DISEÑO DE UNA INFRAESTRUCTURA AVANZADA DE MEDICIÓN (AMI) A NIVEL RESIDENCIAL

MICHAEL ANDERSON TOVAR MELO

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C.
2022

DISEÑO DE UNA INFRAESTRUCTURA AVANZADA DE MEDICIÓN (AMI) A NIVEL RESIDENCIAL

MICHAEL ANDERSON TOVAR MELO

Proyecto de grado presentado como requisito para optar al título de INGENIERO ELECTRÓNICO

Director: Edwin Francisco Forero García

UNIVERSIDAD SANTO TOMÁS DE AQUINO
FACULTAD DE INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C.
2022

Dedicatoria

“To live without hope is to cease to live”
– Fyodor Dostoevsky

A mis padres, hermanos,
hermanos de otra madre, compañeros
y a cada una de esas personas que directa
o indirectamente estuvieron allí tendiéndome una mano
siempre tendrán un espacio en mi corazón.

En memoria de mi gran amiga Sandra Sacristán (Q.E.P.D).

AGRADECIMIENTOS

Este proyecto de grado es una forma de plasmar una pequeña parte del conocimiento que he adquirido; gracias a mis padres que siempre han estado ahí y con su basta sabiduría me guiaron para crecer como persona, mi madre madrugando todas las mañanas para que su hijo fuera ingeniero. Mis hermanos que me ayudaron a forjar mi carácter. Mis hermanos de otra madre que caminaron junto a mí en situaciones difíciles. Los profesores que plantaron esa semilla de curiosidad en mi alma enseñándome sus experiencias e inspirándome, a mi director de tesis y a Sandra quien con su elocuencia y muchas otras virtudes me enseñaron a ser mejor persona y que sin el apoyo de ella no estaría en este lugar (Q.E.P.D).

CONTENIDO

INTRODUCCIÓN 13
1 PROBLEMA 14
2 ANTECEDENTES 15
3 JUSTIFICACIÓN 17
4 OBJETIVOS 19
4.1 Objetivo general 19
4.2 Objetivos específicos 19
5 MARCO TEÓRICO 20
5.1 INTEROPERABILIDAD AMI 20
5.1.1 Redes inteligentes (Smart grids) 20
5.1.2 AMI 22
5.2 ESP (ESPRESSIF) 31
5.2.1 Modos 31
5.3 FIREBASE 34
5.4 IEC 61968-9 35
5.5 XAMARIN FORMS 35
5.5.1 MVVM (Model-View-ViewModel) 36
5.5.2 Microcharts 37
5.6 SISTEMA INTELIGENTE 39
5.6.1 Google Colaboratory 39
5.7 AMI EN COLOMBIA 39
6 DISEÑO 43
6.1 HAN 43
6.1.1 Medidor inteligente 44
6.2 NAN 45
6.2.1 Gateway (Firebase - ESP) 45
6.2.2 XAMARIN 62
6.3 FIREBASE 80
6.3.1 Servicio energía eléctrica 80
6.3.2 Servicio de agua 81
6.3.3 Servicio de gas 82

6.4 SISTEMA INTELIGENTE 82
7 RESULTADOS 87
7.1 REDES HAN Y NAN 87
7.2 CURVAS DE CONSUMO DE SERVICIO 89
7.3 SISTEMA INTELIGENTE 92
7.4 ANÁLISIS ECONÓMICO 94
7.4.1 Ahorro 94
7.4.2 Implementación del proyecto 94
7.5 PROYECCIÓN 97
8 CONCLUSIONES 100
9 BIBLIOGRÁFICAS 102
ANEXOS 105
9.1 ANEXO A 105
9.2 ANEXO B 111
9.3 ANEXO C 121
9.3.1 Model 121
9.3.2 View 124
9.3.3 View-Model 137
9.4 ANEXO D 168
9.4.1 Código QR de medidor con MAC: **a4:cf:12:d9::3f:b7** 168
9.4.2 Código QR de medidor con MAC: **a4:cf:12:d9:3e:88** 169
9.4.3 Código QR de medidor con MAC: **a4:cf:12:d9:3f:1b** 170
9.5 ANEXO E 171
9.5.1 Importar librerias 171
9.5.2 Importar Data 171
9.5.3 Información básica de la data 171
9.5.4 Creacion modelo 172
9.5.5 Ajuste de modelo 172
9.5.6 Score 172
9.5.7 Validación 172
9.5.8 Matriz de confusión 173
9.5.9 Reporte de clasificación 173
9.5.10 Entrenando solo con data de entrenamiento 173

9.5.11 Resultados 174
9.5.12 Conexión Firebase 175
9.5.13 Prueba Firebase 175
9.5.14 Resultado 176

FIGURAS

Figura 1 Modelo de comunicación de smart grids [3] 21
Figura 2 Evolución de la interoperabilidad de smart grids [3] 21
Figura 3 Esquema general de comunicación de AMI [1] 23
Figura 4 Comunicaciones para medición inteligente [4] 24
Figura 5 Componentes principales de AMI [5] 25
Figura 6 IEC 61968-9 Modelo de referencia parcial con información del cliente y sistema de facturación. [3] 26
Figura 7 Topologías de red [11] 28
Figura 8 Tecnologías de telecomunicaciones [4] 30
Figura 9 Modo estación en ESP8266 [15] 32
Figura 10 Modo punto de acceso en ESP8266 [15] 33
Figura 11 Modo cliente seguro en ESP8266 [15] 33
Figura 12 Precios Firebase (Izquierda plan gratis, derecha plan de paga) [13] 35
Figura 13 Xamarin [40] 36
Figura 14 Patrón MVVM [17] 36
Figura 15 Gráfico de barras [18] 37
Figura 16 Gráfico de línea [18] 37
Figura 17 Gráfico de puntos [18] 38
Figura 18 Gráfico Radial Gauge [18] 38
Figura 19 Gráfico circular (Pie) [18] 38
Figura 20 Gráfico de radial [18] 38
Figura 21 Funcionalidades de medidores inteligentes [25] 41
Figura 22 Funciones mínimas de medidores inteligentes [25] 42
Figura 23 Infraestructura AMI 43
Figura 24 Autenticación en Firebase [13] 46
Figura 25 Usuario en Firebase [13] 46
Figura 26 Reglas Realtime Database [13] 47
Figura 27 Tablas en Realtime Database [13] 49
Figura 28 Tabla *meterTable* en Firebase [13] 50
Figura 29 Tabla *measureTable* en Firebase [13] 51
Figura 30 Tabla *InfoTable* en Firebase [13] 52
Figura 31 Tabla *statusTable* en Firebase [13] 53
Figura 32 Tabla *sesionTable* en Firebase [13] 53
Figura 33 IDE de Arduino [39] 54
Figura 34 IDE de Platform io [19] 54
Figura 35 ESP8266 NodeMCU v3 55
Figura 36 ESP8266 ESP – 01 56
Figura 37 Topología de red 56
Figura 38 Estructura de mensaje enviada por Gateway 57
Figura 39 Estructura de mensaje enviada por Medidor 59
Figura 40 Validación para sStatus 61
Figura 41 Inicio de USTAPG app. 64

Figura 42 Validación conectividad USTAPG app 65
Figura 43 Validación correo USTAPG app 66
Figura 44 Validación clave USTAPG app 67
Figura 45 Validación clave USTAPG app 68
Figura 46 Código QR por medidor [40] 69
Figura 47 Base de datos en Firebase [13] 69
Figura 48 Ejemplo QR MAC de equipo [40] 70
Figura 49 Validación MAC medidor USTAPG app 72
Figura 50 Validación ingreso MAC medidor USTAPG app 72
Figura 51 Página tabulada USTAPG app 74
Figura 52 Validación conectividad USTAPG UWP 76
Figura 53 Validación clave USTAPG UWP 76
Figura 54 Validación correo USTAPG UWP 77
Figura 55 Validación MAC QR USTAPG UWP 78
Figura 56 Validación MAC medidor USTAPG UWP 78
Figura 57 Página tabulada USTAPG UWP 79
Figura 58 Curva de carga por estrato [20] 80
Figura 59 Consumo por estrato de servicio de agua potable Bogotá [21] 81
Figura 60 Data para entrenamiento [41] 83
Figura 61 Detalle data [41] 84
Figura 62 Posibles salidas [41] 84
Figura 63 Regresión logística en Colab [41] 85
Figura 64 Entrenamiento sin caer en Overfitting [41] 85
Figura 65 Matriz de confusión [41] 85
Figura 66 Reporte de clasificación [41] 86
Figura 67 Valores por default de los medidores [41] 88
Figura 68 Valores de variables en Firebase [13] 88
Figura 69 Valores de variables enviados desde el medidor [41] 89
Figura 70 Curva de carga por estrato energía eléctrica (escala de hora) 90
Figura 71 Curva de consumo de agua potable (escala de hora) 90
Figura 72 Curva de consumo de gas (escala de hora) 91
Figura 73 Curva de carga por estrato energía eléctrica (escala de día) 91
Figura 74 Curva de consumo de agua potable (escala de día) 92
Figura 75 Curva de consumo de gas (escala de día) 92
Figura 76 Test algoritmo de regresión logística [41] 93
Figura 77 Evaluación sobre datos de Firebase [41] 94
Figura 78 Impacto industrial vs Actividad inventiva en campos relacionados con la distribución y gestión de la energía eléctrica [28] 98
Figura 79 Impacto industrial vs Actividad inventiva de las empresas más representativas [28] 98
Figura 80 Aspectos fundamentales redes inteligentes. [27] 99

GLOSARIO

ADA – Automatización de la red de distribución
ADC – Analog Digital Converter
AMI – Advanced Metering Infrastructure (Infraestructura de medición avanzada)
AP – Access Point (Punto de acceso)
COB - Permite la comunicación bidireccional
COP – Pesos colombianos
CREG - La Comisión de Regulación de Energía y Gas
CRUD – Create, Read, Update, Delete (Crear, Leer, Actualizar, Borrar)
DER - Recursos distribuidos
DevKit – Kit de desarrollo
DNP - Departamento Nacional de Planeación
EPS - Electric Power System (Sistema de energía eléctrica)
FNCER - Fuentes no convencionales de energía renovables
FRA - Prevención y detección de fraudes
Gateway – Puerta de enlace
GD - Soporta la importación y exportación de energía
GNC – Gas Natural Comprimido
GPU - Graphics processing unit (Unidad de Procesamiento Gráfico)
HAN - Home Area Network (Red de Área Doméstica)
IDE – Integrated Development Environment
IEC - International Electrotechnical Commission (Comisión Electrotécnica Internacional)
IEEE - Institute of Electrical and Electronics Engineers (Instituto de Ingenieros Eléctricos y Electrónicos)
IoT – Internet of things (Internet de las cosas)
kWh – Kilo vatio por hora
LAN - Local Area Network (Red de Área Local)
LRM - Lectura Remota del medidor
ML – Machine Learning
MME - Ministerio de Minas y Energía
MVVM - Model-View-ViewModel
NAN - Neighborhood Area Network (Red de Área de Vecindario)
PLC - Power Line Communications (Comunicaciones por Línea de Potencia)
RTDB – Realtime Database
SCADA - Supervisory Control and Data Acquisition (Supervisión, Control y Adquisición de Datos)
Smart grids - Redes inteligentes
Smart meter - Medidor inteligente
Two-way – Dos caminos o bidireccional
UI – User Interface (Interfaz de Usuario)
UPME - Unidad de Planeación Minero-Energética

US – Dólar estadounidense
USU - Acceso del usuario a la información del medidor
UWP – Universal Windows Platform (Plataforma Universal de Windows)
VE - Vehiculos electricos
WAN - Wide Area Network (Red de Área Amplia)
WiFi – Wireless Fidelity

RESUMEN

El presente proyecto brinda dar una alternativa de bajo costo a las tecnologías que actualmente se utilizan para la construcción de una Infraestructura de Medición Avanzada diseñando y implementando algunas de las partes que componen esta infraestructura como lo son los medidores inteligentes dando un enfoque a la comunicación que estos presentan con el concentrador, puerta de enlace o concentrador detallando su funcionamiento y el alcance como la conexión a la nube y actualizaciones que este podría tener en un futuro para mejorar su funcionalidad, la comunicación con el usuario ofreciendo alternativas al momento de dar a conocer el consumo, tarifas y estado de su medidor actual y sistemas inteligentes capaces de detectar un consumo anormal por cada cliente de manera personalizada para tener una gestión adecuada del suministro y de igual forma tener alternativas diferentes a las que ofrecen las grandes compañías a un costo menor ya que se realiza un análisis de los costos que tendría una implementación de este tipo y un análisis de escalabilidad del mismo.

INTRODUCCIÓN

AMI comprende amplia variedad de conceptos como lo son los medidores inteligentes que por sí solos comprenden un gran caso de estudio y un gran mercado al cual grandes corporaciones de talla mundial como Qualcomm Inc, Toshiba Corp y LG Electronics Inc están dispuestas a invertir ya que ven una gran oportunidad debido a que la mayoría de los países están viendo en AMI una oportunidad al momento de tener toda la red de suministro de energía eléctrica controlada y monitoreada y los beneficios que esta pueda traer. Colombia hace parte de estos países que planea bajo la agenda 2030 un desarrollo sostenible para mejorar la calidad de vida de los colombianos es así como surgen iniciativas por parte del gobierno para que tanto las empresas comercializadoras de los servicios de energía eléctrica, agua y gas (con un enfoque en la energía eléctrica) como terceros empiecen la implementación de infraestructuras de medición avanzada a lo largo del Sistema Interconectado Nacional para mantener completamente controlada la red de distribución. Es así que empresas como Enel-Codensa, Grupo EPM, Surtigas, Air-e, Vanti, Gases del Caribe, Efigas, Afinia, Celsia han empezado la instalación de medidores inteligentes a sus usuarios. Uno de los objetivos principales por el cual se implementa una red AMI es porque tanto como el distribuidor y el cliente cuentan con información sobre el estado de las redes de distribución del domicilio con la cual se puede prever y detectar fallos en esta misma; además de que los usuarios y la empresa pueden realizar la comparación entre la energía comprada, entregada y facturada. Implementar una red AMI es la solución si se quiere reducir los costos en la distribución y comercialización de energía, tener un control en tiempo real y remoto de una red de distribución, sin dejar de proporcionar constantemente información para el usuario. Sin embargo, al ser tecnologías que son relativamente nuevas en el país tienen un alto precio a la hora de implementarlas por lo que en este proyecto se diseña he implementan algunas de las partes que comprende una red AMI dando primero una contextualización de todo lo que comprende una Infraestructura de Medición Avanzada a nivel de normas internacionales, seguido de esto una descripción de las tecnologías utilizadas actualmente a la hora de implementar este tipo de infraestructuras y pasando al diseño empezando por la comunicación de los medidores inteligentes al concentrador, la gestión de información de los medidores por parte del concentrador y su conexión a la base de datos alojada en la nube para que tanto el usuario y l empresa que gestiona el suministro estén al tanto de dicho consumo. A nivel del usuario será netamente informativa por medio de una solución multiplataforma disponible en iOS, Android y UWP, solo testeada e estas dos últimas; y de carácter necesario para la empresa que gestiona el suministro ya que serán estos datos los utilizados para el entrenamiento del sistema inteligente que dará alertas si existe un consumo anormal. Al final se hará un análisis sobre los resultados obtenidos y las proyecciones y escalabilidad que podría llegar a tener este proyecto.

1 PROBLEMA

Uno de los pilares más importantes para la distribución inteligente y eficiente del suministro de energía eléctrica es el monitoreo, el hecho de tener claro cuáles son los momentos en los que se tienen los picos de mayor demanda y en cuáles no, para así tener una adecuada gestión de la red y tomar las respectivas decisiones en cuanto a la generación; una red AMI (Advanced Metering Infrastructure o Infraestructura de Medición Avanzada) tiene como función básica el monitoreo inteligente, es decir que a partir de las diferentes mediciones que se realicen, una red inteligente recopila estos datos, los analiza y posteriormente en un centro de control se modifica la generación de acuerdo a la demanda.

Al realizar esta simple acción de suministrar la energía de acuerdo con la demanda, reduce costos de mantenimiento que además de proteger la integridad del personal encargado, evita el reemplazo temprano de las redes de distribución por deterioro y a su vez el costo de las pérdidas en la transmisión se reduce previniendo así sobrecargas al sistema, y todo esto en tiempo real y de manera remota lo que también reduce personal necesario para realizar el control y monitoreo de la red.

Pero ¿Cómo reducir los costos por pérdidas no técnicas (hurto de energía, manipulación de indebida medidores o de sistemas de facturación para reducir el registro de consumo) en la distribución y comercialización de servicios públicos a nivel residencial?

2 ANTECEDENTES

En Ecuador existen dificultades en cuanto a la distribución de la energía eléctrica por lo que es necesario implementar un sistema que gestione de forma eficiente el suministro, por lo que se realiza un estudio completo para la implementación de una red AMI [1].

En [2] se realiza el diseño e implementación una micro red AMI para la Universidad Nariño para la gestión adecuada del suministro de energía eléctrica y su posterior evaluación y desempeño de esta.

La Empresa Eléctrica de Quito S.A. pierde bastante dinero por pérdidas al momento de la distribución de la energía eléctrica. Una solución a estos inconvenientes es el estudio y diseño de una red AMI capaz de gestionar el suministro de energía eléctrica según la demanda [3].

Uno de los grandes problemas en las redes AMI es la comunicación entre medidores, el [4] propone una solución para este problema en particular y posteriormente evaluando su desempeño.

En el estudio realizado en [5] se realiza el estudio de las necesidades eléctricas industriales y domésticas en la localidad de Fontibón cumpliendo con los reglamentos establecidos por las diferentes Normas Técnicas Colombianas o NTC.

En [6] se realiza el estudio y diseño de una red AMI para las necesidades eléctricas de Ecuador teniendo en cuenta los cambios regulatorios nacionales.

Una red AMI está compuesta por sensores, medidores inteligentes, entre otros, todo esto para realizar la distribución automatizada de la energía eléctrica, además de la comunicación utilidad (sistema) y el consumidor [7].

Para realizar la gestión remota de un sistema de suministro eléctrico es necesario la utilización de un controlador inteligente, en [8] se muestra de una manera general un controlador inteligente.

La comunicación bidireccional, es decir entre el usuario final y las aplicaciones de utilidad, es un pilar fundamental en una red AMI, por lo que en [9] se realizará el estudio de los diferentes protocolos de comunicación para hacer una comunicación efectiva.

Lahore y Karachi son dos ciudades de Teherán que tienen varios problemas en cuanto a red eléctrica se refiere, por lo que en [10] se propone un modelo básico y funcional de una red AMI para la solución de estos problemas.

Al manejar información sobre el consumo de energía eléctrica y analizar esta información para su posterior procesamiento y control, existe el riesgo de que esta información sea filtrada para otros fines por lo que es necesario de asegurar la integridad de los datos [35].

En [36] se presenta un modelo de referencia en el cual se tiene en cuenta el almacenamiento de la energía, la distribución y finalmente simplificar su lectura para la toma de decisiones mediante el uso de TIC’s.

Para la detección de anomalías en un Sistema de distribución eléctrica es necesario que exista un “supervisor” que periódicamente observe el comportamiento de la red y tome decisiones para contrarrestar si existiera algún problema, para esto se utilizan técnicas de inteligencia artificial, específicamente Machine Learning evitando también así los problemas de ciberseguridad que conlleva tener una red interconectada [37].

