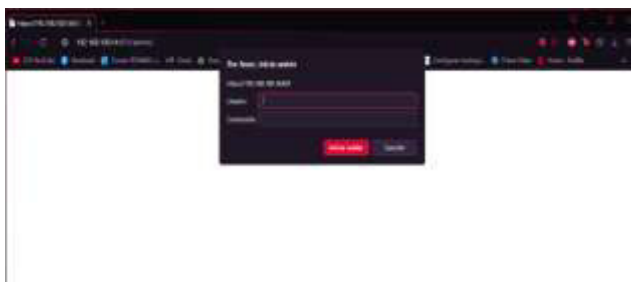


Figura 3.32 Credenciales de verificación

Al verificar las credenciales, se pasará al registro de la impresora, en una lista aparecerán las impresoras dentro de la red que se encuentran disponibles, en la Figura 3.33 se detallan las impresoras que se encuentran dentro de la red.



Figura 3.33 Lista de impresoras disponibles

Cuando se tiene la impresora escogida se procede al registro de la impresora, este proceso consta de la configuración de la impresora, como el nombre con el que se la va a poder ver en la red o la ubicación donde se va a encontrar la impresora, en la Figura 3.34 se muestra la configuración de la impresora, se debe mencionar que la casilla de compartición debe estar marcada para que la impresora pueda ser compartida.



Figura 3.34 Configuración de la impresora

El siguiente paso en el proceso, es la elección de los *drivers* con los que la impresora pueda funcionar, en la Figura 3.35 se ve la lista de drivers con los que cuentan la aplicación con respecto a la marca y modelo de la impresora, cabe mencionar que, la aplicación CUPS permite al usuario subir archivos propios de los *drivers* si no se encontrasen en la lista antes mencionada.

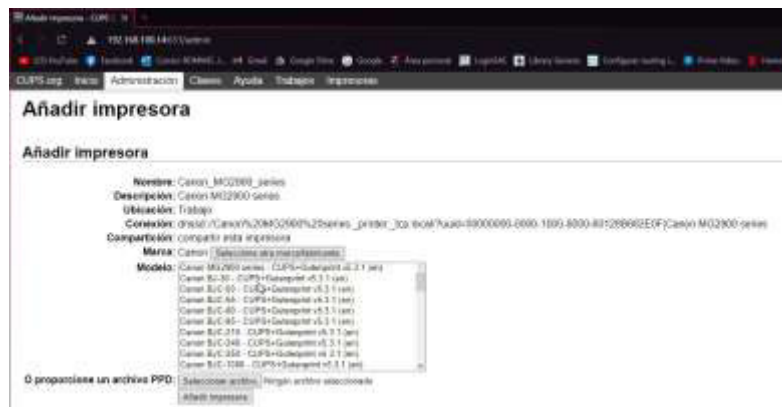


Figura 3.35 Drivers de modelos de impresoras

Luego del registro de la impresora, estará lista para ser ocupada, en la sección de Impresoras, se podrá ver que la impresora ya se encuentra en la lista de las impresoras

agregadas. En la Figura 3.36 se detalla la impresora agregada a las impresoras disponibles.



Figura 3.36 Impresoras disponibles dentro del servicio de impresión

Etapa de impresión

Para agregar el servicio de impresión de la impresora agregada, se necesita actualizar la lista de dispositivos disponibles al momento de imprimir. Al actualizar, se podrá observar que la impresora agregada ya podrá ser visualizada y lista para ser agregada a lista de impresoras disponibles para el proceso de impresión como se ve en la Figura 3.37.



Figura 3.37 Adición de impresora en el editor de texto

Al momento de haber concluido el proceso de impresión, el servicio en conjunto con el programa enviará un mensaje de verificación a la dirección de correo electrónico con la que el usuario se registró en la plataforma web. En la Figura 3.38 se muestran los parámetros con los que cuenta el mensaje, detallándolos a continuación:

