

PROTOTIPO SBC BASADO EN IoT PARA AUTOMATIZACION Y LA GESTION
DE CAMAS O CAMILLAS HOSPITALARIAS



CESAR DAVID RAMIREZ GUERREO
EDWIN ORLANDO PARDO LASSO

INSTITUCION UNIVERSITARIA COLEGIO MAYOR DEL CAUCA
FACULTAD DE INGENERIA
INGENIERIA INFORMATICA
POPAYÁN – CAUCA

2019

PROTOTIPO SBC BASADO EN IoT PARA AUTOMATIZACIÓN Y LA GESTIÓN
DE CAMAS O CAMILLAS HOSPITALARIAS



CESAR DAVID RAMÍREZ GUERREO
EDWIN ORLANDO PARDO LASSO

TRABAJO DE GRADO PRESENTADO COMO REQUISITO PARCIAL PARA
OPTAR AL TÍTULO DE INGENIERÍA INFORMÁTICA

DIRECTOR

Mg.(c) STIVENS ANTONIO DIONIZIO SOLARTE

INSTITUCIÓN UNIVERSITARIA COLEGIO MAYOR DEL CAUCA
FACULTAD DE INGENIERÍA
INGENIERÍA INFORMÁTICA
POPAYÁN – CAUCA

2019

TABLA DE CONTENIDO

CAPITULO 1. PLANTEAMIENTO DEL PROYECTO	7
1.1 INTRODUCCIÓN	7
1.2 PROBLEMA	8
1.3 JUSTIFICACIÓN	10
1.4 OBJETIVOS.....	11
1.4.1 OBJETIVOS GENERAL	11
1.4.2 OBJETIVOS ESPECÍFICOS.....	11
CAPITULO 2. METODOLOGÍA	11
2.1 GENERACIÓN DE BASE CONCEPTUAL	11
2.2 DISEÑO DEL SISTEMA.....	18
2.2.1 DISEÑO PRELIMINAR	30
2.2.2 DISEÑO CONCEPTUAL	37
2.2.3 DISEÑO DETALLADO.....	45
CAPITULO 3. IMPLEMENTACIÓN DEL SISTEMA.....	51
CAPITULO 4. VERIFICACIÓN DEL SISTEMA	78
4.1 PRUEBAS DE FUNCIONAMIENTO DEL PROTOTIPO	78
4.2 PRUEBAS DE TEMPERATURA	81
4.3 PRUEBAS DE CONSUMO ENERGETICO	84
CAPITULO 5. MANTENIMIENTO DEL SISTEMA.....	85
CAPITULO 6. CONCLUSIONES	89
REFERENCIAS	92

INDICE DE ILUSTRACIONES

Ilustración 1. ESP 32 características,.....	31
Ilustración 2. FSR 406,.....	33
Ilustración 3. Características FSR406,	34
Ilustración 4 Tarjeta RFID-RC522,	35

Ilustración 5. RFID - RC522,	36
Ilustración 6. Diseño Arquitectura N° 1 del sistema con HTTP POST, servidor LAMP y MySQL	40
Ilustración 7. Diseño Arquitectura N°2 del sistema con MQTT + Node Red	42
Ilustración 8. Arquitectura MQTT.....	43
Ilustración 9. Filtro de Mensajes MQTT,.....	44
Ilustración 10. Diagrama de Flujo del prototipo,	46
Ilustración 11. Conexión electrónica ESP32 y FSR,	47
Ilustración 12. Conexión electrónica ESP32 y RFID - RC522.....	48
Ilustración 13. Esquema preliminar del prototipo	49
Ilustración 14. Esquema final del prototipo, Fuente: Propia.....	49
Ilustración 15. Conectividad del sistema IoT, Prototipo y Servidor LAMP	50
Ilustración 16. Conectividad completa del Sistema IoT	51
Ilustración 17. Primera fase de implementación, ESP32 + RFID RC522 + LED'S de notificación, Fuente: Propia	53
Ilustración 18. Conexión de los módulos ESP32 + RFID RC522 + LED'S de notificación	54
Ilustración 19. Conexión del FSR 406 al prototipo en desarrollo	54
Ilustración 20. Conexión del sensor FSR406 al ESP32	55
Ilustración 21. Implementación de switch de encendido y adaptador de entrada de energía del prototipo	55
Ilustración 22. Prototipo preliminar, Dispositivo SBC + RaspBerry Pi B+ servidor LAMP	56
Ilustración 23. Prototipo final	56
Ilustración 24. Interfaz, instalador SO,	58
Ilustración 25. Actualización de los paquetes de instalación para el sistema del servidor,	59
Ilustración 26. Actualización del gestor de paquetes., Fuente: Propia	60
Ilustración 27. Instalación del servidor Apache en la RaspBerry,	61
Ilustración 28. Finalización de la instalación del servidor Apache en la RaspBerry,	62
Ilustración 29. Modificación del archivo apache2.conf para agregar una capa de seguridad al servidor,	63

Ilustración 30. Instalación del lenguaje PhP en el servidor,	64
Ilustración 31. Instalación del servidor y el cliente de MariaDB,	65
Ilustración 32. Instalación de la base de datos MariaDB en el servidor,	66
Ilustración 33. comandos para ingresar al gestor de la base de datos MariaDB, Fuente: Propia	66
Ilustración 34. Pantalla principal para la creación de la base de datos en MariaDB, Fuente: Propia	67
Ilustración 35. Instalación de PhpMyAdmin en el servidor, Fuente: Propia.....	68
Ilustración 36. Configuración del PhpMyAdmin, Relación entre MariaDB y PhpMyAdmin, Fuente: Propia	69
Ilustración 37. Configuración de contraseña para PhpMyAdmin, Fuente_ Propia	70
Ilustración 38. Comando para ingresar al fichero dhcpd.conf para la configuración del canal de comunicación, Fuente: Propia.....	71
Ilustración 39. Configuración del canal de comunicación mediante una IP estática, Fuente: Propia	72
Ilustración 40. Librerías implementadas en el prototipo IoT, Fuente: Propia	73
Ilustración 41. Diagrama Entidad Relación de la base de datos del sistema IoT, Fuente: Propia	74
Ilustración 42. Pantalla Principal del framework de NodeRed,	75
Ilustración 43. Diagrama de flujos del proceso del sistema IoT, Fuente: Propia ...	76
Ilustración 44. Diagrama de flujos para la visualización de los datos registrados, Fuente: Propia	77
Ilustración 45. interfaz gráfica del sistema IoT con los datos de la camillas o camas, Fuente: Propia	77
Ilustración 46. Estadística de ocupación de las camas o camillas	78
Ilustración 47. Código fuente. Reinicio automático del prototipo por falla de la conexión a la red,.....	80
Ilustración 48. Posicionamiento del sensor de presión FSR406,	86
Ilustración 49. Posicionamiento del prototipo en la camilla,	86
Ilustración 50. Reconexión del sensor FSR406 al prototipo	87
Ilustración 51. Adaptador de energía del prototipo	88
Ilustración 52. Switch de encendido y apagado del prototipo	88

INDICE DE TABLAS

Tabla 1. Entidades hospitalarias públicas y número de camillas	12
Tabla 2. Cantidad de habitantes en la ciudad de Popayán, datos tomados: de la página web de la Departamento Administrativo Nacional de Estadística (DANE) 13	13
Tabla 3. Fórmulas para el cálculo de camas hospitalarias por mil habitantes	13
Tabla 4. Camas necesarias para los años 2015-2016.....	14
Tabla 5. Camas o camillas necesarias para el año 2019. Fuente: DANE	14
Tabla 6. Características placas SBC.....	20
Tabla 7. Normalización de datos de las Placas SBC's	23
Tabla 8.Resultado de análisis de Placas SBC.....	24
Tabla 9. Características Placas Raspberry, Fuente Propia	26
Tabla 10. Normalización de datos RaspBerry Boards, Fuente: Propia	28
Tabla 11. Resultado de análisis de la placa Raspberry	28
Tabla 12. Descripción de los componentes del Módulo ESP 32,.....	32
Tabla 13. Especificaciones Técnicas del FSR 406,.....	33
Tabla 14. Requisitos Funcionales del prototipo IoT,	39
Tabla 15. Componentes Hardware del dispositivo SBC	52
Tabla 16. Componentes Software del sistema IoT	57

INDICE DE GRAFICAS

Gráfica 1. Estados de funcionamiento del prototipo,	79
Gráfica 2. Porcentaje de Estados del prototipo en prueba,	81
Gráfica 3. Temperatura del Prototipo,	82
Gráfica 4. Temperatura del Servidor,	82
Gráfica 5. Temperatura y estados del prototipo.....	83

CAPITULO 1. PLANTEAMIENTO DEL PROYECTO

1.1 INTRODUCCIÓN

En la actualidad desde que se creó el internet, un protocolo estándar para el transporte de datos entre distintos ordenadores, las personas alrededor del mundo se han podido comunicar de manera mucho más eficiente y rápida, con el crecimiento del internet y la tecnología se han logrado grandes avances en diferentes disciplinas tales como la educación, salud, economía, etc. Y a través del crecimiento que ha tenido estos dos elementos que se han hecho parte fundamental de la vida cotidiana del ser humano, facilitando todo tipo de herramientas que ayudan a solucionar parte de las cuestiones cotidianas de las personas. Es gracias a la combinación de estos factores que se llegó al término denominado internet de las cosas o IoT

El Internet de las Cosas es una idea que se comenzó a desarrollar desde los años 90s a partir del ingreso del concepto de computación Ubicua que introdujo Mark Weiser director científico del Xerox Palo Alto Research Center (Parc), este concepto viene dado por la integración de los diferentes elementos cotidianos, por un medio tecnológico y en conjunto con la gestión de redes desde un simple ordenador o PC, pero el término en sí “Internet de las Cosas” fue creado por Kevin Ashton, dado a conocer en una presentación en la compañía Procter & Gamble, en 1998: “Adicionar identificación por radio frecuencia y otros sensores a objetos cotidianos creará una Internet de las Cosas, y sentara las bases de una nueva era de percepción de la máquina.”. [1]

La funcionalidad del internet de las cosas se ve involucrada en varias de las disciplinas de nuestra sociedad como ya se había mencionado con anterioridad y una de ellas es la salud la cual no está fuera del alcance de la IoT, implementándose en cada una de estas instituciones o planteles ya sea en el cuidado general de los pacientes o facilitando las tareas de cada uno de los involucrados dentro de la estructura del hospital, así creando un entorno de trabajo mucho más eficiente y eficaz para la prestación de los diferentes servicios que se le dan a la comunidad.

Uno de los procesos que se presenta dentro de los hospitales es la administración de los recursos que este tiene y que hace uso durante todo el año para que los procesos sean óptimos dentro de la misma organización y de la prestación de servicio. En el caso de Colombia, el sistema de salud atraviesa una crisis por la incontenible demanda de los servicios de salud, que supera la capacidad de los hospitales para atender pacientes, solamente en la ciudad de Popayán, de catorce instituciones hospitalarias habilitadas, disponemos cuatro de carácter público y solo tres cuentan con camas o camillas a disposición para la prestación de servicio médico. Según la REPS el número de insumo que está en funcionamiento es de 30, el cual representa la mayor cantidad de recursos en una sola instalación médica y siendo cero (0) la cantidad mínima que se encuentra en la ciudad de Popayán [2], analizando esta situación se denota la desproporción entre oferta y demanda de servicios de salud, que conlleva la saturación y colapso en la entidad.

La eficiencia es una condición importante debido a que cada uno de estos elementos son limitados en los hospitales y la cama como fuente central es de gran importancia para la atención y el cuidado. Por eso, entre las medidas globales de la eficiencia hospitalaria están aquellas que se relacionan con el uso que se le dé a este recurso. [3]. Es por lo que a partir de este suceso se pretende realizar un prototipo para la gestión automática de las camas o camillas de hospital y mejorar la eficiencia en la gestión que se le da a este escaso recurso en estas instituciones.

1.2 PROBLEMA

Las instituciones hospitalarias que prestan el servicio de salud y el cuidado hacia el paciente, enfrentan diversas situaciones diarias por posibles motivos o factores que generan complicaciones a la hora de brindar atención médica a la comunidad, como la saturación de cada uno de los servicios ofertados por la institución, la mala organización, la falta de recursos económicos, recursos estructurales y de suministros, son uno de los muchos factores que llegan a impedir el buen funcionamiento administrativo de un hospital o brindar un servicio de atención de calidad al ciudadano Colombiano.

Frente a diversos casos que enfrenta las instituciones médicas, uno de los que más resalta es la limitación que hay de los recursos o herramientas de apoyo y la mala gestión que se le dan a estos, para ser más precisos: la gestión del recurso de camas o camillas, puesto que por la falta de este instrumento fundamental de trabajo se ha dejado de atender pacientes, en donde su estado vital es fortuito o deplorable, por otro lado la saturación del servicio de salud por la fuerte demanda que hay de este mismo recurso. Según estudios del diario el tiempo “El departamento de Cauca, con una población de 1'415.933 personas, cuenta solo con 0,9 camas, por cada 1000 habitantes. El 58,5% del total de camas hospitalarias corresponde a entidades públicas. Por esto la gestión administrativa de las instituciones hospitalarias han decaído en la ineficiencia de la prestación de salud dando perdidas tanto humanas como económicas en el país”. [7]

Teniendo en cuenta la consigna de la constitución política de Colombia y la situación que se presenta en cada uno de los hospitales, los cuales confrontan la problemática que genera los recursos limitados brindados por el gobierno y que por ello se identifica la necesidad de implementar nuevas estrategias que contribuyan a la agilidad de la gestión, “El número de camas hospitalarias necesarias para la correcta atención de la población no puede definirse universalmente ya que no sólo depende de la eficiencia en su gestión sino también del desarrollo de otras modalidades de atención y de la incorporación de nuevas tecnologías que influyen en su tasa de utilización.” [8]. Teniendo en cuenta lo anterior se identifica la necesidad de mejorar la atención médica, sobre todo en la parte tecnológica la cual contribuye en la mejora de tareas cotidianas del personal hospitalario y puedan generar una mayor productividad en cuanto a la gestión y en el buen aprovechamiento de los recursos, ya que hasta la fecha se ha llevado un control de manera manual y no sistémica.

1.3 JUSTIFICACIÓN

La finalidad de este proyecto en la investigación y potencial desarrollo de un prototipo SBC basado en IoT, a consecuencia de los inconvenientes en los procesos de gestión en los recursos que tiene una demanda alta frente a la prestación de servicios dentro de los hospitales, que en este caso se está hablando de las camas o camillas dentro de las instituciones médicas, puesto que es uno de los elementos físicos más importantes con el que los hospitales brindan un servicio y confort obligatorio, según la ley estatutaria N° 1751 del 16 de febrero del 2015, que regula el derecho fundamental a la salud y se dictan a otras disposiciones. teniendo como referencia el ARTÍCULO 6°. ELEMENTOS Y PRINCIPIOS DEL DERECHO FUNDAMENTAL A LA SALUD. Determina que, El derecho fundamental a la salud incluye los siguientes elementos esenciales e interrelacionados: disponibilidad, aceptabilidad, accesibilidad calidad e idoneidad profesional y principios entre ellos la Eficiencia: El sistema de salud debe procurar por la mejor utilización social y económica de los recursos, servicios y tecnologías disponibles para garantizar el derecho a la salud de toda la población. [1]. Por otra parte, el ministerio de la salud también propone en la ley estatutaria N° 1751 en el ARTÍCULO 22. POLÍTICA DE INNOVACIÓN, CIENCIA Y TECNOLOGÍA EN SALUD: El Estado deberá establecer una política de Innovación, Ciencia y Tecnológica en Salud, orientada a la investigación y generación de nuevos conocimientos en salud, la adquisición y producción de las tecnologías, equipos y herramientas necesarias para prestar un buen servicio de salud que permita el mejoramiento de la calidad de vida de la población. [1]. Tal y como podemos ver la implementación en Bogotá, Colombia, de un sistema con el enfoque del internet de las cosas “La industria de la salud es quizás uno de los sectores más adecuados para integrar las tecnologías de IoT. Este documento presenta un enfoque de IoT en curso para implementar una Red de Sensores Inalámbricos (WSN) aplicada al monitoreo ambiental de la temperatura y la humedad relativa dentro de hospitales o laboratorios clínicos.”. [2] y por el hecho de que en Colombia se tiende a la poca inversión en planes o medios para contrarrestar esta situación en el área de la salud, se decide frente a esta situación donde la gestión adecuada del recurso de las camas o camillas es lo esencial para

una buena atención a la comunidad, solamente en la ciudad de Popayán se presenta este proyecto como un posible acercamiento al mejoramiento de la gestión de las camas o camillas dentro de los centros hospitalarios, el cual mediante el uso de las tecnologías y el concepto del internet de las cosas "con el objetivo de demostrar todas las ventajas que proporciona la IoT y lo inevitable de la implementación en el campo de la atención médica y la salud para crear sistemas inteligentes dirigida a la atención de pacientes". [3] Se proporcionará una herramienta prototipo que permita al personal de las instituciones de salud, llevar una administración y control de un recurso tan valioso para ellos y para toda la sociedad.

1.4 OBJETIVOS

1.4.1 OBJETIVOS GENERAL

Diseño y construcción de un prototipo SBC basado en IoT para la gestión automatizada de camas o camillas hospitalarias.

1.4.2 OBJETIVOS ESPECÍFICOS

1. Diseñar un sistema hardware-software para la gestión de camas o camillas hospitalarias.
2. Implementar un prototipo del sistema SBC para la gestión de camas o camillas hospitalarias.
3. Verificar el funcionamiento del prototipo SBC para la a gestión de camas o camillas hospitalarias.

CAPITULO 2. METODOLOGÍA

2.1 GENERACIÓN DE BASE CONCEPTUAL

En los hospitales públicos los recursos tales como las camas o camillas no abundan en cantidades por la falta de inversión por parte del gobierno colombiano y departamental, generando así un gran déficit de atención a la comunidad payanesa. El número de camas hospitalarias necesarias para la correcta atención de la población no puede definirse universalmente ya que no sólo depende de la

eficiencia en su gestión sino también del desarrollo de otras modalidades de atención y de la incorporación de nuevas tecnologías que influyen en su tasa de utilización. [4]. Según la REPS (Registro Especial de Prestadores de Servicios de Salud) la cual cuenta con una base de datos con información reportada por las Entidades Departamentales y Distritales de Salud en ella se puede evidenciar como vemos a continuación en la siguiente tabla el déficit que se encuentra en las diferentes instituciones hospitalarias públicas de Popayán.

depa_nombre	muni_nombre	sede_nombre	naju_nombre	nivel	caracter	grupo_capacidad	coca_nombre	cantidad
Cauca	POPAYÁN	HOSPITAL MARIA OCCIDENTE	Pública	1	DEPARTAMENTAL	CAMAS	Pediátrica	4
Cauca	POPAYÁN	HOSPITAL MARIA OCCIDENTE	Pública	1	DEPARTAMENTAL	CAMAS	Adultos	9
Cauca	POPAYÁN	HOSPITAL MARIA OCCIDENTE	Pública	1	DEPARTAMENTAL	CAMAS	Obstetricia	4
TOTAL CAMA								17
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Pediátrica	27
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Adultos	60
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Obstetricia	33
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Cuidado Intermedio Neonatal	12
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Cuidado Intensivo Neonatal	10
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Cuidado Intermedio Pediátrico	3
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Cuidado Intensivo Pediátrico	6
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Cuidado Intermedio Adulto	2
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Cuidado Intensivo Adulto	3
Cauca	POPAYÁN	HOSPITAL SUSANA LOPEZ DE VALENCIA	Pública	2	DEPARTAMENTAL	CAMAS	Cuidado básico neonatal	4
TOTAL CAMA								160
Cauca	POPAYÁN	HOSPITAL TORIBIO MAYA	Pública	1	DEPARTAMENTAL	CAMAS	Pediátrica	0
Cauca	POPAYÁN	HOSPITAL TORIBIO MAYA	Pública	1	DEPARTAMENTAL	CAMAS	Adultos	0
Cauca	POPAYÁN	HOSPITAL TORIBIO MAYA	Pública	1	DEPARTAMENTAL	CAMAS	Obstetricia	0
TOTAL CAMA								0
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Pediátrica	16
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Adultos	137
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Obstetricia	27
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Cuidado Intermedio Neonatal	15
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Cuidado Intensivo Neonatal	15
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Cuidado Intermedio Pediátrico	2
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Cuidado Intensivo Pediátrico	6
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Cuidado Intermedio Adulto	24
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Cuidado Intensivo Adulto	29
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Unidad de Quemados Adulto	6
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Psiquiatria	9
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Unidad de Quemados Pediátrico	7
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Cuidado Agudo Mental	5
Cauca	POPAYÁN	IVERSITARIO SAN JOSE DE POPAYAN EMPRESA SOCIA	Pública	3	MUNICIPAL	CAMAS	Cuidado básico neonatal	5
TOTAL CAMA								303

Tabla 1¹. Entidades hospitalarias públicas y número de camillas

Como se puede ver en la tabla 1. Se encuentran cada uno de los hospitales del departamento del Cauca de carácter público, donde su ubicación de funcionamiento y de atención al ciudadano se encuentra en la capital Popayán, en cada una de las instituciones cuentan con una cantidad de camas redistribuidas en sus diferentes áreas de atención, el cual el hospital universitario san José de nivel 3 sería el que tiene la mayor cantidad de este recurso con un total de 303 camas, y siendo el de mayor capacidad para la atención medica publica dentro de la ciudad, en contra

¹ Tomada y adaptada del: registro Especial de Prestadores de Servicios de Salud (REPS), según información reportada por las Entidades Departamentales y Distritales de Salud

parte la institución que cuenta con la menor cantidad del recurso de camas o camillas es el hospital Toribio Maya siendo de cero (0) el total con el que se manejan.

Según la OMS (organización mundial de la salud) la gestión de un recurso tan vital como lo es las camas y camillas dentro de las instituciones hospitalarias puede ayudar a salvar más de una vida. Teniendo en cuenta estas palabras se pueden calcular el número de camas o camillas para poder cubrir la necesidad de atención pública a la población, este cálculo toma como referencia la cantidad de camas por cada Mil (1000) habitantes de una población de una región determinada, aplicando la operación correspondiente en la cual toma otros factores como los egresos del hospital, la estancia media de un paciente en el hospital y el índice óptimo de ocupación que corresponde al 85% a una institución de salud pública, dará como resultado que tantas camas habrá por habitante y si esa cantidad es suficiente para la atención de la población, donde la OMS dicta que la cantidad óptima serían 2 o más camas o camillas por habitante.

Para la población de la ciudad de Popayán la situación es regular, viendo el resultado obtenido al aplicar la fórmula y cómo podemos ver en la siguiente imagen:

CIUDAD	POBLACIÓN	AÑO
Popayán - Cauca	280.107	2015-2016
Popayán - Cauca	318.059	2019

Tabla 2. Cantidad de habitantes en la ciudad de Popayán, datos tomados: de la página web de la Departamento Administrativo Nacional de Estadística (DANE)

EGRESOS :	$\text{población} \times \text{frecuencia} / 1000$
ESTANCIA MEDIA :	7 días
CAMAS NECESARIAS:	$\text{Egresos} \times \text{estancia media} / 365 \times \text{índice de ocupación óptimo (85\%)}$

Tabla 3². Fórmulas para el cálculo de camas hospitalarias por mil habitantes

EGRESOS :	$280.107 \times 100 / 1000 =$	28011
------------------	-------------------------------	-------

² Datos tomados: E. Dr. Quesada, *CANTIDAD, USO Y GESTION DE LAS CAMAS HOSPITALARIAS. TENDENCIAS EN EL MUNDO Y SITUACIÓN EN MENDOZA*, Mendoza, Argentina, 2011.

CAMAS NECESARIAS:	$28011 \times 7 / 365 \times 0.85 =$	457
CAMAS POR CADA MIL HABITANTES EN POPAYÁN :	1,6	

Tabla 4. Camas necesarias para los años 2015-2016

Para los años 2015-2016 la población de Popayán solamente contaba con una cantidad de 1,6 camas por cada mil habitantes, un promedio más que por debajo de lo que recomienda la OMS como ya se había mencionado.

Para el año de 2019, la situación ha mejorado, con una población de 318.059 habitantes según la DANE, se calculó la necesidad del recurso de las camas o camillas para satisfacer la demanda de prestación del servicio hospitalario, como se observa a continuación

EGRESOS :	$318.059 \times 100 / 1000$	31805,90
CAMAS NECESARIAS:	$31805,9 \times 7 / 365 \times 0.85$	718
CAMAS POR CADA MIL HABITANTES EN POPAYÁN :	2,3	

Tabla 5. Camas o camillas necesarias para el año 2019. Fuente: DANE

Promediando una cantidad de egresos a las instituciones hospitalarias de los habitantes de la ciudad de Popayán de un valor de 31805.90, es necesario tener una cantidad que pueda cubrir la atención médica para cada uno de ellos, es por esto por lo que el número del recurso debe ser de 718 camas o camillas, acorde con la estancia media dentro de un hospital por paciente que es de 7 días sobre el transcurso de un año por el índice de ocupación del 85%. Obteniendo como resultado que por habitante en la ciudad de Popayán debe haber 2,3 de camas o camillas para satisfacer la necesidad de atención hospitalaria.

Teniendo un promedio de 2,3 camas por cada mil habitantes para la atención médica. Frente a esta situación la gestión del recurso en cuestión tiene que ser optima por parte del personal hospitalario para generar una movilidad y tener conocimiento de su ubicación de las camas o camillas dentro de las instalaciones del hospital y así brindar una atención al ciudadano de calidad.

La Tecnología, ha ingresado a la vida cotidiana del hombre para su desarrollo tanto personal como laboral, generando grandes cambios a lo largo del tiempo y una de esas tecnologías que se han involucrado son las tecnologías IOT que por sus siglas en ingles traducen INTERNET DE LAS COSAS. El cual trata de la conexión de dispositivos y objetos de la vida cotidiana a través de Internet. IoT permite integrar sensores y dispositivos con objetos que quedan conectados a Internet a través de redes fijas e inalámbricas [5], es a través de este concepto que al integrarlo en las diferentes tareas de las personas ayudan a simplificar el grado de complejidad a la hora de dar una solución a un determinado problema. Tal es el caso de la domótica una de las aplicaciones más reconocidas y por la que la IOT tuvo un reconocimiento dentro del público tanto educativo como general, esta rama del internet de las cosas trata de reducir las tareas dentro del entorno doméstico ya sea en la parte de seguridad, gestión de energía, comunicación y como en la de entretenimiento. Es así la aplicabilidad de esta tecnología en estos campos que se han propuesto diferentes soluciones como la de un sistema IoT de monitoreo y alarma para personas mayores, tomando ventaja de las tecnologías actuales como sistemas embebidos con sensores de movimiento, sensores biométricos, conexión inalámbrica, geoposicionamiento, etc. Haciendo uso de estos dispositivos y de la computación en la nube, se busca brindar a las personas adultas mayores y de su entorno mayor comodidad, autonomía, independencia, reducción de costos en los cuidados, mejores controles de la salud y mayor celeridad en la respuesta ante emergencias. Se busca crear una solución que monitoree la salud del usuario y la reporte a los cuidadores, familiares y médicos a través de internet durante las 24hs los 7 días de la semana. Además, emita alertas en los casos que el usuario requiera atención inmediata. [6]. Esta propuesta dirigida a la automatización de una casa para las personas mayores edad, queda catalogada dentro de una de las aplicaciones de la domótica como lo es la seguridad. Otro ejemplo en el que se evidencia la implementación de los sistemas domóticos en que da una solución al consumo energético de un campus, es el desarrollado por el autor Sócrates Martínez Giménez, en el que se realizará el diseño de una red de acceso óptica IoT para la gestión eficiente de la energía, situada en los edificios correspondientes a

la Ciudad Politécnica de la Innovación (CPI) del Campus de Vera en la Universidad Politécnica de Valencia. Donde se identificarán los componentes comerciales necesarios para esta aplicación concreta, cerciorándose de que permitirán proponer soluciones que den lugar a una gestión eficiente de la energía y, por tanto, supongan un ahorro considerable. Este ahorro se cuantificará, tras el correcto dimensionado tanto de los edificios como de dichos componentes, mediante un análisis energético en el que se comparará el consumo actual de los edificios de la CPI con el ahorro que conllevaría el utilizar las diferentes soluciones inteligentes propuestas. [7]. Siguiendo con las diferentes aplicaciones que tiene la IoT en el entorno doméstico y para ser más precisos en la reducción del consumo del servicio eléctrico se encuentra la siguiente propuesta de proyecto el cual describe el diseño e implementación de un prototipo de control inteligente del sistema de iluminación LED con suministro eléctrico en corriente continua de baja tensión, LVDC, en una plataforma IoT. El objetivo de la implementación de un control inteligente del sistema de iluminación artificial es garantizar la reducción del consumo eléctrico debido al máximo aprovechamiento de la luz natural y al apagado del sistema de iluminación al detectar la ausencia de ocupantes en el espacio interior. El dispositivo de control de las luminarias LED atenúa el nivel de luminosidad, maximizando el aprovechamiento de la luz natural exterior y garantizando los mínimos niveles de iluminación, establecidos en la normativa de iluminación en espacios interiores UNE-EN 12464-1. La monitorización, gestión y visualización de la información del sistema de Iluminación LED se realiza a través de la plataforma IoT. El acceso y gestión de la información se puede realizar desde cualquier dispositivo electrónico que cuente con acceso a internet. [8]

Permitiendo a las personas cambiar su estilo y mejorar la calidad de vida, ya que este sistema IOT recolecta información pertinente que al recopilarlos se podrían analizar y transformar esos datos en información importante y conocimiento.