ZigBee es una tecnología que se encarga de realizar tareas de domótica, es decir, monitoreo y control, además que toda esta información es posteriormente procesada para el análisis y distribución inteligente según la demanda, pero tiene limitaciones de tráfico de alta demanda y de comunicación, por lo que en [38] se propone una solución mediante el uso de WIFI.

Los medidores inteligentes son una parte importante en lo que se refiere a redes AMI, ya que estos son los actuadores (salidas) y sensores (entradas) de la red AMI. Al tener la información de entrada se tomas las respectivas decisiones y se procede a activar los actuadores necesarios para cumplir la demanda [39].

3 JUSTIFICACIÓN

En esta sección se presenta la necesidad de realización del proyecto, las ventajas del trabajo y los aportes (si aplica) que hace al mundo moderno relacionado con la electrónica. La infraestructura energética en Colombia necesita expandirse para cubrir las futuras necesidades en cuanto a eficiencia energética y gestión inteligente se refiere, según el grupo Bancolombia junto con estudios realizados por la UPME "El consumo de energía eléctrica anual del país está cerca de alcanzar los 70.000 GWh/año y para los próximos 11 años, según las proyecciones realizadas por la UPME, se espera un incremento promedio del 2% anual, teniendo en cuenta las expectativas de dinámica del sector industrial, la electrificación de la economía y un incremento en el número de vehículos eléctricos, que para 2030 se estima sean 400.000 en circulación en las vías colombianas" [18], por esta razón surge la necesidad de realizar una correcta gestión de la energía eléctrica. Uno de los problemas más comunes en cuanto a la distribución y comercialización de la energía eléctrica, son los contadores o medidores de servicios públicos, por ejemplo "La entidad (SIC - Superintendencia de Industria y Comercio) sostiene que se tiene identificado que una gran porción de la población manifiesta desconfianza de las mediciones que se le realizan" [16]; esta información hace parte de un artículo publicado en la revista Portafolio en donde se evidencia la inconformidad por parte de varios usuarios del servicio de agua y energía eléctrica por costos elevados en la factura de cada servicio que muchas veces no corresponde al consumo real, lo que genera desconfianza por parte de la comunidad hacia la empresa encargada de la distribución ya que existe una escasa reglamentación técnica con respecto al uso y ajuste metrológico de los contadores; además de que estos pueden ser fácilmente manipulados para tener una lectura errónea, lo que conlleva a que la empresa prestadora del servicio no tenga en cuenta esto al momento de cobrar, además de que estos registros se utilizan posteriormente para realizar una correcta distribución del servicio para cada conexión domiciliaria. Pero la modificación de los contadores no es la única razón por la cual una conexión domiciliaria podría tener una lectura irregular en el consumo; fallas en las conexiones o fugas y el mal estado en las redes de distribución también hacen parte de los factores que afectan en gran proporción las diferentes mediciones. Si dichos contadores estuvieran conectados a una Smart Grids, estas anomalías se podrían identificar de una manera rápida y eficaz, lo que reduciría en gran medida la pérdida, de manera que el costo tanto para el usuario como para la empresa se reduciría.

En consecuencia, surge la necesidad de invertir en el campo de eficiencia energética por ejemplo la empresa de servicios de telecomunicaciones Claro invirtió alrededor de 5.1 millones de dólares en soluciones de IoT, todo esto para suplir

necesidades en “Ciudades inteligentes, agricultura, telemetría y servicios públicos, entre otros, son casos de uso que requieren contar con una tecnología que les supla sus requerimientos en transmisión y consumos de energía” [17].

Esta situación, hace surgir la necesidad de diseñar un sistema que pueda de manera automatizada y remota gestionar en tiempo real la distribución de la energía eléctrica a cada conexión domiciliaria, además de realizar un correcto monitoreo de los demás servicios públicos sin dejar de proporcionar información a cada cliente.

4 OBJETIVOS

4.1 Objetivo general

Diseñar una infraestructura avanzada de medición (AMI) a nivel residencial

4.2 Objetivos específicos

- Parametrizar medidores inteligentes de servicios públicos localizados en viviendas obteniendo medidas de referencia para el sistema inteligente.
- Diseñar un algoritmo que funcione como puerta de enlace (Gateway) de bajo costo que logre gestionar de manera adecuada los diferentes medidores inteligentes de una residencia.
- Desarrollar una aplicación móvil multiplataforma capaz de realizar el monitoreo de los diferentes datos proporcionados por el Gateway.
- Simular e implementar un sistema inteligente capaz de gestionar el suministro de servicios públicos de una residencia.

5 MARCO TEÓRICO

5.1 INTEROPERABILIDAD AMI

5.1.1 Redes inteligentes (Smart grids)

La Infraestructura de medición avanzada (AMI) es una parte importante de las redes de distribución eléctrica inteligente o conocidas también como Smart grids.

Smart grids es el término utilizado para hacer referencia a las redes de distribución eléctrica que son capaces de transmitir energía en ambos sentidos, es decir que no solo la central eléctrica es la que proporciona el suministro, si no que a su vez los consumidores comunes, es decir viviendas y distintos negocios, pueden o no ser también pequeños productores de energía. Además, para que todo esto funcione debidamente se deben implementar diferentes tecnologías que sean capaces de asegurar la fiabilidad y calidad en el suministro de energía eléctrica lo que conlleva a detectar averías de una manera más sencilla y a su vez aislar el problema; optimizar el consumo de los usuarios, entre otros muchos beneficios como la sostenibilidad ambiental y almacenamiento de energía eléctrica.[7]

El Instituto de Ingenieros Eléctricos y Electrónicos o IEEE es una asociación mundial la cual se encarga de entre otras cosas la creación de estándares internacionales para la industria. En el estándar IEEE 2030 llamado *"Guide for Smart Grid Interoperability of Energy Technology and Information Technology Operation with the Electric Power System (EPS), End-Use Applications, and Loads"* ó *"Guía para la Interoperabilidad de Redes Inteligentes de Tecnología Energética y Operación de Tecnologías de la Información con el Sistema Eléctrico de Potencia (EPS), Aplicaciones de Uso Final y Cargas"* se describen las mejores prácticas para la interoperabilidad de una red inteligente. Este estándar abarca todo sobre la interoperabilidad de las redes inteligentes permitiendo su extensibilidad, la escalabilidad y la capacidad de actualización. [6]

En IEEE 2030 se definen tres perspectivas: sistemas de energía, tecnología de comunicaciones y tecnología de la información. En la figura 1 se puede ver de manera general el modelo de comunicación que establece IEEE 2030 para una red inteligente.

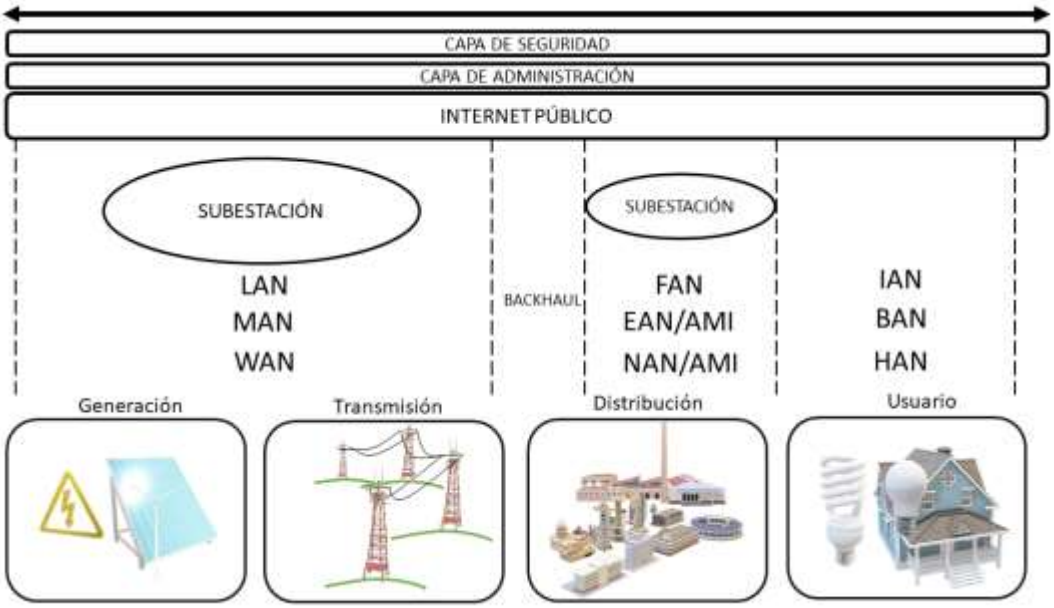


Figura 1 Modelo de comunicación de smart grids [3]

Fuente: IEEE. 2030.2-2015 - IEEE Guide for the Interoperability of Energy Storage Systems Integrated with the Electric Power Infrastructure. 2015.

Se puede apreciar que AMI es una parte para el funcionamiento de las redes inteligentes como arquitectura de aplicación como se muestra en la figura 2 donde se encuentra la evolución de la interoperabilidad de las redes inteligentes de distribución eléctrica.

Modelo de referencia de Smart Grid	Modelos conceptuales de referencia (NIST, IEC, etc)		
IEEE 2030 Interoperabilidad de Smart Grid	Arquitectura de comunicación	Arquitectura del sistema de potencia	Arquitectura de la tecnología de la información
Aplicaciones de Smart Grid	Aplicación de arquitectura AMI	Aplicación de arquitectura PEV	Aplicación de arquitectura ...

Figura 2 Evolución de la interoperabilidad de smart grids [3]

Fuente: IEEE. 2030.2-2015 - IEEE Guide for the Interoperability of Energy Storage Systems Integrated with the Electric Power Infrastructure. 2015.

5.1.2 AMI

AMI (Advanced Metering Infrastructure) o infraestructura de medición inteligente es en esencia la integración de diferentes tecnologías para que exista una conexión inteligente mediante un flujo de información bidireccional entre consumidores o medidor inteligente y operadores del sistema de gestión, distribución y suministro de energía eléctrica o empresa de servicios, esto con el objetivo de brindar información actualizada (tiempo real) para la gestión adecuada de dicho suministro y con esto el ahorro de recursos económicos para ambas partes. [1]

En una infraestructura de medición avanzada además de realizar la captura actualizada del consumo de energía, permite utilizar otras aplicaciones como la reconexión y desconexión de manera remota, detección de interrupciones y de posibles fallas. Esto se hace por medio del medidor inteligente el cual debe tener o no las capacidades anteriormente mencionadas. Estas capacidades que puede o no tener el medidor inteligente además del intercambio de información entre sistemas de distribución eléctrica se definen en el estándar de la comisión electrotécnica comercial IEC 61968. [2]

5.1.2.1 COMUNICACIÓN

La infraestructura de medición avanzada es un sistema de recolección de datos medidos, ya sea con una periodicidad de tiempo predefinida o en cualquier momento a solicitud de la empresa de servicios. Este canal por el cual viajan los datos recolectados por los diferentes medidores debe ser bidireccional (Two-way), es decir que no solo el medidor debe enviar los datos si no que a su vez debe de poder recibir órdenes solicitadas por la empresa prestadora del servicio.

Esta comunicación bidireccional que se maneja en la infraestructura de medición avanzada se divide en tres zonas o secciones las cuales dependen del tipo de red. En la figura 3 se muestra el esquema general de comunicación de estas redes de acuerdo con el flujo de información.

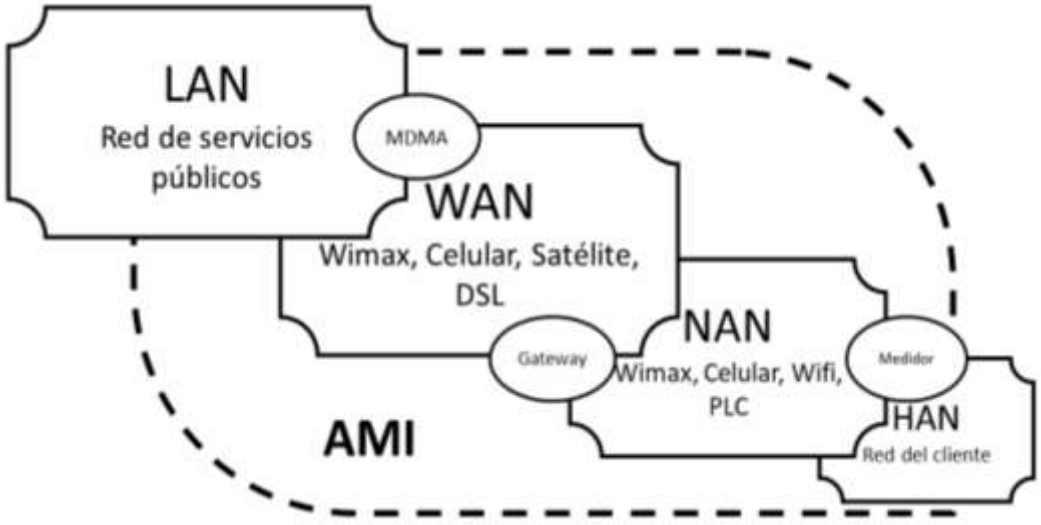


Figura 3 Esquema general de comunicación de AMI [1]
Fuente: A framework for intrusion detection system in advanced metering infrastructure, Article in Security and Communication Networks, January 2014

HAN - Home Area Network (Red de Área Doméstica)
Este tipo de red se encuentra, como lo dice su nombre, con una cobertura netamente en el hogar, residencia o instalaciones del cliente y permite la comunicación entre los diferentes dispositivos de IoT como sensores o electrodomésticos y el medidor inteligente, o simplemente el medidor inteligente. Esta red está a cargo del cliente y al contar con el medidor inteligente se puede recibir instrucciones sobre el uso de la electricidad o activar y desactivar cargas. Toda la información necesaria para la gestión de la energía eléctrica es enviada posteriormente al concentrador.[4]

LAN - Local Area Network (Red de Área Local)
El alcance de esta red se limita a un espacio físico reducido, a lo sumo un edificio. Son de uso muy cotidiano en negocios, empresas y hogares.[4]

NAN - Neighborhood Area Network (Red de Área de Vecindario)

Este tipo de red tiene como función servir como puente de comunicación entre los medidores inteligentes y el centro de control, esto por medio del concentrador de datos o Gateway. Es decir que permite la conexión entre múltiples HAN’s, lo que implica que los medidores se pueden conectar entre ellos facilitando los mensajes de diagnóstico y supervisión de las condiciones de los medidores.[4]

WAN - Wide Area Network (Red de Área Amplia)

Esta red conecta varias NAN con la red de backhaul para remitir la información al centro de gestión. La información de las redes NAN son recolectadas por el Gateway, es decir que desde el Gateway se envía la información de múltiples viviendas por lo que al pasar esta información desde el Gateway hasta el centro de control se envía bastante información por lo que en esta red se debe tener un gran ancho de banda, alta confiabilidad, privacidad y seguridad. Esto no quiere decir que en las redes anteriormente mencionadas no se deba de tener en cuenta estas características, pero por el flujo de información la red WAN es en la que se deben evaluar estos aspectos con mayor prioridad. [4]

Aunque el esquema general de comunicaciones comprende cuatro tipos de redes HAN, NAN, WAN y LAN, esta última se puede omitir que si se compara la utilidad según la cobertura y el propósito entre los tipos de red HAN, WAN y LAN se puede observar que con una red HAN se obtiene la cobertura suficiente para las medidas de un hogar, es decir, se pueden tener, como se mencionó anteriormente, varios dispositivos conectados como el medidor inteligente indispensable para AMI y otros con un enfoque dirigido a IoT; y con una red NAN, en la que se encuentra el concentrador o Gateway se obtiene la conexión a múltiples redes HAN, enviando posteriormente toda esta información por alguno de los diferentes medios utilizados en una red WAN hacia la empresa de servicios públicos por lo que utilizar un tipo de red LAN sería innecesario ya que con estas tres mencionadas anteriormente se puede implementar toda una infraestructura avanzada de medición como se observa en la figura 4. [4]

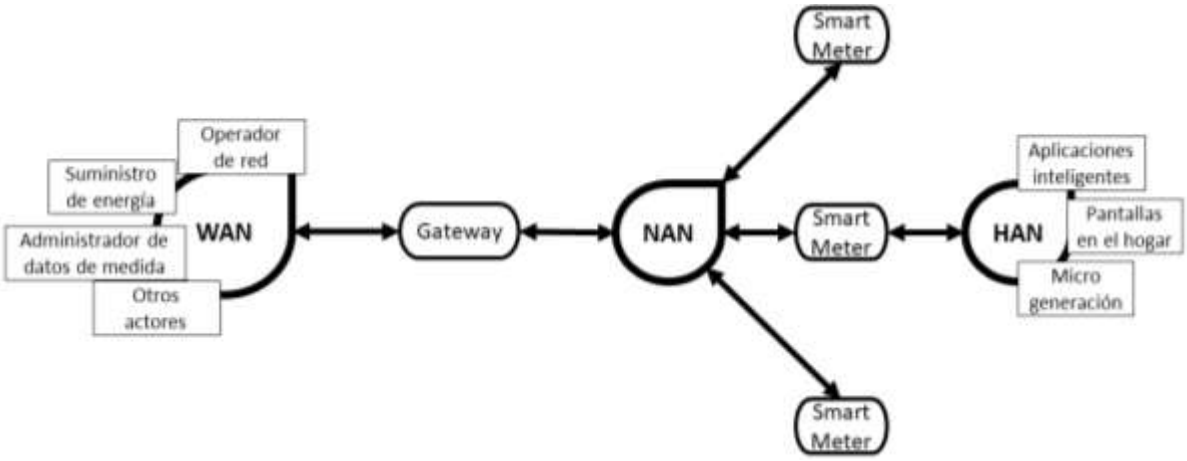


Figura 4 Comunicaciones para medición inteligente [4]

Fuente: Javier Bernardo Cabrera Mejía. Diseño de la red de telecomunicaciones para el sistema de medición avanzada (AMI) de energía eléctrica en el centro histórico de la ciudad de Cuenca. Quito,Ecuador, 2014

5.1.2.2 COMPONENTES

Como se puede observar en la figura 5 además de las diferentes redes que componen a AMI como las redes HAN, NAN y WAN, también existen diferentes componentes que utilizan estas redes para comunicarse

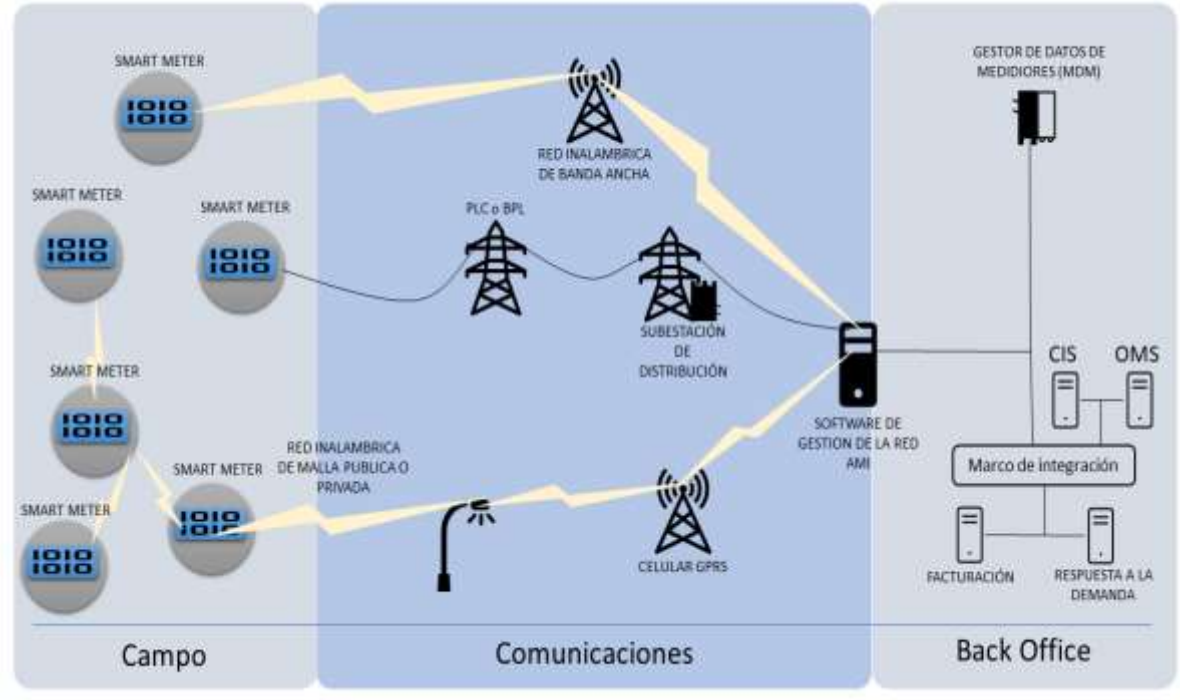


Figura 5 Componentes principales de AMI [5]
Fuente: Javier Revelo-Fuelagán Álvaro José Cervelion-Bastidas Guefry L. AgredoMéndez. Diseño y evaluación de desempeño de la infraestructura AMI para la microrred de la Universidad de Nariño. Pasto, Colombia, 2018.