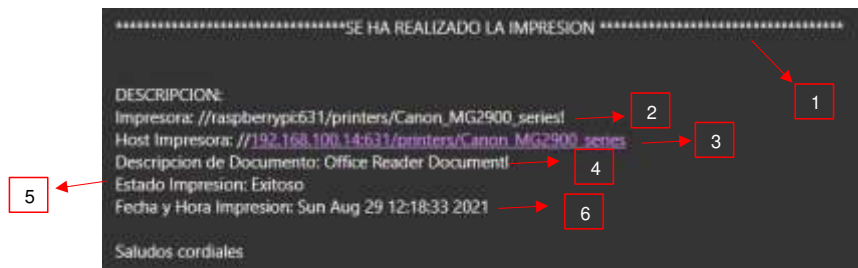


Figura 3.38 Parámetros del mensaje de verificación

1. Título del mensaje
2. Impresora registrada desde la que se realiza la impresión
3. Dirección de donde se envía la orden de impresión
4. Tipo de documento
5. Estado de la impresión
6. Fecha y hora en la que fue realizado el proceso de impresión

3.6 Optimización del dispositivo

Diseño de la cubierta

A partir de las consideraciones y especificaciones de la placa *Raspberry Pi* y tomando en cuenta la adaptación del puerto *LPT1* se decidió realizar una cubierta o carcasa para la protección de la placa y del adaptador mencionado.

El diseño comenzó en base a las especificaciones y medidas de la placa de desarrollo *Raspberry Pi*. En la Figura 3.39 se observa las medidas que tiene la placa.

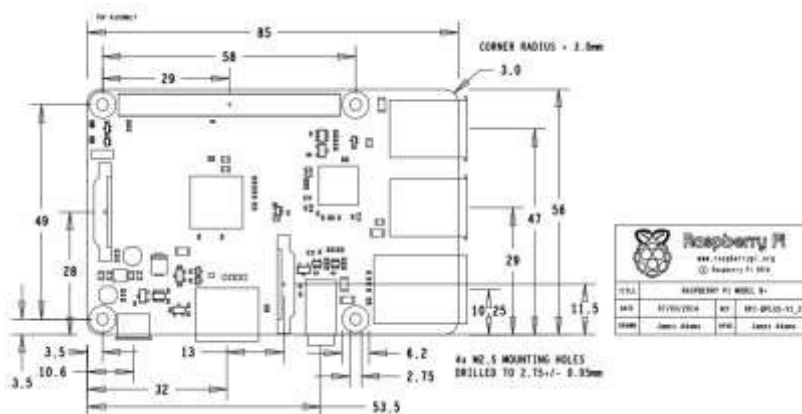


Figura 3.39 Medidas de la *Raspberry Pi*

Con las medidas predispuestas se procedió a digitalizar la placa, para ello se utilizó el *software* de diseño asistido por computadora *AutoCAD* el cual será utilizado para modelar tanto en 2D como en 3D. Ya obtenida la digitalización de la placa se empieza a diseñar una cubierta que cumpla con los requerimientos anteriormente mencionados, por lo que se propuso realizar una cubierta con dos secciones, tanto para la placa como para el adaptador *USB a LPT1* tomando en cuenta que este último es de gran tamaño debido a la cantidad de pines que tiene el puerto (Figura 3.39) en consecuencia es necesario contar con un espacio donde se acomode dicho adaptador.

Para el diseño inicial, se realizó el modelo que se observa en la Figura 3.40, cumplía con los requerimientos pedidos anteriormente, además de contar con el espacio abierto para los puertos de la placa, tanto *USB*, *Ethernet*, como *HDMI* y la entrada de alimentación.

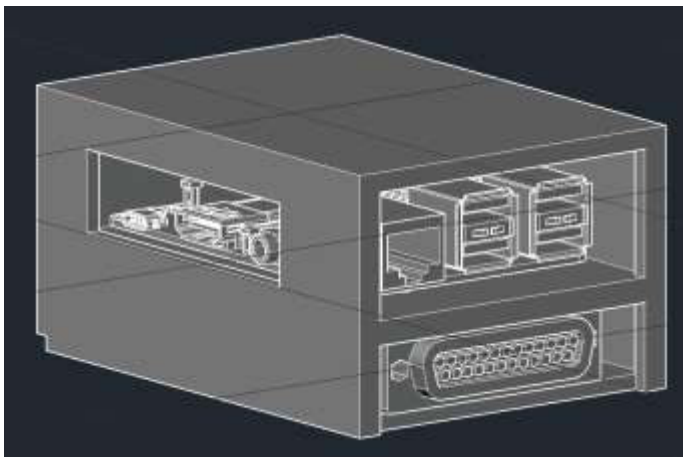


Figura 3.40 Primer modelo de la cubierta

Durante el desarrollo del prototipo, se encontraron varios problemas con la refrigeración de la placa, ni el ventilador ni los disipadores colocados en el procesador daban el abasto suficiente para que el aire caliente encerrado salga de la cubierta. Otro de los factores que se tomó en cuenta fue el tiempo durante el que pasaría encendido el prototipo, debido a que ofrece un servicio dentro la red necesita estar continuamente funcionando durante periodos largos de tiempo, es por eso por lo que se realizó pruebas de funcionamiento del sistema de ventilación durante un periodo de ocho horas.