El Internet de las cosas aplicado en el área de la salud permitirá que muchas personas, independientemente de su clase social, utilicen los servicios que por medio del IoT se podrían ofrecer y que en muchos países ya se están ofertando.

Servirá para llevar un control constante de nuestra salud, teniendo en cuenta que hay muchas enfermedades en las cuales síntomas son silenciosos y que un diagnóstico temprano permitiría la prevención y posibles soluciones a las enfermedades que pueden resultar mortales. Las diferentes soluciones/aplicaciones del IoT en la salud pueden ser sectorizadas en los servicios de Telemedicina, Emergencia, Medicación, Redes sociales para la salud, salud en el hogar, paquetes farmacéuticos inteligentes y en dispositivos biomédicos. [5]. Así como podemos ver en la siguiente propuesta de trabajo la implementación de la IoT en el sector de la salud, en la que se realizó el diseño e implementación de una Arquitectura de un Sistema de Tele monitorización para Hospitalización Domiciliaria de Adultos Mayores Apoyada en Tecnologías de Internet de las Cosas (IoT). Esta arquitectura se encarga de gestionar de manera eficiente el monitoreo de los adultos mayores que se encuentran en hospitalización domiciliaria, evitando que sucedan situaciones potencialmente peligrosas que pongan en riesgo la salud de estas personas. [9]. Es tal implementación de la IoT en el sector de salud ya que permite la interoperabilidad de diferentes dispositivos y el envío de datos en tiempo real para el monitoreo de las pacientes, tal es el caso que beneficiándose del desarrollo de Internet de las cosas de atención médica (IoHT) en los últimos años, el reconocimiento del modo de locomoción mediante sensores portátiles desempeña un papel importante en el campo de la rehabilitación en el hogar. En este documento, se presenta un sistema de detección inteligente que utiliza sensores de electromiografía flexible (EMG) y sensores de fuerza de reacción del suelo (GRF) para el reconocimiento del modo de locomoción, junto con su uso bajo la arquitectura IoHT. Es de gran importancia controlar la información física, como la marcha y la condición muscular, de los pacientes que se recuperan de lesiones o enfermedades en diferentes modos de locomoción para hacer un plan de rehabilitación adecuado y evaluar el efecto de rehabilitación. [10]

Para el desarrollo del sistema, uno de los componentes principales a utilizar son los denominados SBC por sus siglas en inglés (Single Board Computer). Son plataformas electrónicas de desarrollo que permiten su programación, con el fin de implementar proyectos computacionales de diverso tipo, para integrar y controlar

todas sus funciones de hardware, así como para ejecutar correctamente los códigos realizados en los diferentes lenguajes soportados. Existen SBC con capacidad de programación en lenguajes como Python, Scratch, Sonic Pi, Java, Matemática, C++ y otros. [11] Dos de las más reconocidas marcas de los cuales sus costos son asequibles son Arduino y RaspBerry que además contienen en sus diferentes placas diversas características y funcionalidades que proporcionan a los diferentes proyectos diversas soluciones apropiadas. Así como podemos ver a continuación la implementación de un Diseño de una herramienta de medición de ruidos basados en tecnologías Arduino-RaspBerry PI que ayuda al ser humano a resolver los problemas que aquejan a la sociedad, por tal razón se diseña esta herramienta que permite medir el nivel de contaminación auditiva o acústica en recintos cerrados, aplicando Internet de las cosas (IoT), el IoT permite una mejor calidad de vida, ya que tiene la capacidad de recopilar, analizar datos que reunidos entre sí puedan convertirse en información importante y de conocimiento [12]. Otro de los campos en los que incursionan estos dispositivos es en de la domótica tal como lo proponen el siguiente artículo, el cual tiene como objetivo diseñar e implementar un sistema de seguridad para el hogar con la capacidad de detección humana. El sistema tradicional de seguridad para el hogar, es decir, circuito cerrado de televisión (CCTV) solo puede capturar y grabar el video sin poder dar una respuesta de advertencia si hay algún objeto sospechoso. Por lo tanto, se requiere un método adicional de detección y advertencia de objetos si hay un intruso. El diseño propuesto se implementa utilizando Raspberry Pi 3 y Arduino, que están conectados por un cable USB. El sensor PIR está instalado en el Arduino y la cámara web está montada en Raspberry Pi 3. [13]

2.2 DISEÑO DEL SISTEMA

Para el desarrollo del diseño del sistema a implementar, se realizará una investigación que permite conocer cuáles son los diferentes elementos hardware a utilizar en el desarrollo del prototipo IoT. Al terminó de esta investigación y obteniendo todos los datos necesarios, se procederá a la ejecución de la implementación del dispositivo que se llevará a cabo en tres diferentes etapas, la etapa del diseño conceptual, el diseño preliminar y la etapa del diseño detallado, las

cuales describen paso a paso y de manera detalla el comportamiento y funcionamiento del prototipo, teniendo en cuenta los objetivos planteados.

▪ IDENTIFICACIÓN DE LOS ELEMENTOS HARDWARE

Para dar Inicio con el diseño del prototipo, primero se planteará una identificación de los diferentes elementos hardware que componen el desarrollo del sistema, para esto se implementara una comparativa entre sí de cada una de las características de los componentes que se tendrán en cuenta y que puedan cumplir con la línea del proyecto y satisfacer los objetivos.

CARACTERISTICAS								
Placas SBC´s	WIFI	Bluetooth	SRAM(MB)	Consumo de energía operacional	Dimensiones			Precio COP
					L(mm)	A(mm)	P(gr)	
Arduino Yún	Si	Si	64 MB	90 mA	Largo: 68.6 mm	Ancho: 53.4 mm	Peso: 25 g	\$ 200.983
ESP32	Si	Si	0,52 MB	80 mA	Largo: 59,76 mm	Ancho:28,05 mm	Peso: 9,75 g	\$ 40.000
ARDUINO UNO Wifi REV2	Si	No	6.1e^-6 MB	46 mA (Min)	Largo: 68.6 mm	Ancho: 53.4 mm	Peso: 25 g	\$ 128.765
Arduino MKR1000 WIFI	Si	No	0.032 MB	120 mA	Largo: 61,5 mm	Ancho: 29 mm	Peso: 32 g	\$ 119.193
MKR WAM 1300	Si	No	0.032 MB	120 mA	Largo: 67,64 mm	Ancho: 25 mm	Peso: 32 gr	\$ 157.500
MKR VIDOR 4000	Si	No	0.008 MB	120 mA	Largo: 83 mm	Ancho: 25 mm	Peso: 43.5 gr	\$ 250.000
Arduino uno	No	No	0.002 MB	46 mA (Min)	Largo: 68.6 mm	Ancho: 53.4 mm	Peso: 25 g	\$ 100.000

Arduino Mega	No	No	0.008 MB	93 mA (Min)	Largo: 101.5 mm	Ancho: 53.3 mm	Peso: 37 g	\$ 131.150
Arduino DUE	No	No	0.096 MB	75 mA (Min)	Largo: 101.5 mm	Ancho: 53.3 mm	Peso: 36 g	\$ 131.150

Tabla 6. Características placas SBC.

Una vez generada la información obtenida y demostrada en la tabla 6, se crearon los siguientes criterios cualitativos de acuerdo con los objetivos planteados, dichos criterios ayudaran a la selección de los elementos hardware óptimos para la implementación del prototipo IoT que se plantea desarrollar.

- Consumo de energía: El prototipo debe ser un sistema de bajo consume de energía para que garantice la durabilidad de la batería-pila. ya que el prototipo es un dispositivo que no está conectado a la corriente eléctrica permanentemente.
- Portabilidad: El prototipo debe contar con una dimensionalidad de tamaño adecuado para que en el momento de funcionar no afecte la movilidad de las camas o camillas, la comodidad del paciente y las labores del servicio médico.
- Costo: Es importante el valor monetario ya que el prototipo debe ser accesible para prestar su servicio.

De esta forma también se desarrollaron métricas a nivel cuantitativo para seleccionar el SBC que con sus características sea el adecuado para su implementación.

- Wifi, se estableció un 10% de las métricas ya que en la implementación del sistema es de importancia para desarrollar el canal de comunicación entre los diferentes componentes.

- Bluetooth, representó un 10%, dado que es de gran importancia tener una conexión inalámbrica para la transmisión de datos o para el uso de periféricos.
- SRAM, represento un 10%, ya que es la encargada de llevar a cargo la ejecución de los diferentes procesos del prototipo de forma rápida y concisa.
- Consumo de energía la cual represento un 15%, debido a que el prototipo no cuenta con una carga portable de sustento energético para su funcionamiento, de tal manera estará siempre conectado a una fuente de corriente y es necesario el mínimo consumo de esta en los centros hospitalarios.
- Dimensiones, se determinó un 15% del total de las métricas, ya que al tener una proporcionalidad de tamaño reducido no interferirá en las acciones del personal hospitalario, como también de la comodidad del paciente.
- Precios, se determinó un porcentaje del 40%, debido a que se realizará un dispositivo de bajo costo, en cumplimiento de los objetivos establecidos con anterioridad

Obtenida la información necesaria de cada una de las tarjetas SBC's, el preprocesamiento de estos datos pretende ser utilizados en tareas de análisis o descubrimiento de conocimiento conservando su coherencia. Este procedimiento se desarrollará a partir de la normalización de datos. Esta técnica es necesaria para adecuar los datos a los problemas de clasificación, debido a que estos no están en las mismas escalas numéricas y en algunos casos siguen diferentes formas de distribuciones [14]. A partir de esto se procederá desarrollar un análisis para la selección de la placa adecuada a utilizar en la implementación del prototipo. haciendo uso de la siguiente formula:

$$V' = \frac{V - \mathit{min}_A}{\mathit{max}_A - \mathit{min}_A} * (\mathit{newmax}_A - \mathit{newmin}_A) + \mathit{newmin}_A$$

Fórmula empleada para normalización de los siguientes atributos RAM, consumo Energético, Dimensiones y precio, en una escala de 1 a 5.

Para ello se procedió a enumerar (transformar valores categóricos en numéricos) y normalizar la tabla de datos de los SBC's, para lo cual se tomó una escala entre 0 y 1.

Para los indicadores donde se estableció en la tabla, los valores SI y NO, se tomó el SI con un puntaje de 1 y el NO con un puntaje de 0 (cero). [15]

Para los demás indicadores se aplicó la fórmula de normalización, con los siguientes valores de mínimo y máximo:

Memoria RAM:

- Valor Máximo: 64 MB – En la escala representa un puntaje de 5
- Valor Mínimo: 0.0000061 MB – En la escala representa un puntaje de 1

Consumo Energético:

- Valor Máximo: 120 mA – En la escala representa un puntaje de 5
- Valor Mínimo: 46 mA – En la escala representa un puntaje de 1

Dimensiones:

Largo:

- Valor Máximo: 101,5 mm – En la escala representa un puntaje de 5
- Valor Mínimo: 59,76 mm – En la escala representa un puntaje de 1

Ancho:

- Valor Máximo: 53,4 mm – En la escala representa un puntaje de 5
- Valor Mínimo: 25 mm – En la escala representa un puntaje de 1

Peso:

- Valor Máximo: 43,5 gr – En la escala representa un puntaje de 5
- Valor Mínimo: 9,75 gr – En la escala representa un puntaje de 1

Precio del SBC:

- Valor Máximo: 250.000 COP – En la escala representa un puntaje de 5
- Valor Mínimo: 40,000 COP – En la escala representa un puntaje de 1

CARACTERISTICAS									
Placas SBC's	WIFI 10%	Bluetooth 10%	SRAM (MB) 10%	Consumo de energía operacional (mA) 15%	Dimensiones 15%			Precio COP 40%	Puntaje (1 a 5)
					L(mm) 5%	A(mm) 5%	P(gr) 5%		
Arduino Yún	5	5	5,000	3,378	1,847	5,000	2,807	4,066	4,012
ESP32	5	5	1,032	2,838	1,000	1,430	1,000	1,000	2,287
Arduino Uno Wifi REV2	5	1	1,000	1,000	1,847	5,000	2,807	2,691	2,543
Arduino MKR1000 WIFI	5	1	1,002	5,000	1,167	1,000	3,637	2,508	2,539
MKR WAM 1300	5	1	1,002	5,000	1,755	1,000	3,637	3,238	2,704
MKR VIDOR 4000	5	1	1,000	5,000	3,227	1,000	5,000	5,000	3,278
Arduino uno	1	1	1,000	1,000	1,847	5,000	2,807	2,143	1,975
Arduino Mega	1	1	1,000	3,541	5,000	4,986	4,230	2,736	2,937
Arduino DUE	1	1	1,006	2,568	5,000	4,986	4,111	2,736	2,801

Tabla 7. Normalización de datos de las Placas SBC's

De las nueve (9) placas SBC's analizadas solo tres (3) cumplen con los requisitos para la implementación del prototipo, teniendo en cuenta las características y las métricas establecidas, cuyos resultados fueron:

Placa SBC	Puntaje
Arduino Mega	2,937
Arduino MKR1000 WIFI	2,539
ESP32	2,287

Tabla 8.Resultado de análisis de Placas SBC

Se determinó que la placa que cumple con las condiciones la placa ESP 32, donde contiene las características tales como un módulo de WIFI integrado WROOM32, chip el cual está diseñado para ser escalable y adaptable, que reduciría el tamaño, consumo de energía y sobre todo el costo de mercado, a su vez su fácil manejo de configuración son especificaciones que se ajustan necesariamente a los objetivos de este proyecto y por tal motivo se implementara para el desarrollo del proyecto.

A demás de la placa SBC se incorporará otro tipo de hardware que proporcionara distintas funcionalidades y ayudaran a la implementación y desarrollo del prototipo, estos dispositivo hardware denominado Raspberry que entran en la categoría de placas SBC, prometen simular las mismas capacidades que tienen un computador todo incluido en una tarjeta de proporciones reducidas en la que se puede presenciar varios componentes como puertos USB, Ethernet y HDMI, los cuales permitirán conectar otros dispositivos como un teclado, un ratón y entre otros, ayudando a controlar este miniordenador.

Para la selección de este dispositivo se tomará como referencia el mismo proceso que se desarrolló para determinar la placa SBC óptima para la implementación del proyecto. Se realizará una investigación de los diferentes tipos de Boards RaspBerry con cada una de sus características que hay en mercado. Como podemos ver en la siguiente tabla.

Métricas cuantitativas para la determinar la RaspBerry con las características adecuadas para la implementación del sistema IoT basado en SBC, así dando la mejor solución a los objetivos planteados, de esta forma se crearon las siguientes métricas:

- CPU se estableció un 10% del total de las métricas, debido a que es de gran importancia el buen rendimiento de la CPU dentro el sistema IoT
- RAM represento un porcentaje del 10%, ya que con una cantidad suficiente se ejecutarán de forma apropiada todos los componentes software que se encuentren dentro de la placa RaspBerry.

- Wifi represento un 10% del total de las métricas, esto debido a la estabilidad del canal de comunicación del sistema para todos los componentes que la conforman.
- Bluetooth se estableció un 5%, ya que es una alternativa para comunicación y para la conectividad de periféricos.
- Puertos USB representa un 10% de las métricas ya que se podrán conectar elementos tales como el teclado, mouse u otro periférico para la dada modificación o manejo de la placa.
- Ethernet se estableció un 10%, debido a sustitución del canal principal de comunicación dado por el modem, se podrá establecer por este medio.
- OS se estableció un 5% del total de las métricas, dado que se encargará de brindar un espacio de trabajo para instalar, modificar alguna característica del funcionamiento de la RaspBerry
- Precio, se estableció un 40 %, ya que este sistema se desarrollará con el mínimo costo posible, pero con las mismas garantías para satisfacer los objetivos.

CARACTERISTICAS								
RASPBERRY PI BOARDS	CPU (GHz)	RAM (Gb)	Wifi	Bluetooth	Puertos USB	Ethernet	OS	Precio COP
R. Pi Zero	BCM2835 ARM 7 1GHz	0,512 GB	No	No	No	No	Si	\$118.678
R. Pi Zero W	Broadcom BCM2835 1GHz	0,512 GB	Si	Si	No	Si	Si	\$138.793
R. Pi 1 Model A+	ARM11 ARMv6 0,7GHz	0,512 GB	No	No	Si, 1unidad	Si	Si	\$109.000

R. Pi 1 Model B+	ARM11 ARMv6 0,7 GHz	0,512 GB	No	No	Si, 4 unidades	Si	Si	\$135.572
R. Pi 2 Model B	quad-core ARM Cortex-A7, 0,9GHz	1 GB	No	No	Si, 4 unidades	Si	Si	\$148,100
R. Pi 3 Model B	ARM Cortex-A53, 1.2GHz	1 GB	Si	Si	Si, 4 unidades	Si	Si	\$166.600
R. Pi 3 Model B+	ARM Cortex A53, 1.4GHz	1 GB	Si	Si	Si, 4 unidades	Si	Si	\$184.450
R. Pi 3 Model A+	Cortex-A53 (ARMv8), 1.4 GHz	0,512 GB	Si	Si	Si, 1 unidad	No	Si	\$122,700
R. Pi 4 Model B	Quad core Cortex-A72 (ARM v8) 1,5 GHz	2 GB	Si	Si	Si, 4 unidades	Si	Si	\$184,450

Tabla 9. Características Placas Raspberry, Fuente Propia

Obtenidos los resultados de la investigación, se procederá a crear los distintos criterios cualitativos además de los cuantitativos, esto con él fin de reducir las opciones de selección de la placa Raspberry y proporcionar una mirada más clara para determinar que conceptos son los necesarios para la implementación del sistema y cumplir con los objetivos planteados.

- Costo: La placa deberá tener un valor monetario accesible, de bajo costo para la implementación del prototipo
- WIFI: La placa Raspberry deberá constar con la tecnología de conexión a la red WIFI
- USB: Es importante que la placa contenga este tipo de puertos para la conexión de los periféricos como mínimo el teclado y el mouse, con los que se puede configurar y dar un uso adecuado para el prototipo.

Siguiendo el mismo proceso empleado anteriormente para la selección de la tarjeta del prototipo IoT, se realizará los mismos pasos para determinar la RaspBerry correcta a implementar en sistema.

Fórmula descrita con anterioridad será empleada para la normalización de los siguientes atributos RAM, CPU y precio, en una escala de 1 a 5.

Para ello se procedió a enumerar (transformar valores categóricos en numéricos) y normalizar la tabla de datos de los SBC's, para lo cual se tomó una escala entre 0 y 1.

Para los indicadores donde se estableció en la tabla, los valores SI y NO, se tomó el SI con un puntaje de 1 y el NO con un puntaje de 0 (cero). [15]

Para los demás indicadores se aplicó la fórmula de normalización, con los siguientes valores de mínimos y máximos:

CPU

- Valor Máximo: 1,5 GHz – en la escala representa un puntaje de 5
- Valor Mínimo: 0,7 GHz - en la escala representa un puntaje de 1

Memoria RAM

- Valor Máximo: 2 GB – en la escala representa un puntaje de 5
- Valor Mínimo: 0,512 GB – en la escala representa un puntaje de 1

Costo

- Valor Máximo: \$184,450 – en la escala representa un puntaje de 5
- Valor Mínimo: \$109.000 – en la escala representa un puntaje de 1

Raspberry Pi Boards	CARACTERISTICAS								
	CPU (GHz) 10%	RAM (Gb) 10%	Wifi 10%	Bluetooth 5%	Puertos USB 10%	Ethernet 10%	OS 5%	Precio COP 40%	Puntaje (1 a 5)
R. Pi Zero	2,500	1,000	1	1	1	1	5	1,5131	1,752
R. Pi Zero W	2,500	1,000	5	5	1	5	5	2,5795	3,385

R. Pi 1 Model A+	1,000	1,000	1	1	5	5	5	1,0000	2,500
R. Pi 1 Model B+	1,000	1,000	1	1	5	5	5	2,4087	2,676
R. Pi 2 Model B	2,000	2,312	1	1	5	5	5	3,0729	3,048
R. Pi 3 Model B	3,500	2,312	5	5	5	5	5	4,0537	4,358
R. Pi 3 Model B+	4,500	2,312	5	5	5	5	5	5,0000	4,601
R. Pi 3 Model A+	4,500	1,000	5	5	5	5	5	1,7263	4,028
R. Pi 4 Model B	5,000	5,000	5	5	5	5	5	5,0000	5,000

Tabla 10. Normalización de datos RaspBerry Boards, Fuente: Propia

Según los resultados obtenidos en el proceso de normalización de datos la placa con los mejores puntajes es:

Placa RaspBerry	Puntaje
R. Pi 4 Model B	5,000
R. Pi 3 Model A+	4,601
R. Pi 3 Model B	4,358

Tabla 11. Resultado de análisis de la placa RaspBerry

siguiendo los criterios creados anteriormente la placa que obtuvo el puntaje más alto fue la RaspBerry Pi 4 modelo B, sin embargo es presenta una gran desventaja comparada con las otras dos placas, aunque cuentan casi con las mismas características de funcionalidad y hardware, el costo de mercado que tiene la placa en cuestión un poco más elevado que las otras RaspBerry, con esto dicho se decidió por la segunda mejor para ser implementada en el sistema IoT y que de igual forma cumple a cabalidad con los requisitos y criterios planteados y esa es la RaspBerry Pi 3 modelo B+.

Además de la placa SBC se utilizarán módulos electrónicos extras que ayudarán al desarrollo del proyecto. La integración de sensores, actuadores y conexiones inalámbricas de ágil acoplamiento, que permitirán una interconexión fácil con la placa SBC, Estos dispositivos tales como el sensor RFID son tecnologías compuestas por un microchip en forma de tarjeta o tag y por un lector que utilizan

ondas de radio son utilizadas para la identificación automática de personas u objetos, Los sistemas RFID son muy útiles para sistemas de control de acceso, seguridad electrónica, trazabilidad. Unas de las principales ventajas son

- Resistencia al medio.
- Facilidad de lectura.
- Rapidez de Respuesta (100 ms).
- Fácil implementación

Unas de sus principales características son:

- Lector-Grabador sistema de modulación y demodulación RFID 13.56MHz
- comunicación SPI lo que permite trabajar fácilmente con la mayoría de los microcontroladores.
- Rango de detección de tags RFID es de aprox. 5-7cm.

Su principio de funcionamiento consiste en pasar un TAG, cerca de un lector RFID, el TAG tiene la capacidad de enviar información al lector. Dicha información puede ser desde un simple código o todo un paquete de información guardado en la memoria del Tag.

A demás del RFID, otro dispositivo hardware que se plantea usar es el sensor de fuerza o Force Sensing Resistor por sus siglas en Ingles (FSR), son dispositivos robustos de película gruesa de polímero (PTF) que exhiben una disminución de la resistencia con el aumento de la fuerza aplicada a la superficie del sensor. Esta sensibilidad a la fuerza está optimizada para su uso en el control táctil humano de dispositivos electrónicos tales como electrónica automotriz, sistemas médicos y en aplicaciones industriales y robóticas.

Principales Características del FSR406:

- película gruesa de polímero (PTF) cuadrado de 1.72 " es flexible,
- Peso, liviana de 1.2 g
- Extremadamente delgada (0.02 "),
- Área de detección activa de 1.56 " × 1.56 ".

Ya seleccionados los diferentes dispositivos hardware de acuerdo con los criterios establecidos cada uno de ellos dispondrá de una parte software libre que ayudará a la comunicación esta se realizará a través de librerías y funciones escritas en el lenguaje de programación Arduino y conformaran así el prototipo IOT para la automatización y gestión de las camas y camillas de un hospital

2.2.1 DISEÑO PRELIMINAR

Generada la base conceptual requerida tanto hospitalaria denotando la gestión del recurso las camas o camillas dentro de las instituciones de salud, y la parte hardware mostrando los diferentes características y funcionalidades de los dispositivos que serán implementados para construcción del prototipo SBC, y que en conjunto dará paso a desarrollar una propuesta de solución basada en IoT para la problemática planteada.

De esta forma y basados en la investigación de los diferentes conceptos, los dispositivos seleccionados a utilizar en el prototipo y de acuerdo con esta se presenta a continuación las especificaciones a fondo de cada uno de ellos.

- **ESP 32**

ESP32 es una solución altamente integrada para aplicaciones de IoT con Wi-Fi y Bluetooth, con alrededor de 20 componentes externos. ESP32 integra un interruptor de antena, balun RF, amplificador de potencia, amplificador de recepción de bajo ruido, filtros y módulos de administración de energía. Como tal, toda la solución ocupa un área mínima de placa de circuito impreso (PCB)

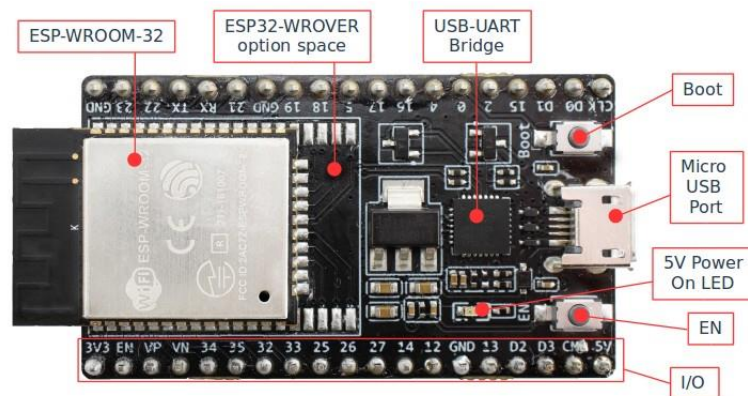


Ilustración 1. ESP 32 características,
 Fuente: <https://co.mouser.com/new/espressif/espressif-esp32-devkit-boards/>

Como podemos ver en la ilustración anterior el ESP32 puede alimentarse a través del puerto USB, 5V: se puede suministrar 5V, que es regulado a 3.3V para alimentar la placa.