En la figura 5 se pueden observar diferentes componentes, pero entre los principales, es decir los componentes indispensables para el correcto funcionamiento de AMI son:

Equipo de medición: Según el estándar IEEE 61968-9 es “una especialización dispositivo final. El dispositivo final puede contener una capacidad de metrología, puede contener una capacidad de comunicaciones, puede ser una unidad de control de carga y puede contener una combinación de muchos tipos diferentes de funciones”, como dice en el estándar, una de las capacidades es la de transmitir los datos y eventos a un sistema de adquisición de datos a un concentrador cada cierto periodo de tiempo. Para el usuario/cliente es posible verificar y validar cómo se está utilizando la energía eléctrica del inmueble que le suministra la respectiva empresa prestadora del servicio y le ayuda a controlar su consumo a través del medidor inteligente. [5]

Además, dicho medidor debe de tener las siguientes características:

- Tiene una identidad única
- Se gestiona como activo físico
- Puede emitir eventos
- Puede recibir solicitudes de control
- Puede recopilar e informar valores medidos
- Puede participar en procesos comerciales de servicios públicos

Más específicamente en el modelo de referencia que se observa en la figura 6 se observa cual es el flujo de información que debe haber en él y hacia el medidor inteligente.

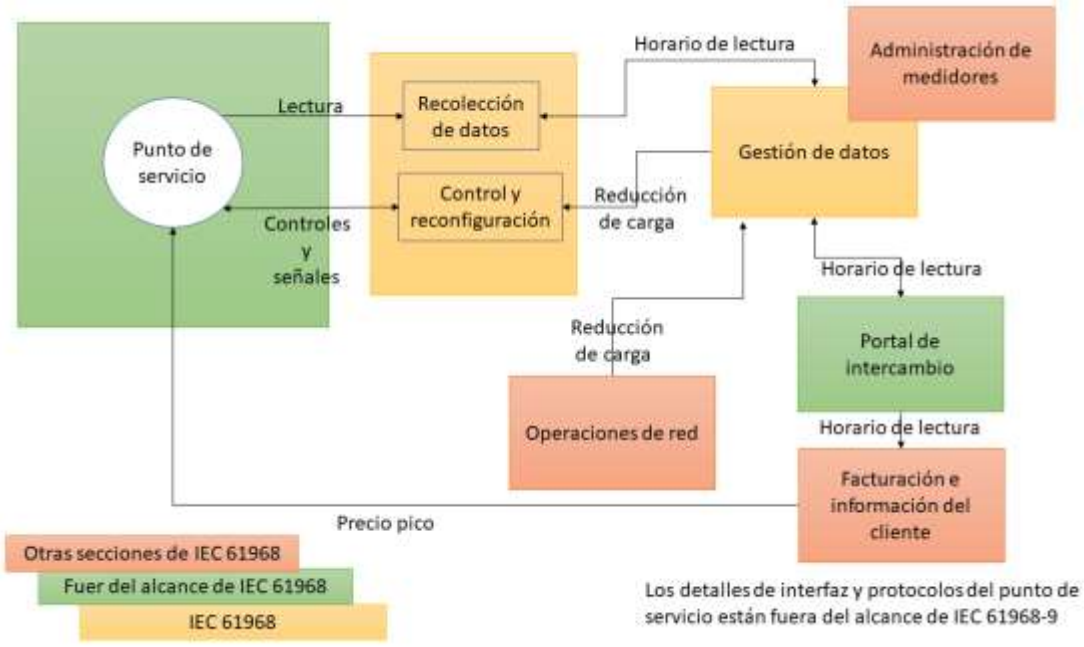


Figura 6 IEC 61968-9 Modelo de referencia parcial con información del cliente y sistema de facturación. [3]

Fuente: IEEE. 2030.2-2015 - IEEE Guide for the Interoperability of Energy Storage Systems Integrated with the Electric Power Infrastructure. 2015.

Puerta de enlace (Gateway): El Gateway o concentrador de datos, cuenta con capacidad de almacenamiento además de recibir información enviada desde los smart-meter (medidores inteligentes) y luego envía dicha información hasta el Sistema de Gestión ubicado en el centro de control. [5]

Red de telecomunicaciones: Sobre la red de telecomunicación se realiza el intercambio de información entre la puerta de enlace y el sistema de gestión. La infraestructura de medición avanzada (AMI) puede contener varias tecnologías que le permitan la transmisión de datos entre las cuales se destacan la fibra óptica, redes inalámbricas, microondas, entre otras, siendo flexible dependiendo de los requerimientos del cliente y de la empresa de servicios públicos. [5]

5.1.2.3 ARQUITECTURA Y SEGURIDAD

La infraestructura de medición avanzada presenta nuevos desafíos de seguridad porque se compone de dispositivos que se colocan en ubicaciones físicamente inseguras y hace uso de la comunicación inalámbrica que puede estar posiblemente corrupta. Estos recursos pueden ser accedidos por usuarios descuidados o malintencionados. Cleveland en él se discute la seguridad, requisitos y amenazas relacionadas de los principales componentes de un AMI. Los problemas de seguridad para AMI se pueden clasificar en los siguientes:

- La confidencialidad y privacidad, ya que pueden llegar a afectar fuertemente la visión de los clientes sobre el despliegue de la red inteligente ya que llegado el caso de que algún tercero no autorizado conozca los patrones de uso de cada residencia se pueden revelar hábitos de vida e incluso la presencia/ausencia de residentes que podrían ser utilizados por ladrones. Si no se satisfacen las preocupaciones de las personas, es posible que se nieguen o afecten el correcto despliegue de la red inteligente; es decir, pueden negarse a permitir que los proveedores de servicios públicos instalen medidores inteligentes en sus lugares de residencia.

- La integridad en los sistemas AMI, significa evitar cualquier cambio en los datos de medición recibidos de los medidores y los comandos de control enviados a los medidores. Un escenario que puede ocurrir es cuando un pirata informático envía un comando de desconexión al violar un sistema de administración de medidores y luego desconectar millones de contadores inteligentes.

- La disponibilidad, se considera el requisito más crucial en AMI porque algunos sistemas o aplicaciones son en tiempo real y están directamente relacionados con la disponibilidad de energía, que es el factor más esencial en la red inteligente ya que es mediante el cual se obtiene la información.

En AMI, la disponibilidad e integridad de los datos tienen prioridad, requisitos y amenazas relacionadas de los principales componentes. Los problemas de seguridad para AMI se pueden clasificar en los siguientes: La red de área amplia es una red multipropósito que proporciona conectividad desde los colectores de datos hasta las unidades de control en el centro de servicios públicos. Conecta varias subestaciones y puntos de control locales con el centro de servicios principal. Lo que forma una columna vertebral de comunicación que interconecta los centros de servicios públicos a las subestaciones y puntos finales (medidores), por lo que es sumamente importante mantener la confidencialidad de los datos. [1]

5.1.2.4 TOPOLOGÍAS DE RED

El término topología de red se refiere a aquella estructura de intercomunicación (arreglo físico) en la que se pueden organizar las redes de transmisión de datos entres diferentes dispositivos o nodos. Las topologías de red más utilizadas en AMI son bus, estrella, árbol y malla, cada una de estas topologías está compuesta por una parte física que es la que define la manera en la que se deben interconectar los elementos de red, y una parte lógica que dependiendo de la topología gestiona el conjunto de reglas en la transmisión de los datos. [5][11].

Debido a la extensión de una red AMI es común que se utilice una topología híbrida (la unión de dos o más topologías) ya que se logra acondicionar a los diferentes tipos de red y/o componentes presentes en esta misma, asimismo la utilización de una topología influye en la velocidad de transmisión, tiempos de llegada, cantidad de trabajo sobre los nodos o dispositivos, entre otras características en cuanto al flujo de información se refiere [11]. Las topologías más utilizadas en una red AMI son (figura 7):

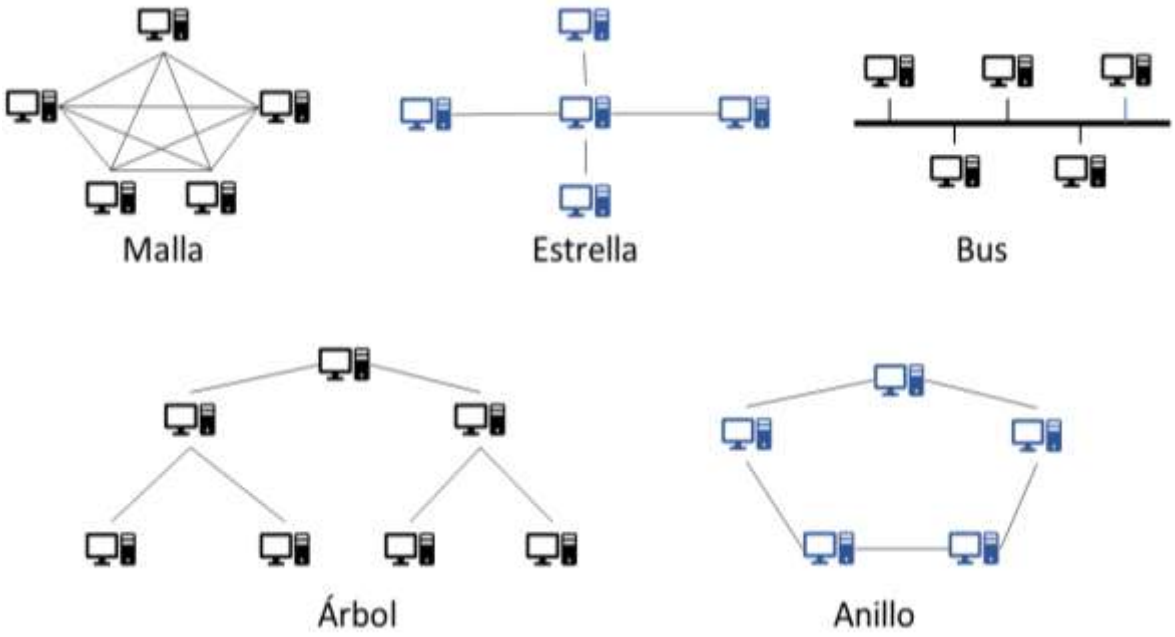


Figura 7 Topologías de red [11]

Fuente: Sistemas Industriales Distribuidos: una filosofía de la automatización. Ingeniería Técnica Telecomunicación. 3er curso. Dpto. Ingeniería electrónica, Universidad de Valencia. Alfredo Rosado Muños

- Estrella

La característica más relevante de esta topología es que tiene un nodo central conectado directamente con cada uno de los otros nodos que componen la red, esto permite que exista mucha más velocidad de comunicación entre los nodos. La ventaja de implementar este tipo de red es que en el caso de que exista un fallo en alguno de los nodos periféricos, este fallo no afecta el funcionamiento de toda la red, sin embargo, toda la carga del tráfico de datos lo maneja el nodo central, por lo

que un mal funcionamiento de este nodo puede dejar afectar la operatividad de toda la red [5].

- Bus

La topología de bus tiene una sola conexión de comunicaciones al cual se le denomina bus de datos o simplemente bus, en el cual se conectan cada uno de los nodos, por lo que todos los dispositivos comparten el mismo canal de comunicación. Entre sus ventajas, el despliegue de esta topología requiere menos cableado si se le compara con una topología de estrella, además resulta más sencillo conectar nuevos dispositivos, lo que implica una fácil escalabilidad. Sin embargo, por su distribución es bastante difícil de descubrir el origen de alguna falla, además si se compromete la integridad del bus de datos toda la red dejaría de funcionar [5].

- Árbol

Esta topología es una especie de unión entre las topologías de estrella y la topología de bus ya que es una agrupación de redes en estrella interconectadas por un bus de datos. Debido a su distribución es una de las redes que se pueden escalar de manera más sencilla. Si existe una falla en un enlace que se interconecta con el nodo central, este nodo permanecería desconectado incluyendo sus nodos periféricos.[5].

- Malla o total

La conexión entre los dispositivos que conforman una topología de malla es de cada nodo periférico a todos los demás nodos periféricos (todos a todos). Lo que quiere decir que la información perteneciente a determinado nodo podría llegar tomar distintas rutas por lo que, en una posible falla de un nodo, la información tendrá la posibilidad de tomar cualquiera de las otras rutas a través de los otros nodos para llegar al destino. Sin embargo, una de las desventajas de esta topología es que cada punto de conexión se encuentra con la sobrecarga de información de los demás nodos, lo que implica un aumento de retardos en la comunicación [5].

5.1.2.5 PROTOCOLOS DE COMUNICACIÖN

Existen infinidad de protocolos de comunicación y distintas tecnologías que pueden ser implementadas en la medición inteligente, sin embargo, se pueden dividir en tres áreas: HAN, WAN, NAN. Los protocolos que se utilizan en una red de comunicación NAN son los encargados de interconectar los medidores eléctricos inteligentes entre sí. En la figura 8 se describen distintos tipos de protocolos de comunicación que se pueden llegar a utilizar para la implementación de una red inteligente. [4]

Subredes	Tecnologías de comunicación
HAN	Ethernet, Wireless ethernet, Power Line Carrier (PLC), Broadband over power line (BPL), ZigBee
NAN	PLC, BPL, Metro Ethernet, Digital Subscriber Line (DSL), EDGE, HSPA, UMTS, Long Term Evolution (LTE), WiMax, Frame relay
WAN	Multiprotocol Label Switching (MPLS), WiMax, LTE, Frame relay

Figura 8 Tecnologías de telecomunicaciones [4]

Fuente: Javier Bernardo Cabrera Mejía. Diseño de la red de telecomunicaciones para el sistema de medición avanzada (AMI) de energía eléctrica en el centro histórico de la ciudad de Cuenca. Quito,Ecuador, 2014

5.1.2.5.1 PLC

La tecnología de comunicaciones por la línea de potencia o PLC por sus siglas en inglés Power Line Communications; utiliza la infraestructura de la red de suministro eléctrico como red base de comunicaciones. La información se transmite a través de la línea de alimentación como una señal de alta frecuencia de baja tensión que está acoplada a la señal de potencia de baja frecuencia de alta tensión. Entre las desventajas de esta tecnología se encuentran los errores de transmisión de datos por las distintas interferencias de campos electromagnéticos en la red, la velocidad de transmisión y el ancho de banda son muy bajos por lo que es ideal para redes NAN y HAN ya que se limitan a un rango de 1 km [4] [7].

5.1.2.5.2 FIBRA ÓPTICA

La fibra óptica funciona mediante el envío de impulsos de luz la cual se modula para llevar la señal de información a través de un núcleo de vidrio el cual se une a un receptor que recibe dichos impulsos y realiza el debido tratamiento y adecuación de la señal. Generalmente se emplea para comunicar centros de gestión con subestaciones. La mayor ventaja de la fibra óptica es que proporciona una tasa de transmisión bastante alta, alrededor de 10 Gigabits si se utiliza una longitud de onda y de 40 Gbps a 1.600 Gbps multiplexado por división de longitud de onda (WDM). También tiene alto rendimiento y alta confiabilidad. La fibra óptica se utiliza comúnmente para conectar zonas alejadas, de largo recorrido tales como los que se encuentran en redes WAN, MAN. [4] [7].