En la Tabla 3.8 se detallan las temperaturas obtenidas en cada hora que pasó el prototipo en funcionamiento.

Tabla 3.8 Control de la temperatura durante 8 horas

Hora	1	2	3	4	5	6	7	8
Temperatura	35°C	38°C	42°C	46°C	49°C	51°C	55°C	60°C

Considerando el problema de refrigeración de la placa, se propuso realizar unos cambios en el diseño del primer modelo de la cubierta en donde se pueda abastecer de la suficiente refrigeración y expulsión del aire caliente. Los principales cambios realizados al primer modelo se pueden evidenciar en: el corte realizado a la altura de los pines *GPIO* de la placa y el corte circular en la parte superior de la cubierta donde se expulsará el aire caliente, el cambio en comparación al primer modelo se puede observar en la Figura 3.41.

Con los cambios realizados a la cubierta se procedió a realizar otro control de temperatura para verificar si existen variaciones diferentes a las obtenidas en la Tabla 3.8. Los datos recogidos con el nuevo sistema de ventilación se detallan en la Tabla 3.9.

Tabla 3.9 Control de temperatura de los dos modelos durante 8 horas

Hora	1	2	3	4	5	6	7	8
Primer modelo	35 °C	38°C	42 °C	46 °C	49 °C	51 °C	55 °C	60 °C
Segundo modelo	33 °C	34 °C	35 °C	33 °C	34 °C	36 °C	34 °C	35 °C

El rendimiento del segundo modelo en cuestiones de temperatura mejoró sustancialmente en comparación al rendimiento del primero modelo, la temperatura promedio que se obtiene es mucho menor por lo que se puede concluir que el prototipo puede estar en funcionamiento durante periodos prolongados de tiempo sin que la temperatura sea un factor que afecte su rendimiento.

Fabricación de la cubierta

Para elegir el material y el tipo de fabricación de la cubierta se tomó en cuenta ciertos materiales de los cuales el escogido, debía poseer características que faciliten el desarrollo del prototipo. Los materiales a comparar fueron: filamento de material termoplástico (material usado en impresiones 3D), tableros de densidad media o *MDF* y, por último, metacrilato o *PMMA*.

Estos materiales serán comparados en base a ciertos factores que son importantes para la fabricación de la cubierta, en la Tabla 3.10 se realiza una comparación entre estos tres materiales.

Tabla 3.10 Comparación entre posibles materiales de fabricación de la cubierta

Características	Filamento termoplástico	MDF	Metacrilato
Costo promedio	\$20 c/u rollo	\$11 c/u plancha	\$50 c/u lámina
Maleabilidad	Únicamente se realiza a través de la impresora	Corte Láser	Herramientas especiales para su corte
Tipo de acabado	Una sola pieza con el acabado diseñado	El corte láser genera quemaduras en bordes	Liso por el tipo de material
Tiempo de fabricación	10 horas promedio	24 horas promedio	50 horas promedio
Durabilidad	Material con rigidez media, pero baja durabilidad	Material consistente y durable	Material altamente durable

Concluyendo la comparación entre los posibles candidatos de material de fabricación de la cubierta, se intuyó que la madera *MDF* es el material más adecuado por las características que brinda como: su bajo costo, maleabilidad, tiempo de fabricación y durabilidad.

Una vez seleccionado el material, se diseñaron las piezas en el *software AutoCAD*, ya que la cubierta no se iba a realizar en una sola pieza, se ideó diseñar pieza por pieza, tanto paredes laterales con sus respectivos cortes para el sistema de ventilación, piso superior e inferior al igual que la parte frontal. En la Figura 3.41 se observa el diseño de cada una de las piezas, para finalmente ser pasadas por un corte láser.