Componente	Descripción
ESP32-WROOM-32	Modulo con ESP32 en el núcleo
EN	Botón de reset
Boot	Botón de descarga. Si mantiene presionado Boot y luego presiona EN, se inicia el modo de descarga de firmware para descargar el firmware a través del puerto serie.
USB-to-UART Bridge	Single USB-UART bridge chip provides transfer rates of up to 3 Mbps.
Micro USB Port	Interfaz USB. Fuente de alimentación para la placa, así como la interfaz de comunicación entre una computadora y el módulo ESP32
5V Power On LED	Se enciende cuando el USB o una fuente de alimentación externa de 5V

	está conectada a la placa. Para más detalles.
I/O	La mayoría de los pines del módulo ESP están divididos en los encabezados de los pines de la placa. Puede programar ESP32 para habilitar múltiples funciones

Tabla 12. Descripción de los componentes del Módulo ESP 32,
Fuente: propia

El ESP32 está diseñado para aplicaciones móviles, de dispositivos usables y de Internet de las cosas (IoT). Por ejemplo, en un escenario de múltiples sensores IoT de baja potencia, El ciclo de trabajo bajo se utiliza para minimizar la cantidad de energía que gasta el chip. La salida del amplificador de potencia también es ajustable, lo que contribuye a un intercambio óptimo entre el rango de comunicación, la velocidad de datos y el consumo de energía. [16]

- FSR 406 (Force Sensing Resistor)

Los FSR son sensores exhiben una disminución en la resistencia con un aumento en la fuerza aplicada a la superficie de estos, Son fáciles de usar y de bajo costo. Este sensor es un modelo Interlink 406 FSR con una región de detección cuadrada de 38 mm².

dispositivos que exhiben una disminución en la resistencia con un aumento en la fuerza aplicada a la superficie del sensor. Esta sensibilidad a la fuerza está optimizada para su uso en el control táctil humano de dispositivos electrónicos tales como electrónica automotriz, sistemas médicos y en aplicaciones industriales y robóticas. Estos sensores son bastante económicos y fáciles de usar, pero rara vez son precisos. También varían algunos de un sensor a otro, básicamente, para obtener rangos de respuesta.

Si bien los FSR pueden detectar peso, son una mala opción para detectar exactamente cuántas libras de peso tienen.

Los FSR están hechos de plástico y la lengüeta de conexión está engarzada en un material delicado. La mejor manera de conectarse a estos es simplemente enchufarlos en una placa de pruebas o usar un conector de estilo de abrazadera como pinzas de cocodrilo, encabezado hembra o un bloque de terminales.

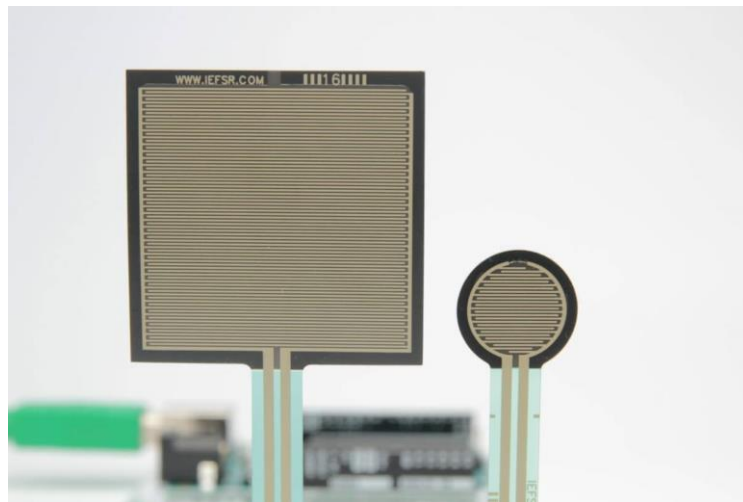


Ilustración 2. FSR 406,
Fuente: <https://www.makerguides.com/fsr-arduino-tutorial/>

Especificaciones Técnicas

FSR 406 Especificaciones Técnicas	
Longitud total	88 mm
Ancho total	43 mm
Área de detección	38 x 38 mm
Rango temperatura de funcionamiento	-30 ~70C
Fuerza de actuación	0.1 Newtons
Rango de sensibilidad de fuerza	0.1 - 10.0 ² Newtons

Tabla 13. Especificaciones Técnicas del FSR 406,

Características FSR 406

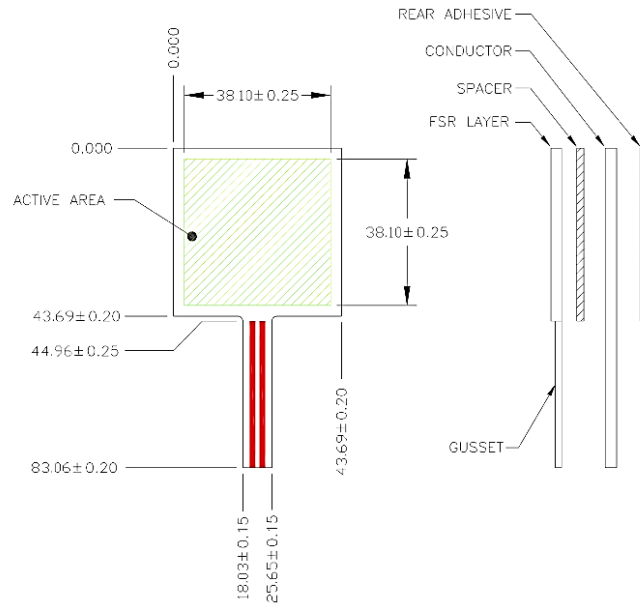


Ilustración 3. Características FSR406,
Fuente: Interlink Electronics

- RFID-RC522

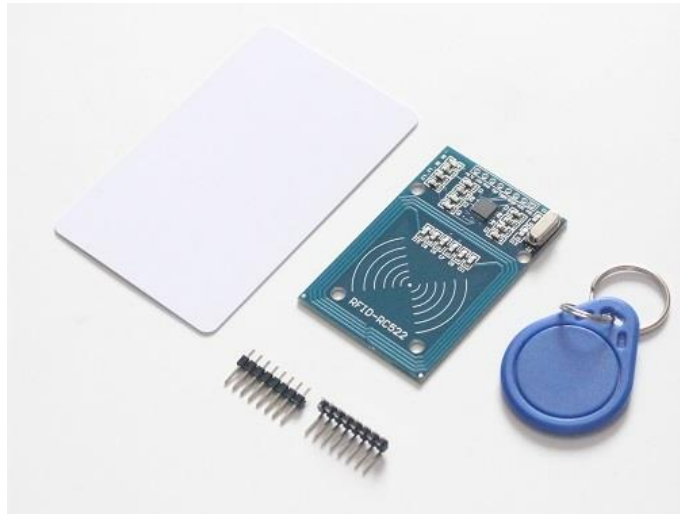


Ilustración 4 Tarjeta RFID-RC522,
Fuente: <http://www.roboticatemuco.cl/inicio/43-modulo-rfid-rc522.html>

El Módulo RC522 es Lector-Grabador RFID, posee comunicación SPI lo que permite trabajar fácilmente con la mayoría de microcontrolador. Utiliza un sistema de modulación y demodulación para todo tipo de dispositivos pasivos RFID de 13.56MHz. El dispositivo maneja el ISO14443-A y soporta el algoritmo de encriptación Quick CRYPTO1 y MIFARE. El rango de detección de tags RFID es de aprox. 5-7cm. Compatible con Arduino, Pic, Raspberry Pi y más.

Los sistemas RFID son muy útiles para sistemas de control de acceso, seguridad electrónica, trazabilidad. Su principio de funcionamiento consiste en pasar un TAG, cerca de un lector RFID, el TAG tiene la capacidad de enviar información al lector. Dicha información puede ser desde un simple código o todo un paquete de información guardado en la memoria del Tag.

ESPECIFICACIONES TÉCNICAS:

- Voltaje de Operación: 3.3V DC
- Corriente de Operación: 13-26mA/3.3V DC
- Corrientes de Stand by: 10-13mA/3.3V DC

- Transferencia de datos: Max. 10Mbit/s
- Dimensiones RFID-RC522: 40 mm x 60 mm
- Dimensiones Tarjeta: 85 mm x 54 mm
- No requiere de conexión a fuente de alimentación externa.
- Cuenta con un período de almacenamiento de 10 años, puede sobrescribirse 100000 veces y leerse ilimitadamente.



Ilustración 5. RFID - RC522,
Fuente: <https://ja-bots.com/producto/rfid-rc522/>

La tarjeta RFID cuenta con una serie de pines, así como lo podemos ver en la ilustración anterior, los pines SPI connector, la interfaz de comunicación con los otros componentes, son mediante ellos que se realizara la conexión directa entre la placa ESP 32.

El módulo RC522 tiene un total de 8 pines que lo conectan con el mundo exterior. Las conexiones son las siguientes: [17]

Vcc pin: suministra energía para el módulo. Esto puede ser de 2.5 a 3.3 voltios. Puede conectarlo a la salida de 3.3V de su Arduino. ¡Recuerde que conectarlo a un pin de 5V probablemente destruirá su módulo!

RST pin: es una entrada para restablecer y apagar. Cuando este pin se pone bajo, se habilita el apagado total. Esto apaga todos los sumideros de corriente internos, incluido el oscilador, y los pines de entrada están desconectados del mundo exterior. En el flanco ascendente, el módulo se reinicia.

GND: es el Pin de tierra y necesita estar conectado al pin GND en el Arduino.

IRQ: es un pin de interrupción que puede alertar al microcontrolador cuando la etiqueta RFID se acerca.

MISO/SCL/TX: el pin actúa como Master-In-Slave-Out cuando la interfaz SPI está habilitada, actúa como reloj en serie cuando la interfaz I2C está habilitada y actúa como salida de datos en serie cuando la interfaz UART está habilitada.

MOSI (Master Out Slave In): es la entrada SPI al módulo RC522.

SCK (Serial clock): acepta pulsos de reloj proporcionados por el bus maestro SPI, es decir, Arduino.

SS/SDA/Rx : el pin actúa como entrada de señal cuando la interfaz SPI está habilitada, actúa como datos en serie cuando la interfaz I2C está habilitada y actúa como entrada de datos en serie cuando la interfaz UART está habilitada. Este pin generalmente se marca encerrando el pin en un cuadrado para que se pueda usar como referencia para identificar los otros pines. [17]

2.2.2 DISEÑO CONCEPTUAL

En esta fase de la metodología, una vez obtenida la información necesaria se dispondrá a elaborar los procedimientos para el desarrollo del prototipo, comenzando por establecer las diferentes funcionalidades que el dispositivo realizará, a partir de ello, sus limitantes, la arquitectura a utilizar y todo lo que conlleva su estructuración desde un punto conceptual.

2.2.2.1 REQUISITOS FUNCIONALES

En esta primera parte del diseño del sistema, se analizarán y describirán las principales características y comportamientos particulares del prototipo IOT también conocidos como los requisitos funcionales del sistema.

REQUISITO FUNCIONAL	NOMBRE	DESCRIPCION
RF1	Conexión y verificación	El dispositivo se conectará a la red y verificará en qué estado está la cama o camilla
RF2	Activación prototipo	El dispositivo se activará para la lectura de las tarjetas o tag cuando el lector de presión de fuerza se active
RF3	Lector de Datos	El dispositivo podrá leer los datos de la tarjeta o tag del usuario por medio del RFID para separar o no la cama o camilla
RF4	verificación de estado	El sistema tendrá la capacidad de verificar el estado ocupado o libre de la cama o camilla cuando esta esté a punto de ser utilizada
RF5	Modificador de estado	El prototipo modificara el estado de la cama o camilla a ocupada o libre
RF6	Almacenamiento de información	El sistema podrá almacenar la información de la cama o camilla, de sus diferentes estados y como también de las diferentes tarjetas en un

		servidor remoto implementado en una RaspBerry
RF7	Visualización de información	El sistema permitirá la visualización de las diferentes camas o camillas con sus estados

Tabla 14. Requisitos Funcionales del prototipo IoT,
Fuente: propia

A continuación, se expondrán las limitaciones que tendrá el prototipo

- El dispositivo SBC no estará involucrado con los sistemas de información o base de datos del hospital.
- El dispositivo SBC no interferirá en los procesos médicos y en el monitoreo de los signos vitales del paciente.
- El dispositivo SBC no interferirá con los demás dispositivos que manejan la salud del paciente.

2.2.2.2 DISEÑOS DE LA ARQUITECTURA IOT DEL DISPOSITIVO.

En esta etapa de diseño, se realizarán dos tipos de arquitecturas para el prototipo que darán solución a la problemática planteada, esto con el fin de generar un mejor rendimiento de respuesta que permita obtener buenos resultados, como podemos ver a continuación dos ilustraciones de arquitectura IoT con el sistema SBC en conjunto con otras tecnologías que expresan el diseño que se propuso para dar solución al problema.

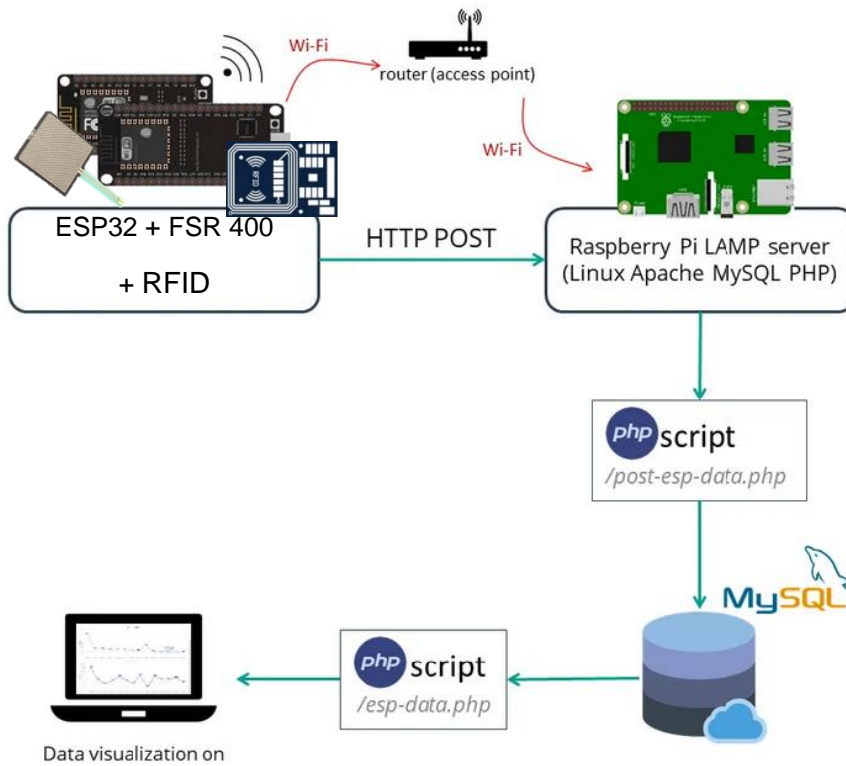


Ilustración 6. Diseño Arquitectura N° 1 del sistema con HTTP POST, servidor LAMP y MySQL

La propuesta de diseño de arquitectura IoT con el sistema SBC de la ilustración uno (1), se podrá evidenciar la interconectividad que hay entre el prototipo conformado por los dispositivos ESP32, el RFID y El sensor de fuerza FSR406 con la Raspberry que empleara la función de servidor, esta plataforma compuesta por una infraestructura de herramientas software de libre uso como Linux, apache, MySQL y PhP son una solución económica y principalmente para mostrar información a la que es fácil acceder en cualquier momento, estos dos componentes de la arquitectura los cuales están conectados a la red wifi dada por un modem que servirá como canal de comunicación utilizando el protocolo HTTP POST el cual es una de las principales formas de comunicación en la informática es a través de HTTP (del inglés, Hypertext Transfer Protocol, abreviado HTTP), es el protocolo de transmisión de información de la World Wide Web, es decir, el código que se establece para que el computador solicitante y el que contiene la información solicitada puedan “comunicarse” un mismo Lenguaje de programación a la hora de transmitir información por la red. Este protocolo establece las pautas a seguir, los métodos de petición (llamados “verbos”) y cuenta con cierta flexibilidad para

incorporar nuevas peticiones y funcionalidades, en especial a medida que se avanza en sus versiones.

La comunicación de los diferentes dispositivos será a través del protocolo previamente explicado, los cuales se encargarán de recoger y transmitir la información y con la ayuda de funciones escritas en PHP del inglés Hypertext Preprocessor (preprocesador de hipertexto) el cual es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el pre procesamiento de texto plano, así de esta forma se emplearan scripts que permitirán el tratamiento de los datos y a su vez almacenar esta información en una base datos alojada en el servidor potenciada por el gestor de base de datos MySQL bajo la licencia GNU GPL software de uso libre si ningún costo y es una base de datos muy rápida en la lectura lo cual es una de las características por las que se eligió este gestor de datos ya que para este ámbito hospitalario el tener la información al tiempo es de vital importancia.

Teniendo en cuenta el diseño de arquitectura para el sistema propuesto anteriormente, se indago sobre nuevos softwares que permitan la comunicación y el tratamiento de datos y que en conjunto con el prototipo IoT halla una sinergia dando como resultado una arquitectura que pueda dar una solución al problema de la gestión de las camas o camillas de un hospital. Así como se puede se obtuvo el siguiente diseño.

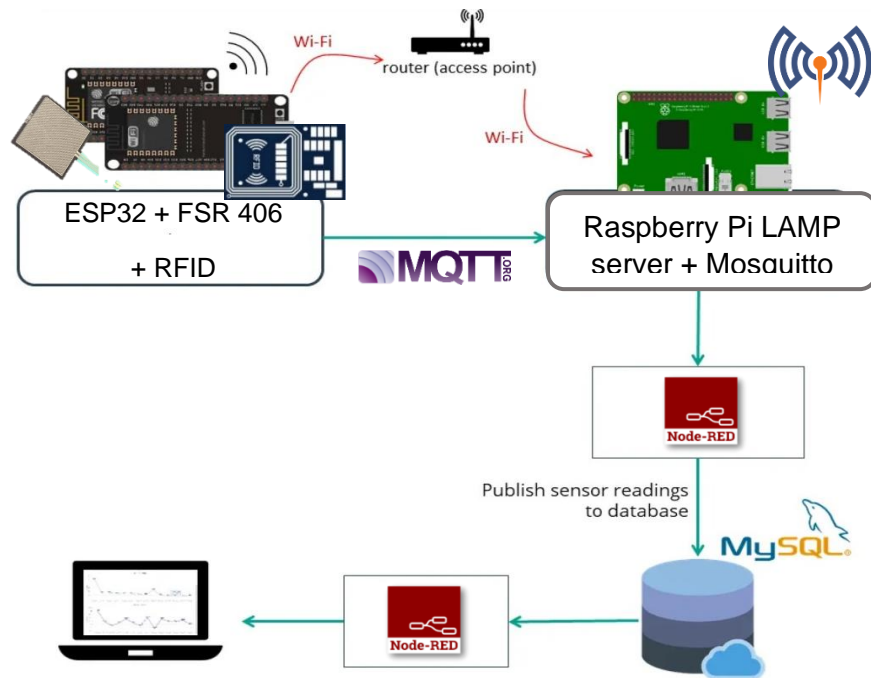


Ilustración 7. Diseño Arquitectura N°2 del sistema con MQTT + Node Red

Para este nuevo diseño de la arquitectura IoT para el sistema SBC el cual se tuvo en cuenta los mismos dispositivos y módulos, tales como el ESP32 , el RFID y la Raspberry PI 3 actuando como servidor, la forma de comunicar estos dos componentes del diseño es a través de MQTT.

MQTT significa MQ Telemetry Transport, es un protocolo de conectividad máquina a máquina (M2M) / "Internet de las cosas". Fue diseñado como un transporte de mensajes de publicación / suscripción extremadamente ligero. Es útil para conexiones con ubicaciones remotas donde se requiere una pequeña huella de código y / o el ancho de banda de la red es muy importante. Por ejemplo, se ha utilizado en sensores que se comunican con un Reuter a través de un enlace satelital, o sobre conexiones de acceso telefónico ocasionales con proveedores de atención médica y en una variedad de escenarios de automatización del hogar y dispositivos pequeños. También es ideal para aplicaciones móviles debido a su pequeño tamaño, bajo consumo de energía, paquetes de datos minimizados y distribución eficiente de información a uno o varios receptores. [18]

Está basado en el protocolo TCP/IP como base para la comunicación. En el caso de MQTT cada conexión se mantiene abierta y se "reutiliza" en cada comunicación. Es una diferencia, por ejemplo, a una petición HTTP 1.0 donde cada transmisión se realiza a través de conexión. [19]

En una arquitectura de un sistema MQTT se conectan los dispositivos. Utilizan una topología en estrella, es decir, todos los clientes se conectan directamente a un punto central que hace de servidor. En protocolo MQTT este servidor se llama Broker. Este tipo de arquitecturas lleva asociada otra interesante característica: la comunicación puede ser de uno a uno o de uno a muchos. [20]

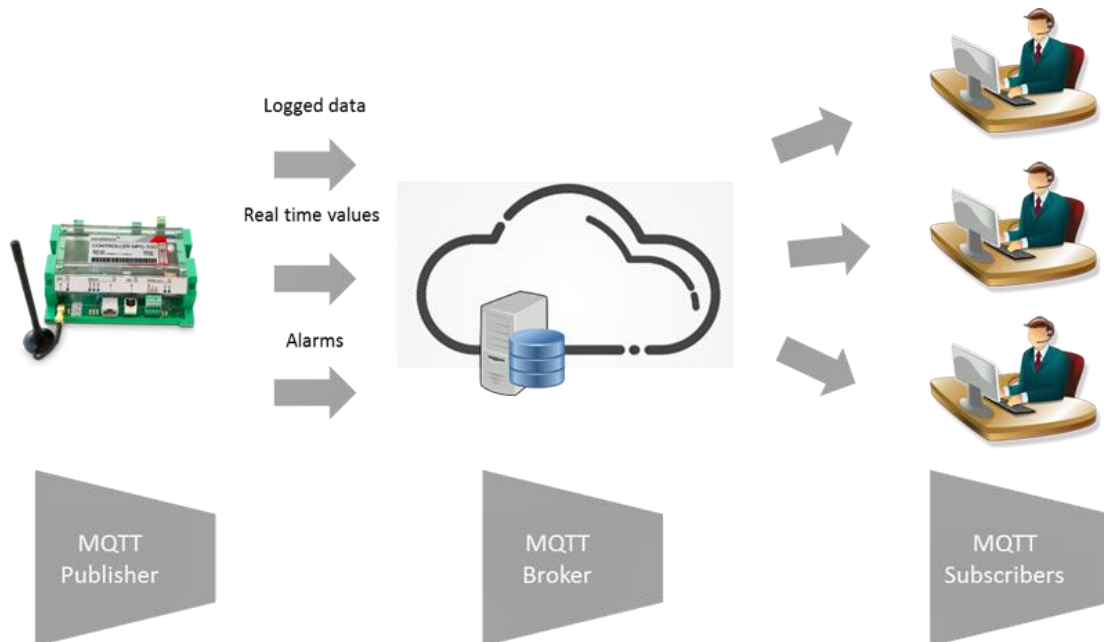


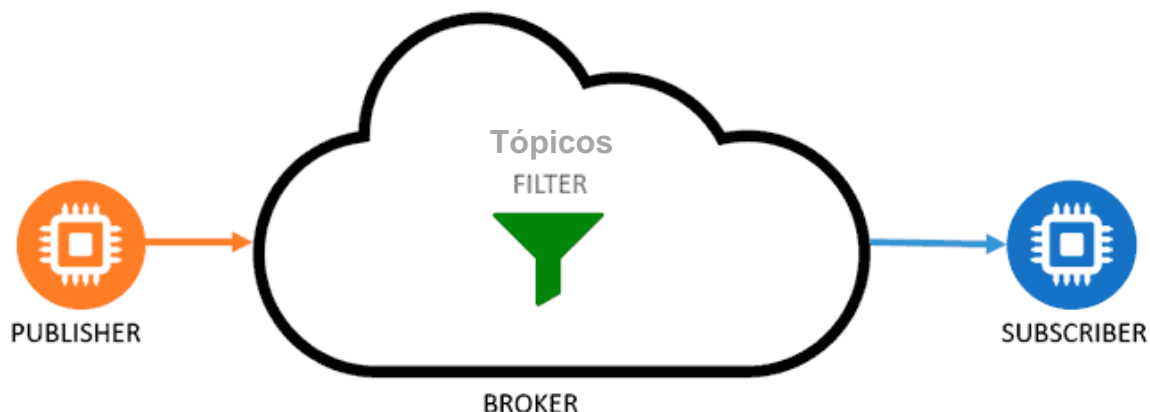
Ilustración 8. Arquitectura MQTT

Fuente: <http://www.casadomo.com/2017/02/22/pasarela-industrial-integracion-concentradores-datos-modbus-cualquier-plataforma-oit>

El funcionamiento del MQTT es un servicio de mensajería push con patrón publicador/suscriptor (pub-sub), en este tipo de infraestructuras los clientes se conectan con un servidor central denominado broker. Para filtrar los mensajes que son enviados a cada cliente, estos se disponen en topics³ organizados

³ "topic" o "tema" en español ya que a través de estos "topics" se articula la comunicación puesto que emisores y receptores deben estar suscritos a un "topic" común para poder **entablar la comunicación**.

jerárquicamente. Un cliente puede publicar un texto en un determinado topic. otros clientes pueden suscribirse a este topic, y el broker le hará llegar los mensajes suscritos.



*Ilustración 9. Filtro de Mensajes MQTT,
Fuente: <https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/>*

Los clientes inician una conexión TCP/IP con el broker, el cual mantiene un registro de los clientes conectados. Esta conexión se mantiene abierta hasta que el cliente la finaliza.

A demás para la construcción del diseño de la ilustración 2, otra de las tecnologías involucradas que complementa de forma exacta en funcionalidad con las demás utilizadas, es mosquitto. Mosquitto es a su vez un mediador de mensajes que incluye el protocolo MQTT y también un broker OpenSource ampliamente utilizado debido a su ligereza lo que nos permite, fácilmente, emplearlo en gran número de ambientes, incluso si éstos son de pocos recursos.

Los datos obtenidos y transmitidos por el protocolo MQTT y almacenados previamente en un servidor OpenSource por parte de Mosquito y haciendo uso de una herramienta que proporcionara con sus funcionalidades la comunicación y almacenaje de los datos en MySQL de manera permanente, es a través de Node-RED.

Node-RED es una herramienta de programación basada en flujo, desarrollada originalmente por el equipo de Servicios de Tecnología Emergente de IBM y ahora

parte de la Fundación JS. Node-RED es una herramienta de programación para conectar dispositivos de hardware, API y servicios en línea. Proporciona un editor basado en navegador que facilita la conexión de flujo de datos utilizando la amplia gama de nodos que permiten implementar en su tiempo de ejecución con un solo clic, donde se pueden crear funciones de JavaScript dentro del editor de texto enriquecido. Node-RED se basa en Node.js, aprovechando al máximo su modelo. Esto lo hace ideal para ejecutarse en el borde de la red en hardware de bajo costo, como Raspberry Pi, así como en la nube. La programación basada en flujo es una forma de describir el comportamiento de una aplicación como una red de cajas negras o "nodos" como se les llama en Node-RED. Cada nodo tiene un propósito bien definido; se le ingresan algunos inputs o entrada de datos, procesa los datos y a su vez estos datos son prerequisites de los nodos siguientes. La red como tal es responsable del flujo de datos entre los nodos. [21]

2.2.3 DISEÑO DETALLADO

El comportamiento del prototipo SBC basado en IoT para la gestión de las camas o camillas y su iteración entre los diferentes módulos que lo componen cumplir con los objetivos, siguiendo el diseño seleccionado en la anterior etapa, se configuro el prototipo de la siguiente forma.