5.1.2.5.3 RED CELULAR

Lo que se buscaba solucionar con la creación de las redes celulares era la comunicación de voz y enlace de datos, sin embargo, se han implementado en sistemas de medición avanzada AMI, puntualmente en sistemas SCADA para el monitoreo de dispositivos de manera remota. La mayor ventaja a la hora de utilizar esta tecnología es su gran cobertura, ya que se cuenta con la infraestructura ya construida y no es necesario crear instalaciones adicionales. Esto es lo ideal para aplicaciones WAN [4]. Tener la infraestructura de comunicación creada infiere en un

bajo costo de mantenimiento siendo una solución competitiva en términos económicos. [7]

5.1.2.5.4 ZigBee

ZigBee es el nombre con el cual se le conoce a un conjunto de protocolos de comunicación inalámbrica de alto nivel con radiodifusión digital, es ideal para soluciones de bajo consumo energético y costo. La tecnología ZigBee sienta sus bases en el estándar IEEE 802.15.4 y normalmente es utilizado para soluciones que tengan tasas bajas de transferencia de datos siendo fiable y permitiendo la creación de redes de diferentes topologías, como estrella, árbol y malla por lo que ZigBee es ideal para aplicaciones utilizadas en redes HAN [4]

5.1.2.5.5 WiFi

Es una de las tecnologías diseñadas para diferentes ambientes ya sean industriales y hogareños. La tecnología WiFi o estándar 802.11b permite la conexión de una cantidad alta de usuarios utilizando una misma banda de frecuencias con muy poca interferencia. En términos económicos y de eficiencia resulta más beneficioso a la hora de implementar una red LAN que esta sea inalámbrica y no cableada ya que permite la movilidad de los dispositivos conectados. Aunque presenta algunas limitaciones debido a interferencias por radiaciones electromagnéticas sigue siendo la más utilizada en aplicaciones de redes NAN [4]

5.2 ESP (ESPRESSIF)

Espressif Systems es una multinacional originaria de Shanghai, China que se dedica al desarrollo de tecnología enfocado a soluciones de IoT de comunicaciones inalámbricas de bajo consumo. Espressif Systems ha desarrollado una amplia familia de chips que cumplen con los objetivos de esta empresa, entre estas familias de chips existe la serie de ESP8266 que tiene varios DevKits (kits de desarrollo que tiene todos los dispositivos y elementos necesarios para facilitar programación e implementación) y que por su preció, disponibilidad del mercado y tareas que puede desarrollar se utilizará el Devkit NodeMCU en este proyecto [14]

El kit de desarrollo NodeMCU v3 cuenta con un Soc (Sistema en Chip) ESP8266 que es el que proporcionará la conexión WiFi (IEEE 802.11), también posee un microcontrolador Tensilica L106 de 32 bits, memoria RAM 50KB que funciona a una frecuencia de 160MHz, con una entrada analogica de 10 bits (ADC) y 17 GPIO. Además, este kit también cuenta con un conversor serie-USB.[14]

5.2.1 Modos

Todos los elementos que se interconectan en una red WiFi se les conoce como estaciones (STA) y a su vez es proporcionado por un Access Point que funciona como punto central de comunicación para variedad de estaciones. [15]

La placa ESP8266 tiene versatilidad en su funcionamiento ya que puede actuar como una estación conectándose a red WiFi y como Access Point Wireless estableciendo su propia red por lo que podemos conectar otros puntos (estaciones). Sin embargo, no es necesario elegir entre una o la otra ya que puede operar en los dos modos, modo estación y modo AP con lo que se puede construir una topología de malla.[15]

5.2.1.1 Estación

Cuando se requiere conectar la placa (ESP8266) a una red WiFi se debe configurar como station mode o modo estación:

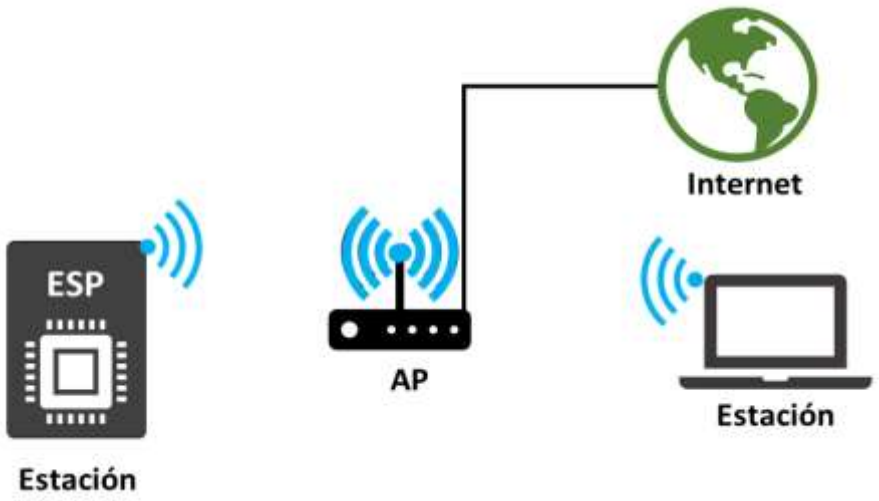


Figura 9 Modo estación en ESP8266 [15]
Fuente: Grokhotkov, Ivan, “ESP8266 Documentación Arduino Core”, 2017.

Si se llega a perder la conexión o se reinicia el módulo, éste se conecta de nuevo al último punto de acceso configurado ya que se guardan los datos del último AP en la memoria flash (ROM). Utilizando estos datos previamente almacenados la placa también volverá a conectarse siempre y cuando no se haya modificado el sketch, siempre y cuando el código no altere el modo WiFi o las credenciales.[15]

5.2.1.2 Punto de acceso (AP - Access Point)

Por definición un punto de acceso es un dispositivo electrónico inalámbrico que suministra un acceso a la red a distintas estaciones. La máxima cantidad de stations

que pueden estar conectadas al mismo tiempo al SoftAP puede establecerse de 0 a 8, pero por defecto es 4.[15]

Figura 10 Modo punto de acceso en ESP8266 [15]
Fuente: Grokhotkov, Ivan, “ESP8266 Documentación Arduino Core”, 2017.

5.2.1.3 Cliente seguro (Client BearSSL)

La seguridad en las conexiones de los servidores y clientes necesitan mucha más memoria y capacidad de procesamiento si se les compara con conexiones no tan seguras ya que se requiere habilitar algoritmos criptográficos. Se debe tener en cuenta que solo se puede realizar una conexión segura entre el servidor o cliente a la vez ya que la placa ESP8266 presenta poca memoria RAM [15]

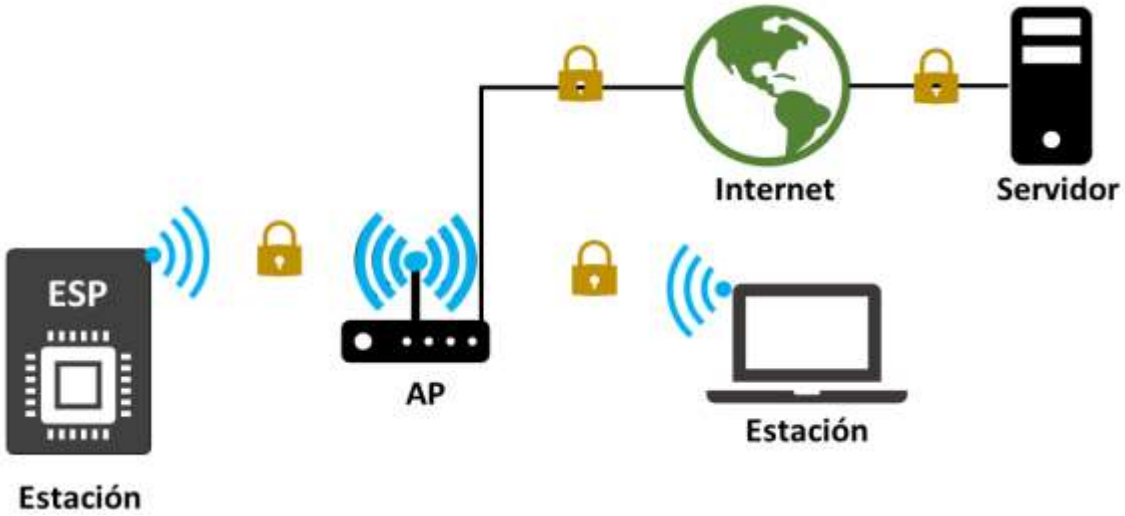


Figura 11 Modo cliente seguro en ESP8266 [15]
Fuente: Grokhotkov, Ivan, “ESP8266 Documentación Arduino Core”, 2017.

5.2.1.4 ESP-Now

ESP-NOW es uno de los muchos protocolos desarrollados por Espressif System, que es la empresa que fabrica los módulos ESP; y que permite la conexión de múltiples dispositivos entre sí (cualquier topología de red) sin usar Wi-Fi, por lo que este método es aún más eficiente en términos de energía. [16]

Este protocolo, al igual que el WiFi utiliza la banda de los 2,4 GHz. Por lo que se debe realizar el emparejamiento entre dispositivos antes de comunicarse. Una vez emparejado, la conexión es segura y se realiza el envío de paquetes cortos de máximo 250 bytes de manera unidireccional o bidireccional, que para alcances e implementación de este proyecto serán suficientes [16].
Utilizando ESP Now, como se mencionó anteriormente, se pueden implementar varias topologías de red entre los dispositivos conectados, desde una topología de malla hasta topología en estrella que será la que se implementará en este proyecto.

Algunas de las ventajas que ofrece ESP Now son:

- Comunicación unidifusión cifrada y no cifrada;
- Devolución de llamada de envío que puede configurarse para el intercambio de información a la capa de aplicación sobre el fracaso o éxito de la transmisión.

La tecnología ESP-NOW también tiene las siguientes limitaciones [16]:

- Pares cifrados limitados. Se admiten 10 pares cifrados como máximo en el modo Estación; 6 máximo en configuración de SoftAP o SoftAP+Estación;
- Admiten múltiples pares sin cifrar, sin embargo, el total de conexiones debe estar por debajo de los 20, incluidos los pares cifrados;
- La carga útil está limitada a 250 bytes.

5.3 FIREBASE

En esencia Firebase es una plataforma móvil de Google. Fue adquirida en 2014 y permite el desarrollo de aplicaciones móviles y aplicativos web que integran todas las funciones de Google Cloud Platform y busca la fácil integración de todos los servicios y herramientas que ofrece Google para los desarrolladores, además de una basta documentación y gran comunidad que hacen que sea una alternativa bastante fiable.

Algunos de sus clientes más importantes son empresas como The New York Times, The Economist, Alibaba, Todoist que utilizan entre muchas de los servicios y herramientas que ofrecen para el desarrollo como Analytics, Authentication, Cloud Firestore, Cloud Functions, Cloud Messaging (FCM), Hosting, Firebase ML, Performance Monitoring, Realtime Database, Cloud Storage, entre muchos otros y existentes y en desarrollo [13].

Una de las ventajas de Firebase es que es gratis hasta ciertos límites de uso y luego de que se sobrepasan estos límites se cobra por uso. En la figura 12 se muestra uno de los valores que tiene el producto, más exactamente el producto de Realtime Database que es uno de los que más se utilizará en este proyecto.

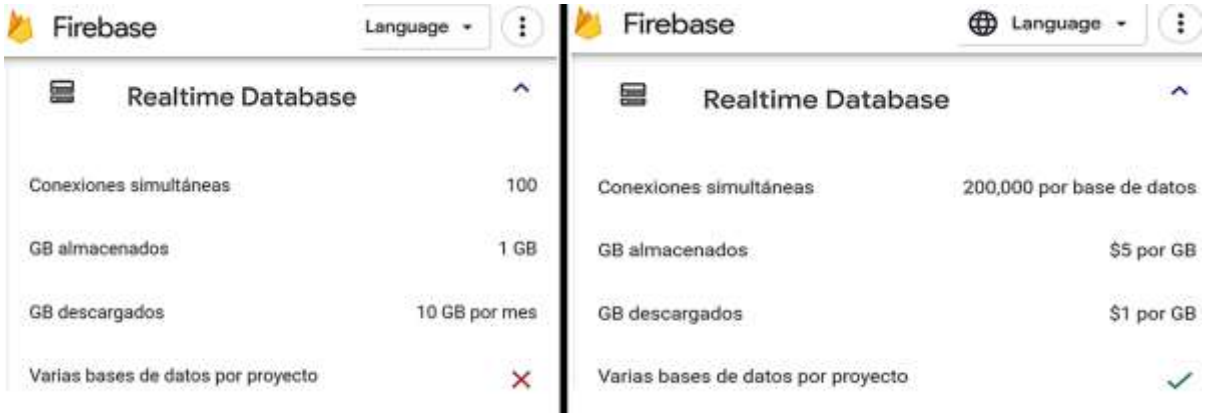


Figura 12 Precios Firebase (Izquierda plan gratis, derecha plan de paga) [13]
Fuente: Google Inc, Firebase, 2022.

Para efectos de este proyecto no se sobrepasa el límite mensual de uso por lo que la utilización de Firebase para el alcance de este proyecto no genera costo adicional.

5.4 IEC 61968-9

IEC o también conocida por su sigla en español CIE (Comisión Electrotécnica Internacional) es una organización líder presente en más de 170 países que se encarga del desarrollo y publicación de los estándares en los campos de la electrónica, eléctrica y tecnologías relacionadas, conocidas como electrotecnia, para sustentar la infraestructura de calidad y comercio internacional de productos eléctricos y electrónicos.

En el estándar IEC 61968 se define la integración de sistemas de medición que incluirá los sistemas de lectura de medidores automatizados tradicionales con otros sistemas y funciones comerciales dentro del alcance de la IEC 61968, el alcance de este estándar es el intercambio de lectura de medidores, información de eventos y control entre sistemas. Los detalles específicos de los protocolos de comunicación que emplean los sistemas están fuera del alcance.[12]

5.5 XAMARIN FORMS

Xamarin ofrece dos maneras de integrar las soluciones multiplataforma, una llamada Xamarin Native que como su nombre lo indica su nombre permite diseñar la parte de la vista (interfaz de usuario UI) con código nativo de cada plataforma (Android, UWM - Universal Windows Platform, iOS); y Xamarin Forms, la cual es la

que se implementará ya que existe un código compartido que permite tener la misma solución para las diferentes plataformas sin necesidad de programar en código nativo cada una de estas plataformas (figura 13).

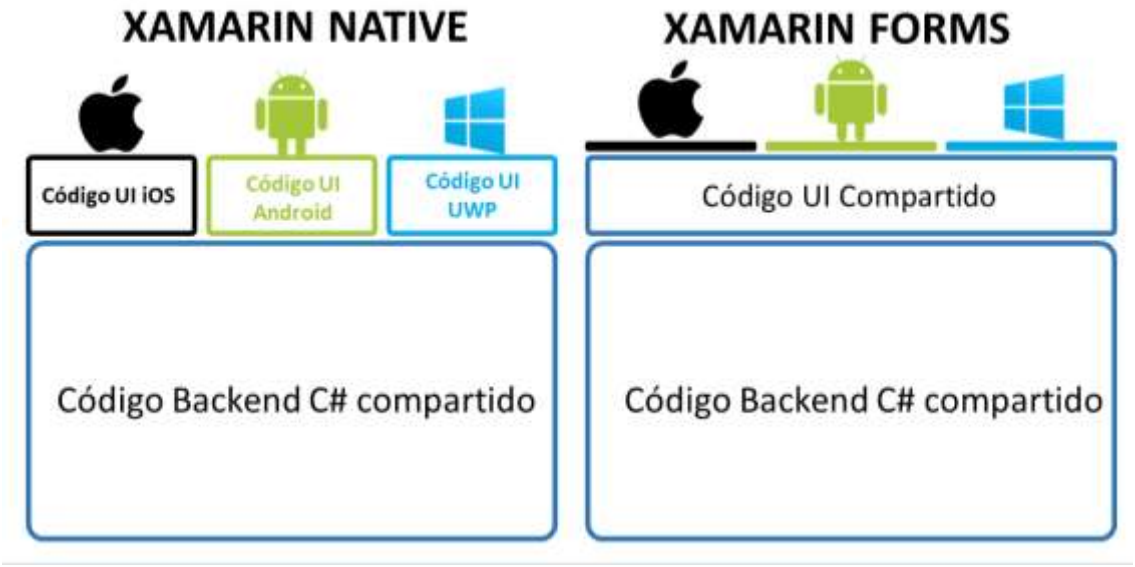


Figura 13 Xamarin [40]
Fuente: David Britch, Craig Dunn, Hemant Arya, Justin Johnson, “What is Xamarin? - Xamarin | Microsoft Docs”, Diciembre, 2021.

5.5.1 MVVM (Model-View-ViewModel)

El patrón MVVM (Model-View-ViewModel) es un patrón de arquitectura de software que separa la lógica comercial y la interfaz de usuario (UI) lo que hace que la escalabilidad y el mantenimiento sea más fácil de implementar (figura 14). [17]

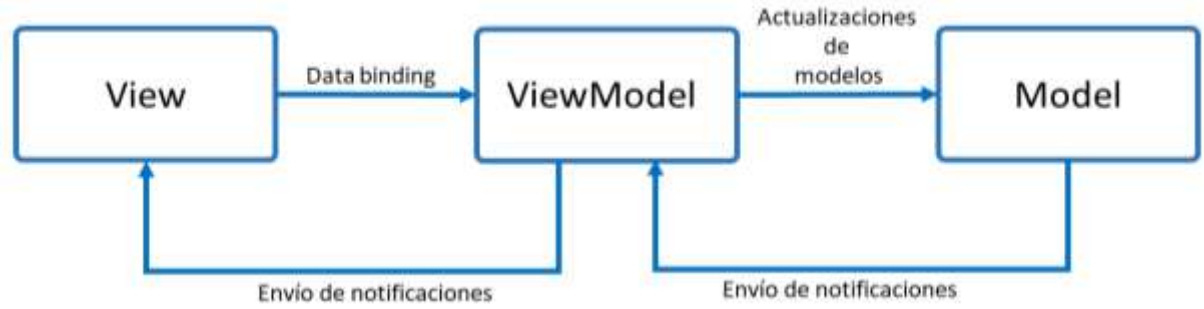


Figura 14 Patrón MVVM [17]
Fuente: Britch David; Schonning, Nick; Dunn, Craig; JohnCOsborne, “The Model-View-ViewModel Pattern”, Agosto, 2021.

En la Vista (View) en donde se implementa la interfaz de usuario (UI). En el caso de Xamarin cada vista se realiza en XAML que es un lenguaje de formato utilizado en .NET. La Vista-Modelo (ViewModel) es la que enlaza las propiedades y comandos

que se ejecutan en la vista (View) y notifica los cambios realizados por el usuario, es decir los gestos que este realice, como scroll, que es deslizar la pantalla ya sea vertical u horizontalmente; tap que simplemente es tocar la pantalla, swipe, entre otros. Estás propiedades se enlazan (binding) con atributos propios de cada control contenido en el archivo de vista (formato XAML). Por último, la sección (clases) de modelo (Model) que encapsula todos los datos de la aplicación.

Algunas de las buenas prácticas para implementar este patrón MVVM es utilizando un patrón singleton a la hora de instanciar los diferentes objetos, como clases de ViewModel lo que aseguraría tener solo una instancia de dicha clase y proporcionar un acceso global. A su vez también utilizar una clase que sea el punto de acceso global anteriormente mencionado. Crear también una clase en ViewModel por cada página creada en la sección de View y enlazarlas mediante gestos o atributos como binding context.

5.5.2 Microcharts

Mircrocharts es una librería para Xamarin creada por el consultor de Xamarin en Orange Applications for Business, Alois Deniel para mostrar gráficas de manera sencilla. Deniel utilizó hizo uso SkiaSharp (se utiliza en .NET y C# como sistema de gráficos 2D) por lo que se tiene variedad en gráficos que se pueden implementar como se muestra en la figura 15, 16, 17, 18, 19 y 20 en donde se puede observar los distintos gráficos que se pueden implementar utilizando esta librería.[18]

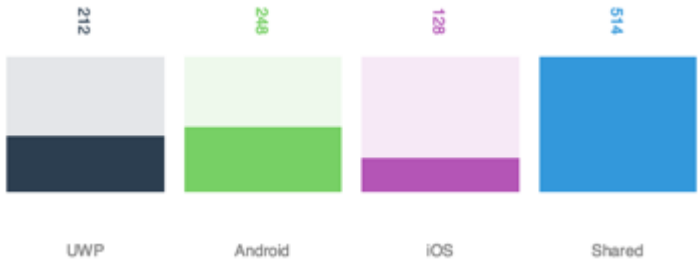


Figura 15 Gráfico de barras [18]
Fuente: Aloïs Deniel, “Microcharts: Elegant Cross-Platform Charts for Every App”, Octubre, 2017.

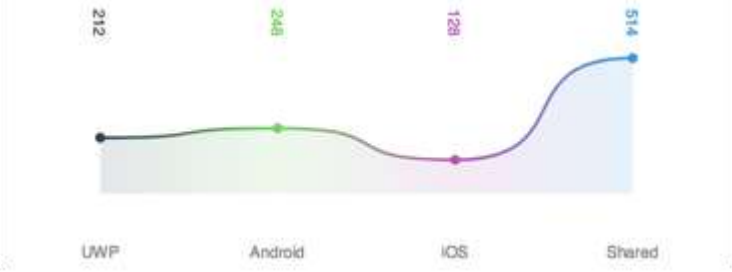


Figura 16 Gráfico de línea [18]
Fuente: Aloïs Deniel, “Microcharts: Elegant Cross-Platform Charts for Every App”, Octubre, 2017.

Figura 17 Gráfico de puntos [18]
Fuente: Fuente: Aloïs Deniel, “Microcharts: Elegant Cross-Platform Charts for Every App”, Octubre, 2017.