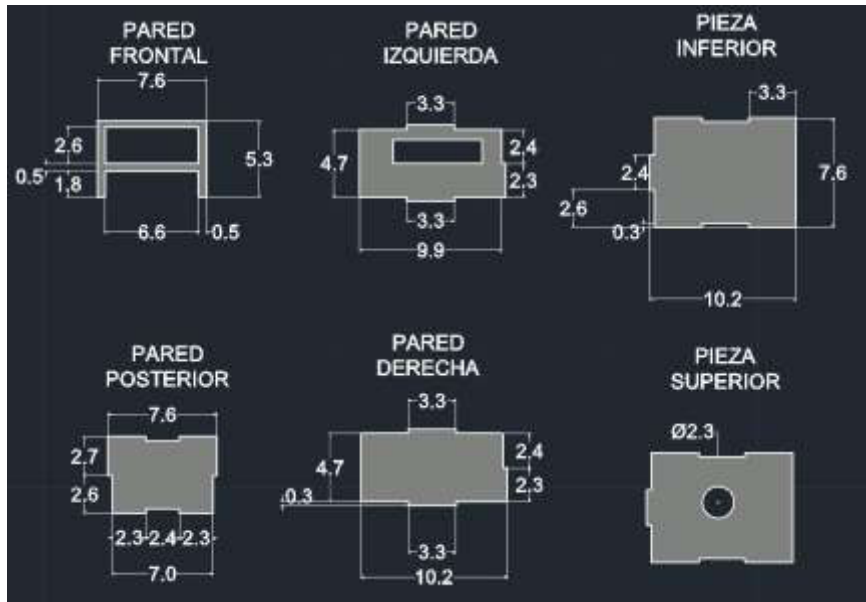


Figura 3.41 Diseño de las piezas de la cubierta

Con las piezas cortadas se procedió a unir cada una de las piezas, como se muestra en la Figura 3.41, cada pieza tiene un fleco o espacio, esto se lo realizó para armar la cubierta tal y como un rompecabezas donde las piezas embonan de tal modo que al finalizar dan la forma que se diseñó en el modelo, las piezas fueron unidas con un pegamento especial para evitar que se suelten y aumentar la durabilidad de la estructura, en la Figura 3.42 se puede observar el producto terminado digitalmente y en la Figura 3.43 como se ve físicamente.

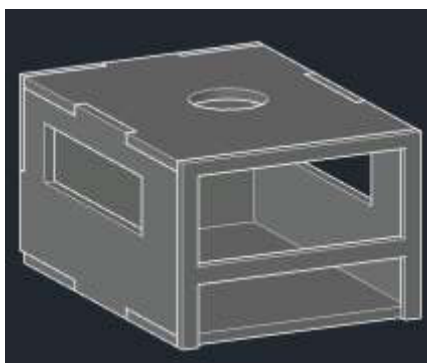


Figura 3.42 Diseño de la cubierta armada digitalmente



Figura 3.43 Diseño de la cubierta armada físicamente

Configuración del autodescubrimiento para la *Raspberry Pi*

Para la fácil navegación del usuario y que el dispositivo sea visible en la red se activó en base a comandos el autodescubrimiento, esto para detectar los recursos de una red local al conectarse a ella, haciendo accesible al dispositivo desde cualquier nodo con el uso de la dirección *Raspberrypi.local*, además de informar las peticiones o salidas de un servicio. Para activar esta característica, se instala en la terminal de la *Raspberry Pi* los siguientes comandos:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

Estos primeros comandos para verificar si existen o no actualizaciones o mejoras que se deban instalar en la placa.

```
sudo apt-get install avahi-daemon
```

Para hacer las pruebas correspondientes de que se está resolviendo nombres de dominio, se realiza un *ping* a cualquier dirección o sitio *web*, como se muestra en la **Figura 3.44** se realizó un *ping* a *google.com* concluyendo que si se está ejecutando la resolución de nombres.

```

pi@raspberrypi:~$ ping google.com
PING google.com[142.250.190.100] 56 data bytes
64 bytes from 142.250.190.100: icmp_seq=1 ttl=117 time=93.9 ms
64 bytes from 142.250.190.100: icmp_seq=2 ttl=117 time=93.3 ms
64 bytes from 142.250.190.100: icmp_seq=3 ttl=117 time=93.3 ms
64 bytes from 142.250.190.100: icmp_seq=4 ttl=117 time=93.9 ms
64 bytes from 142.250.190.100: icmp_seq=5 ttl=117 time=94.9 ms
64 bytes from 142.250.190.100: icmp_seq=6 ttl=117 time=93.4 ms
^C
--- google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 30ms
rtt min/avg/max/mdev = 93.000/93.711/94.578/6.460 ms
pi@raspberrypi:~$

```

Figura 3.44 Ping realizado a google.com

Costo del prototipo

El costo del prototipo ya incluida la fabricación de la cubierta se contempla en la Tabla 3.11 donde se adicionará los costos tantos de la placa de desarrollo, programación, fabricación y armado de la cubierta y costos de aditamentos integrados al prototipo.