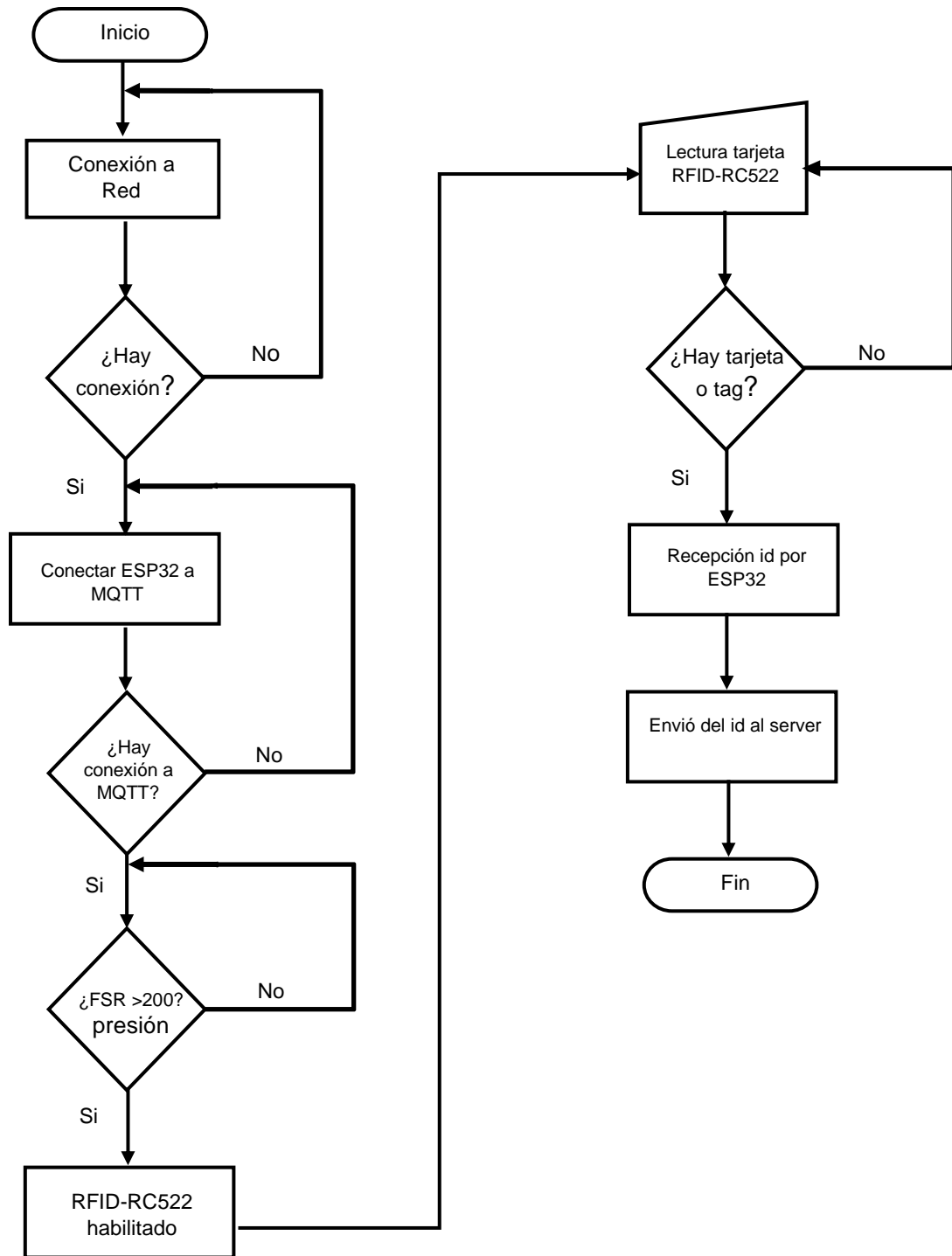


Ilustración 10. Diagrama de Flujo del prototipo,
Fuente: Propia

La conexión del prototipo a la red se da al instante en el que se enciende, este a su vez conectará con el protocolo MQTT, habilitando el funcionamiento de todos los módulos y componentes que conforman el prototipo dando paso a su uso.

El sensor FSR (Force Sensing Resistor) uno de los componentes involucrados en el desarrollo del prototipo y cuya función al activarse es el de proporcionar los primeros datos aplicando una presión mayor igual a 200Ω el cual representa la resistencia del circuito abierto que integra el FSR al aplicarle una determinada presión, donde esta señal análoga proporcionada por el mismo sensor de presión de fuerza será recibida por la placa ESP32, de esta manera el prototipo analizará desde la parte lógica dicha señal y activará los demás componentes con sus respectivas funcionalidades.

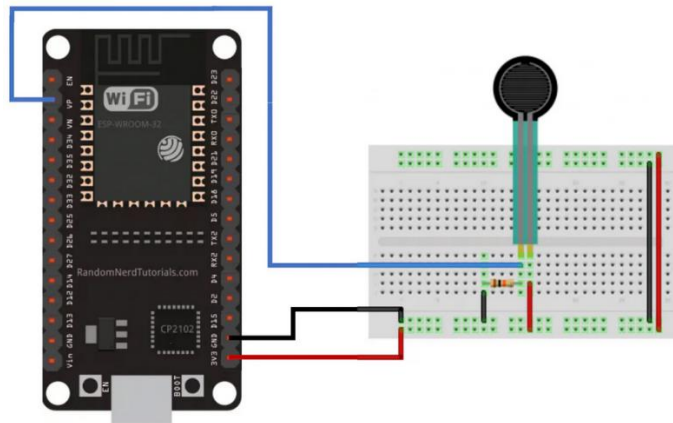


Ilustración 11. Conexión electrónica ESP32 y FSR,
Fuente: <https://www.hackster.io/jmlopezdona/Smart-saddle-a1eff9>

La conexión electrónica empleada para el funcionamiento explicado anteriormente es la que podemos ver en la ilustración anterior, de esta forma se estableció la comunicación entre el sensor y la placa para obtener los resultados apropiados siguiendo específicamente el diseño elegido. Una vez el sensor de presión haya emitido la señal por su activación, El ESP32 activa la función de habilitar módulo RFID-RC522, este dispositivo tiene la capacidad de leer la información, la cual ayudará con facilidad a obtener el id que el tag tiene por defecto, una vez obtenida dicha información esta será

enviada a la placa ESP32 para su interpretación ya que esta es enviada de forma encriptada. Para que esta comunicación se dé entre dos dispositivos se seguirá una conexión electrónica como podemos ver en la ilustración N°12, entre estos elementos y teniendo en cuenta las especificaciones técnicas de cada uno de ellos para su mejor desempeño en conjunto.

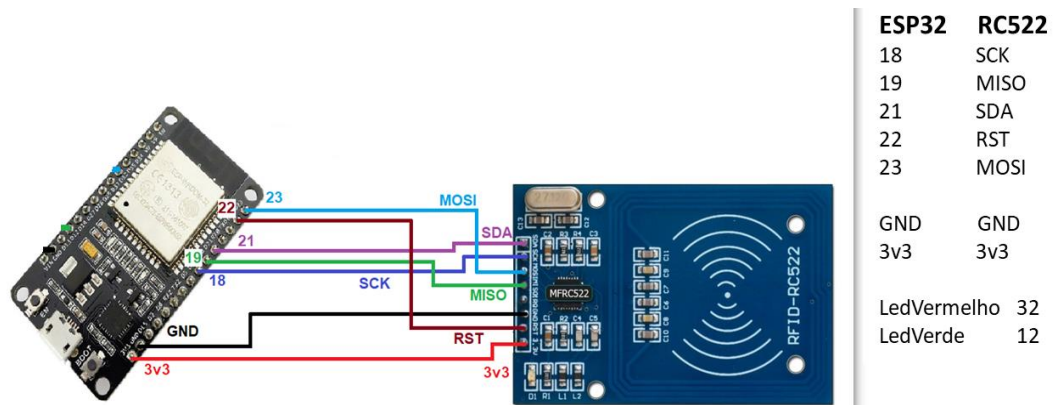


Ilustración 12. Conexión electrónica ESP32 y RFID - RC522

La recepción de los datos y siguiendo el esquema de conexión, el ESP32 interpretará dicha información transmitida por el RFID, traduciéndola en formato hexadecimal para su mejor lectura y **Posteriormente su envío al servidor:**

Una vez entendido las características de cada uno de los componentes, el funcionamiento interno y su comportamiento, esto permite crear una comunicación dinámica desde la parte electrónica y lógica entre los diferentes módulos, fortaleciendo el funcionamiento en común para el propósito que fue implementado, es así como resultado final se obtuvo la siguiente configuración del prototipo, vista en la ilustración número N°13.

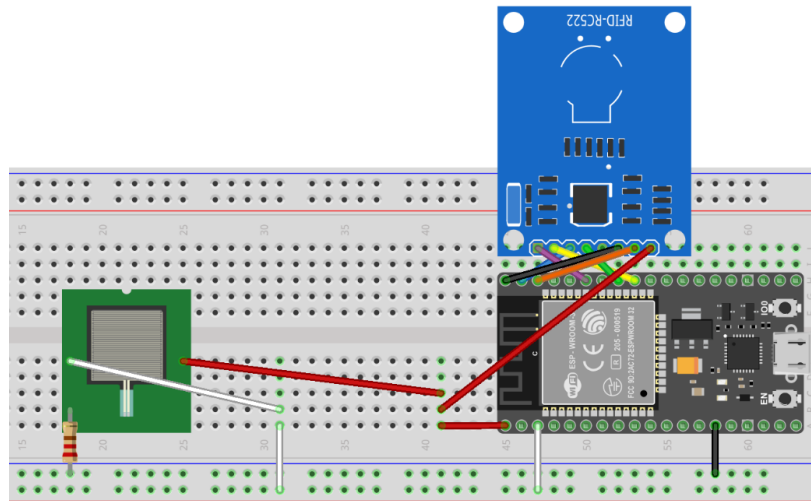


Ilustración 13. Esquema preliminar del prototipo

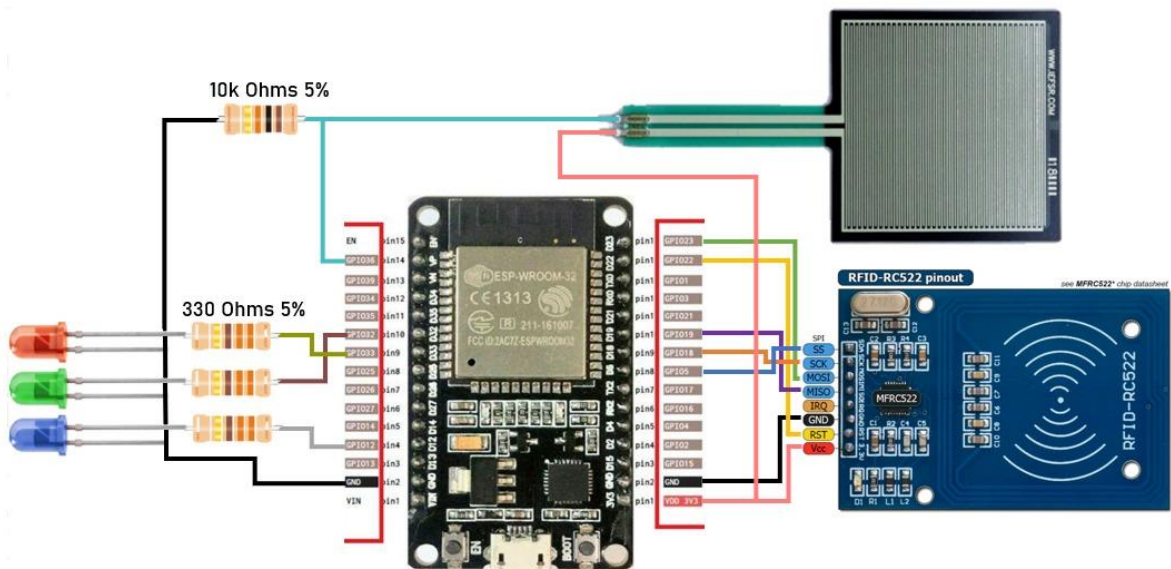


Ilustración 14. Esquema final del prototipo, Fuente: Propia

Generada la comunicación entre los diferentes elementos hardware que conforman el prototipo IoT, se procederá a realizar el envío de los datos que se obtuvieron del RFID y es a través de unas funciones programables y haciendo uso de las herramientas MQTT y Node-RED que servirán como canal de comunicación, dicha información se enviará por parte del ESP32 en donde se almacenarán en un registro en la base de datos MySQL alojada en un servidor LAMP implementada en la RaspBerry Pi 3 b+ la cual está

configurada dentro de una subred con una IP estática. Así como podemos ver en la siguiente ilustración.

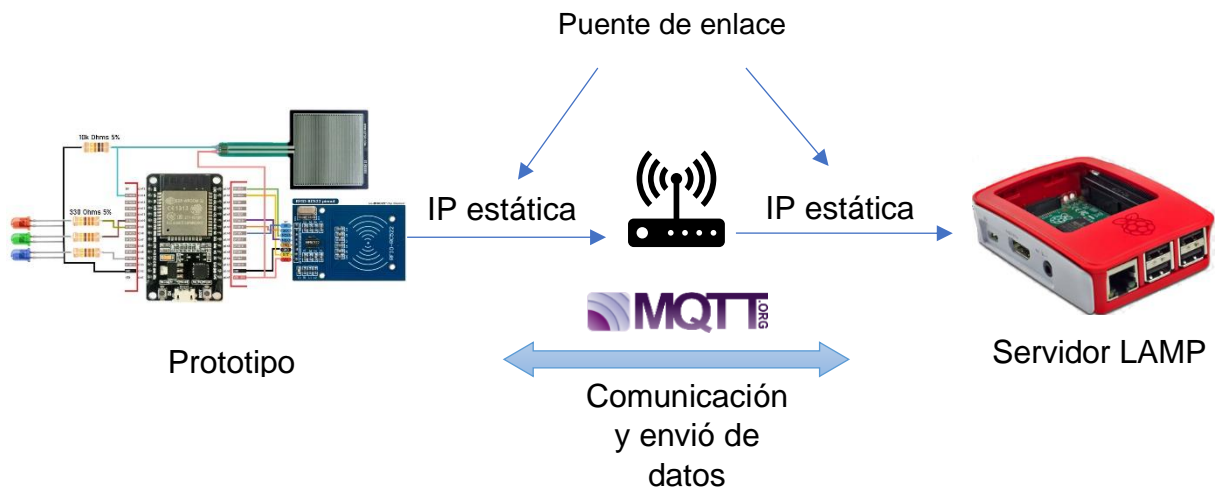


Ilustración 15. Conectividad del sistema IoT, Prototipo y Servidor LAMP

IoT hasta el servidor donde se aloja una base de datos para almacenar toda la información requerida, que representan el código alfanumérico de cada una de las tarjetas y el número de la cama o camilla del hospital, solamente quedaría visualización como parte del sistema para dar una retroalimentación al usuario final de cual es estado final en el que se encuentra este recurso, y es a través de la herramienta NodeRed que además de hacer el procesamiento de los datos y las consultas a la base de datos en complemento con MQTT, proporciona una funcionalidad de generar una interfaz gráfica mostrando los datos que se le requiera, es así como los usuarios se podrán acceder de forma remota a la información obtenida por el dispositivo IoT.

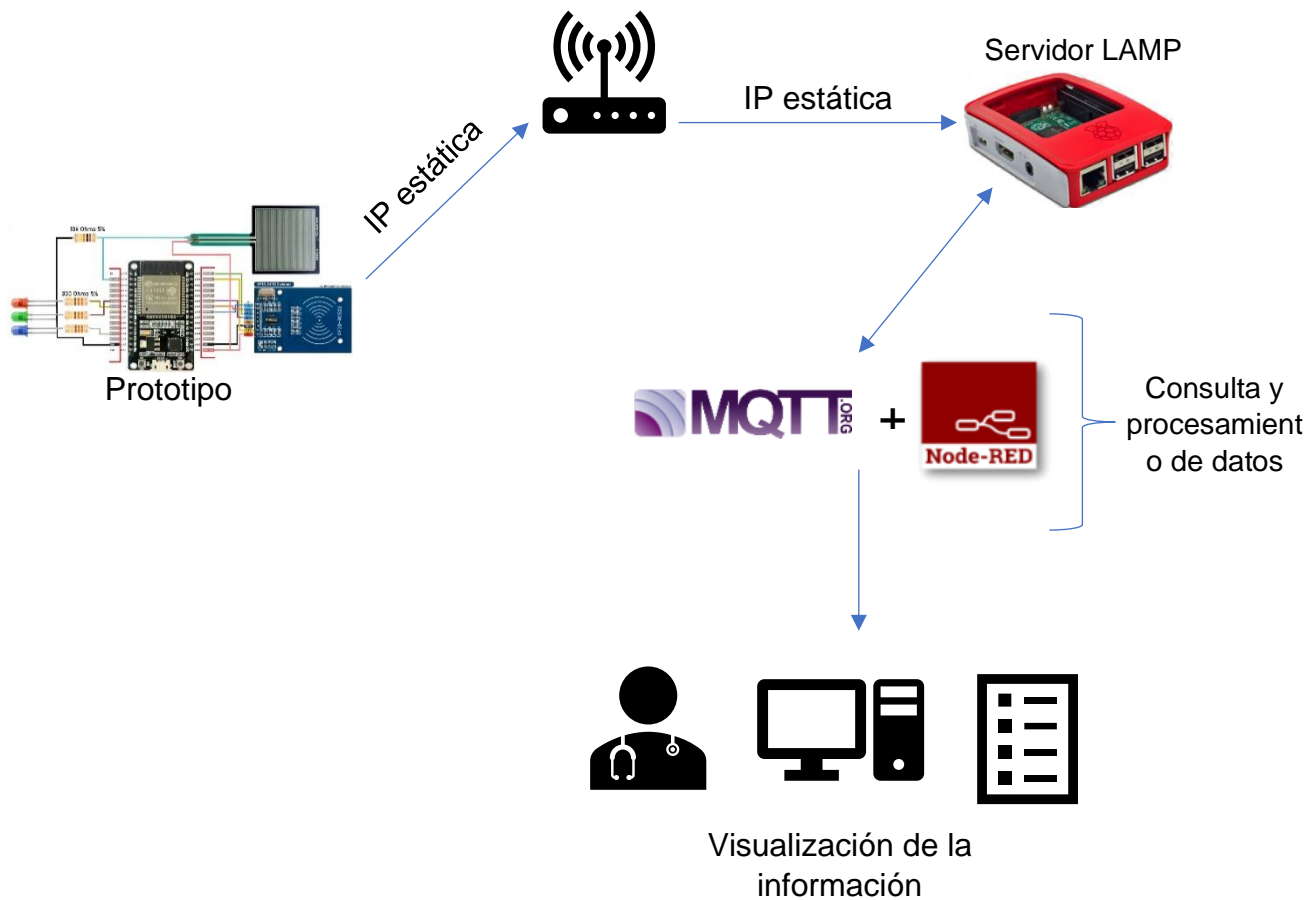


Ilustración 16. Conectividad completa del Sistema IoT

CAPITULO 3. IMPLEMENTACIÓN DEL SISTEMA

En este capítulo como su nombre lo indica se construirá el prototipo SBC basado en IoT para la gestión de camas o camillas. En esta fase del proyecto donde los componentes o módulos seleccionados en el diseño creado en los capítulos anteriores y con el análisis previo de los esquemas de las conexiones eléctricas, electrónicas y de software de cada uno de ellos, se inició la construcción del prototipo.

- **IMPLEMENTACIÓN DE HARDWARE**

Teniendo en cuenta el estudio realizado previamente de la selección de los componentes y sus características se procedió a la realización de la

construcción del prototipo, los elementos a utilizar para este desarrollo son los siguientes

Componentes o módulos hardware:

COMPONENTE O MODULO HARDWARE
ESP 32
RFID RC522
FSR 406
LED's de notificación
RASPBERRY Pi 3 B+
MODEM
RESISTENCIAS

Tabla 15. Componentes Hardware del dispositivo SBC

La primera parte implementada es el servidor LAMP y la configuración de la red por la cual se pretendía establecer la comunicación, para este desarrollo se utilizó la placa SBC RaspBerry PI 3 B+ las placas RaspBerry son prácticamente computadores con características menores en procesamiento y almacenaje que un Pc normal pero lo suficientemente poderoso y adaptable que lo hacen esencial para este tipo de proyectos, es por esto por lo que este tipo de placa se adaptó fácilmente nuestro prototipo. Configurando la RaspBerry y transformándola en un servidor remoto, se inició con la instalación del sistema operativo Raspbian,” Raspbian es un sistema operativo gratuito basado en Debian optimizado para el hardware Raspberry Pi. Sistema operativo con un conjunto de programas básicos y utilidades que hacen que su Raspberry Pi funcione. Sin embargo, Raspbian proporciona más que un sistema operativo puro: viene con más de 35,000 paquetes, software precompilado incluido en un formato para una fácil instalación en su Raspberry Pi” [22].

La implementación del hardware del prototipo se realiza a la par con la implementación del software, ya que se necesita de los componentes y de sus características propias para ser leídas e incorporadas al código donde se utilizarán

o modificarán según el uso que se le quiera dar para cada una de las funcionalidades creadas.

En la primera fase de implementación del prototipo de gestión de camas y camillas, se realizó la conexión de los módulos ESP32, el lector de tarjetas o tags REFID y RC522, siguiendo a pie las conexiones pertinentes establecidas en el capítulo anterior en la ilustración N°12 se procedió a su desarrollo, así proporcionando en parte una de las principales funcionalidades para este para la gestión del recurso en cuestión. El resultado final de esta fase del prototipo fue el siguiente, tal como lo podemos ver en la siguiente ilustración

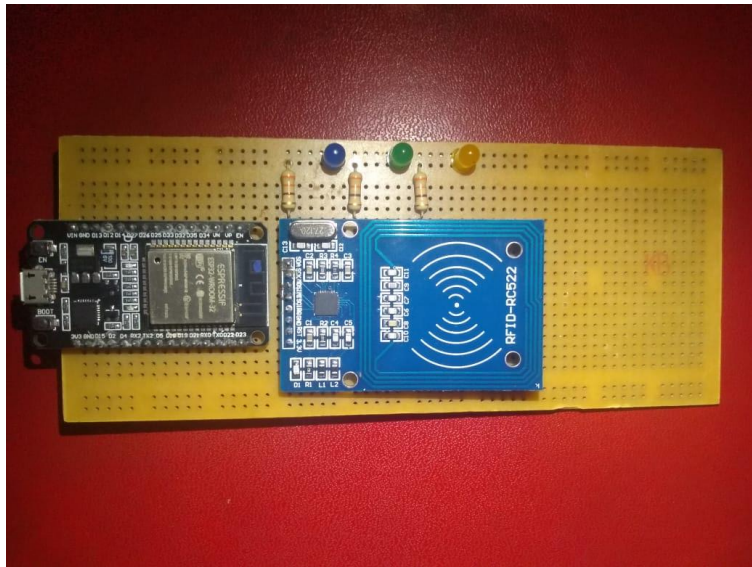


Ilustración 17. Primera fase de implementación, ESP32 + REFID RC522 + LED'S de notificación, Fuente: Propia

Además de la comunicación de los componentes ESP32 y el RFID RC522, se implementó un sistema de notificación hacia el usuario mediante la instalación de luces led que darán a conocer el estado en el que se encuentra la cama o camilla, como también la transición de cambio de estado del recurso. Esta conexión se implementó siguiendo las especificaciones técnicas de la ilustración N° 14, que mediante resistencias de 330Ω para cada led, permitirá una iteración directa y dinámica con los dos componentes ya previamente instalados.

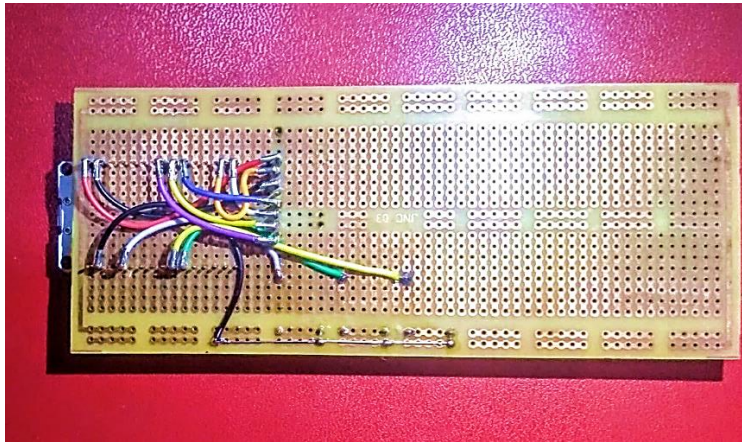


Ilustración 18. Conexión de los módulos ESP32 + RFID RC522 + LED'S de notificación

una vez completada esta fase donde los diferentes componentes se comunican de forma apropiada y eficiente, el siguiente paso a implementar es la conexión del sensor de presión FSR 406 al dispositivo, como se muestra en la siguiente ilustración:

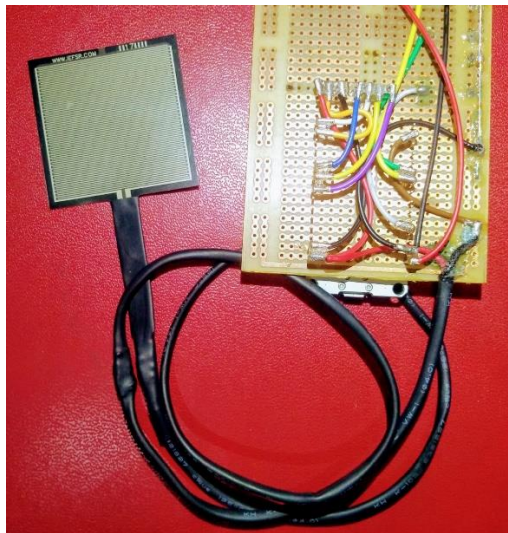


Ilustración 19. Conexión del FSR 406 al prototipo en desarrollo

Siguiendo las especificaciones técnicas para su conexión al ESP32 , se establecerá esta comunicación a través de cable desde la punta del sensor hasta la base del prototipo el cual tiene una longitud de 70,7 cm, este a su vez se conectará a una resistencia de 10k Ω que hará puente con él modulo ESP32, la implementación del sensor cuya funcionalidad principal es activar las otras funciones del dispositivo.

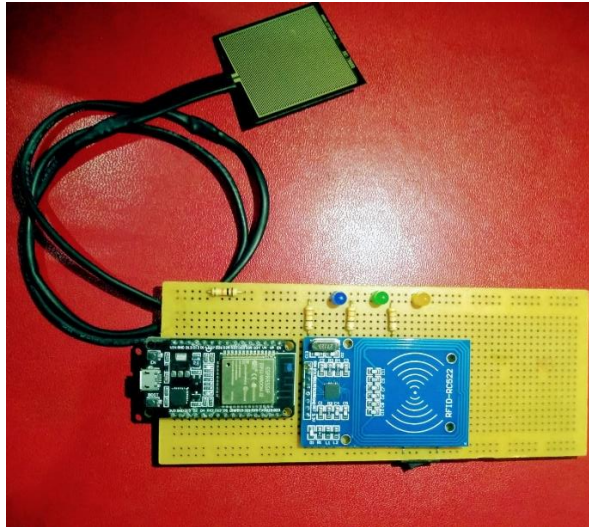


Ilustración 20. Conexión del sensor FSR406 al ESP32

En la última fase de implementación del prototipo, se incorpora un switch de encendido y apagado, como también un adaptador de entrada para la alimentación de energía hacia el dispositivo.