Figura 18 Gráfico Radial Gauge [18]
Fuente: Aloïs Deniel, “Microcharts: Elegant Cross-Platform Charts for Every App”, Octubre, 2017.

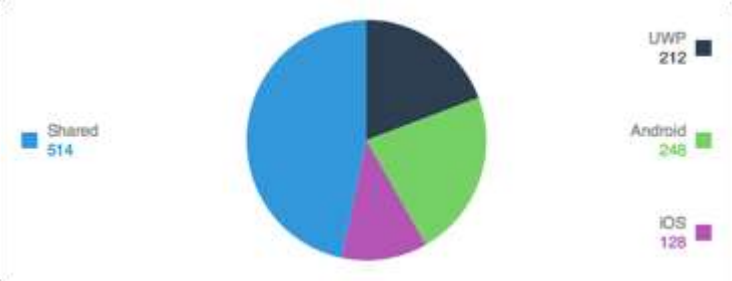


Figura 19 Gráfico circular (Pie) [18]
Fuente: Aloïs Deniel, “Microcharts: Elegant Cross-Platform Charts for Every App”, Octubre, 2017.

Figura 20 Gráfico de radial [18]
Fuente: Aloïs Deniel, “Microcharts: Elegant Cross-Platform Charts for Every App”, Octubre, 2017.

5.6 SISTEMA INTELIGENTE

Los términos Machine Learning e inteligencia artificial a menudo son confundidos y en algunos casos dándoles el mismo significado, pero recordemos que el Machine Learning es una rama de la inteligencia artificial. En un principio el Machine Learning se utilizaba como programa de entrenamiento para la inteligencia artificial, pero hubo una época en la que la mayoría de las investigaciones sobre la inteligencia artificial, o al menos las más importantes habían sido detenidas, pero al contrario de lo que pasaba en las investigaciones de la inteligencia artificial, en las relacionadas con Machine Learning empezaron a aumentar [33] [34]. EL área de Machine Learning empezó a resaltar con la resolución de problemas prácticos mediante la teoría de la probabilidad y estadística y no tanto al conocimiento (knowledge-driven) como se venía realizando [32], es decir que se deben de tener una cantidad considerable de datos para que al ser entrenadas estas redes tengan una solución más precisa.

Existen en Machine Learning tres tipos de algoritmos: Aprendizaje supervisado que se refiere al aprendizaje mediante ejemplos los cuales ya se conoce la entrada y la salida deseadas, Aprendizaje no supervisado en las cuales se identifican patrones y correlaciones que tienen los datos para determinar la agrupación de los mismos ya que en este tipo de aprendizaje no se tienen las salidas; y por último el aprendizaje por refuerzo que consiste en tener procesos reglamentados ya sea por acciones, valores finales y/o parámetros haciendo que la maquina aprenda por ensayo y error. A su vez, entre estos tipos de algoritmos existen otros tipos de algoritmos como de regresión en el que se estima la salida centrándose en la variable dependiente y teniendo una predicción y pronósticos de esto datos; algoritmos de agrupación que son utilizados en aprendizaje no supervisado ya que no se tienen definidos los grupos al que pertenecen los datos por lo que el algoritmo tomara cada uno como un grupo diferente y encontrando correlaciones entre estos.

5.6.1 Google Colaboratory

Es una herramienta desarrollada por Google Inc en la cual se pueden ejecutar programas escritos en Python con la ventaja de tener acceso gratuito a GPU por lo que no es necesario tener un equipo de cómputo sofisticado ya que este solo deberá tener acceso a un buscador y el programa se ejecutará en GPU’s remotas. Esta es la herramienta que se utilizará para el desarrollo del sistema inteligente.

5.7 AMI EN COLOMBIA

La Comisión de Regulación de Energía y Gas (CREG) presenta varios documentos sobre medición avanzada (AMI) que son públicos en los cuales se pueden encontrar resoluciones y análisis realizados por la misma CREG y otros estudios contratados [24], además del marco legislativo y regulatorio que debe estar presente para la

implementación de este tipo de infraestructura. En el marco legislativo se encuentra la Resolución 40072 de 2018 en la cual se establecen los mecanismos que se deben tener en cuenta al momento de implementar una infraestructura de medición avanzada específicamente sobre el servicio de energía eléctrica [23].

En esta resolución se hace alusión a estudios realizados por la Unidad de Planeación Minero-Energética (UPME) sobre Smart Grids (Smart Grids Colombia Visión 2030), en la cual junto con la Universidad Nacional de Colombia se definen las funcionalidades mínimas que deben tener los medidores inteligentes en Colombia.

Es este estudio realizado en primera instancia se identifican las funcionalidades mínimas que tienen normalmente los medidores inteligentes entre las cuales se definen dos grupos de funciones, las inherentes al medidor y las soportadas por el medidor. En este primer grupo están las funcionalidades relacionadas a la operación bajo el esquema AMI que son necesarias para que dicho medidor se pueda integrar a la red y en el último grupo (las soportadas por el medidor) se encuentra todo lo relacionado con esas funciones que producen información necesaria para el funcionamiento de la red. En la figura 21 se observan todas estas funcionalidades según el grupo.

Grupo funci.	Símbolo	Descripción de la funcionalidad
INHERENTES AL MEDIDOR	ALM	Almacenamiento de datos en el medidor
	COB	Permite la comunicación bidireccional por diferentes medios
	SEG	Soporta unas comunicaciones de datos seguras
	SIN	Permite la sincronización de tiempos del medidor con el sistema de medida
	A&C	Permite la configuración (intervalos de lectura, tarifas, etc.) y actualización (Software, Firmware, etc.) remota del medidor
SOPORTADAS POR EL MEDIDOR	USU	Acceso del usuario a la información del medidor
	LRM	Lectura remota del medidor
	TAR	Soporta esquemas de tarificación avanzada
	CDL	Conexión y desconexión del suministro de energía y/o limitación de potencia de forma remota
	FRA	Prevención y detección de fraudes
	GD	Soporta la importación y exportación de energía
	CAL	Proporciona medidas de calidad de potencia
	PRE	Soporta la implementación de modo prepago
	HAN	Soporta integración de redes de automatización del hogar (HAN)

Figura 21 Funcionalidades de medidores inteligentes [25]
Fuente: Universidad Nacional de Colombia. Facultad de Ingeniería, Departamento de Energía Eléctrica y Electrónica, "Definición de Funcionalidades Mínimas de un Medidor Inteligente en Colombia", Bogotá, Universidad Nacional, 2016

Sin embargo, las funcionalidades de la figura anterior son las que pueden tener los medidores inteligentes. Las funciones que se muestran en la figura 22 son las funciones que debe tener como mínimo los medidores inteligentes según ocho criterios de evaluación: Funcionalidad para agentes involucrados (comercializador, cliente y operador de la red), seguridad, comunicaciones, interoperabilidad, beneficio, costo, marco regulatorio que rodea la funcionalidad en Colombia y la priorización y peso de algunas funcionalidades sobre otra. En cuanto a la relación de los grupos y funciones, algunas de estas han sido cambiadas de grupo como la función LRM (Lectura Remota del medidor) y GD (soportan la importación y exportación de energía) que anteriormente era soportada por el medidor, pero ahora aparece en el grupo de funciones inherentes, caso contrario con la función COB (Comunicación bidireccional) que pasó de estar en el grupo de inherentes al medidor a funciones soportadas por el medidor [25].

FUNCIONALIDADES SELECCIONADAS		
Grupo funci.	Símbolo	Descripción de la funcionalidad
INHERENTES AL MEDIDOR	LRM	Lectura remota del medidor
	GD	Soporta la importación y exportación de energía
SOPORTADAS POR EL MEDIDOR	COB	Permite la comunicación bidireccional por diferentes medios
	USU	Acceso del usuario a la información del medidor
	FRA	Prevención y detección de fraudes

Figura 22 Funciones mínimas de medidores inteligentes [25]
Fuente: Universidad Nacional de Colombia. Facultad de Ingeniería, Departamento de Energía Eléctrica y Electrónica, "Definición de Funcionalidades Mínimas de un Medidor Inteligente en Colombia", Bogotá, Universidad Nacional, 2016

Las funciones mínimas que deben tener están definidas por la norma NTC 6019, excepto la función GD que se describe según lo establecido en la resolución CREG 038-2014.

- LRM (lectura Remota del medidor)

"6.3.1.2 Requisitos para administración de información: El software del sistema AMI debe permitir la lectura local y remota, acompañado de la fecha y hora." [25]

- GD (Soporta la importación y exportación de energía)

Artículo 27. *"Sellado de los elementos del sistema de medición: a) Suministrar e instalar sellos y mantener el registro correspondiente, para detectar manipulaciones e interferencias sobre los medidores, los transformadores de medida, las borneras de prueba y demás elementos susceptibles de afectación y protección mediante un sello".* [25]

- FRA (Prevención y detección de fraudes)

"6.3.3 Requisitos de gestión de eventos y alarmas: el software de gestión del sistema AMI debe permitir a detección de intervenciones no autorizadas al equipo". [25]

- COB (Permite la comunicación bidireccional)

"6.3.2 Requisitos de configuración, control y operación de componentes: el sistema debe estar orientado a servicios que permitan la conectividad e interfaces con otros sistemas bajo los modelos de integración de información CIM (IEC 61968, IEC 61979) o MultiSpeak." [25]

- USU (Acceso del usuario a la información del medidor)

"6.1.3 Las unidades de medida pueden contar con visualizaciones para que el cliente pueda leer su consumo, algunos de estos están incorporados en el medidor y otros son dispositivos separados del medidor." [25]

6 DISEÑO

La infraestructura de medición avanzada (AMI) en esencia es la unión de tres tipos de redes: HAN, NAN y WAN por lo que el diseño se basa en cómo estos tres tipos de redes se interconectan, qué componentes conforman cada red, las tecnologías de comunicación entre ellas, tipologías de red, protocolos de comunicación y demás elementos necesarios para la constitución de una red AMI a nivel residencial que se irán detallando en este documento. En la figura 23 se muestra un esquema general de la red AMI a Implementar, los elementos que se encuentran con línea discontinua son elementos que se encuentran fuera del alcance de este proyecto o que se implementarán parcialmente.

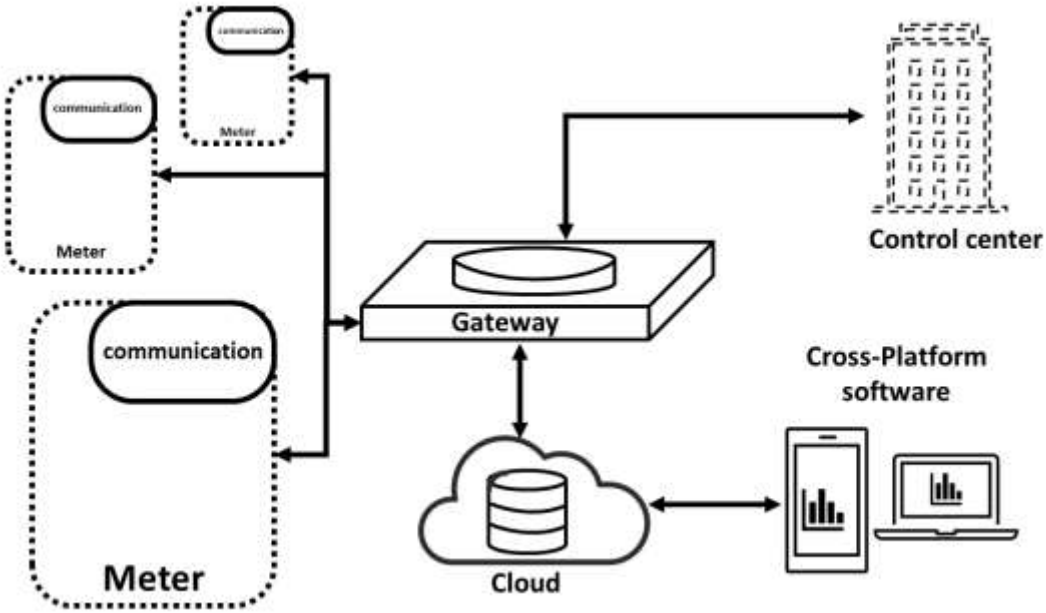


Figura 23 Infraestructura AMI
Fuente: “El Autor”.

6.1 HAN

La red HAN (Red de Área Doméstica) tiene muy poca cobertura, lo que concierne al hogar, es decir que las comunicaciones se extienden tan solo unos pocos metros en la residencia o instalaciones del cliente y permite la comunicación entre los diferentes dispositivos de IoT como sensores o electrodomésticos y el medidor inteligente. Esta red está a cargo del cliente y al contar con el medidor inteligente se puede recibir instrucciones sobre el uso de la electricidad o activar y desactivar cargas mediante el uso de eventos según lo establecido en la IEC 61968-9.[4][12]

6.1.1 Medidor inteligente

Para el estándar IEC 61968-9 los medidores hacen parte de los dispositivos finales, estos dispositivos finales, dependiendo de su tecnología, pueden o no tener diferentes capacidades, sin embargo, la de metrología es fundamental, así como se menciona en la norma IEC 61968-9 en el modelo de referencia: *"El medidor es una especialización de un dispositivo final. El dispositivo final puede contener una capacidad de metrología, puede contener una capacidad de comunicaciones, puede ser una unidad de control de carga y puede contener una combinación de muchos tipos diferentes de funciones … intenta describir el concepto de esencialmente una lista de compras de funcionalidad que puede estar disponible en el dispositivo final (lógico o físico) ...*

- *Tiene una identidad única*
- *Se gestiona como activo físico*
- *Puede recibir solicitudes de control*
- *Puede recopilar e informar valores medidos*
- *Puede participar en procesos comerciales de servicios públicos"* [12]

6.1.1.1 Dispositivo de medición

La función del medidor es registrar las diferentes medidas y utilizar estas medidas para la tarificación. En la IEC 61968-9, los medidores son un subtipo de dispositivo final tiene capacidad de metrología, puede o no tener capacidad de comunicación, puede o no puede tener capacidad de conexión / desconexión, o una serie de otras capacidades que son definidas por los diferentes proveedores. Hay que aclarar que la norma no restringe el uso de nuevas tecnologías ni busca definir un medidor, sino que simplemente da parámetros que podrían o no cumplir estos medidores, claro, respetando algunas funciones básicas como lo es la capacidad de metrología en el medidor. [12]

Teniendo en cuenta lo mencionado en el estándar IEC 61968-9, en cuanto a funciones que debe tener los medidores inteligentes, los establecidos por la UPME junto con la Universidad cumplen con estos criterios por lo que serán los que se utilizan de guía para el módulo de comunicación del medidor exceptuando algunas funciones.

El alcance de este proyecto en lo que refiere al medidor inteligente, solo se tiene en cuenta la transmisión de la información desde el módulo de comunicaciones del medidor (comunicación bidireccional) más no su capacidad de metrología, adecuación de la señal y demás funcionalidades que no estén estrictamente relacionadas con el módulo de comunicaciones por lo que al final se tendrán estas funcionalidades:

- LRM (lectura Remota del medidor)

- COB (Permite la comunicación bidireccional)
- USU (Acceso del usuario a la información del medidor)

6.1.1.2 Módulo de comunicaciones

El medidor inteligente debe ser capaz, según lo está estipulado en el estándar IEC 61968-9, de: Puede emitir eventos, puede recibir solicitudes de control, puede recopilar e informar valores medidos; por lo que el módulo de comunicaciones debe ser capaz de enviar y recibir información (comunicación bidireccional).[12]

Teniendo en cuenta la red a la que pertenece este componente (HAN) la cual no es de un alcance extenso y que dicha comunicación debe ser bidireccional se utiliza un módulo WiFi ESP8266-01 en modo de estación ya que cumple con los requerimientos de seguridad y requerimientos mínimos establecidos.

6.2 NAN

Este tipo de red es más extensa que una red HAN por lo que sirve de puente de comunicación entre los medidores inteligentes y el centro de control. Este puente de comunicación está dado por el concentrador de datos o Gateway. Es decir que permite la conexión entre múltiples HANs.

6.2.1 Gateway (Firebase - ESP)

Firebase es una excelente alternativa gratuita (hasta cierto límite, pero suficiente para el desarrollo de este proyecto) de base de datos en tiempo real en nube en la cual se almacenarán la información de cada medidor inteligente implementado.

El producto de Realtime Database de Firebase es una base de datos NoSQL alojada en la nube. Los datos se almacenan en formato JSON y se sincronizan en tiempo real con cada cliente conectado.

La comunicación entre el medidor inteligente y la base de datos alojada en Firebase no va a ser directa, esta tiene que pasar por el Gateway ya que al mismo tiempo que se almacena esta información en la base de datos de Firebase, dicha información también tiene que ser enviada para su debido procesamiento en el sistema de control y ser visible para el respectivo usuario en la aplicación [13].

6.2.1.1 Configuración en Firebase

En primera instancia, al estar la base de datos en tiempo real alojada en un servidor en la nube, se debe tener una forma de autenticación. Firebase ofrece variedad de opciones de autenticación como se muestra en la figura 24.

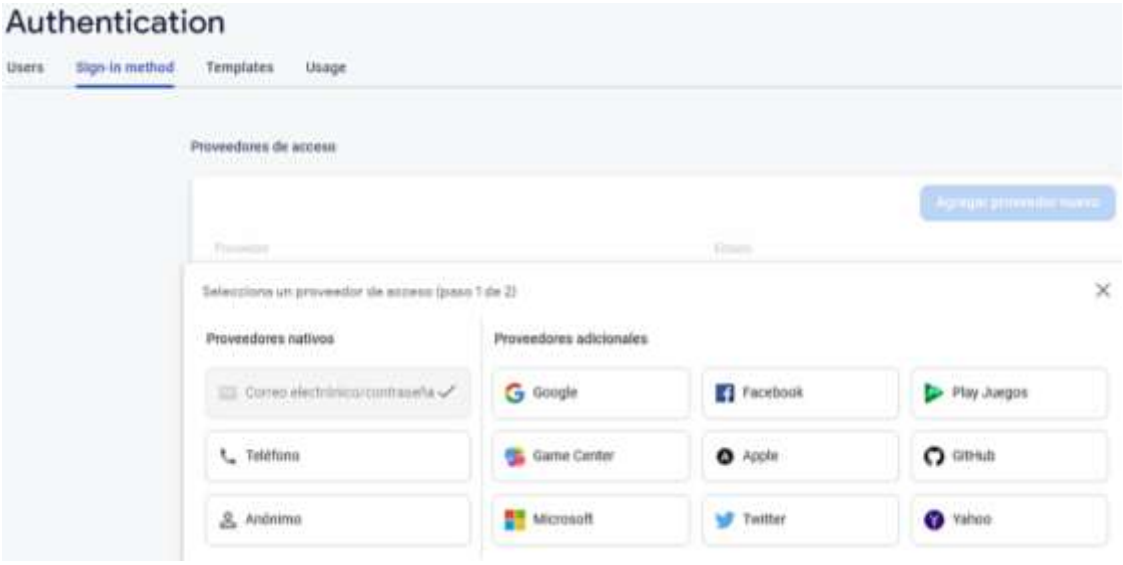


Figura 24 Autenticación en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

Aparte de la autenticación existen más validaciones como una clave de API Web, validación con cuenta de servicio y reglas de seguridad de la base de datos en tiempo real por lo que en temas de seguridad Firebase es una buena opción. Se elige la autenticación por correo electrónico en la cual se genera un ID de usuario (figura 25) el cual se utilizará para identificar el medidor ya que cada correo electrónico va a representar uno de estos.