Tabla 3.11 Tabla de costos del prototipo

Cantidad	Detalle	Precio Unitario	Precio Total
90	Programación en <i>Python</i> (por hora)	\$10,00	\$900,00
1	<i>Raspberry Pi Model 3 B+</i>	\$75,00	\$75,00
1	Fabricación de la cubierta en material <i>MDF</i>	\$20,00	\$20,00
1	Memoria <i>MicroSD Kingston 16</i> (GB)	\$6,00	\$6,00
1	Carcasa original para <i>Raspberry Pi</i>	\$7,50	\$7,50
1	Cable adaptador <i>USB</i> a serial <i>LPT1</i> macho	\$8,75	\$8,75
1	Cable adaptador <i>USB</i> a serial <i>LPT1</i> hembra	\$8,75	\$8,75
1	Adaptador <i>USB</i> hembra - hembra	\$4,00	\$4,00
1	Ventilador para <i>Raspberry Pi</i> (5 [V])	\$5,75	\$5,75
3	Disipadores de aluminio para <i>Raspberry Pi</i>	\$1,00	\$3,00
Costo Total			\$1.038,75

Manual de uso y mantenimiento

Para presentar el manual de uso y mantenimiento, se realizaron dos videos interactivos donde se muestra cada uno de los aspectos del dispositivo implementado. En la Figura 3.45 se muestra el enlace del video de uso y funcionamiento y en la Figura 3.46 el enlace del video de mantenimiento del dispositivo.



Figura 3.45 Enlace al video del manual de uso



Figura 3.46 Enlace al video del manual de mantenimiento

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- Con la adaptación del prototipo para permitir utilizar impresoras con puerto paralelo LPT1 se alarga el tiempo de vida de este tipo de impresoras que han caído en el desuso por la llegada de nuevas tecnologías que se agregan a este tipo de dispositivos, teniendo así la posibilidad de que puedan estar disponibles para ser empleadas como impresoras dentro de la red.
- Uno de los principales motivos de seleccionar una *Raspberry Pi* como placa de desarrollo fue el número de puertos, ya que este proyecto se centró en la posibilidad de brindar capacidades de red a impresoras que se encontraban en desuso debido a la falta de estas características, por tal motivo, la cantidad de puertos *USB* del *Raspberry* permite convertirlo en un dispositivo que permita la conexión de más de una impresora facilitando así la centralización de estas. Además, se logró una adaptación de otros puertos usados por impresoras más antiguas como el puerto paralelo LPT1 para lo cual se hizo uso de un adaptador.
- Se eligió la aplicación CUPS para administrar las impresoras ya que además de ser un sistema gratuito, cuenta con una interfaz intuitiva para el usuario, además de permite registrar hasta cuatro impresoras, siempre y cuando estas estén conectadas a la *Raspberry* para su debido funcionamiento, en el caso que se deseen conectar más impresoras se deberá borrar las impresoras que no se encuentran en uso y reemplazarlas por las que se quieren agregar.
- La selección del sistema operativo se basó en un criterio fundamental, ser fácil de instalar y optimizar los recursos con características que sean justas y necesarias para el desarrollo del proyecto. Al analizar todas sus versiones de *Raspbian*, la versión seleccionada fue la versión reducida debido a que cuenta con solo el sistema operativo ya que su instalación es fácil y sus características son suficientes para el desarrollo del proyecto, además de tomarse en cuenta un factor crítico como lo es el almacenamiento.
- Desarrollar el proyecto dentro de un ambiente virtual o un ambiente encapsulado permitió contar con varias funcionalidades como incluir dependencias y librerías sin la necesidad de completar algún proceso de instalación, por lo que complementa el objetivo de optimizar los recursos para la ejecución de los servicios que se proporcionarán en el dispositivo.