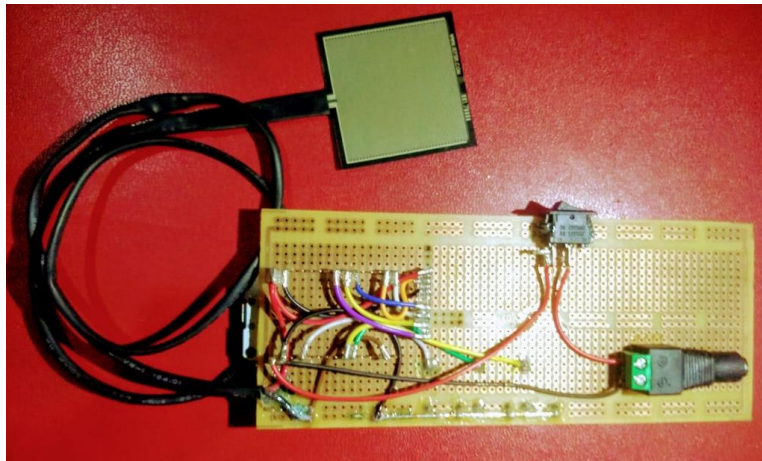


Ilustración 21. Implementación de switch de encendido y adaptador de entrada de energía del prototipo

Esta parte se hace con el fin de conectar el dispositivo a una fuente de alimentación inalámbrica que proporciona la suficiente corriente para que el dispositivo trabaje con normalidad y no sufra anomalías en su funcionamiento diario.

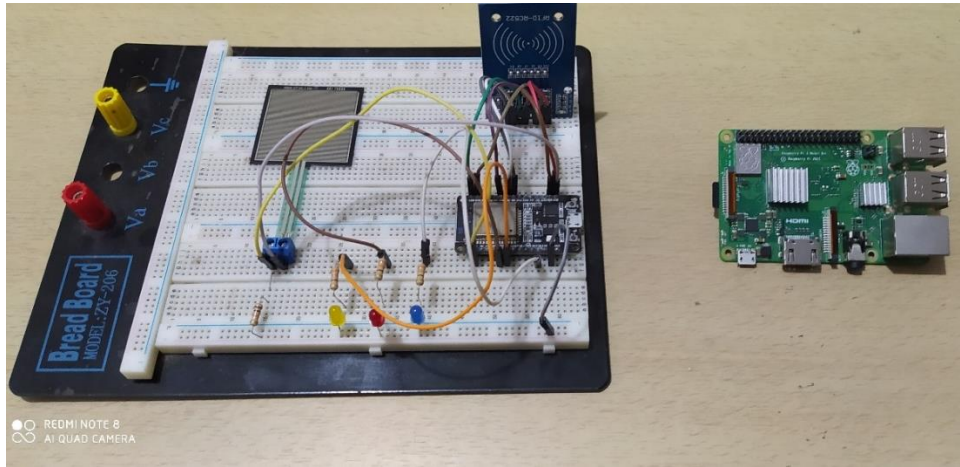


Ilustración 22. Prototipo preliminar, Dispositivo SBC + RaspBerry Pi B+ servidor LAMP

Realizada la implementación del prototipo correspondiendo a los objetivos planteados en el proyecto, se tuvo como resultado un sistema electrónico de proporciones adecuadas teniendo en cuenta el contexto en el que se implementará dando como acertada la selección de cada uno de los dispositivos para el desarrollo de este proyecto. Así como se observa en la ilustración N° 23 con el prototipo final.

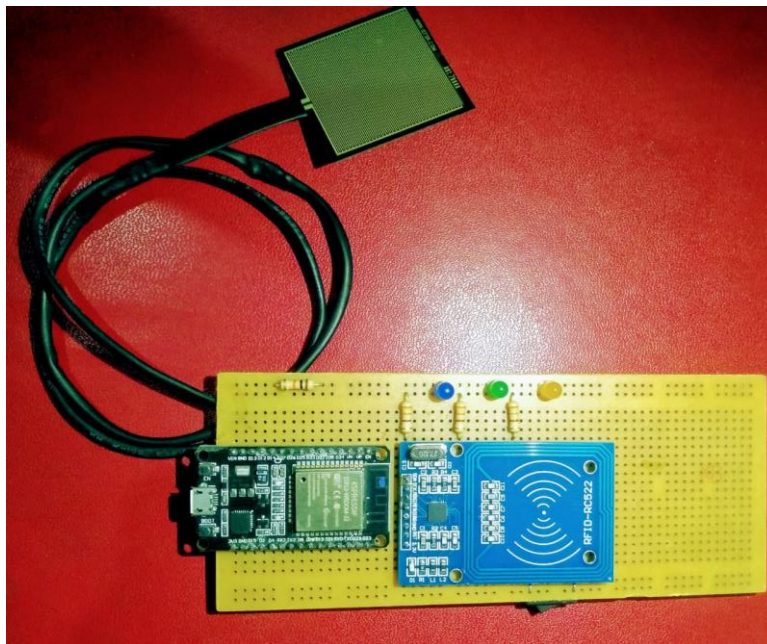


Ilustración 23. Prototipo final

- **IMPLEMENTACIÓN DE SOFTWARE**

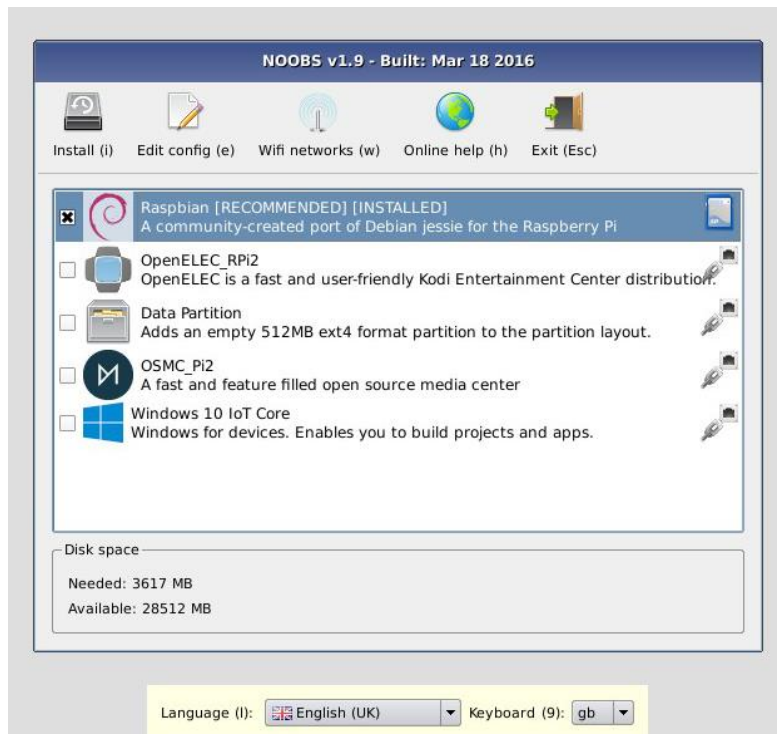
Componentes Software:

COMPONENTE SOFTWARE
Arduino IDE
RASPBIAN
APACHE
MySQL
PHP
MQTT
NODERED

Tabla 16. Componentes Software del sistema IoT

Para la instalación del sistema operativo Raspbian, la organización que construye las placas Raspberry, ofrece una herramienta llamada **NOOBS** que son las siglas en inglés para ‘New Out Of Box Software’, NOOBS está diseñado para facilitar la selección e instalación de sistemas operativos para Raspberry Pi sin tener que preocuparse por obtener imágenes de su tarjeta SD manualmente.

En el primer arranque, NOOBS volverá a particionar su tarjeta SD y le permitirá seleccionar qué SO quiere instalar de una lista. Esta lista de SO se genera automáticamente a partir de los SO disponibles localmente (es decir, los contenidos en el directorio / os en el disco) o los disponibles desde nuestro repositorio remoto (se requiere conexión de red).



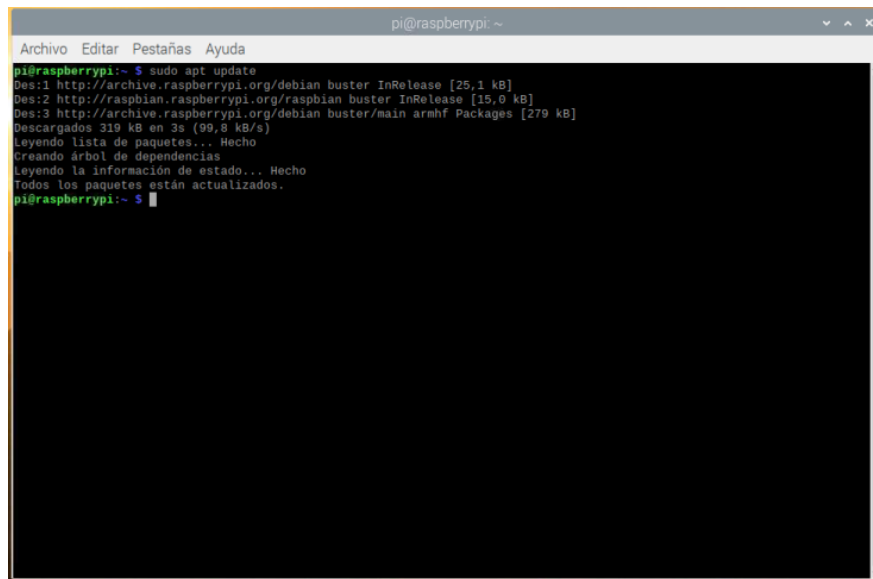
*Ilustración 24. Interfaz, instalador SO,
Fuente: <https://github.com/raspberrypi/noobs/blob/master/README.md>*

Como podemos ver en la imagen anterior, solo se mostrará la última versión de cada sistema operativo, lo que significa que puede estar seguro de que ha instalado la versión más actualizada de su sistema operativo seleccionado. [23]

Iniciando la instalación del sistema operativo y la que solo tardará unos cuantos minutos, la Raspberry quedará lista con una imagen para su manejo y eventual transformación en un servidor. Otra de las herramientas a utilizar es Apache. Es un servidor HTTP de código abierto para sistemas operativos modernos, incluidos UNIX y Windows. El objetivo de este proyecto es proporcionar un servidor seguro, eficiente y extensible que proporcione servicios HTTP sincronizados con los estándares HTTP actuales [24].

Ingresando al panel de comandos del sistema operativo previamente instalado, uno de los primeros comandos a ejecutar es **sudo apt update**, el cual nos permitirá descargar las listas de paquetes de los repositorios y las "actualiza" para obtener

información sobre las versiones más recientes de paquetes y sus dependencias. Así como vemos en la siguiente ilustración.

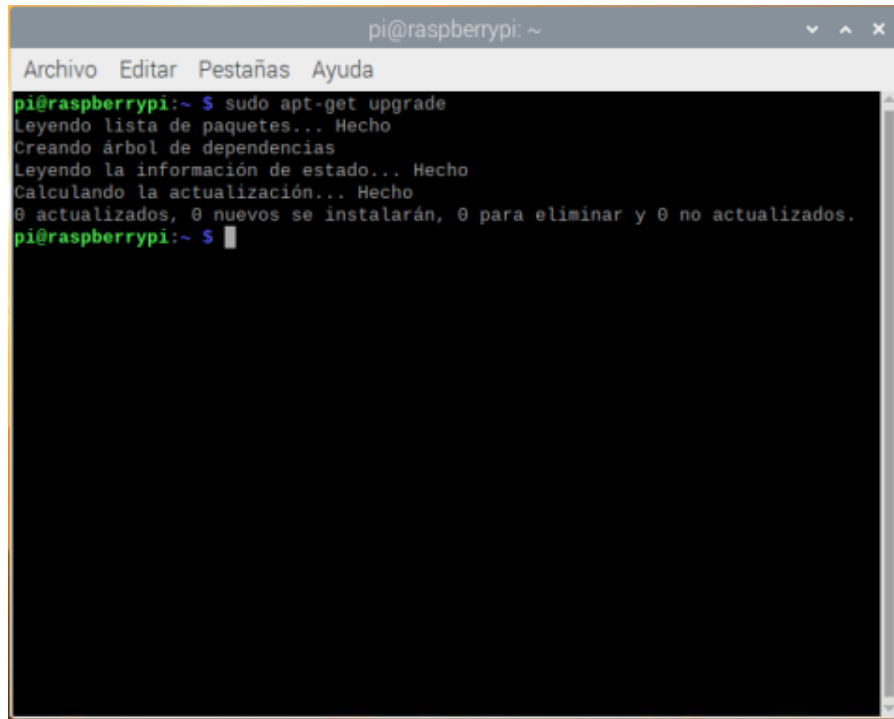


```
pi@raspberrypi:~$ sudo apt update
Des:1 http://archive.raspberrypi.org/debian buster InRelease [25,1 kB]
Des:2 http://raspbian.raspberrypi.org/raspbian buster InRelease [15,0 kB]
Des:3 http://archive.raspberrypi.org/debian buster/main armhf Packages [279 kB]
Descargados 319 kB en 3s (99,8 kB/s)
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Todos los paquetes están actualizados.
pi@raspberrypi:~$
```

Ilustración 25. Actualización de los paquetes de instalación para el sistema del servidor,

Fuente: Propia

Una vez se haya actualizado la lista, procedemos a actualizar el sistema de gestor de paquetes donde su finalidad es a la instalación y eliminación de los diferentes programas de los sistemas operativos Linux. Ejecutando la siguiente línea de comando el resultado será como se ve a continuación en la ilustración.



```
pi@raspberrypi: ~  
Archivo Editar Pestañas Ayuda  
pi@raspberrypi:~ $ sudo apt-get upgrade  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Calculando la actualización... Hecho  
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.  
pi@raspberrypi:~ $
```

Ilustración 26. Actualización del gestor de paquetes., Fuente: Propia

Al término de la actualización del gestor de paquetes se procederá a la instalación del apache, donde se ejecutará la siguiente línea de comando **sudo apt-get install apache2 -y**, esta línea de comando permitirá instalar la herramienta de trabajo, así como vemos en la siguiente ilustración.

```
pi@raspberrypi: ~
Archivo  Editar  Pestañas  Ayuda
pi@raspberrypi:~ $ sudo apt-get install apache2 -y
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
apache2-bin apache2-data apache2-utils libapr1 libaprutil1
libaprutil1-dbd-sqlite3 libaprutil1-ldap ssl-cert
Paquetes sugeridos:
apache2-doc apache2-suexec-pristine | apache2-suexec-custom
openssl-blacklist
Se instalarán los siguientes paquetes NUEVOS:
apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1
libaprutil1-dbd-sqlite3 libaprutil1-ldap ssl-cert
0 actualizados, 9 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 1.992 kB de archivos.
Se utilizarán 6.229 kB de espacio de disco adicional después de esta operación.
Des:1 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf libapr1 arm
hf 1.6.5-1 [83,3 kB]
Des:2 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf libaprutil1
armhf 1.6.1-4 [81,7 kB]
Des:3 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf libaprutil1
-dbd-sqlite3 armhf 1.6.1-4 [17,3 kB]
Des:4 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf libaprutil1
-ldap armhf 1.6.1-4 [16,3 kB]
Des:5 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf apache2-bin
armhf 2.4.38-3+deb10u3 [1.121 kB]
Des:6 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf apache2-dat
a all 2.4.38-3+deb10u3 [165 kB]
Des:7 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf apache2-uti
ls armhf 2.4.38-3+deb10u3 [235 kB]
Des:8 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf apache2 arm
hf 2.4.38-3+deb10u3 [251 kB]
Des:9 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf ssl-cert al
l 1.0.39 [20,8 kB]
Descargados 1.992 kB en 9s (230 kB/s)
Preconfigurando paquetes ...
Seleccionando el paquete libapr1:armhf previamente no seleccionado.
(Leyendo la base de datos ... 156557 ficheros o directorios instalados actualmen
te.)
```

Ilustración 27. Instalación del servidor Apache en la RaspBerry,

Fuente: Propia

Una vez finalizada de la instalación se subirá el apache con la siguiente línea de comando **sudo nano apache start**, este comando permitirá iniciar de forma inmediata el apache, además para confirmar si fue correcto el proceso de instalación ejecutaremos el siguiente comando en el símbolo del sistema **sudo nano /var/www/index.html**, o una forma más sencilla de verificación es ingresar al navegador web de la RaspBerry e ingresar en la barra de búsqueda **http://localhost** , y esta te redireccionara a la página principal de apache para rapsbian así como podemos ver en la siguiente ilustración.

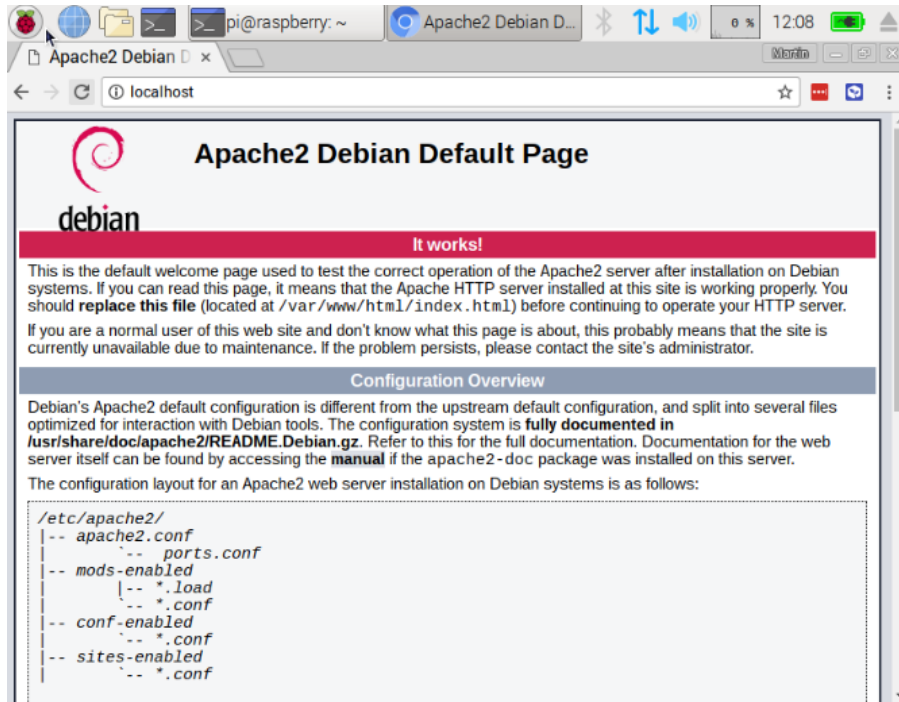


Ilustración 28. Finalización de la instalación del servidor Apache en la RaspBerry,

Fuente: Propia

Finalizando la instalación del apache se modificará el archivo `apache2.conf`, donde se le cambiara unas características que permitirán agregarle una capa de seguridad extra en nuestro servidor. Para esta configuración en el símbolo del sistema ejecutaremos la línea de comando **`sudo nano /etc/apache2/apache2.conf`**, el cual nos redireccionará al archivo `apache2.conf`, y en donde se modificará dos de sus atributos tales como `ServerSignature` y `ServerTokens`, El primero, `ServerSignature` lo pondremos en `Off`, el cual le dice a apache que no muestre la versión del servidor en las páginas de error u otras páginas que genera. El segundo `ServerTokens` lo modificaremos a `Prod`, el cual le dice a apache que solo devuelva Apache en el encabezado del servidor, devuelto en cada solicitud de página. Así como podemos ver en la siguiente ilustración.

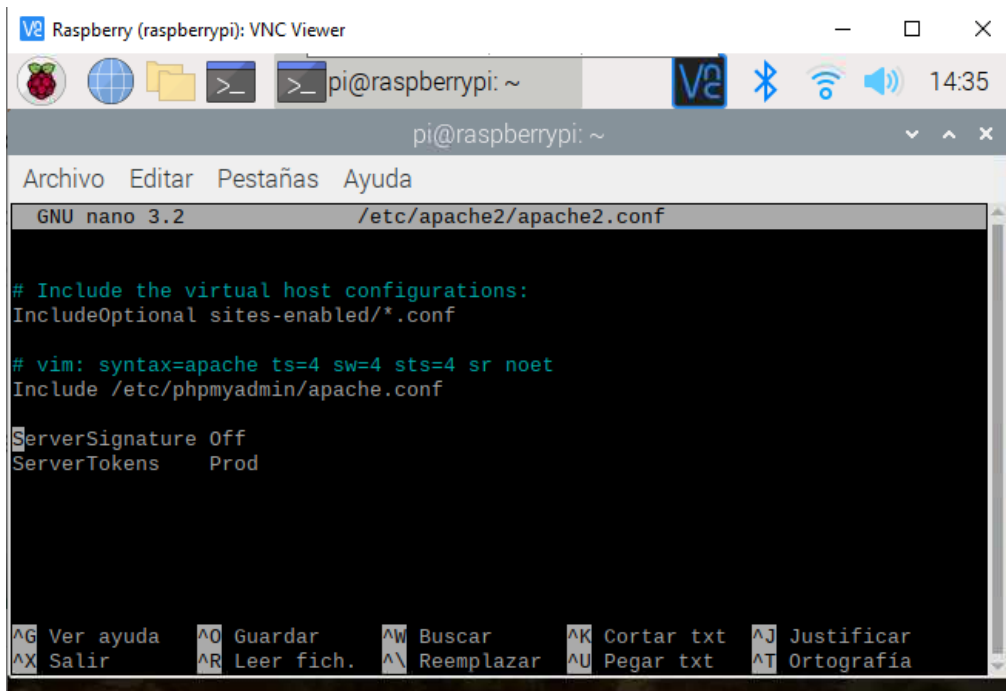
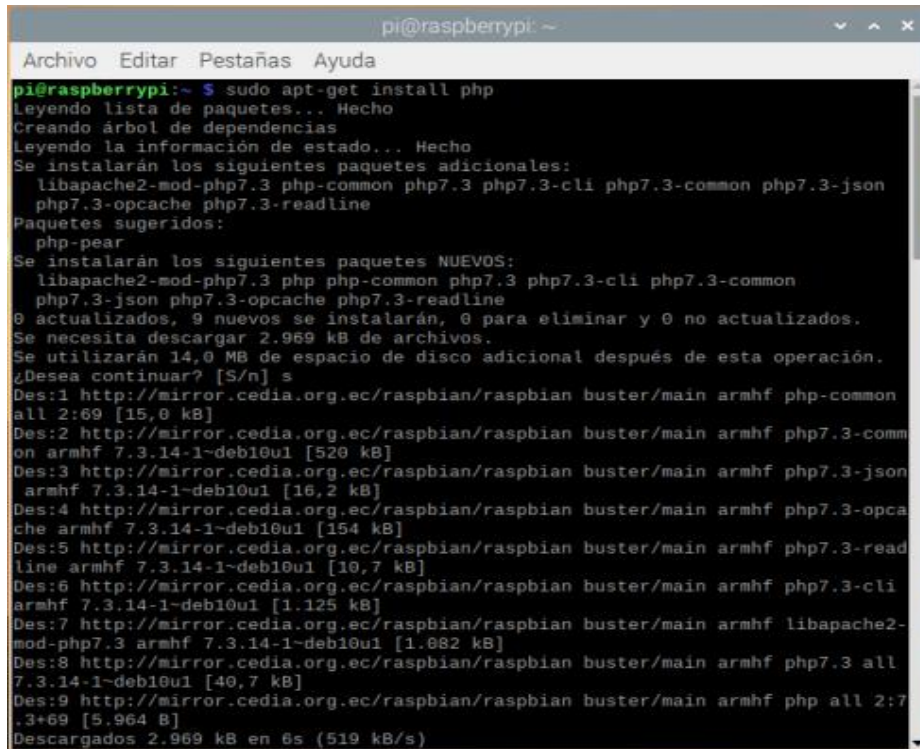


Ilustración 29. Modificación del archivo apache2.conf para agregar una capa de seguridad al servidor,

Fuente: Propia

Otra de las herramientas a utilizar es PHP, para su instalación se utilizó el siguiente comando, **sudo apt-get install php**, como podemos ver en la siguiente ilustración



```
pi@raspberrypi: ~  
Archivo Editar Pestañas Ayuda  
pi@raspberrypi:~$ sudo apt-get install php  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Se instalarán los siguientes paquetes adicionales:  
  libapache2-mod-php7.3 php-common php7.3 php7.3-cli php7.3-common php7.3-json  
  php7.3-openssl php7.3-readline  
Paquetes sugeridos:  
  php-pear  
Se instalarán los siguientes paquetes NUEVOS:  
  libapache2-mod-php7.3 php php-common php7.3 php7.3-cli php7.3-common  
  php7.3-json php7.3-openssl php7.3-readline  
0 actualizados, 9 nuevos se instalarán, 0 para eliminar y 0 no actualizados.  
Se necesita descargar 2.969 kB de archivos.  
Se utilizarán 14,0 MB de espacio de disco adicional después de esta operación.  
¿Desea continuar? [S/n] s  
Des:1 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf php-common  
all 2:69 [15,0 kB]  
Des:2 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf php7.3-comm  
on armhf 7.3.14-1-deb10u1 [520 kB]  
Des:3 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf php7.3-jso  
n armhf 7.3.14-1-deb10u1 [16,2 kB]  
Des:4 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf php7.3-opca  
che armhf 7.3.14-1-deb10u1 [154 kB]  
Des:5 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf php7.3-read  
line armhf 7.3.14-1-deb10u1 [10,7 kB]  
Des:6 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf php7.3-cli  
armhf 7.3.14-1-deb10u1 [1.125 kB]  
Des:7 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf libapache2-  
mod-php7.3 armhf 7.3.14-1-deb10u1 [1.082 kB]  
Des:8 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf php7.3 all  
7.3.14-1-deb10u1 [40,7 kB]  
Des:9 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf php all 2:7  
.3+69 [5.964 B]  
Descargados 2.969 kB en 6s (519 kB/s)
```