Figura 25 Usuario en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

Además, también se deben configurar las reglas para el acceso al CRUD en el apartado de Reglas de Realtime database se debe configurar como se muestra en la figura 26.

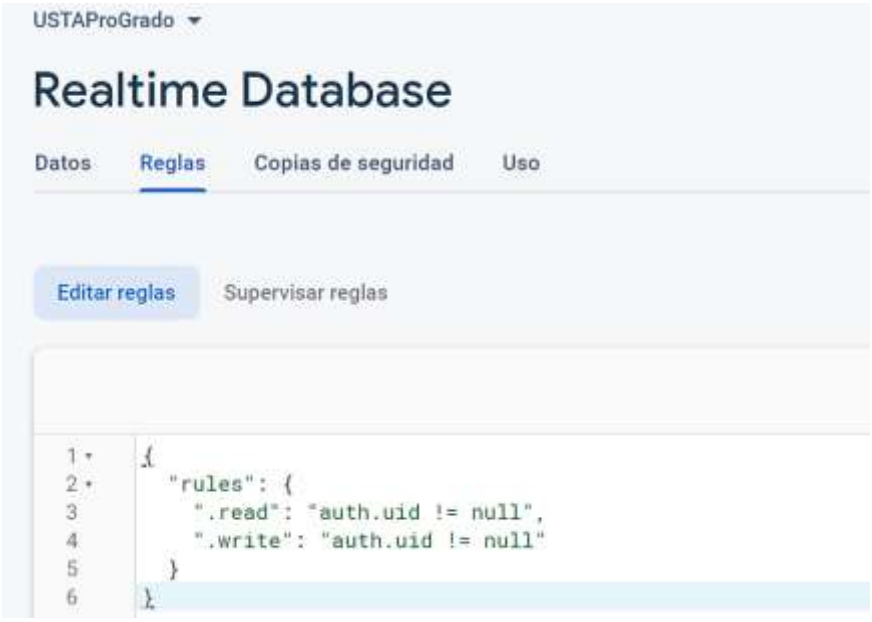


Figura 26 Reglas Realtime Database [13]
Fuente: Google Inc, Firebase, 2022.

Esta configuración de la base de datos se establece para que tanto como en la lectura y escritura se deba de tener autenticado con usuario y contraseña para poder acceder a dichas funciones. Esta autenticación se obtiene mediante un correo y clave que deben estar previamente registrados en las configuraciones de autenticación de Firebase. Cabe aclarar que al utilizar este modo de autenticación el tiempo de transmisión de datos es un poco más lento (unos cuantos milisegundos) que si se utilizara el método clásico que es con un token heredado ya que en este solo se utilizaban 40 bytes mientras que para la autenticación por correo (que es la que se va a implementar) se utilizan 900 bytes, pero esta última es más segura. En promedio, la velocidad de transmisión de datos de la ESP rondará los 200 a 500 milisegundos y para la primera autenticación será de 1 a 2 segundos.

6.2.1.2 Estructura de base de datos

Se crea en Firebase [13] un proyecto llamado “USTA ProGrado” en el cual se aloja la base de datos en tiempo real (se crea utilizando uno de los muchos servicios de Firebase llamado Realtime database). Ver figura 27. Se crearon cinco tablas en las cuales se va a almacenar los datos de cada medidor con la siguiente jerarquía [13]:

- meterTable
    - MAC
        - Gateway
        - Services
            - Service1

            - Service2
            - Service3
        - Status

- measureTable
    - MAC
        - Gateway
        - Service1
            - Status
            - Date
            - Value
        - Service2
            - Status
            - Date
            - Value
        - Service3
            - Status
            - Date
            - Value
- InfoTable
    - MAC
        - Service1
            - AproxMonth
            - BillDate
            - Fare
        - Service2
            - AproxMonth
            - BillDate
            - Fare
        - Service3
            - AproxMonth
            - BillDate
            - Fare
- statusTable
    - MAC
        - Services
            - Service1
            - Service2
            - Service3
        - Status
- sesionTable
    - MAC
        - Correo

Fuente: “El Autor”.

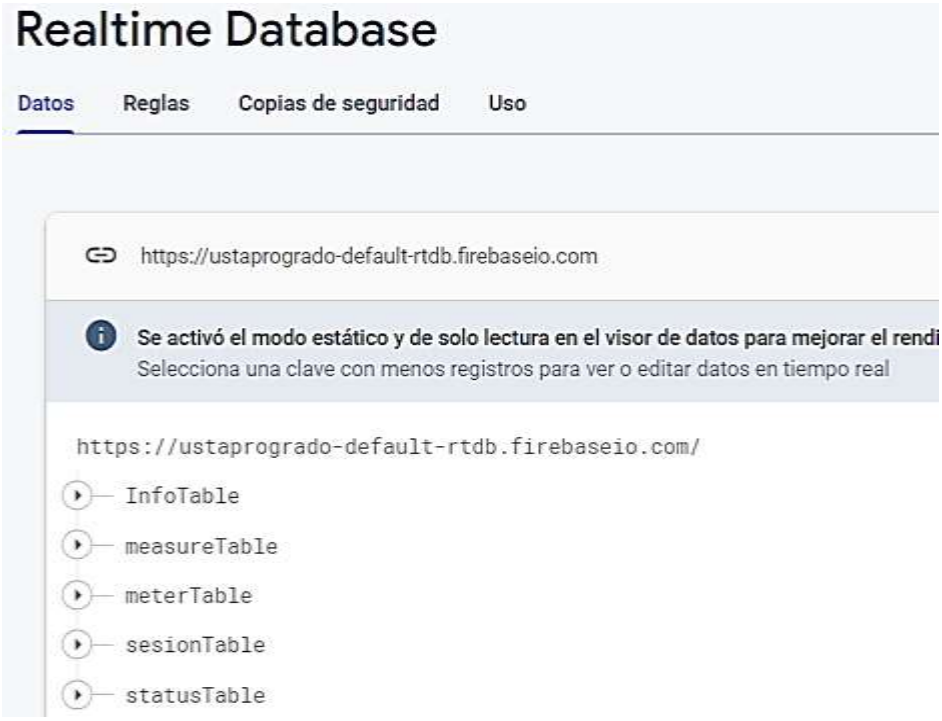


Figura 27 Tablas en Realtime Database [13]
Fuente: Google Inc, Firebase, 2022.

En la tabla meterTable se almacenan los datos necesarios para el sistema de control por medidor, es decir por dirección MAC del módulo de comunicaciones (cada MAC representa un medidor), los datos de Nombre en la variable *Name*, el estado actual del medidor en la variable *Status* y en la variable *Services* se encuentra en que servicios se puede obtener medidas, lo cual depende del medidor que se implemente (figura 28).

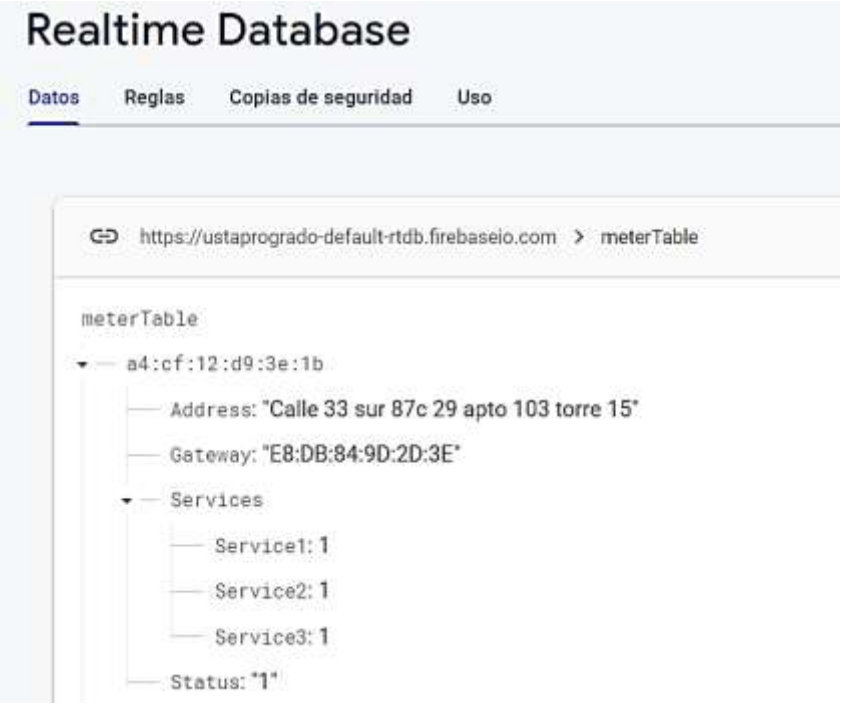


Figura 28 Tabla *meterTable* en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

En la tabla measureTable, al igual que en la tabla meterTable se almacenan los datos por medidor solo que en esta tabla se añaden los datos de las medidas tomadas por cada servicio en la variable *Service1*, *Service2* o *Service3* según sea el caso, en cada variable de servicio se encuentra el estado del servicio en el medidor en la variable *Status,* la fecha en la que se realizó cada medida en la variable *Date* y el valor medido en la variable *Value* (figura 29).

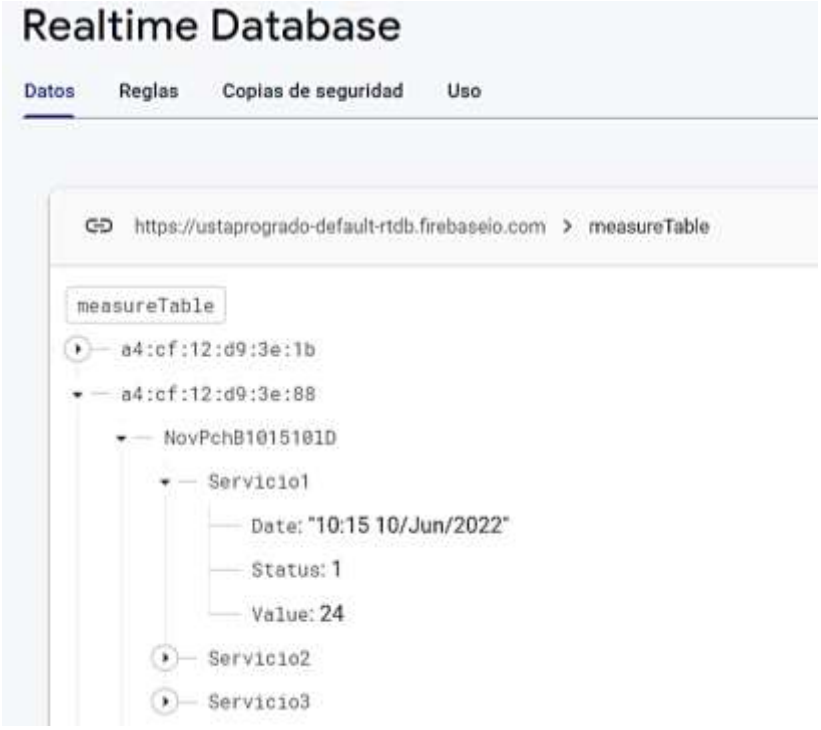


Figura 29 Tabla *measureTable* en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

En la tabla infoTable, al igual que en las tablas anteriores se almacenan los datos por medidor. Las variables que contiene esta tabla son *StatusClass* que indica el estrato donde se encuentra el medidor, y por cada uno de los tres servicios existe la variable *BillDate* que es una variable de tipo string en la que está la fecha de facturación del respectivo servicio, *Fare* que corresponde a la tarifa actual del respectivo servicio y *AproxMonth* que es la variable en donde se almacena el valor estimado del siguiente periodo de tiempo (figura 30).

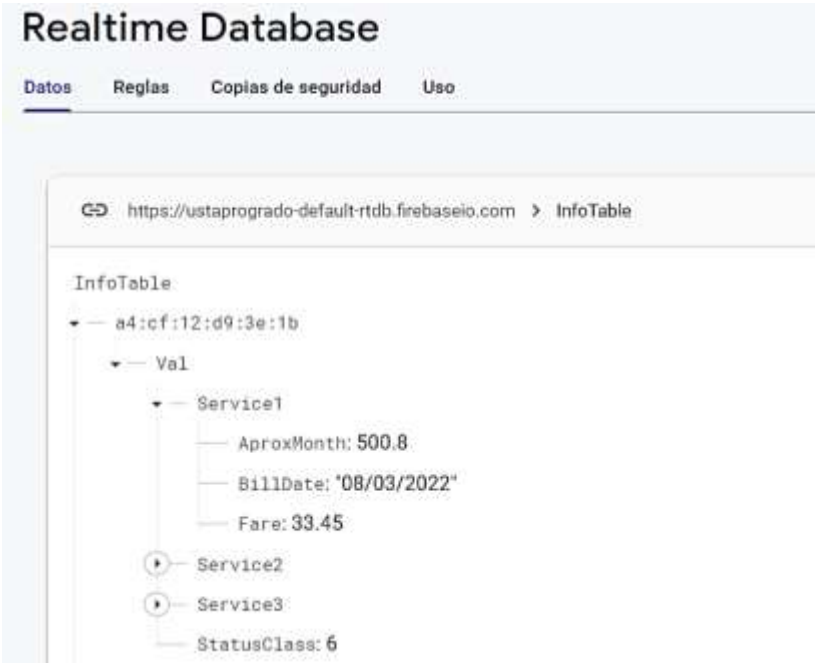


Figura 30 Tabla *InfoTable* en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

La tabla statusTable, será una tabla solo de lectura para el Gateway el cual realiza la consulta (búsqueda) por dirección MAC del medidor. En esta tabla se encuentran almacenados el estado del servicio que debería tener cada medidor, además también es donde se guardan las órdenes que da el Gateway al medidor correspondiente. En la variable *Status* es donde se da la instrucción que se da medidor. Estas instrucciones se explican más a adelante en detalle; y en la variable *Services* que contiene *Service1*, *Service2* y *Service3* se encuentra el estado que debe tener el servicio 1, 2 y 3 respectivamente (figura 31).

Figura 31 Tabla *statusTable* en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

Por último, en la tabla sesionTable se encuentran los correos autorizados para acceder a la información del medidor. La búsqueda también se realiza por dirección MAC y solo contiene una lista de correos (variable *Email*) con los correos autorizados (figura 32).

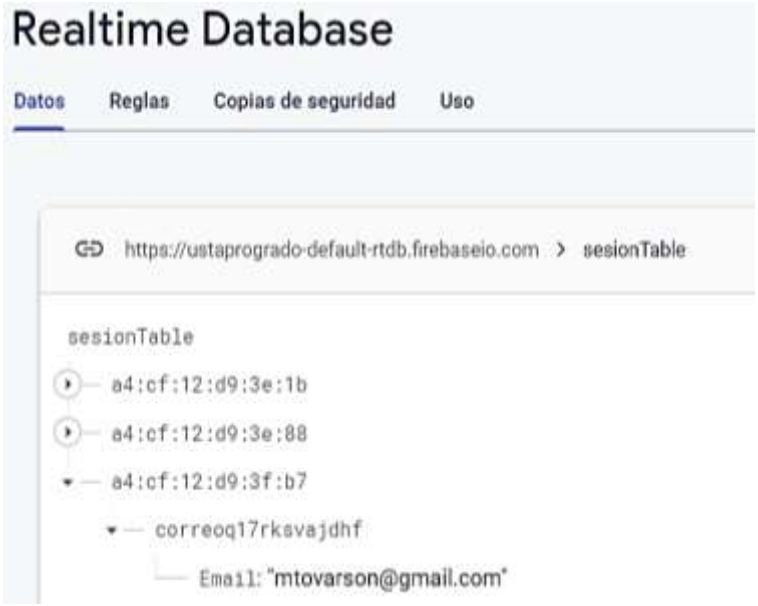


Figura 32 Tabla *sesionTable* en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

6.2.1.3 Configuración de Arduino NodeMCU

Para la programación de las placas ESP existen varios IDE's (ambiente de desarrollo integrado) que son compatibles con el firmware de estas placas. El más utilizado es el IDE de Arduino que con tan solo descargar unas cuantas librerías ya se podrá empezar a programar estas placas, además que cuenta con ventaja sobre otros IDE ya que en este IDE se puede emplear el uso de ejemplos que fueron previamente probados y resultan mucho más fácil y de gran ayuda a la hora de utilizar estas librerías que en un principio son desconocidas; otra de las ventajas que se pueden observar al utilizar el IDE de Arduino es su intuitiva manera de elegir la placa a programar como se muestra en la figura 33.

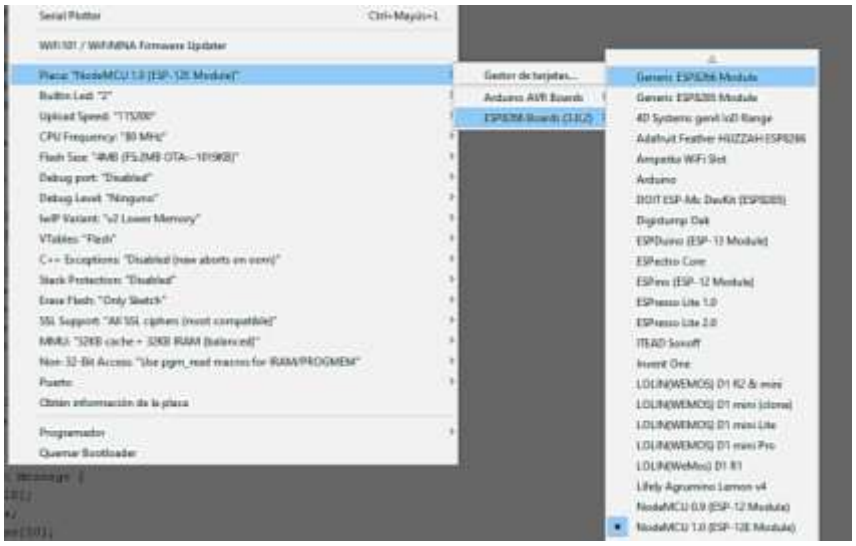


Figura 33 IDE de Arduino [39]
Fuente: Arduino IDE 1.8.19, Software | Arduino, 2022

Por otra parte, Platform io también hace parte de los IDE que son compatibles con el firmware de las placas ESP, así como de muchas otras. Platform io se puede instalar como extensión en el IDE de Visual Studio Code por lo que también cuenta con los beneficios de dicho IDE como IntelliSense que entre otras cosas ayuda a autocompletar (figura 34)

Figura 34 IDE de Platform io [19]
Fuente: Linhart, John; J.Nma; Suwatchai, “Firebase Arduino Client Library for ESP8266 and ESP32”, 2022.