- El aprendizaje de lenguajes de programación como *Python* es importante para el desarrollo de nuevas aplicaciones y dispositivos como el desarrollado, debido a su sencillez y curva de aprendizaje menos exigente en comparación a otros lenguajes de programación lo que lo hacen necesario y complementario para el profesionalismo.
- Debido a que el dispositivo implementado puede ser empleado por largos periodos de tiempo fue necesario considerar la implementación de un sistema de refrigeración para el dispositivo, este sistema consta tanto de disipadores y un control en el ventilador, esto ya que a medida que se incrementa el tiempo en el que el dispositivo se encuentre alimentado generará más calor y por ende su rendimiento decaerá.
- La implementación de una carcasa para poder manipular el dispositivo *Raspberry* en conjunto con el adaptador LPT1 facilita su operación, además de que esta carcasa contiene el sistema de ventilación para evitar el calentamiento del dispositivo, evitando daños a corto y largo plazo.
- La creación de una página *web* facilitó el proceso de registro de nuevos usuarios, cuya información se almacena en una base de datos que trabaja en conjunto con el servicio de correo, además de enlazar a la página principal de *CUPS* para el registro de una impresora. Esto permitió facilitar el proceso de administración de los usuarios registrados en el sistema.
- El hecho de que el dispositivo adquiriera su *IP* de manera dinámica es beneficioso para que pueda funcionar en cualquier lugar, *Raspberry* tiene activo por defecto el servicio de DHCP, pero se tiene que conocer la dirección *IP* del dispositivo que está utilizando el servidor, para ello se sustituye esta dirección con un nombre de dominio aplicado desde DNS local para hacer más fácil el uso del dispositivo.
- El diseño original de la carcasa del dispositivo no cubría los requerimientos de refrigeración para trabajar durante periodos largos de tiempo, por lo que se modificó el diseño en mención para complementar el sistema de refrigeración aplicado, esta modificación consta de cortes realizados en la superficie de la cubierta que sirve como una ventila para expulsar el aire caliente generado por el dispositivo.

4.2 Recomendaciones

- Es importante limpiar tanto el dispositivo como la carcasa, este proceso se realizará cada mes, esto dependerá de la ubicación del dispositivo. Además de ello el dispositivo debe mantenerse en un área seca, evitando así posibles daños de humedad o acumulación de vapor de agua dentro de la placa.
- Para enviar mensajes de verificación, la lista de correos debe contener solo los correos a los cuales se quiere enviar esta notificación, es decir se recomienda borrar o eliminar los usuarios inactivos para no tener un colapso de la base de datos, esto para tener una mayor eficiencia en el envío de correos.
- Debido a la falta de experticia en el campo del desarrollo web, se recomienda mejorar la apariencia de la página HTML ya que solamente se usó una plantilla con colores básicos para la interfaz gráfica del registro de usuario, esta mejora puede incluir nuevas características como, por ejemplo: una nueva plantilla, imágenes, fondos, o diseños que hagan una interfaz mucho más llamativa.
- Para evitar limitaciones en referencia al envío de información, se recomienda conectar el dispositivo a la red de manera cableada ya que será más eficiente el proceso de impresión, no necesariamente es obligatorio usar un cable de red para el dispositivo ya que también puede ser conectado a la red de manera inalámbrica.
- Por motivos ajenos a nuestro campo de especialización y para que el dispositivo se vuelva más comercial, se recomienda modificar el programa realizado en este proyecto, para que en un futuro pueda devolver al usuario parámetros como el nombre del documento a imprimir y el número de hojas, además incluir si es posible el nivel de tinta de la impresora en uso.
- Es importante mantener solamente las impresoras activas dentro del servicio de impresión debido a que la aplicación CUPS se abastece de un número limitado de impresoras, ya que después de sobre pasar el límite CUPS tiende a no publicar las impresoras que estén fuera del registro.
- Si el dispositivo no va a ser usado en un periodo largo de tiempo, es recomendable apagarlo para que los recursos no se desperdicien y alargar la vida útil del dispositivo.