Ilustración 30. Instalación del lenguaje PHP en el servidor,

Fuente: Propia

Continuando con la transformación de RaspBerry Pi modelo B+ en un servidor remoto LAMP se procederá a la instalación de la base de datos que para este caso es una versión de MySQL, la cual se le conoce como MariaDB, es una de las bases de datos relacionales de código abierto más populares. Está hecho por los desarrolladores originales de MySQL y se garantiza que seguirá siendo de código abierto. Es parte de la mayoría de las ofertas en la nube y el valor predeterminado en la mayoría de las distribuciones de Linux [25]. Para su instalación ingresaremos el comando **sudo apt-get install MariaDB-server mariadb-client**. Una vez ejecutado el comando pedirá automáticamente la descarga por el tamaño de los paquetes, como podemos ver a continuación en la ilustración.

```
pi@raspberrypi: ~
Archivo Editar Pestañas Ayuda
pi@raspberrypi:~$ sudo apt-get install mariadb-server mariadb-client
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
 galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl
 libconfig-inifiles-perl libdbd-mysql-perl libdbi-perl libencode-locale-perl
 libfcgi-perl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl
 libhttp-date-perl libhttp-message-perl libio-html-perl
 liblwp-mediatypes-perl libmariadb3 libreadline5 libsigsegv2
 libterm-readkey-perl libtimedate-perl liburi-perl mariadb-client-10.3
 mariadb-client-core-10.3 mariadb-common mariadb-server-10.3
 mariadb-server-core-10.3 mysql-common socat
Paquetes sugeridos:
 gawk-doc libclone-perl libmldbm-perl libnet-daemon-perl
 libsql-statement-perl libdata-dump-perl libipc-sharedcache-perl libwww-perl
 mailx mariadb-test tinyca
Se instalarán los siguientes paquetes NUEVOS:
 galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl
 libconfig-inifiles-perl libdbd-mysql-perl libdbi-perl libencode-locale-perl
 libfcgi-perl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl
 libhttp-date-perl libhttp-message-perl libio-html-perl
 liblwp-mediatypes-perl libmariadb3 libreadline5 libsigsegv2
 libterm-readkey-perl libtimedate-perl liburi-perl mariadb-client
 mariadb-client-10.3 mariadb-client-core-10.3 mariadb-common mariadb-server
 mariadb-server-10.3 mariadb-server-core-10.3 mysql-common socat
0 actualizados, 32 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 18,3 MB de archivos.
Se utilizarán 150 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf libsigsegv2 arm
hf 2.12-2 [32,3 kB]
Des:2 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf gawk armhf 1:4.
2.1+dfsg-1 [598 kB]
Des:3 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf mysql-common al
l 5.8+1.0.5 [7.324 B]
Des:4 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf mariadb-common
all 1:10.3.22-0+deb10u1 [31,9 kB]
Des:5 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf galera-3 armhf
25.3.25-2 [811 kB]
Des:6 http://mirror.cedia.org.ec/raspbian/raspbian buster/main armhf libdbi-perl arm
hf 1.642-1+b1 [766 kB]
```

Ilustración 31. Instalación del servidor y el cliente de MariaDB,

Fuente: Propia

Al finalizar la instalación del MySQL server y client, en el símbolo del sistema se redirigirá a la raíz del sistema operativo entrando como super usuario con el comando **sudo -s**, y es aquí donde se ingresará el siguiente comando en el símbolo del sistema **mysql_secure_installation**,

El cual nos permitirá configurar los diferentes usuarios y contraseñas que administraran las bases de datos, además podremos configurar los permisos y privilegios de cada uno de ellos.

```
root@server:~# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

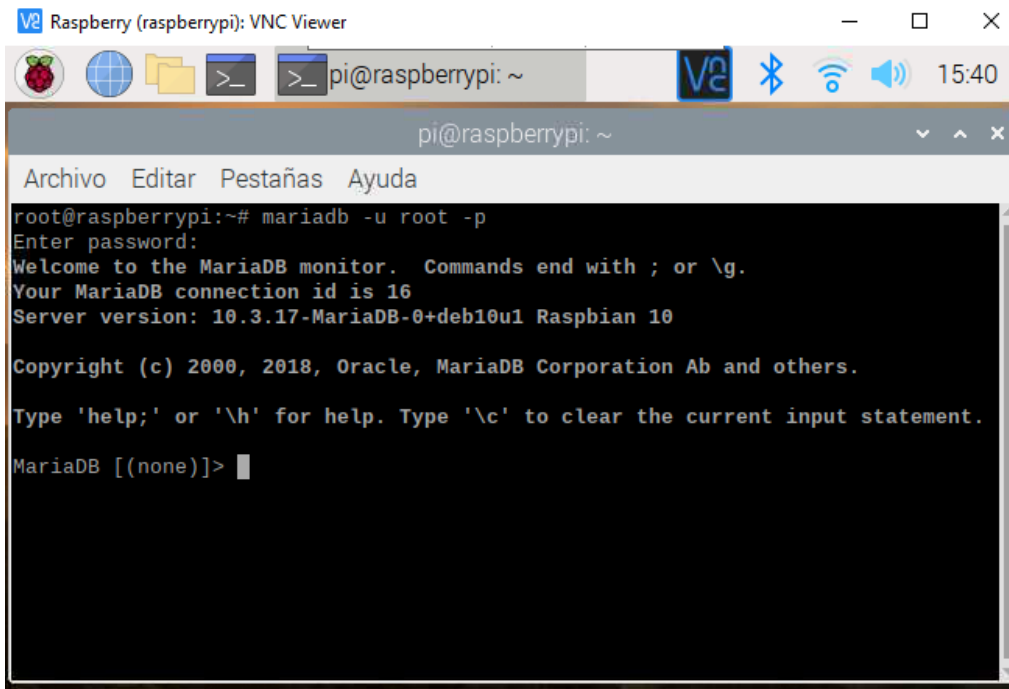
Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] y
New password: |
```

Ilustración 32. Instalación de la base de datos MariaDB en el servidor,

Fuente: Propia

Como podemos ver en la ilustración anterior podremos configurar una capa de seguridad para el ingreso a las diferentes bases de datos que se podrán manejar en MariaDB, en un principio esta contraseña estará en blanco para su posterior cambio.



```
Raspberry (raspberrypi): VNC Viewer
pi@raspberrypi: ~
Archivo Editar Pestañas Ayuda
root@raspberrypi:~# mariadb -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 16
Server version: 10.3.17-MariaDB-0+deb10u1 Raspbian 10

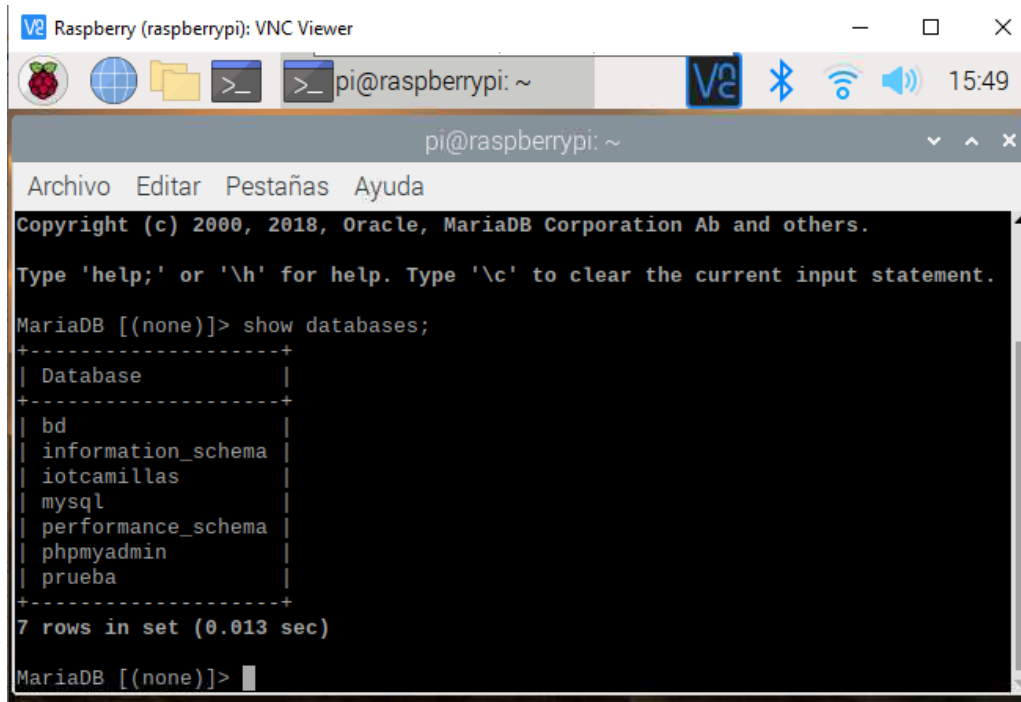
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> |
```

Ilustración 33. comandos para ingresar al gestor de la base de datos MariaDB, Fuente: Propia

Para ingresar a la pantalla principal de MariaDB, es necesario seguir en la raíz del símbolo del sistema y ejecutar el comando **mariadb -u root -p**, así como vemos en la anterior ilustración. De esta manera podremos visualizar, crear, modificar y eliminar bases de datos por medio del símbolo del sistema.



```
pi@raspberrypi: ~
Archivo Editar Pestañas Ayuda
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| bd       |
| information_schema |
| iotcamillas |
| mysql    |
| performance_schema |
| phpmyadmin |
| prueba   |
+-----+
7 rows in set (0.013 sec)
MariaDB [(none)]>
```

Ilustración 34. Pantalla principal para la creación de la base de datos en MariaDB, Fuente: Propia

Para finalizar la configuración del servidor, se instalará otro programa llamado phpMyAdmin, el cual es una herramienta de software gratuita escrita en PHP, destinada a manejar la administración de MySQL en la Web. phpMyAdmin admite una amplia gama de operaciones en MySQL y MariaDB. Las operaciones de uso frecuente (gestión de bases de datos, tablas, columnas, relaciones, índices, usuarios, permisos, etc.) se pueden realizar a través de la interfaz de usuario, mientras aún tiene la capacidad de ejecutar directamente cualquier instrucción SQL. [26]. Así la instalación del phpMyAdmin se hará con la siguiente línea de comando **sudo apt-get install phpmyadmin**.

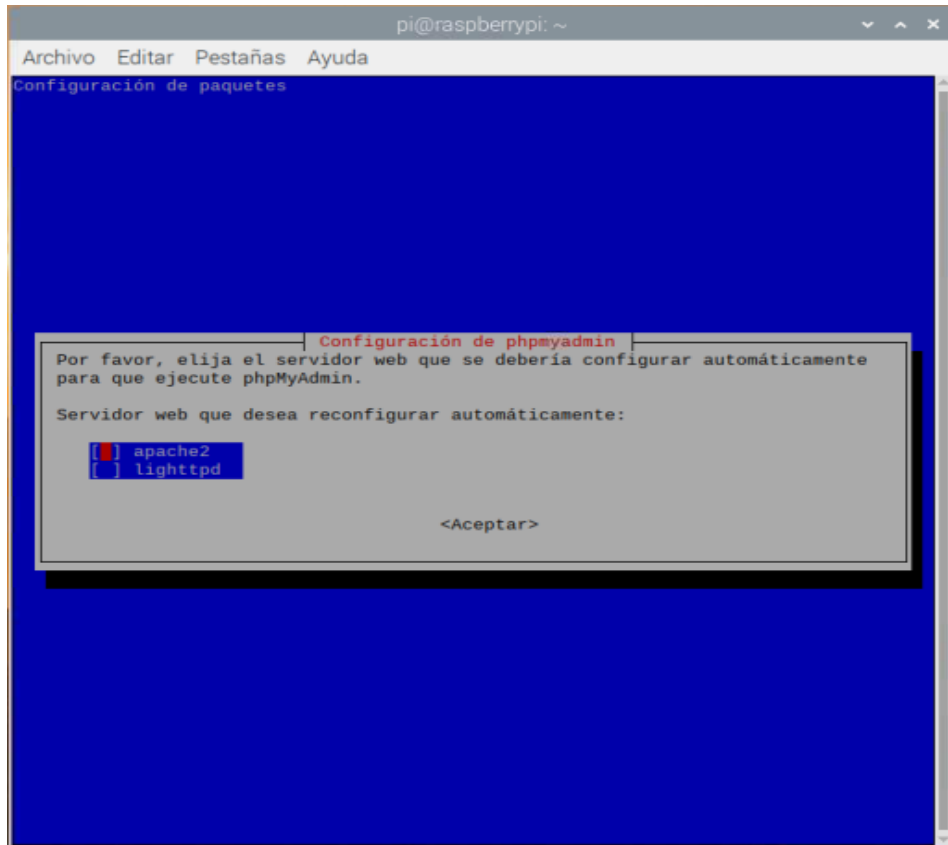


Ilustración 35. Instalación de PhpMyAdmin en el servidor, Fuente: Propia

Al ejecutar el comando anterior, se iniciará la descarga de los paquetes, en donde cierto momento requerirá que seleccionemos un servidor web para su configuración en este caso el servidor web ya está instalado el cual es apache2, como se podrá ver en la ilustración anterior. Escogiendo la opción apache el proceso de instalación y configuración continuará automáticamente.

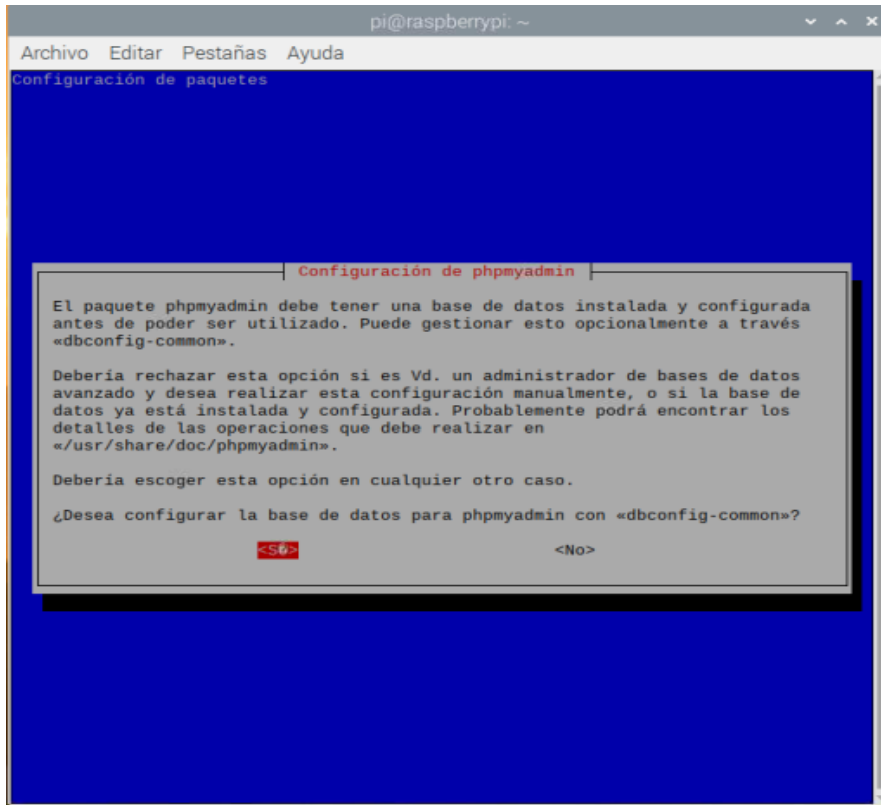


Ilustración 36. Configuración del PhpMyAdmin, Relación entre MariaDB y PhpMyAdmin, Fuente: Propia

Finalmente se podrá realizar una relación entre MariaDB y phpMyAdmin, el cual permitirá administrar las diferentes bases de datos que creamos, para esto la instalación de la herramienta nos proporciona esta interfaz que de una manera sencilla podremos correlacionar estas dos. Continuando con los pasos de la instalación también requiere de una contraseña para que phpMyAdmin se pueda registrar con el servidor de MySQL, como se puede ver en la siguiente ilustración.

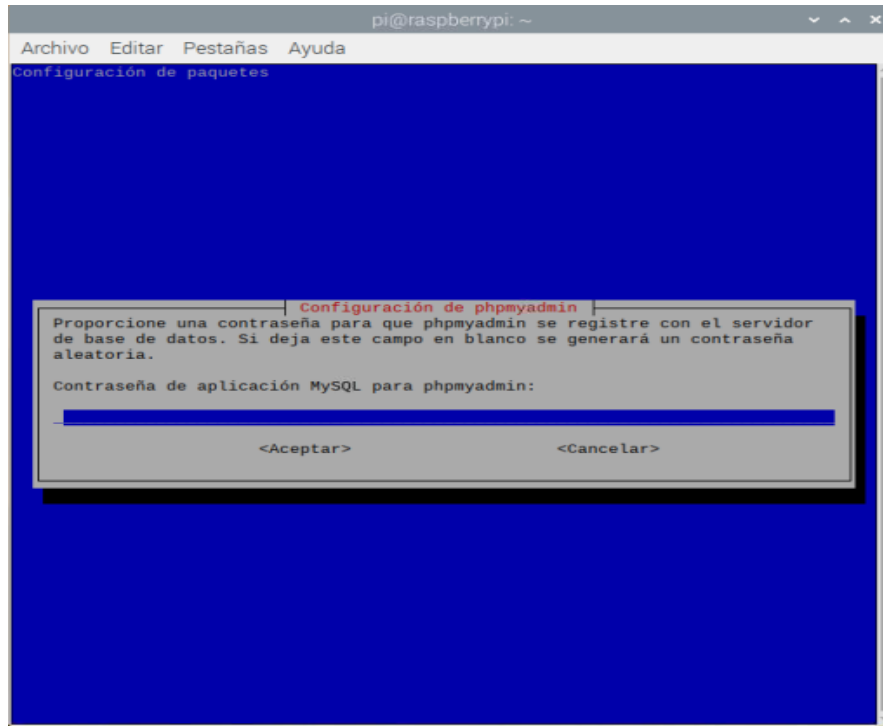


Ilustración 37. Configuración de contraseña para PhpMyAdmin, Fuente_ Propia

Otra de las configuraciones necesarias para el correcto funcionamiento en conjunto del prototipo y el servidor previamente configurado es la asignación de un canal de comunicación bilateral el cual es requerido para él envío de información, para este proceso se implementó una IP estática ya que proporcionara un solo y único medio de iteraciones entre los dos dispositivos.

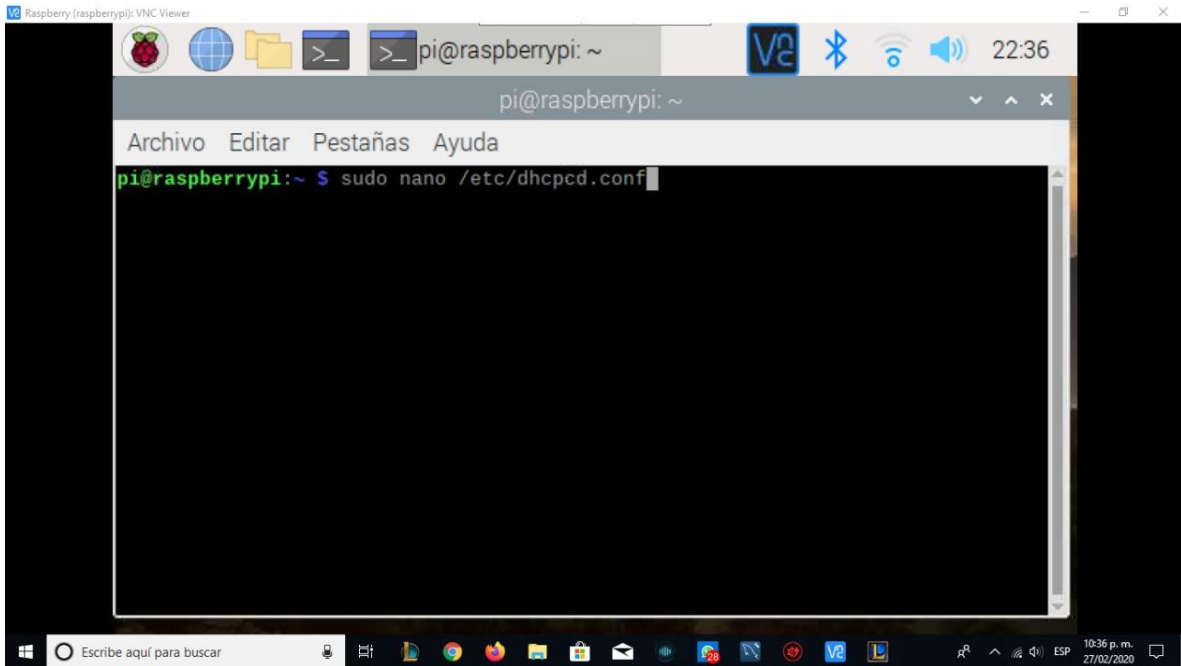
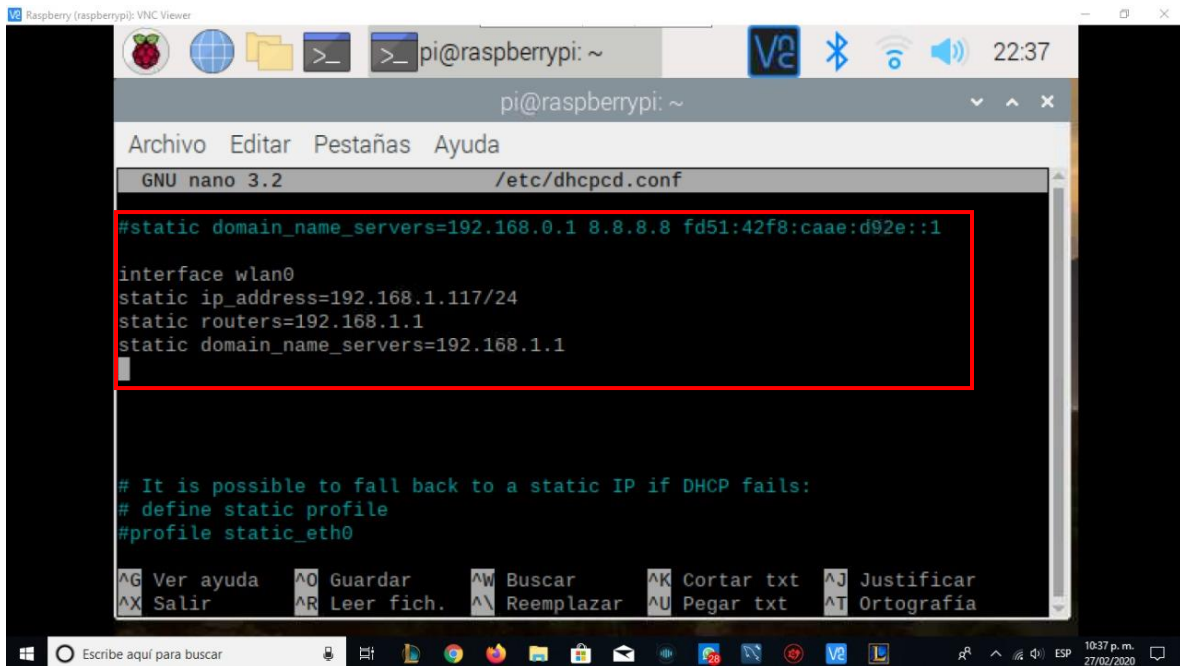


Ilustración 38. Comando para ingresar al fichero dhcpd.conf para la configuración del canal de comunicación, Fuente: Propia

El primer paso para la configuración del canal de comunicación es entrar al símbolo del sistema del sistema operativo Raspbian, y con el comando **sudo nano /etc/dhcpd.conf**, así como se ve en la ilustración anterior.

Al ejecutar el comando anterior rápidamente se redireccionará al fichero dhcpd.conf, el cual se encarga de la asignación dinámica de las direcciones IP para los dispositivos, una vez estando dentro del fichero se dirigirá a la parte que está enmarcada con el nombre **static domain_name_servers**, es aquí donde se gestionara la configuración del canal de comunicación estableciendo la IP estática deseada.



```
pi@raspberrypi: ~  
Archivo Editar Pestañas Ayuda  
GNU nano 3.2 /etc/dhcpd.conf  
#static domain_name_servers=192.168.0.1 8.8.8.8 fd51:42f8:caae:d92e::1  
interface wlan0  
static ip_address=192.168.1.117/24  
static routers=192.168.1.1  
static domain_name_servers=192.168.1.1  
  
# It is possible to fall back to a static IP if DHCP fails:  
# define static profile  
#profile static_eth0  
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar txt ^J Justificar  
^X Salir ^R Leer fich. ^E Reemplazar ^U Pegar txt ^T Ortografía
```

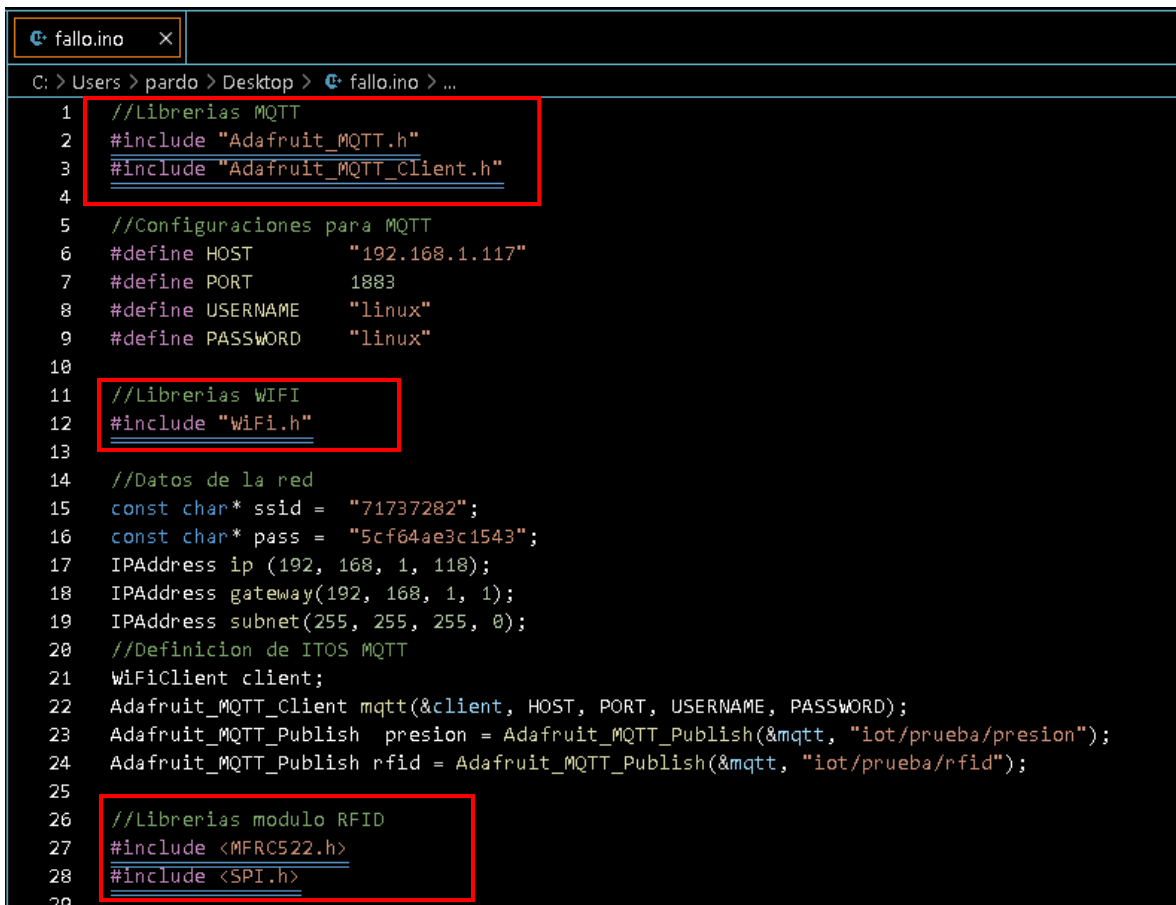
Ilustración 39. Configuración del canal de comunicación mediante una IP estática, Fuente: Propia

Una vez configurado el servidor donde irá toda la información obtenida por el prototipo, empezando por la configuración de los distintos dispositivos que conforman el dispositivo, el principal componente para dar comienzo al desarrollo del proyecto es la placa ESP32, donde se incorporará en ella toda la lógica de programación, que delegará las funcionalidades a los demás modulo, recibirá y enviará de los datos. De esta manera damos por terminado la configuración de cada uno de los componentes software del servidor remoto de nuestro proyecto.

Finalizada la creación y configuración del servidor, se predispone a continuación a la implementación del prototipo con cada uno de los componentes eléctricos, electrónicos y de software, para esto dispondremos de un entorno de desarrollo como lo es el IDE de Arduino que permitirá un espacio de trabajo para el desarrollo de la lógica que involucrara la conexión y el tratamiento de datos entre los dispositivos que conforman el prototipo.