La librería que se utilizará en la ESP8266 NodeMCU es “Firebase-ESP-Client” que es soportada por varios dispositivos de Espressif como Sparkfun ESP32 Thing, NodeMCU-32, WEMOS LOLIN32, TTGO T8 V1.8, M5Stack ESP32, NodeMCU ESP8266, Wemos D1 Mini (ESP8266), Arduino MKR WiFi 1010, LAN8720 Ethernet PHY, ENC28J60 SPI Ethernet module, entre otros, como ESP32 o ESP8266. Y soporta funciones como base de datos Firebase Realtime, Cloud Firestore, almacenamiento Firebase y en la nube de Google, editor y deserializar JSON

integrado, actualizaciones de firmware OTA a través de RTDB, Firebase Storage y Google Cloud Storage. [19]

6.2.1.4 ESP NOW

El emparejamiento entre los dispositivos que conformarán la red ESP Now se realiza utilizando la dirección MAC de cada dispositivo, esta dirección MAC será la que se utilizará como identificador del dispositivo los cuales están distribuidos de la siguiente manera:

1. Gateway - MAC => E8:DB:84:9D:2D:3E
2. Medidor 1 - MAC => A4:CF:12:D9:3F:B7
3. Medidor 2 - MAC => A4:CF:12:D9:3E:88
4. Medidor 3 - MAC => A4:CF:12:D9:3E:1B

El dispositivo llamado Gateway corresponde a una placa de desarrollo ESP8266 NodeMCU versión 1.0 V3 del fabricante Wemos LoLin la cual tiene una memoria ROM de 4MB y 96KB de memoria RAM con conectividad WiFi 2.4GHz 802.11b/g/n y Protocolo TCP/IP integrado (figura 35). Esta placa servirá de puerta de enlace entre los medidores inteligentes y la base de datos.

Figura 35 ESP8266 NodeMCU v3
Fuente: “El Autor”.

Por su parte para los medidores 1, 2 y 3 se utilizará la placa ESP8266 (ESP 01) con una memoria ROM de 1MB y 64KB de memoria RAM con conectividad WiFi 2.4GHz

802.11b/g/n y Protocolo TCP/IP integrado (figura 36). Estos módulos de comunicación serán los que envíen los datos medidos desde el medidor ubicado en la vivienda hasta la puerta de enlace.

Figura 36 ESP8266 ESP – 01
Fuente: “El Autor”.

La topología de red a implementar será en tipo estrella la cual posee un nodo central y que en este caso se utilizará la ESP8266 NodeMCU o Gateway como se muestra en la figura 37.

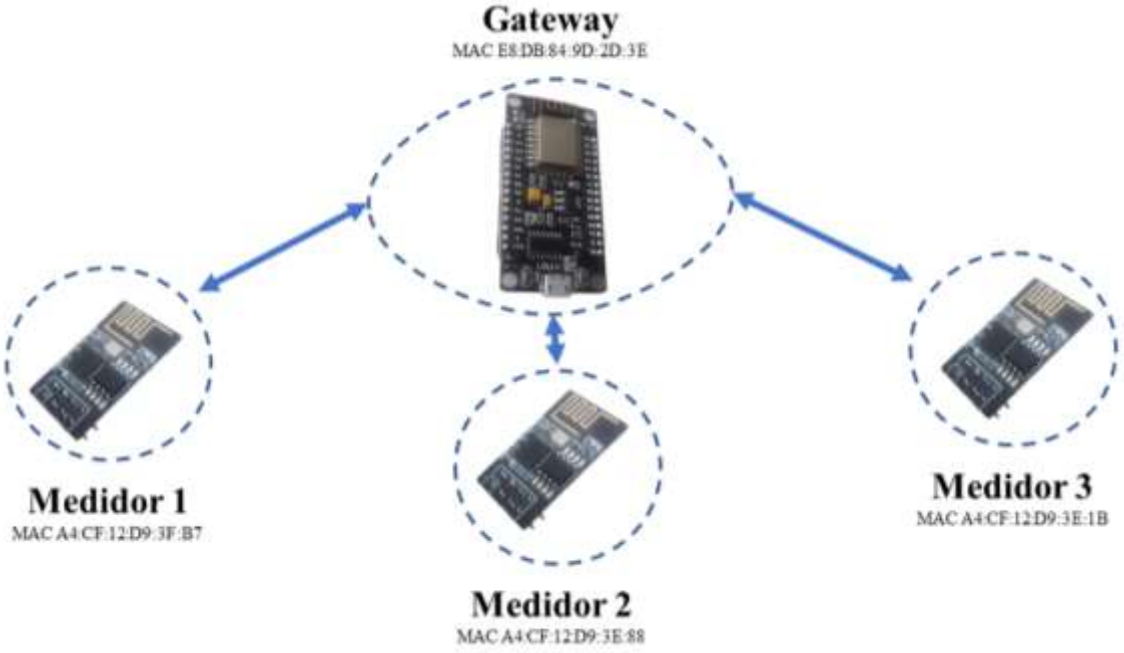


Figura 37 Topología de red
Fuente: “El Autor”.

Como se puede observar en la figura anterior, la comunicación entre los medidores y el Gateway es bidireccional y debe ser así tal y como se establece en el estándar IEC 61968-9 *"...Los detalles específicos de los protocolos de comunicación que emplean estos sistemas están fuera del alcance de esta Norma Internacional. En su lugar, esta norma internacional reconocerá y modelará las capacidades generales que pueden proporcionar potencialmente las infraestructuras de medidores avanzadas y/o heredadas, incluidas las capacidades de comunicación bidireccional, …"* [12] ya que los módulos de comunicación de los medidores no solo deben enviar los datos medidos y el estado del medidor, sino que a su vez debe de recibir órdenes que se envían desde el centro de control para la conexión o reconexión de los servicios como lo dice en la norma IEC 61968-9 que *"esta norma internacional se refiera a un "Medidor", se debe tener en cuenta que un "Medidor" es un dispositivo final que tiene capacidad de metrología, puede o no tener capacidad de comunicaciones, puede o no tener capacidad de conexión/desconexión, o una serie de otras capacidades …"* [12]. En este caso el módulo de comunicaciones podrá enviar los datos medidos y recibir órdenes según lo establecido.

Estas órdenes que se envían desde el centro del control deben de pasar en última instancia por Gateway para finalmente llegar a su destino (módulo de comunicaciones del medidor), por lo que el Gateway debe de enviar dichas órdenes de manera personalizada para cada medidor.

Los módulos de comunicación de los medidores están configurados para que envíen datos cada 15 minutos por defecto o cada que el Gateway envié una solicitud de medida. Y el Gateway lee información de la base de datos cada minuto y de ser necesario algún cambio se envía inmediatamente al medidor correspondiente y escribe en la base de datos cada vez que reciba información del medidor.

Estas órdenes que se enviarán al medidor serán enviadas mediante la tecnología ESP-Now en un paquete (figura 38).

```
typedef struct Order_struct_message {
    char sInOrder[10];
    int sServices[3];
    int Message_len;
} Order_struct_message;
```

Figura 38 Estructura de mensaje enviada por Gateway
Fuente: "El Autor".

La estructura contendrá las siguientes variables:

sServices: Es una variable de tipo de arreglo de enteros que tendrá en cada posición del arreglo un número que será un uno (1) o cero (0) dependiendo del estado que debería tener cada servicio siendo uno para un servicio que debería estar activo y cero para un servicio inactivo. Este tamaño del arreglo de 3 se debe a que un medidor podría llegar a medir hasta tres servicios como lo son electricidad, agua y gas según sea su modelo por lo que a pesar de que en este proyecto no se utilizarán las tres medidas se implementará para su cambio fácil.

Message_len: Esta variable servirá para verificar que el tamaño del mensaje enviado sea el mismo que el tamaño del mensaje recibido.

sOrder: Es una variable de tipo arreglo de caracteres con una longitud de 10 caracteres, lo que supone una longitud máxima de 10 bytes si se llegasen a utilizar los ya mencionados 10 caracteres. Dichos estados enviados causarán un efecto en el medidor al que se le envía dependiendo de dicha orden de la siguiente manera:

- “estado”:

  Realiza la captura del estado del medidor y las medidas e inmediatamente las retorna.

- “inactivar”:

  Inactivar el medidor incluyendo la medida de todos los servicios.

- “servicio”:

  Cambia la configuración de la medida de los servicios según sea el arreglo sServices.

- “n”:

  No realiza ninguna acción.

Así como el Gateway envía órdenes al medidor para su respectivo control, cada medidor envía un paquete de datos (figura 39)

```
typedef struct Message {
    char sName[18];
    char sStatus;
    char sAddress[50];
    int sServices[3];
    float sMeasure[3];
    int Message_len;
} Message;
```

Figura 39 Estructura de mensaje enviada por Medidor
Fuente: “El Autor”.

En los que se incluyen las siguientes variables:

sName: Esta variable contiene la dirección MAC del medidor, ya que es la forma por la cual se identifica el medidor y es único e irrepetible. Esta variable es un arreglo de caracteres de una longitud de 18 caracteres en donde el formato será de esta manera. Ejemplo: “ab:cd:f4:81:23:c4”, el cual tiene un tamaño de 17 caracteres.

sBoard_Id: Es una variable de tipo int que tiene un número irrepetible en las placas ESP que se utilizarán como medidor con el cual se almacenarán las variables en la puerta de enlace.

sServices: Es una variable de tipo de arreglo de enteros que tendrá en cada posición del arreglo un número que será un uno (1) o cero (0) dependiendo del estado que debería tener cada servicio siendo uno para un servicio que debería estar activo y cero para un servicio inactivo. El formato será de esta manera. Ejemplo: [0,1,1] lo que quiere que solo los servicios 2 y 3 están activos ya que tienen un valor de uno (1), y el servicio 1 al tener un cero (0) está desactivado.

sAddress: Es una variable de tipo arreglo de caracteres de un tamaño de 50 caracteres en la que se almacena la dirección en la que se ubica el medidor del que se envían estas variables. El formato será de esta manera. Ejemplo: “Calle 34 sur 78c 28 apartamento 105 torre 4”, el cual tiene un tamaño de 43 caracteres.

sMeasure: Es una variable de tipo de arreglo de número tipo flotante que tendrá en cada posición del arreglo un número decimal que corresponde al valor que se mide en cada servicio, por lo que en la primera posición del arreglo estará la medida correspondiente al servicio 1, en la segunda posición lo obtenido en el servicio 2 y en la última posición la medida correspondiente al servicio 3. El formato será de esta manera. Ejemplo: [0.0, 34.6, 10.0] el cual corresponde a la medida tomada por el servicio 1, 2 y 3 respectivamente.

sStatus: Es una variable de tipo carácter que ocupará un espacio de 1 byte en el paquete que se enviará al Gateway. El formato será de esta manera. Ejemplo: 1 (uno) que quiere decir que el medir está en estado ‘activo’. Los estados están definidos de la siguiente manera:

- 0 (cero)

  Inactivo

- 1 (uno)

  Activo

- 2 (dos)

  Activo con error en toma de medidas servicio 1

- 3 (tres)

  Activo con error en toma de medidas servicio 2

- 4 (cuatro)

  Activo con error en toma de medidas servicio 3

- 5 (cinco)

  Activo con error en toma de medidas de servicios 1 y 2

- 6 (seis)

  Activo con error en toma de medidas de servicios 1 y 3

- 7 (siete)

  Activo con error en toma de medidas de servicios 2 y 3

- 8 (ocho)

  Activo con error en toma de medidas de servicios 1, 2 y 3

- 9 (nueve)

  Activo con error del mensaje anterior

La manera en la cual se llega a estos estados se realiza haciendo la comparación de la medida tomada con el respectivo estado del servicio como se muestra en la figura 40.

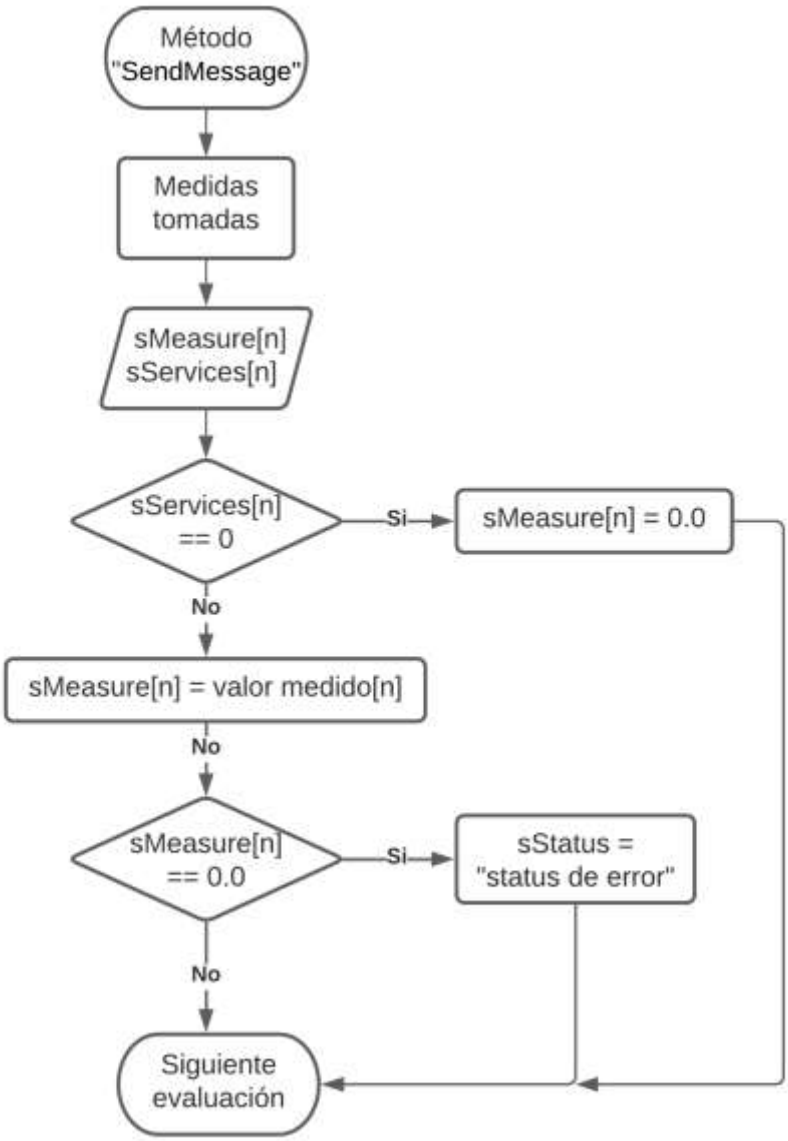


Figura 40 Validación para sStatus
Fuente: “El Autor”.

En cuanto los valores por defecto de los servicios de cada medidor todos se encuentran desactivados, es decir que la variable *sService* de cada medidor está dispuesto como un array con todos sus valores en cero ({0,0,0}), por lo que sólo empezará la medida de los servicios cuando se conecte al Gateway y este mismo le cambie los estados. Los estados que debería tener el medidor se almacenan en la base de datos, en la tabla *statusTable*, en la variable *sService*, que será de donde el Gateway obtiene la información de configuración de cada medidor tanto del estado de los servicios y si hay alguna petición pendiente para su posterior despliegue en la red.

Se debe resaltar que tanto en el medidor como en el Gateway deben existir las dos estructuras de mensajes definidas ya que es la forma en la que se descifrará la información intercambiada.

La tabla *statusTable* solo será de lectura para el Gateway, mientras que las tablas *meterTable* y *measureTable* serán tanto de lectura como de escritura.

Todos los códigos tanto del medidor como del Gateway se encuentran en el anexo A y B respectivamente. El espacio utilizado por el programa del Gateway es de 552880 Bytes (0.55MB) y la memoria disponible en el DevKit NodeMCU es de 4MB, se utilizó tan solo un 11% del almacenamiento total; y el programa que se carga al módulo de comunicaciones del medidor (ESP 01) es de 277312 Bytes 0.27MB y la memoria disponible es de 1MB, es decir un 27% aproximadamente.

6.2.2 XAMARIN

Para aprovechar al máximo todas las funcionalidades que ofrece Xamarin Forms se utilizan librerías de terceros que ya tienen algunas funcionalidades predefinidas por lo que ahorra tiempo al momento de implementar una solución. Los paquetes Nuget utilizados para la implementación de la app USTAPG:

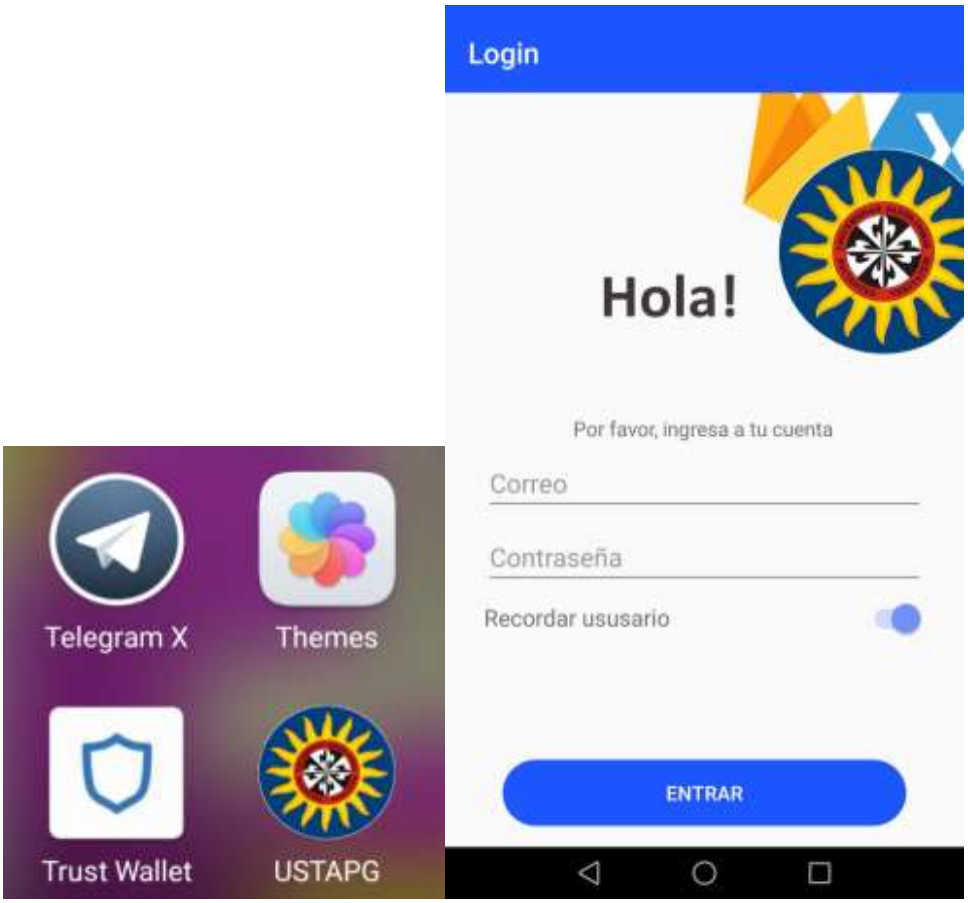
- MVVMLigths
- Firebase.Authentication
- Firebase.Database
- MircroCharts
- Zxing
- Xam.Connectivity
- Newtonsoft.Json

La licencia que utilizan todas las librerías utilizadas es una licencia de código abierto es originaria del MIT que dice *“THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.”* la cual en pocas palabras no ofrece ningún tipo de garantía y da libertad de usar, copiar, modificar, fusionar, publicar, distribuir, sublicenciar, y/o vender copias de este Software sin necesidad de retribuir al creador de dicha librería. [21]

Todo el código relacionado a el software cross platform se encuentra en el anexo C.