5. REFERENCIAS BIBLIOGRÁFICAS

- [1] N. Acero, «¿QUÉ ES LINUX?», *Guarruco*, pp. 1-8, 2015.
- [2] «Programo Ergo Sum,» 2019. [En línea]. Available: <https://www.programoergosum.com/cursos-online/Raspberry-pi/232-curso-de-introduccion-a-Raspberry-pi/instalar-raspbian>. [Último acceso: 4 Junio 2021].
- [3] I. H. P. Tavera, «Software Libre (Ubuntu),» Madrid, 2004.
- [4] «BalenaEtcher,» Balena, [En línea]. Available: <https://www.balena.io/etcher/>. [Último acceso: 15 Junio 2021].
- [5] «Rufus,» [En línea]. Available: <https://rufus.ie/es/>. [Último acceso: 15 Junio 2021].
- [6] E. Upton y G. Halfacree, *Raspberry Pi User Guide*, John Willey & Sons, 2014.
- [7] «Raspberrypi,» [En línea]. Available: <https://Raspberrypi.cl/que-es-Raspberry/>. [Último acceso: 4 Junio 2021].
- [8] J. Finlay, «PyGTK 2.0 Tutorial,» [En línea]. [Último acceso: 5 Junio 2021].
- [9] R. G. Duque, *Python para todos*, Madrid.
- [10] «Tecnología Informática,» 8 Marzo 2020. [En línea]. Available: <https://www.tecnologia-informatica.com/servidor-impresion/>. [Último acceso: 5 Junio 2021].
- [11] J. Rojas, «Intelecto Universal,» [En línea]. Available: <https://intelectouniversal.com/informatica/servidor-de-impresion/>. [Último acceso: 5 Junio 2021].
- [12] «Oracle,» Oracle Technology Network, 2012. [En línea]. Available: https://docs.oracle.com/cd/E37929_01/html/E36601/cups-intro.html. [Último acceso: 5 Junio 2021].
- [13] R. Fielding y R. J. , *Hypertext Transfer Protocol (HTTP/1.1): Authentication*, 2014.
- [14] T. P. Group, «IPP Version 2.0, 2.1, and 2.2,» Nueva Jersey, 2011.

- [15] «CUPS.org,» [En línea]. Available: <http://www.cups.org/doc/spec-design.html>. [Último acceso: 5 Junio 2021].
- [16] «CISCO,» [En línea]. Available: https://www.cisco.com/c/es_mx/tech/wireless-2f-mobility/wireless-lan-wlan/index.html. [Último acceso: 5 Junio 2021].
- [17] L. G. S, «Teoría de Redes de Computadoras,» [En línea]. Available: https://www.oas.org/juridico/spanish/cyber/cyb29_computer_int_sp.pdf.. [Último acceso: 5 Junio 2021].
- [18] F. L. Ortiz, «El estándar IEEE 802.11 Wireless LAN».
- [19] «Arduino Store,» [En línea]. Available: <https://store.arduino.cc/usa/mega-2560-r3>. [Último acceso: 1 Julio 2021].
- [20] «Raspberrypi,» [En línea]. Available: <https://www.Raspberrypi.org/products/Raspberry-pi-3-model-b/>. [Último acceso: 1 Julio 2021].
- [21] «Raspberrypi,» [En línea]. Available: <https://www.Raspberrypi.org/products/Raspberry-pi-3-model-b-plus/>. [Último acceso: 1 Julio 2021].
- [22] «Red Hat,» [En línea]. Available: <https://www.redhat.com/es/topics/linux>. [Último acceso: 1 Julio 2021].
- [23] «Raspberrypi,» [En línea]. Available: <https://www.Raspberrypi.org/software/>. [Último acceso: 1 Julio 2021].
- [24] «Java,» [En línea]. Available: <https://www.java.com/es/>. [Último acceso: 1 Julio 2021].
- [25] «Python,» [En línea]. Available: <https://www.python.org>. [Último acceso: 1 Julio 2021].
- [26] L. R. Alfonso, *Implementación de un servidor Samba con autenticación LDAP como alternativa libre a los servidores de dominio Windows*, Sevilla, 2010.

[28] «Arcade Xpress blog,» 22 Febrero 2020. [En línea]. Available: <https://www.arcadexpress.com/blog/como-conectar-un-mando-arcade-a-la-Raspberry-pi/>. [Último acceso: 15 Julio 2021].

[29] «Amazon,» [En línea]. Available: <https://www.amazon.com/-/es/adaptador-impresora-puerto-paralelo-velocidad/dp/B08KWQY2RJ>. [Último acceso: 15 Julio 2021].

ANEXOS

ANEXO 1: CERTIFICADO DE FUNCIONAMIENTO



ESCUELA POLITECNICA NACIONAL

Campus Politécnico "J. Rubén Orellana R

Quito, -- de ----- de 2021

CERTIFICADO DE FUNCIONAMIENTO DE PROYECTO DE TITULACIÓN

Yo, Leandro Antonio Pazmiño Ortiz, docente a tiempo completo de la Escuela Politécnica Nacional y como director de este trabajo de titulación, certifico que he constatado el correcto funcionamiento del sistema de impresoras para agregar funcionalidades de impresión en red, el cual fue implementado por los estudiantes Raúl Iván Cantuña Oña y Rommel Josué Vega Pazmiño.