Uno de los primeros elementos software a implementar son las distintas librerías que permitirán el control y modificación del comportamiento de los dispositivos entre

ellos el ESP32 y el RFID. A demás se implementarán también librerías adicionales para las herramientas software a utilizar como lo es el MQTT.



```
fallo.ino x
C: > Users > pardo > Desktop > fallo.ino > ...
1 //Librerías MQTT
2 #include "Adafruit_MQTT.h"
3 #include "Adafruit_MQTT_Client.h"
4
5 //Configuraciones para MQTT
6 #define HOST "192.168.1.117"
7 #define PORT 1883
8 #define USERNAME "linux"
9 #define PASSWORD "linux"
10
11 //Librerías WIFI
12 #include "WiFi.h"
13
14 //Datos de la red
15 const char* ssid = "71737282";
16 const char* pass = "5cf64ae3c1543";
17 IPAddress ip (192, 168, 1, 118);
18 IPAddress gateway(192, 168, 1, 1);
19 IPAddress subnet(255, 255, 255, 0);
20 //Definicion de ITOS MQTT
21 WiFiClient client;
22 Adafruit_MQTT_Client mqtt(&client, HOST, PORT, USERNAME, PASSWORD);
23 Adafruit_MQTT_Publish presion = Adafruit_MQTT_Publish(&mqtt, "iot/prueba/presion");
24 Adafruit_MQTT_Publish rfid = Adafruit_MQTT_Publish(&mqtt, "iot/prueba/rfid");
25
26 //Librerías modulo RFID
27 #include <MFRC522.h>
28 #include <SPI.h>
29
```

Ilustración 40. Librerías implementadas en el prototipo IoT, Fuente: Propia

Como se ve en la ilustración anterior en el primer recuadro se incluyen dos librerías para el MQTT, las cuales permitirán acceder a diferentes funciones para la gestión y configuración de los canales de comunicación que estarán dirigidos hacia al broker para la publicación y suscripción hacia los diferentes tópicos por parte de los clientes. Otras de las librerías a utilizar es la de WIFI, así como se podrá ver en el segundo recuadro de la ilustración anterior, al implementar esta librería se crearán instancias de la red en la que se conectara nuestro prototipo IoT. Continuando con el desarrollo de la lógica, la siguiente librería a implementar pertenece a la de los sensores que incorpora el prototipo, una de ellas es la del RFID y la del Force Sensing Resistor 406 que facilitaran el uso correcto de cada una de las características de estos

componentes, y así obtener los diferentes datos necesarios para el buen funcionamiento del sistema.

Tras desarrollar el funcionamiento lógico programable del prototipo y que gracias esto se puede obtener los diferentes datos, se requiere de una base de datos para el almacenamiento de dicha información, para esto hará uso de una de las herramientas para el desarrollo y modelado de la base de datos llamada MySQL Workbench, la cual proporciona modelado de datos, desarrollo de SQL y herramientas de administración integrales para la configuración del servidor y la administración de usuarios.

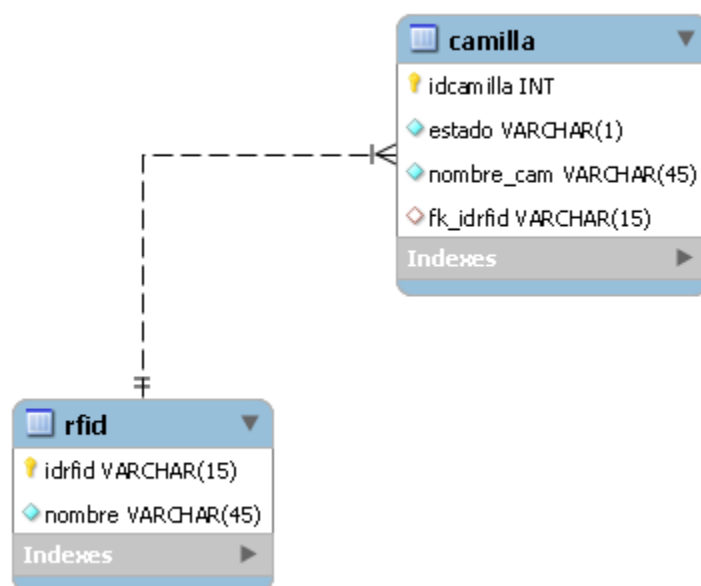


Ilustración 41. Diagrama Entidad Relación de la base de datos del sistema IoT, Fuente: Propia

Como se observa en la anterior ilustración el diseño de la base de datos consta de dos tablas, en la que una de ellas llamada **RFID**, consta de dos atributos, el atributo RFID que es el código alfanumérico de cada una de las tarjetas, el otro corresponde al identificador del personal este identificador es el cargo laboral de los empleados del hospital, la segunda tabla llamada **camilla**, corresponde a la información de las camas que se encuentran ocupadas y libres, y es aquí donde todos los datos convergen dándonos a conocer la situación verdadera del recurso dentro del hospital.

Siguiendo el diseño del sistema, se procede a la implementación del intérprete de las acciones de los diferentes subsistemas que competen el modelo desarrollado y para esto se hará uso de la herramienta ya instalada **Node Red**, ingresando a la IP estática predefinida en la configuración anterior, podremos acceder a la interfaz de trabajo y con cada una de las funciones que se encuentran, definir un modelo de comunicación para la interpretación y procesamiento de los datos.

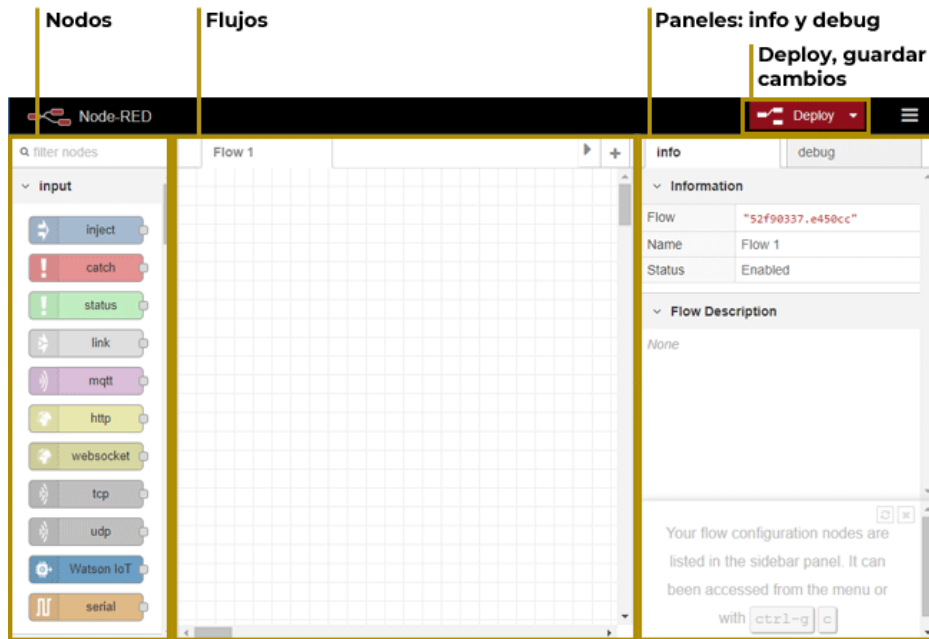


Ilustración 42. Pantalla Principal del framework de NodeRed,

Fuente: <https://aprendiendoarduino.wordpress.com/2018/11/20/node-red/>

En la anterior ilustración se podrá ver los tres módulos que componen la interfaz gráfica de Node Red, en la parte izquierda se encuentran las funciones encapsuladas en nodos, es aquí donde sabiendo el tratamiento que se le quiere dar a los datos, se utilizará el nodo correspondiente para satisfacer esa tarea. En la parte central se encuentra el módulo de trabajo donde se crearán los flujos de datos con los diferentes nodos necesarios para obtener el resultado deseado, y por último encontramos en la parte derecha el módulo de configuración y modificación de los nodos para refinar el resultado individual de cada uno de ellos.

Es así como con el entendido del funcional e integral de la herramienta, de su entorno de trabajo y sus diferentes características que nos proporciona, se creó un diseño

de interpretación y de manejo de datos que nos proporcionara una integración mucho más cómoda y sencilla. Así como se ve en la siguiente imagen.

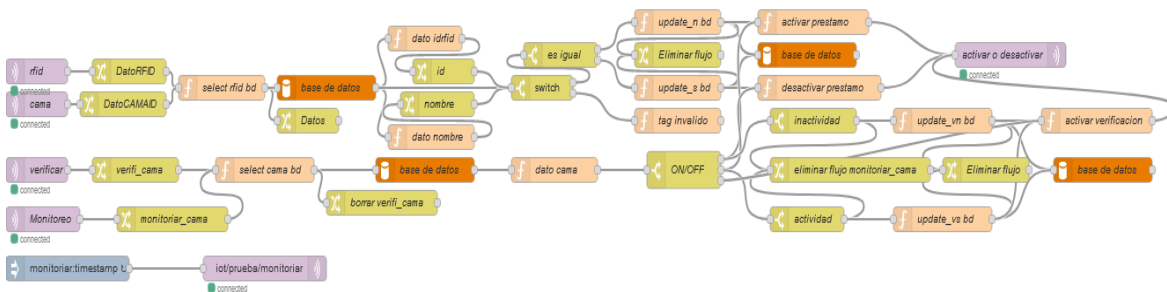


Ilustración 43. Diagrama de flujos del proceso del sistema IoT, Fuente: Propia

En la ilustración anterior, se podrá ver el flujo que permite obtener los datos del prototipo a través de una función que usa una conexión MQTT configurada para que enlace directamente la dirección IP predefinida, estos datos pertenecen a él id de la tarjeta o tag tomada por medio del RFID y a la identificación de la cama o camilla, dicha información es comparada con la ya registrada en la base de datos que esta alojada en el servidor remoto, este procedimiento es realizado con otra de las funciones que contiene NodeRed la cual permite crear consultas SQL para la extracción de información, de esta manera una vez obtenidos los datos de la comparación que determinan si la cama o camilla está ocupada o no, dependiendo del estado de esta el proceso continua con el registro en la base datos.

Además de brindar un manejo de datos para su inserción, modificación y eliminación como también funciones de conectividad, NodeRed proporciona una capa de visualización de la información permitiendo dar conocer al usuario final la situación en el que se encuentra cada cama del hospital, como podemos ver en la siguiente ilustración se crea un flujo para obtener los datos que mostraran al final.

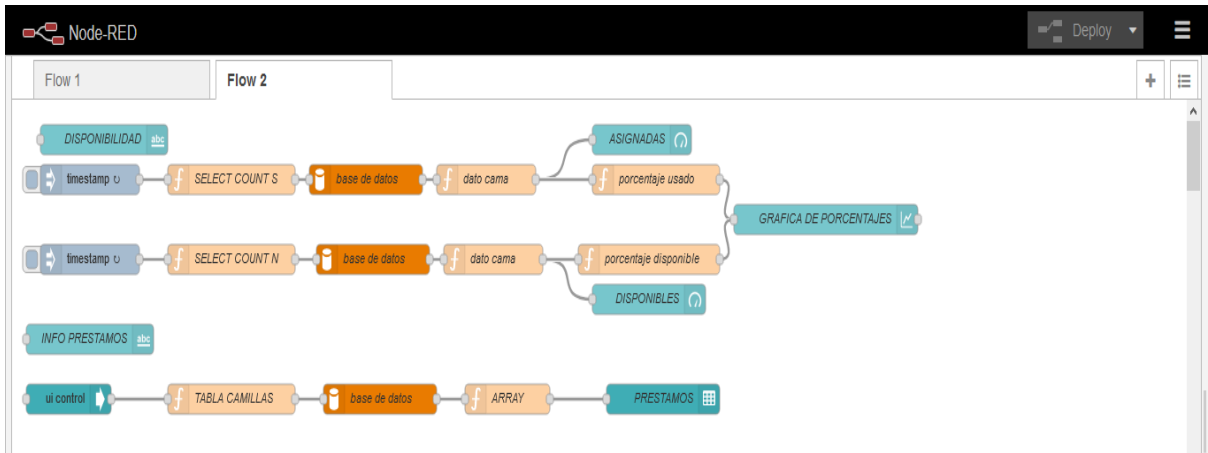


Ilustración 44. Diagrama de flujos para la visualización de los datos registrados, Fuente: Propia

Por medio de una conexión realizada con MQTT al servidor remoto y mediante el uso de consultas SQL se obtiene toda la información requerida desde la base de datos, este proceso mostrará el estado de la cama, el lugar donde pertenece la cama y su identificación, de esta manera el personal del hospital se dará cuenta de forma inmediata en que condición se encuentra la cama o camilla del centro hospitalario. Así como se podrá ver en la siguiente ilustración la interfaz gráfica de visualización de datos

The screenshot shows a web interface titled 'INFO PRESTAMOS' for 'INSTITUCIÓN UNIVERSITARIA COLEGIO MAYOR DEL CAUCA'. It features a sidebar with navigation options: 'UNIMAYOR', 'GRAFICAS DE PRESTAMOS', and 'INFO PRESTAMOS'. The main content area displays a table with the following data:

Id Cama	Estado	Nombre
11	DISPONIBLE	cama 2 pediatria
14	DISPONIBLE	cama 2 emergencias
10	DISPONIBLE	cama 3 pediatria
16	DISPONIBLE	cama 4 emergencias
17	ASIGNADO	cama 5 emergencias
18	ASIGNADO	cama 6 emergencias
19	ASIGNADO	cama 7 emergencias
13	ASIGNADO	cama 1 emergencias
12	ASIGNADO	cama 5 pediatria
15	ASIGNADO	cama 3 emergencias
1	ASIGNADO	cama 1 recuperacion
9	ASIGNADO	cama 4 pediatria
8	ASIGNADO	cama 6 recuperacion

Ilustración 45. interfaz gráfica del sistema IoT con los datos de la camillas o camas, Fuente: Propia



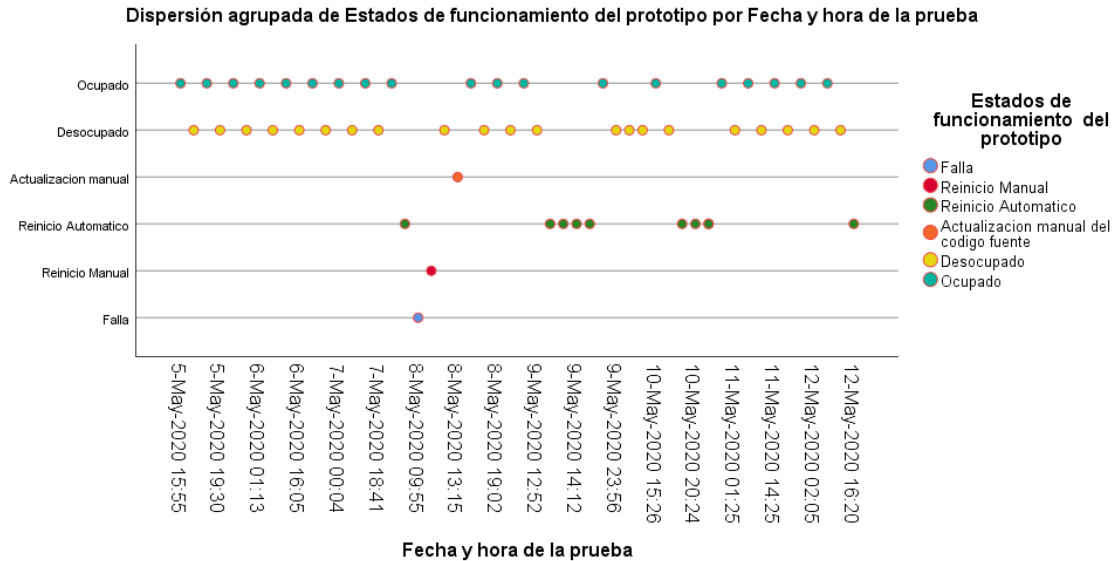
Ilustración 46. Estadística de ocupación de las camas o camillas

CAPITULO 4. VERIFICACIÓN DEL SISTEMA

Para verificar funcionalidades específicas provistas del prototipo, requirió de siete días de pruebas, para ello se realizaron registros manuales en una base de datos, creada, gestionada y analizada, con las herramientas **IBM SPSS** y **Microsoft Excel**. Proceso el cual permitió validar la conectividad a la red inalámbrica y la comunicación con el servidor, para la actualización de datos, así mismo se comprobó los estados de comportamiento del prototipo al mantenerse en funcionamiento constante.

4.1 PRUEBAS DE FUNCIONAMIENTO DEL PROTOTIPO

En esta fase, se trabajó el comportamiento de los estados de funcionamiento del dispositivo durante siete días, realizando registros consecutivos manuales. Obteniendo un total de 53 datos que se ven reflejados en la **Gráfica 1**.



Gráfica 1. Estados de funcionamiento del prototipo,

Fuente: Propia

En la **Gráfica 1** observamos el funcionamiento del prototipo, en la cual es importante denotar los términos:

- **Ocupado y desocupado:** Hace referencia a los registros que realiza en la base de datos.
- **Actualización manual:** Intervención para actualizar el código fuente del prototipo.
- **Reinicio automático:** Es la acción que realiza el prototipo al tener perdida de conexión al servidor o red inalámbrica.
- **Reinicio manual:** Estado en el cual el prototipo requirió intervención humana para reiniciarse.
- **Falla:** el prototipo deja de prestar su servicio, es decir pierde la conectividad a la red inalámbrica y comunicación con el servidor.

En esta gráfica se muestra el comportamiento del prototipo durante los siete días, los cuales corresponden desde el 5 al 12 de mayo de 2020, arrojando lo siguiente:

1. Durante los días 5, 6 y 7 se da un registro estable en la base de datos con una recuperación automática a la conexión de red inalámbrica y comunicación con el servidor.

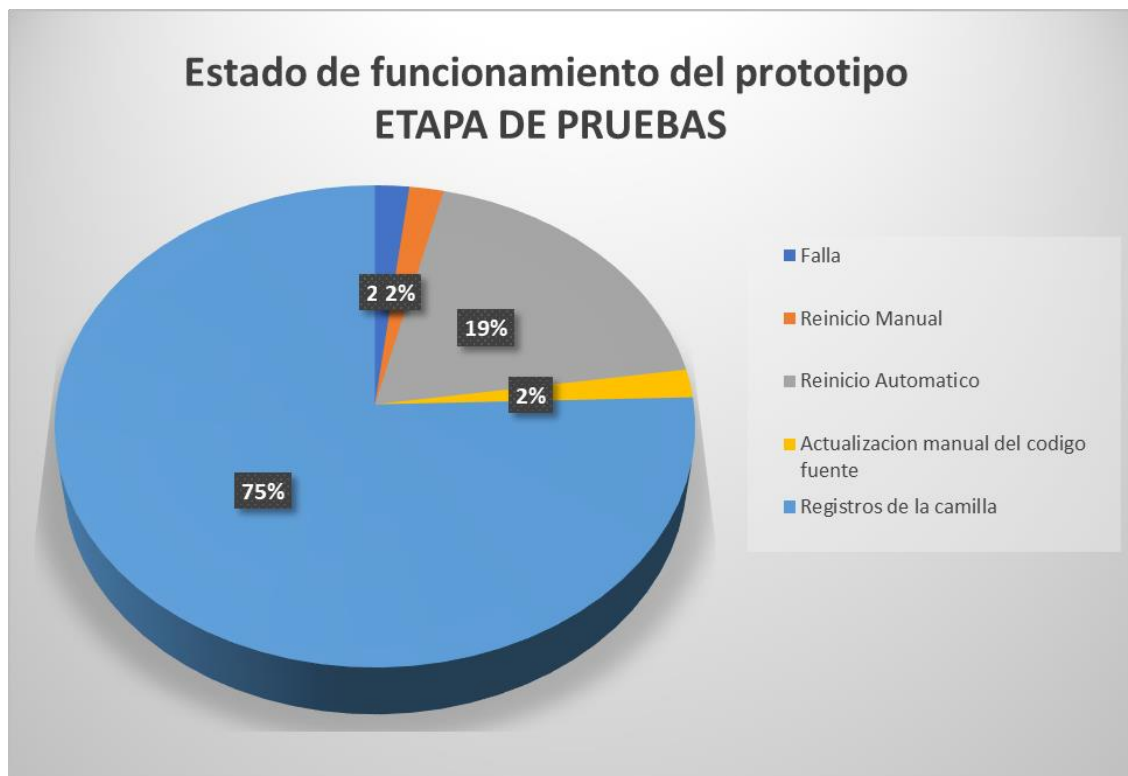
2. El día 8 a las 9:55 a. m. se presentó falla de conexión a la red imposibilitando envío de datos, por lo tanto, se realizó test de acceso y conexión al servidor y prototipo, del cual solo responde el servidor por lo tanto hubo la necesidad de intervención física en el reinicio del prototipo. A raíz de ello se trabajó en el código fuente, lo cual permitió que el dispositivo se reinicie automáticamente ante un fallo de conexión de red inalámbrica o servidor.

```
void loop() {  
    client.loop();  
  
    if (client.isConnected()) {  
        suscripcion();  
    }  
  
    if (millis() > tiempo + periodo) {  
        //retorna si está conectado a la red WiFi y servidor MQTT .  
        if (!client.isConnected()) {  
            led_ON ();  
            //Se reinicia el prototipo  
            ESP.restart();  
        }  
    }  
}
```

Ilustración 47. Código fuente. Reinicio automático del prototipo por falla de la conexión a la red,

Fuente: Propia

3. Después de actualizar el dispositivo el 8 de mayo a la 1:15 pm, en los días restantes hasta el 12 de mayo día final de las pruebas, se da una comunicación estable con el servidor y recuperación automática ante los fallos de conexión inalámbrica.



Gráfica 2. Porcentaje de Estados del prototipo en prueba,

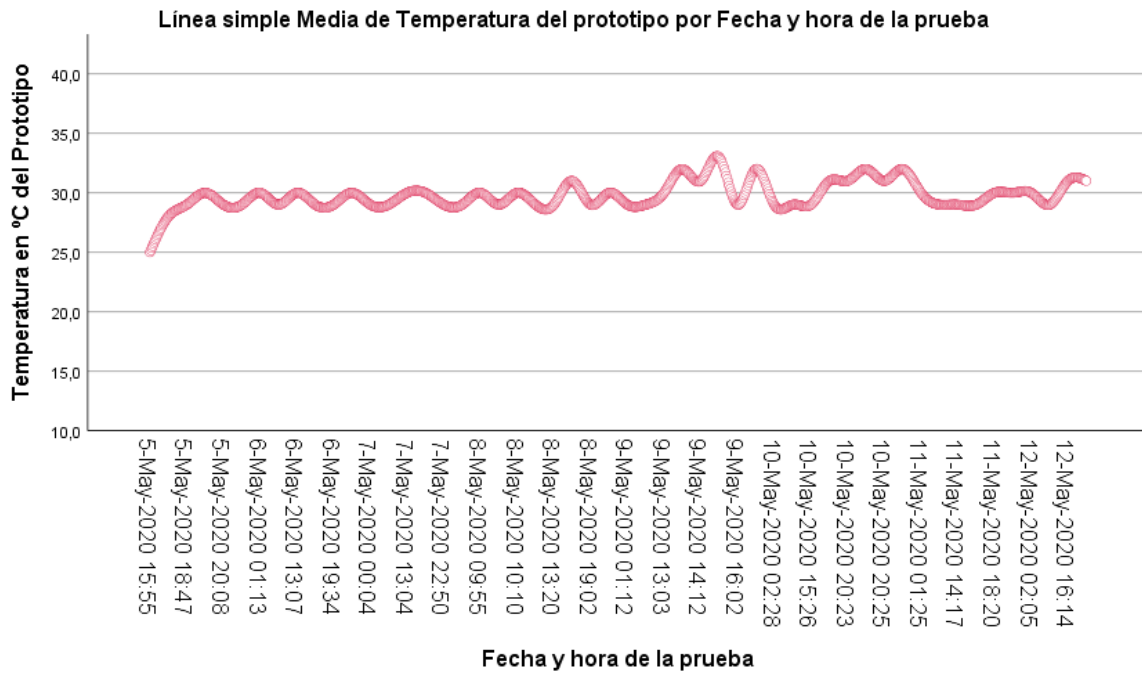
Fuente: Propia

El **19%** del estado **reinicio automático** refleja la recuperación del prototipo para mantener una comunicación con el servidor, estableciendo un canal para el envío de información y registro en la base de datos, reflejado en la funcionalidad del prototipo, condescendiendo en un **75%** de operatividad para realizar los registros de las camas o camillas. Comprobando un comportamiento adecuado del dispositivo, verificando y validando con éxito la actualización del código fuente para el manejo de fallos ante desconexiones de red.

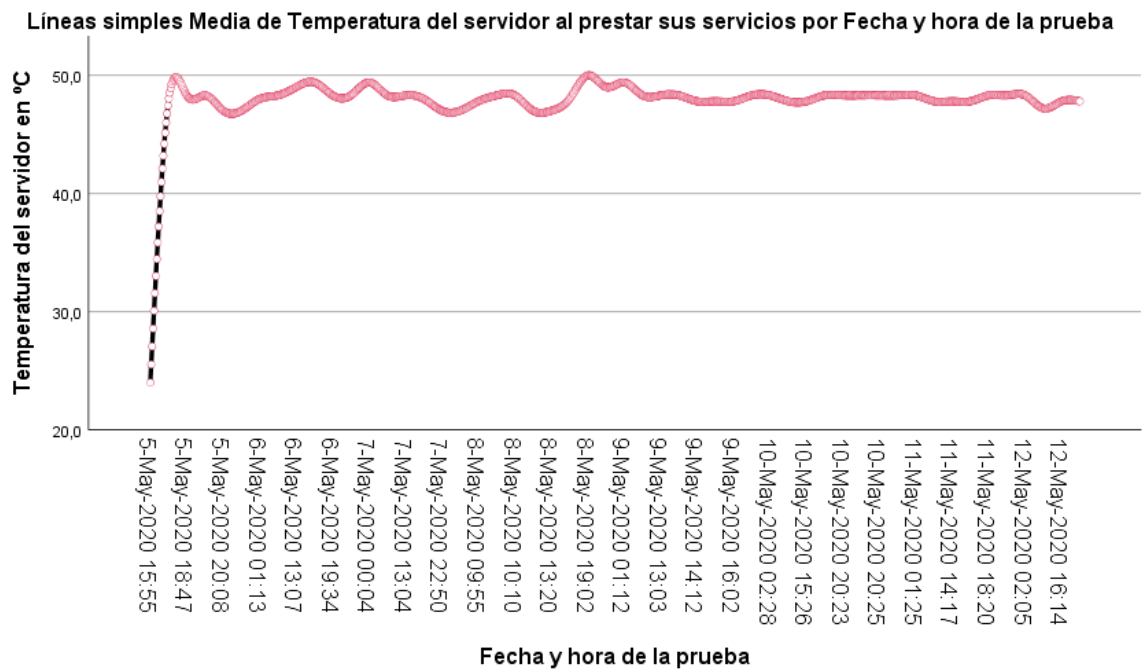
4.2 PRUEBAS DE TEMPERATURA

En esta etapa, se trabajó el comportamiento de la temperatura del prototipo y servidor LAMP, para ello se implementaron siete días, en los cuales se realizaron registros consecutivos y toma de la temperatura de forma manual con un multímetro y termocupla de tipo K, instrumento eléctrico portátil para medir directamente magnitudes eléctricas activas, como corrientes y potenciales, o pasivas, como

resistencias, capacidades y otras. Obteniendo un total de 53 datos que se ven reflejados en la **Gráfica 3** y **Gráfica 4**.

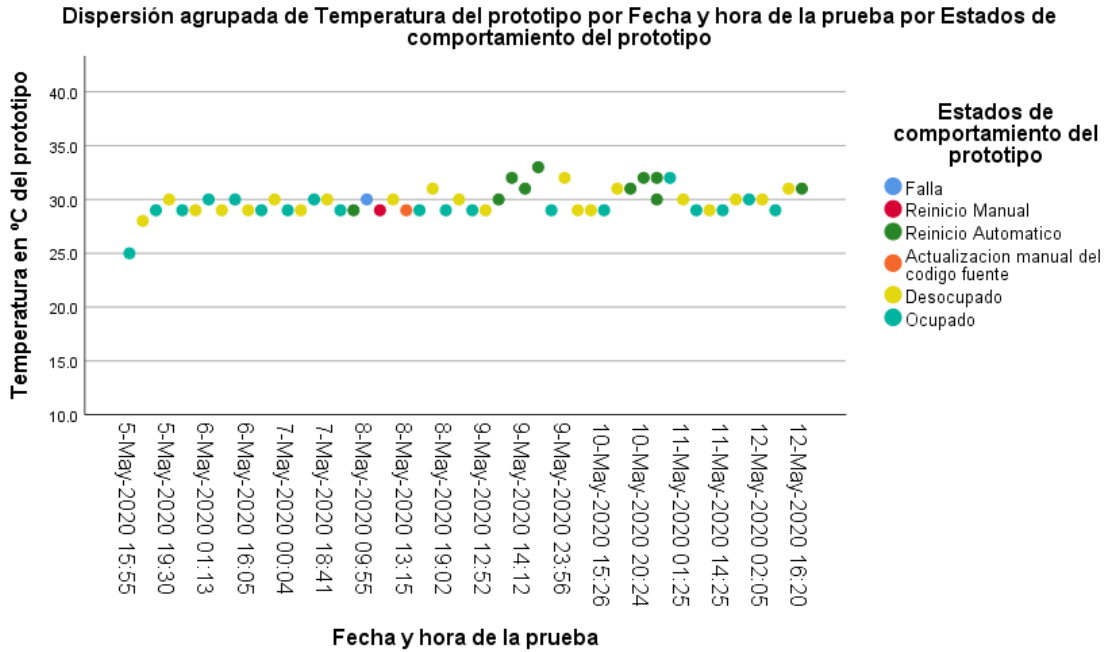


*Gráfica 3. Temperatura del Prototipo,
Fuente: Propia*



*Gráfica 4. Temperatura del Servidor,
Fuente: Propia*

En las pruebas se estudió el comportamiento de la temperatura del prototipo y el servidor con base al tiempo, identificando que los dispositivos trabajan eficientemente y no serán afectados drásticamente por los cambios de temperatura gracias al poco consumo de hardware, ya que según especificaciones técnicas, la placa SBC ESP32-WROOM-32 trabaja de -40 °C a +85 °C de temperatura y la **Gráfica 3** muestra un comportamiento del prototipo entre los +29°C a +30°C arrojando como pico más alto los 33°C, por el lado del servidor tenemos que el rango de temperatura de funcionamiento de la Raspberry Pi 3 Modelo B+ es de 0°C a +70°C, por otro lado en **la Gráfica 4**, se observa que el servidor tiende estabilizarse entre los +47° a +48°C, reflejando como pico más alto los +50°C. Al cotejar los datos técnicos de los dispositivos, con la información de las gráficas se establece que la temperatura y funcionamiento son adecuados.

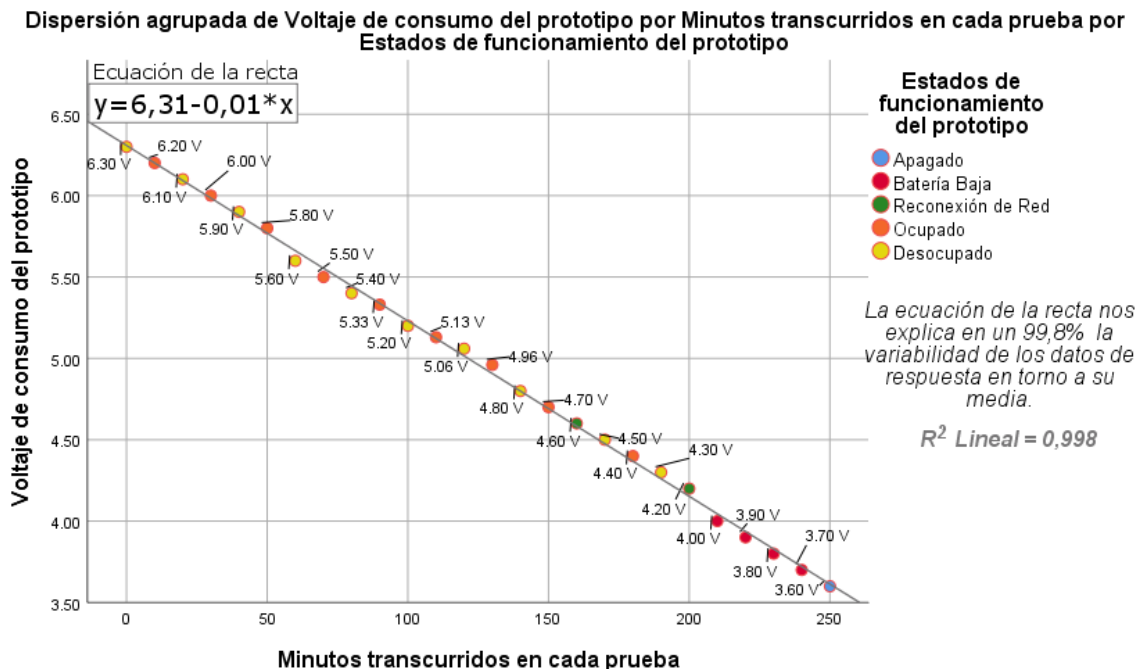


Gráfica 5. Temperatura y estados del prototipo

Con la **Gráfica 5** se pretende comprobar si los estados de comportamiento del prototipo afectan la temperatura por el tiempo de prueba. En la gráfica podemos analizar que los puntos más altos de temperatura son de +31°C a +33°C y están relacionados con el estado **reinicio automático**, al cotejar con los estados restantes que oscilan entre +29°C a +30°C, nos indica que es uno de los procesos con más consumo de hardware ya que involucra reconectar el prototipo a la red inalámbrica y al servidor. Verificando que los estados de comportamiento del prototipo afectan directamente la temperatura.

4.3 PRUEBAS DE CONSUMO ENERGETICO

En esta etapa, se trabajó en el comportamiento del consumo energético del prototipo, para ello se implementó una alimentación DC (Direct Current) de 6.3 Volts en el transcurso de 240 minutos, en la cual, se realizaron registros consecutivos y toma del voltaje de forma manual con un multímetro, instrumento eléctrico portátil para medir directamente magnitudes eléctricas activas, como corrientes y potenciales, o pasivas, como resistencias, capacidades y otras. Obteniendo un total de 26 datos que se ven reflejados en la **Gráfica 6**.



Gráfica 6. Consumo Energético

En la **Gráfica 6** es importante resaltar los términos:

- **Ocupado y desocupado:** Hace referencia a los registros que realiza en la base de datos.
- **Reconexión de red:** Es la acción que realiza el prototipo al tener pérdida de conexión al servidor o red inalámbrica.
- **Batería baja:** Alerta que genera el prototipo, por medio de la intermitencia de leds, informando de los bajos niveles de energía.
- **Apagado:** Es el estado de inanición del prototipo por falta de alimentación energética.

La **Gráfica 6** muestra que el nivel energético afecta directamente al prototipo en su alcance de conectividad inalámbrica, ya que, a menor voltaje, se presenta más estados de **reconexión de red**, revelando que su funcionamiento estable es aproximadamente hasta los 4.5 Volts, punto donde el dispositivo brinda sus servicios y puede actuar ante fallos de conectividad.

Así mismo al hallar la pendiente en la gráfica y su ecuación, para comprender su comportamiento lineal a través del tiempo, se determina que el consumo energético por minuto del prototipo es de **10.8 mV** (Millivolts) por **minuto**, disminuyendo el voltaje de la alimentación DC (Direct Current) en **0.64 Volts** cada **60 minutos** de funcionamiento del prototipo.

CAPITULO 5. MANTENIMIENTO DEL SISTEMA

En este capítulo se brindará unas breves pautas para el mantenimiento del prototipo, aliviando el manejo en si del dispositivo por parte del personal médico, administrativo, sin embargo, se contara con todas las garantías posibles para que este no presente ninguna anomalía o fallas de funcionamiento en ninguna circunstancia de uso.

El sistema está conformado con tecnología de fácil configuración y manipulación ya sea el servidor o el mismo prototipo, no requerirá de una intervención por parte del plantel hospitalario para su modificación estructural interna o de software.

Una de las situaciones que se deberían tener en cuenta es a la hora en el que la cama o camilla sea retirada para el mantenimiento o limpieza requerida para el uso de los pacientes, se debería considerar el sensor de presión, ya que su ubicación se encuentra en la parte media del colchón o en la parte alta debajo de la almohada. Como podemos ver en la siguiente ilustración.

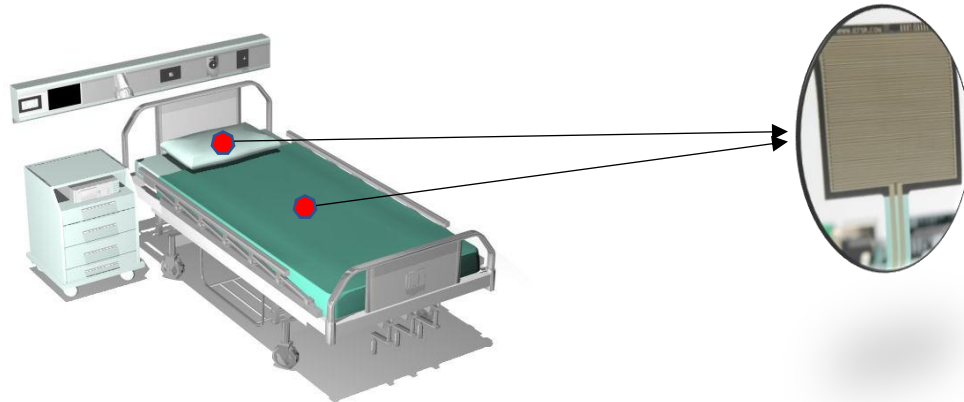


Ilustración 48. Posicionamiento del sensor de presión FSR406,

Fuente ilustración camilla :

https://www.dimensioncad.com/view_category.php?preferred_language=sp&category_number=10&subcategory_number=9&diagram_id=20009415&pageno=1

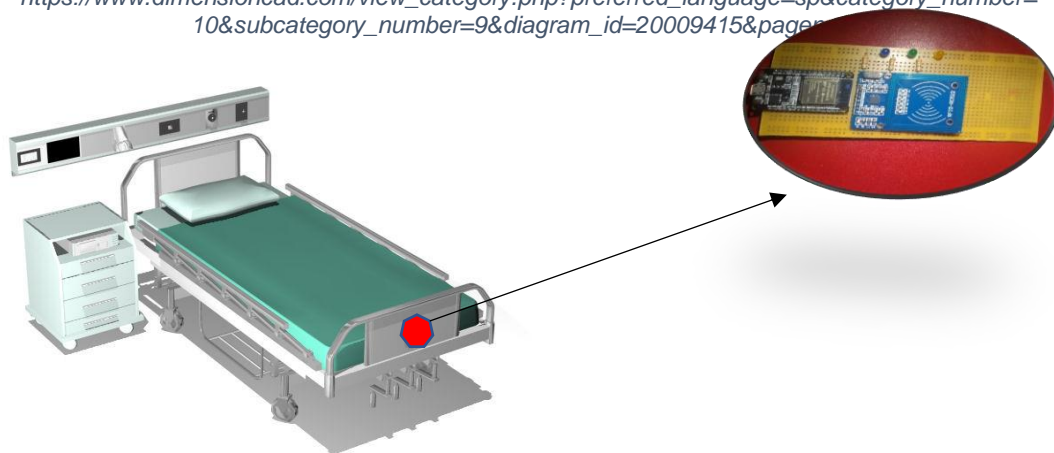


Ilustración 49. Posicionamiento del prototipo en la camilla,

Fuente ilustración camilla :

https://www.dimensioncad.com/view_category.php?preferred_language=sp&category_number=10&subcategory_number=9&diagram_id=20009415&pageno=1

Frente a la manipulación incorrecta del sensor y a su eventual desconexión del prototipo este no podrá activarse y funcionar correctamente. En esta situación las acciones a seguir son las siguientes

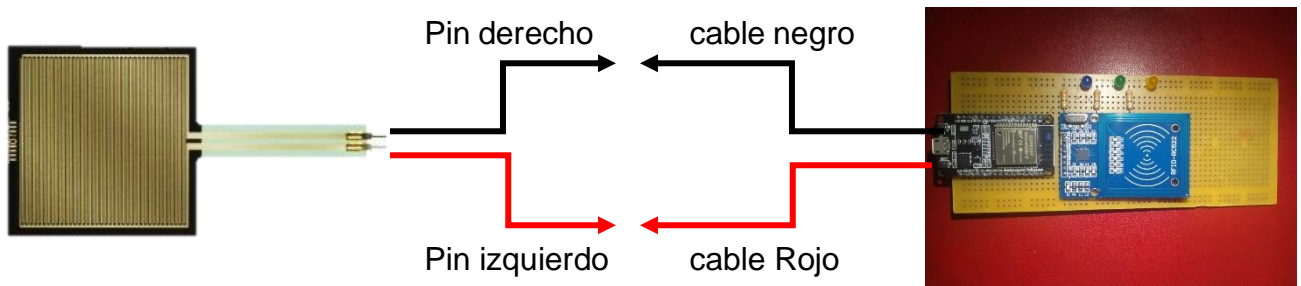


Ilustración 50. Reconexión del sensor FSR406 al prototipo

El prototipo cuenta con un cable, uno de color rojo y otro de color negro, el primero corresponde a la alimentación eléctrica de 5v del sensor fsr406 y el segundo hace referencia a la conexión a tierra, por otro lado, encontramos el sensor FSR406 que en su parte inferior tiene dos pines metálicos por los cuales se puede activar su funcionamiento. Como podemos ver en la ilustración anterior la solución la reconexión es muy sencilla solamente tomar el cable negro y unirlo de al pin derecho del sensor de presión de fuerza como también el pin izquierdo se unirá al cable rojo de esta manera y sin más ajustes él prototipo quedará listo para su uso.

Otro de las situaciones que se puede presentar a medida de que el prototipo sea utilizado es el agotamiento de la batería portátil que este tiene, lo cual puede llegar a afectar el buen funcionamiento del sistema de gestión de las camas o camillas, no obstante el dispositivo tiene una forma de notificación al usuario para cuando este se este quedando sin su fuente de alimentación eléctricas, la cual consiste en, de forma constante los LED'S estarán titilando demostrado la baja entrada de energía.

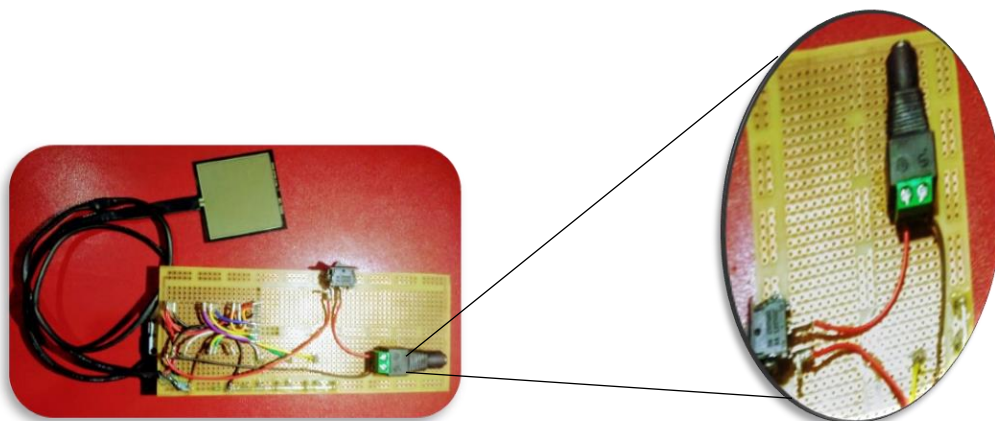


Ilustración 51. Adaptador de energía del prototipo

Para dar solución al agotamiento de la batería y que el dispositivo no falle a la hora de su uso, primero el remplazo inmediato de las baterías, en el caso de que no halla baterías disponibles, el prototipo viene equipado para que este se le pueda conectar un adaptador de 5v para conectar directamente a la red de energía de la institución hospitalaria.

Finalmente, se podrá presentar otra situación que puede que el dispositivo presente un fallo en su funcionamiento, este se trata de la no conexión a la red. El fallo podrá evidenciarse por medio de los LED's de notificación que el prototipo tiene, estos estarán titilando cada 8 segundos demostrando que no se ha podido conectar adecuadamente a la red, por ende, no se podrá cambiar o modificar el estado de la cama o camilla y no se actualizaría esta información en el servidor.

Para solucionar este problema solamente se deberá de reiniciar el dispositivo hasta que este automáticamente se conecte, el usuario se dará cuenta de la conexión si los LED's de notificación dejan de titilar. El reinicio se podrá efectuar fácilmente gracia al switch que el prototipo cuenta, este está ubicado a un costado de él. Así como lo podemos ver en la siguiente ilustración.



Ilustración 52. Switch de encendido y apagado del prototipo

sin embargo, prototipo contará con la calidad y resistencia para soportar su uso diario o eventos extraordinarios que se presente en transcurso del día en el centro hospitalario.

CAPITULO 6. CONCLUSIONES

Frente a la demanda del servicio de atención hospitalaria por parte de la población aumante, el personal médico actuara de manera pertinente a cada situación que se les presente, sin embargo, para dichas acciones se es necesario tener a disposición los elementos o recursos para brindar este servició. Dado esto, después de haber analizado el proceso de la gestión de las camas o camillas dentro de la institución hospitalaria y permitiendo conocer el estado actual en el que se encuentran los diferentes hospitales públicos de la región. Permite ver que el manejo que se lleva a cabo en estos días de un control no automatizado no es el más adecuado para que se brinde un servicio tan importante como el de la salud, generando errores de atención y administración en las instituciones, dentro de estas ideas la implementación de tecnología permite facilitar las tareas diarias en cualquier contexto. Es así como se llegó a la implantación de un prototipo que permitirá la gestión del recurso de las camas o camillas con base en tecnologías como la IoT, favoreciendo en la eficiencia y eficacia de la prestación de este servicio que como

para todos es de gran importancia y como derecho tener acceso a este de forma adecuada y rápida para el beneficio de la población de la región.

Por otro lado es indiscutible que la mala situación económica del país y por ende el sector público tiende a decaer ya que a este no le llega una inyección monetaria adecuada para suplir sus necesidades internas tanto para sus empleados como para adquirir recursos hospitalarios, y es aquí que cuando se ve afectada todo un sistema que a pesar de los esfuerzos realizados por la misma institución no son lo suficiente para poder brindar desde lo más mínimo como una consulta a una hospitalización, ya que se ven obligados a trabajar con los recursos insuficientes adquiridos durante años para atender a una sociedad en crecimiento día tras día. Dado esto, una las maneras confiables para hacer surtir de forma eficiente lo poco de los recursos que se tiene como por ejemplo las camas o camillas, es teniendo un control preciso, seguro y en tiempo real de cómo se está utilizando estos insumos y esto llega de la mano de la tecnología y sus aplicaciones que gracias a ella permiten automatizar la gestión y lograr con tan poco grandes cambios.

Bajo las consideraciones anteriores, es importante optimizar la prestación de los servicios de salud, es decir, fortalecer los procesos internos mediante tecnologías de bajo costo, ágil implementación y rápido desarrollo. De ahí la importancia de este proyecto el cual permitió establecer los siguientes epílogos:

- Se estableció que el prototipo IoT basado en SBC para la gestión de camas o camillas puede prestar un servicio adecuado, ya que permite la conexión, comunicación, lectura y envío de datos al servidor, lo cual es evidenciado en la IU (Interfaz de Usuario). Fundamentado en los siguientes hitos:
- El dispositivo presenta un **75%** de operatividad para realizar los registros de las camas o camillas, sin presencia de errores en la conexión de red, por otro lado, refleja una recuperación **automática** del **19%** ante fallos de red o comunicación al servidor.

- En las pruebas de temperatura del prototipo con base al tiempo y uso, se reflejó que el desplazamiento térmico se encuentra entre los **+29°C a +30°C**, que al ser comparados con los niveles técnicos de especificaciones que va desde los **-40 °C a +85 °C**, la temperatura de funcionamiento del sistema son de buen rendimiento para su funcionalidad.

- Se reflejo que el dispositivo debe tener una alimentación DC (Direct Current) de **6.3 Volts** nominal con un mínimo de operación de **4,5 Volts** ya que su conectividad inalámbrica es afectada directamente por el nivel energético.

- Se comprobó que el dispositivo realiza poco gasto energético, ya que se obtuvo como resultado un consumo de **10.8 mV** (Millivolts) por **minuto**, lo que equivale a **0.64 V** (Volts) por **hora**.

- Se evidenció que es de gran importancia que se implemente un prototipo que permitirá la gestión del recurso de las camas o camillas con base en tecnologías como la IoT, favoreciendo en la eficiencia y eficacia de la prestación de este servicio a la población de la región.

- Es de suma importancia adquirir recursos tecnológicos de ágil implementación, que permitan tener un control preciso, seguro y en tiempo real del uso de las camas o camillas, esto con el fin de automatizar la gestión y lograr un mayor aprovechamiento en la prestación del servicio.

- La experiencia adquirida en este trabajo de grado nos permitió poner en práctica lo aprendido en todo el transcurso de nuestra carrera profesional, así mismo, posibilitar un entregable que genere un beneficio a las personas que lo necesitan. Finalmente, este trabajo nos condesciende a crecer, no solo profesionalmente si no personalmente en cuanto a la implementación y utilidad de la tecnología IoT como apoyo en el desarrollo del prototipo.

REFERENCIAS

- [1] L. E. 1. D. 2015, «Secretaria del senado de Colombia,» 16 febrero 2015. [En línea]. Available: http://www.secretariasenado.gov.co/senado/basedoc/ley_1751_2015.html. [Último acceso: 14 agosto 2019].
- [2] C. J. C. D, C. J, M. D y T. L., «An IoT approach for Wireless Sensor Networks,» de *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom)*, Bogota, Colombia, 2017.
- [3] F. Hu, D. Xie y S. Shen, «On the Application of the Internet of Things in the Field of,» de *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Hubei Province, P.R. China*, 2013.
- [4] E. Dr.Quesada, *CANTIDAD, USO Y GESTION DE LAS CAMAS HOSPITALARIAS. TENDENCIAS EN EL MUNDO Y SITUACIÓN EN MENDOZA*, Mendoza, Argentina, 2011.
- [5] P. S. Mendoza, K. Á. Hernández, C. V. Núñez y D. J. Molinares, «Internet de las cosas y la salud centrada en el hogar,» *Salud Uninorte*, p. 14, 2016.
- [6] S. Barillaro, G. De Luca, W. Valiente, E. Carmuccio, G. Garcia, M. Volker, D. Giulianelli, N. Casas y M. Perez, «Diseño de sistema IoT de monitoreo y alarma para personas mayores,» de *XVIII Workshop de Investigadores en Ciencias de la Computación (WICC 2016, Entre Ríos, Argentina)*, Entre Rios, 2016.
- [7] S. M. Giménez, *RED DE ACCESO ÓPTICA IoT PARA LA GESTIÓN EFICIENTE DE LA ENERGÍA EN EL CAMPUS DE VERA*, Valencia,, 2017.
- [8] D. A. Taipe Manotoa, *PLATAFORMA IoT DE CONTROL INTELIGENTE DE UN SISTEMA DE ILUMINACIÓN LEDCON SUMINISTRO ELÉCTRICO EN CORRIENTE CONTINUA LVDC*, Ambato, Ecuador, 2018.
- [9] M. Á. SEGURA GAZCÓN y R. R. VELÁSQUEZ HERNÁNDEZ, *ARQUITECTURA DE UN SISTEMA DE TELEMONITORIZACIÓN PARA HOSPITALIZACIÓN DOMICILIARIA DE ADULTOS MAYORES APOYADA EN TECNOLOGÍAS DE INTERNET DE LAS COSAS (IOT)*, Cartagena, 2017.
- [10] C. Fang, Y. Wang y S. Gao, *Flexible and Wearable GRF and EMG Sensors Enabled Locomotion Mode Recognition for IoT Based In-home Rehabilitation*, Beijing, China.

- [11] A. SALAZAR-ZULUAGA, L. A. AGUILAR-LONDONO y G. D. a. G.-S. R. ZAPATA-MADRIGAL, «Implementación de Nodos Lógicos DER IEC 61850-7-420 en una Placa Electrónica,» *Scielo*, vol. xx, p. 40, 2017.
- [12] D. Suárez López, R. M. Espinosa, I. C. Gutierrez y L. Schreiner de Oliveira, «Diseño de una herramienta de medición de ruidos basados en tecnologías Arduino-Raspberry PI,» *Scielo*, vol. XII, p. 1, 2017.
- [13] N. Surantha y W. R. Wicaksonob, «Design of Smart Home Security System using Object Recognition and PIR Sensor,» de *3rd International Conference on Computer Science and Computational Intelligence 2018*, Indonesia, 2018.
- [14] W. J. Gil Gonzalez, J. J. Mora Florez y S. M. Perez Londoño, «Análisis del procesamiento de los datos de entrada para un localizador de fallas en sistemas de distribución,» *Tecnura*, vol. 18, nº 41, p. 75, 2014.
- [15] A. Mampotes y M. D. Lopez Romero, *Sistema de gestión de amenazas basado en SBC*, Popayán, 2019.
- [16] Espressif Systems, «ESP 32 DataSheet,» 2019. [En línea]. Available: www.espressif.com. [Último acceso: 11 Diciembre 2019].
- [17] Last Minute Engineers, «Last Minute Engineers,» [En línea]. Available: <https://lastminuteengineers.com/how-rfid-works-rc522-arduino-tutorial/>. [Último acceso: 12 Diciembre 2019].
- [18] «MQTT,» [En línea]. Available: <http://mqtt.org/>.
- [19] L. Llamas, «Luis Llamas,» 17 Abril 2019. [En línea]. Available: <https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/>. [Último acceso: 4 Diciembre 2019].
- [20] L. d. V. Hernández, «ProgramaFacil,» [En línea]. Available: https://programafacil.com/esp8266/mqtt-esp8266-raspberry-pi/#Arquitectura_de_un_sistema_MQTT. [Último acceso: 4 Diciembre 2019].
- [21] «Node-RED,» OpenJS Foundation., [En línea]. Available: <https://nodered.org/>. [Último acceso: Diciembre 10 2019].
- [22] Raspbian org, «Raspbian,» [En línea]. Available: <https://www.raspbian.org/>. [Último acceso: 14 Febrero 2020].
- [23] Raspberry Pi Foundation, «Raspberry Org,» Raspberry Pi Foundation, [En línea]. Available: <https://www.raspberrypi.org/documentation/installation/noobs.md>. [Último acceso: 25 Febrero 2020].

- [24] Apache org , «Apache HTTP Server Project,» [En línea]. Available: <https://httpd.apache.org/>. [Último acceso: 14 Febrero 2020].
- [25] MariaDB Foundation, «MariaDB Foundation,» [En línea]. Available: <https://mariadb.org/>. [Último acceso: 5 Marzo 2020].
- [26] phpMyAdmon, «phpMyAdmin,» [En línea]. Available: <https://www.phpmyadmin.net/>. [Último acceso: 5 Marzo 2020].