El proyecto cumple con los requerimientos de diseño y parámetros necesarios para que los usuarios de la ESFOT puedan usar las instalaciones con seguridad para los equipos y las personas.

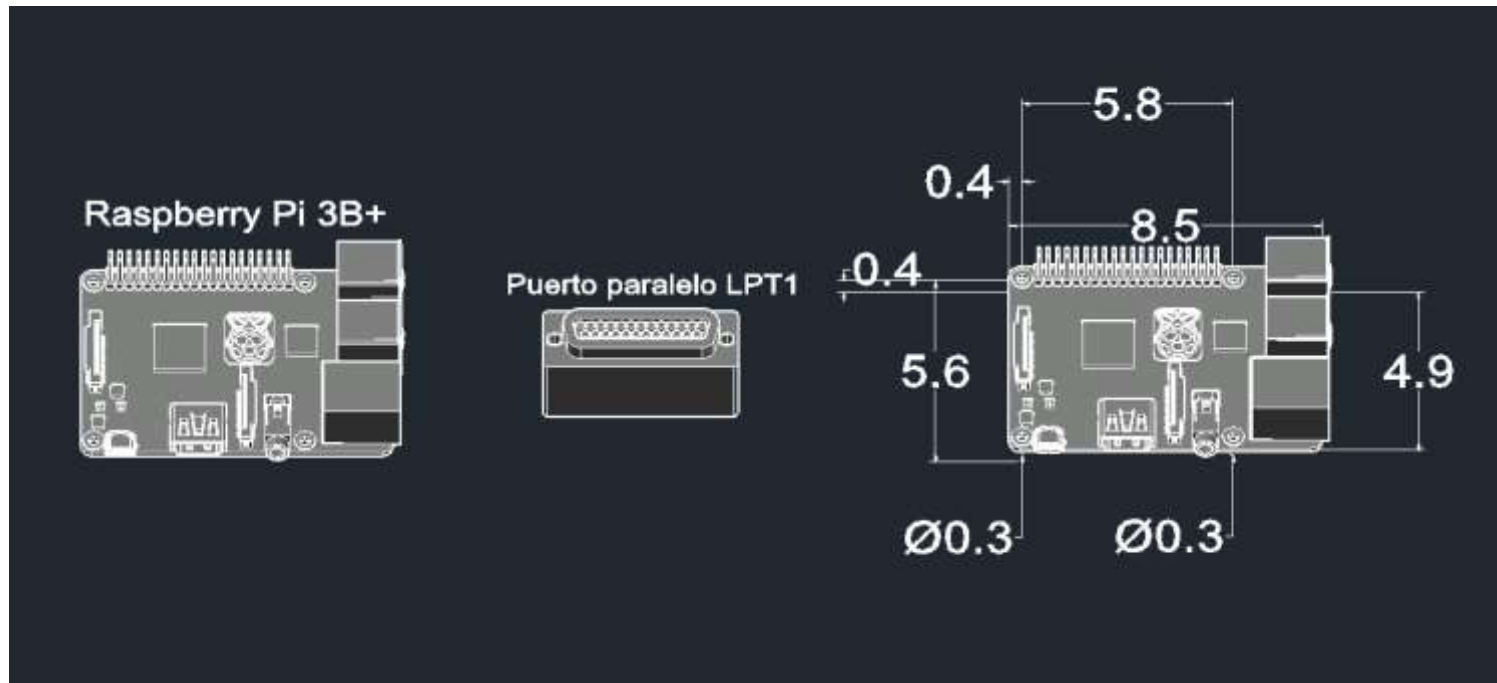
DIRECTOR

Ing. Leandro Antonio Pazmiño Ortiz., Msc.

Ladrón de Guevara E11-253, Escuela de Formación de Tecnólogos, Oficina 28. EXT: 2729
email: pablo.proano@epn.edu.ec

Quito-Ecuador

ANEXO 2: MODELO 3D DE LA *RASPBERRY PI* Y DEL ADAPTADOR LPT1



ANEXO 3: MODELO 3D DE LA ETAPA 1 DE LA CUBIERTA



ANEXO 4: MODELO 3D DE LA ETAPA 2 DEL DISEÑO DE LA CUBIERTA