6.2.2.1 ANDROID

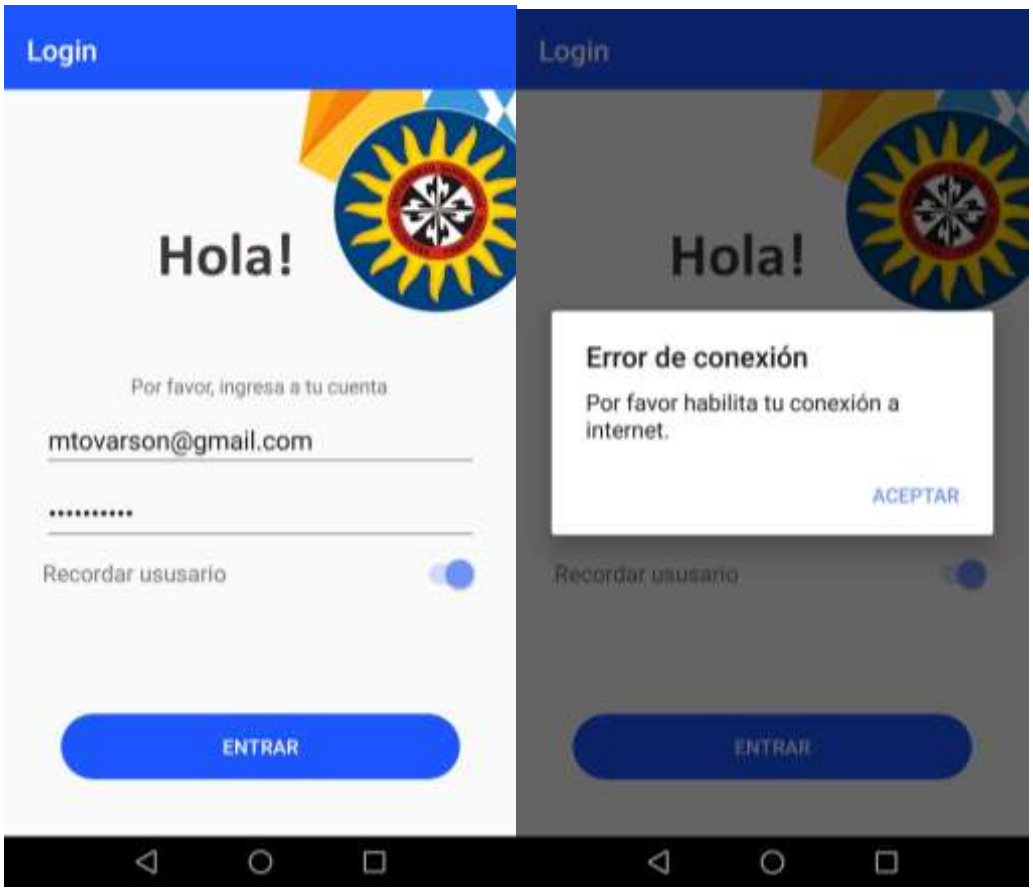
En el sistema operativo Android la app USTAPG tendrá el icono que se muestra en la figura 41.a. La página inicial será una página en la que se debe ingresar con un correo y una contraseña (figura 41.b), estas credenciales deben estar previamente registradas sobre Firebase por lo que no se tendrá acceso hasta que el administrador de la base de datos realice el respectivo registro. En la app el usuario tendrá la opción de guardar las credenciales para un ingreso más rápido.



a) b)

Figura 41 Inicio de USTAPG app.
a) Icono, b) página inicial (Login)
Fuente: “El Autor”.

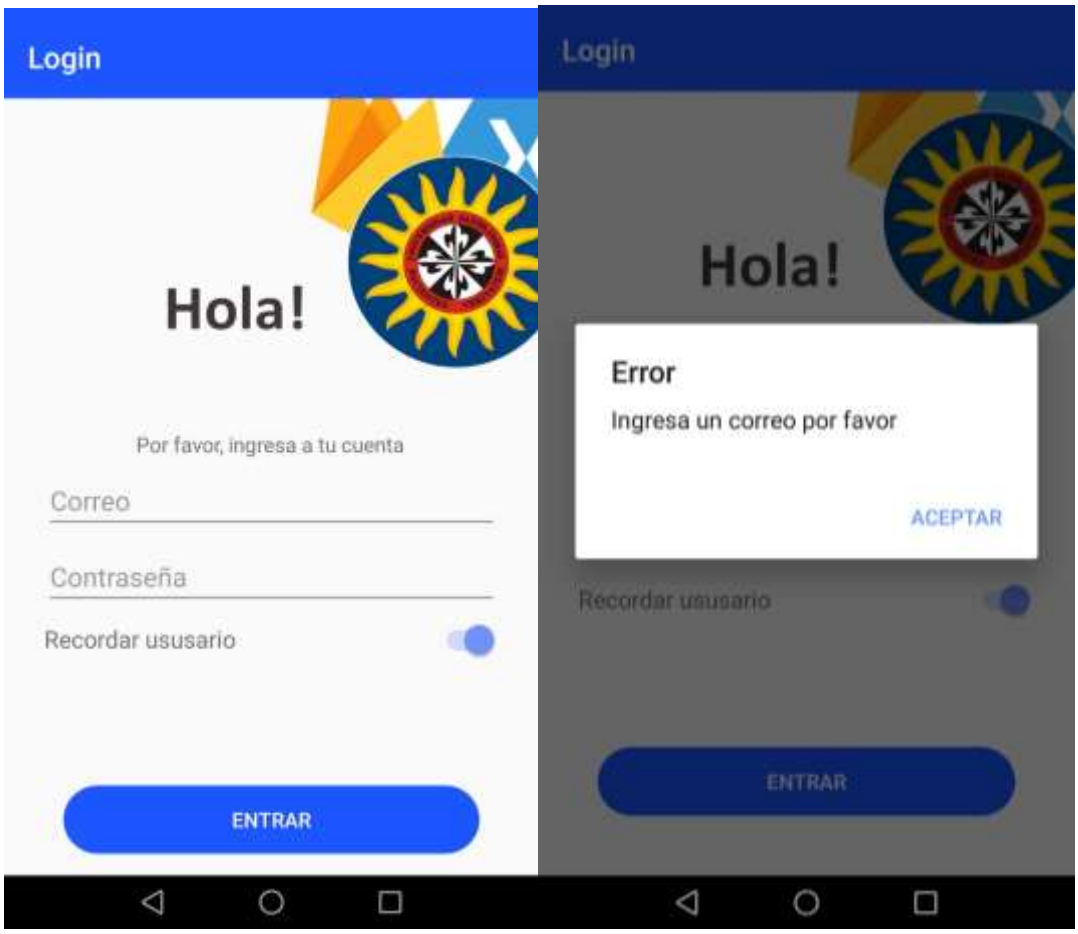
Una vez ingresado el correo y su respectiva contraseña se debe asegurar que el usuario tenga acceso a internet, ya que es indispensable a la hora de conectar con la base de datos en tiempo real de Firebase. Por lo que mediante el uso del plugin “Xam.Connectivity” se verifica primero si el usuario tiene el encendido su modo de acceso a internet, ya sea con conexión WiFi o datos móviles. Pero esta no es la única validación ya que a pesar de que tenga alguno de los anteriormente mencionados modos de conexión encendido, no es certeza de que tenga acceso a internet por lo que también se realiza otra validación haciendo un ping con algún servidor web, de esta forma se asegura de que el usuario tenga que estar conectado (figura 42).



a) b)

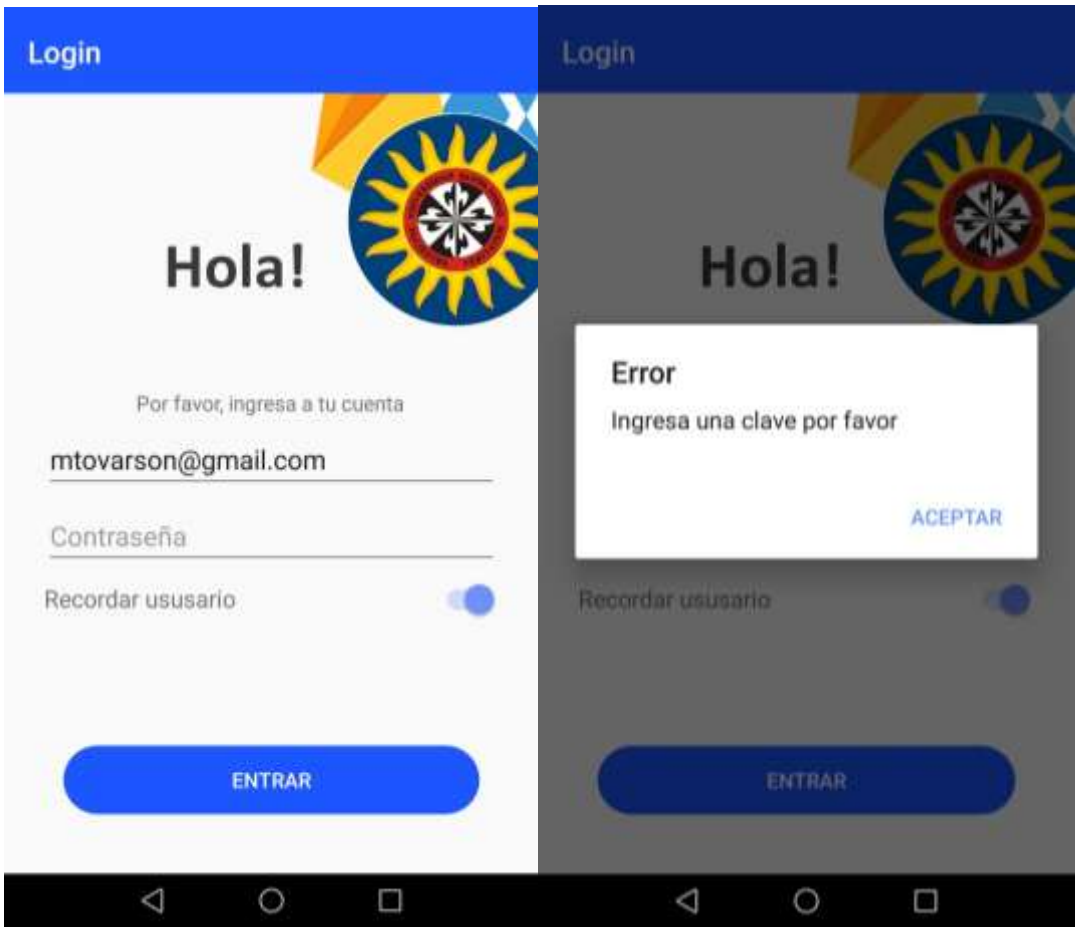
Figura 42 Validación conectividad USTAPG app
a) página inicial, b) Mensaje de error conectividad
Fuente: “El Autor”.

Luego de que se asegure esa conexión a internet se debe de asegurar también que los espacios están correctamente diligenciados por lo que si el usuario no ingresa el correo aparecerá un mensaje como el que se muestra en la figura 43 y si el usuario no ingresa una clave aparecerá un mensaje como el que se muestra en la figura 44.



a) b)

Figura 43 Validación correo USTAPG app
a) página inicial sin campos llenos,
b) Mensaje de error campo correo vacío
Fuente: “El Autor”.



a) b)

Figura 44 Validación clave USTAPG app
a) página inicial campo clave vacío,
b) Mensaje de error campo clave vacío
Fuente: “El Autor”.

Al ingresar las credenciales correctamente ya se tendrá acceso a la base de datos alojada en la nube, de no ser las credenciales correctas aparecerá un mensaje que avisa sobre el ingreso incorrecto de las credenciales (figura 45). De colocar las credenciales correctas y el usuario estar previamente autorizado para entrar en la base de datos en Firebase se tendrá la opción de guardar las credenciales en el dispositivo local para no tener necesidad de ingresarlas nuevamente

Figura 45 Validación clave USTAPG app
Fuente: “El Autor”.

Ahora se debe especificar qué información se debe leer ya que en la base de datos se encuentran todos los datos de los medidores por lo que es necesario especificar cuál es el medidor. Cada medidor será identificado con un código QR que contiene la dirección MAC del equipo para realizar la consulta en la base de datos ya que es la manera en la que se busca la información. Esta dirección MAC es de suma importancia ya que, al tener esta información, junto con las credenciales se tiene acceso a la información del usuario, por lo cual se realiza una codificación UTF-16 de caracteres Unicode, y esta cadena de bytes se convierten en dígitos de base 64 organizados en una matriz de enteros de 8 bits, convirtiendo la dirección MAC de menos legible.

Si se aplica la codificación anteriormente explicada a la cadena “Hola”, esta cadena especificada de caracteres se convertiría en una cadena de bytes separada por ceros ‘0’ [72, 0, 111, 0, 108, 0, 97, 0], como se observa el byte de la primera posición es 72 que el correspondiente ASCII es ‘H’, 111 a ‘o’ y así con los demás caracteres. Posterior a convertirlos en una cadena de bytes se convierten a un string en base64 quedando la cadena de la siguiente manera “SABvAGwAYQA=”, la cual no es legible a simple vista y solo será codificada con la aplicación USTAPG.

En la figura 46 se puede observar cómo se genera el código QR y en la figura 47 como es la estructura de la base de datos.

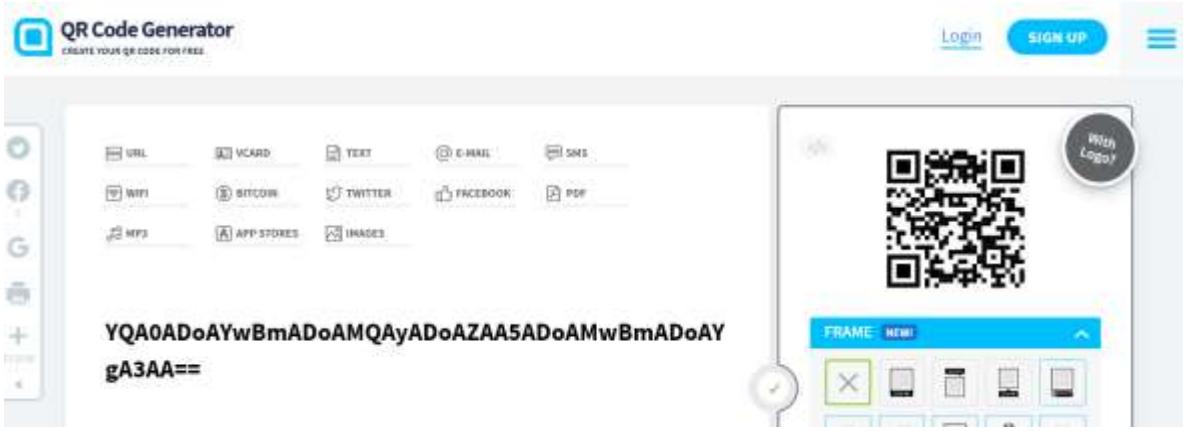


Figura 46 Código QR por medidor [40]
Fuente: Crea códigos QR gratis con nuestro generador (qr-code-generator.com)

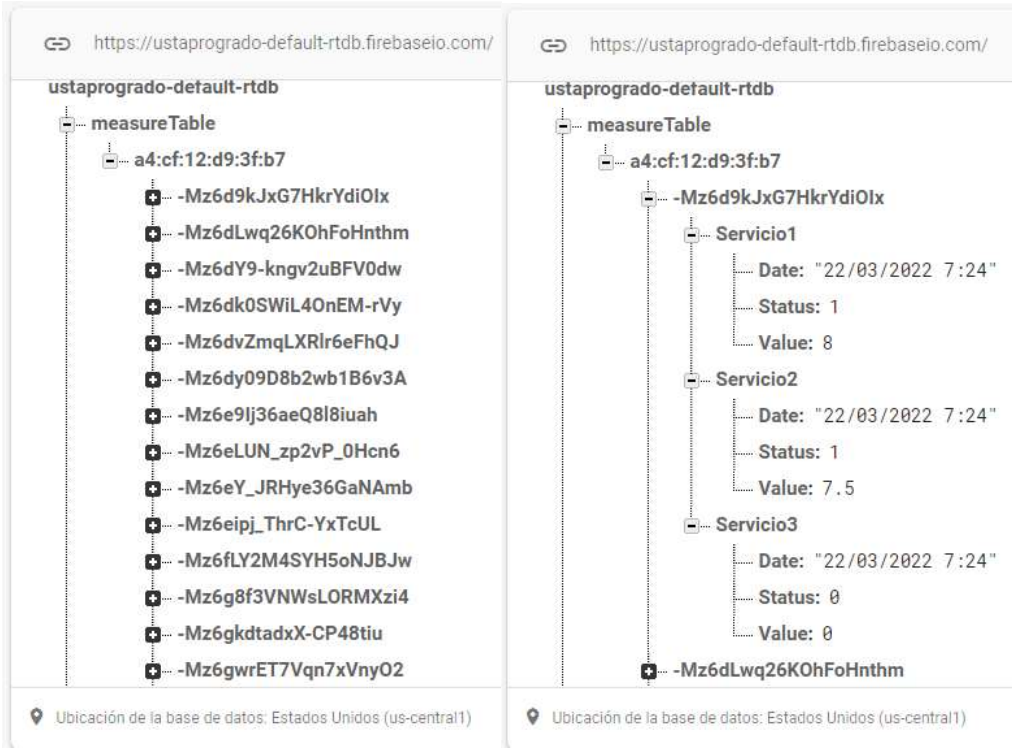


Figura 47 Base de datos en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

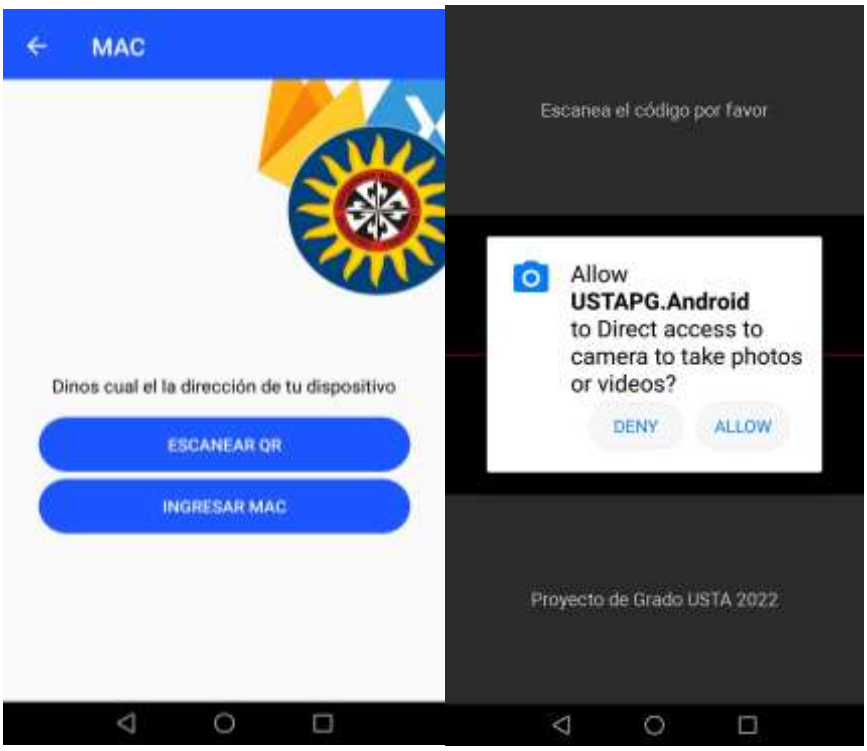
En este proyecto grado se hará uso de tres módulos de comunicación diferentes, es decir tres placas ESP cada una con su correspondiente dirección MAC a la cual

se le realizó la respectiva codificación y se generó el código QR como el que se muestra en la figura 48.

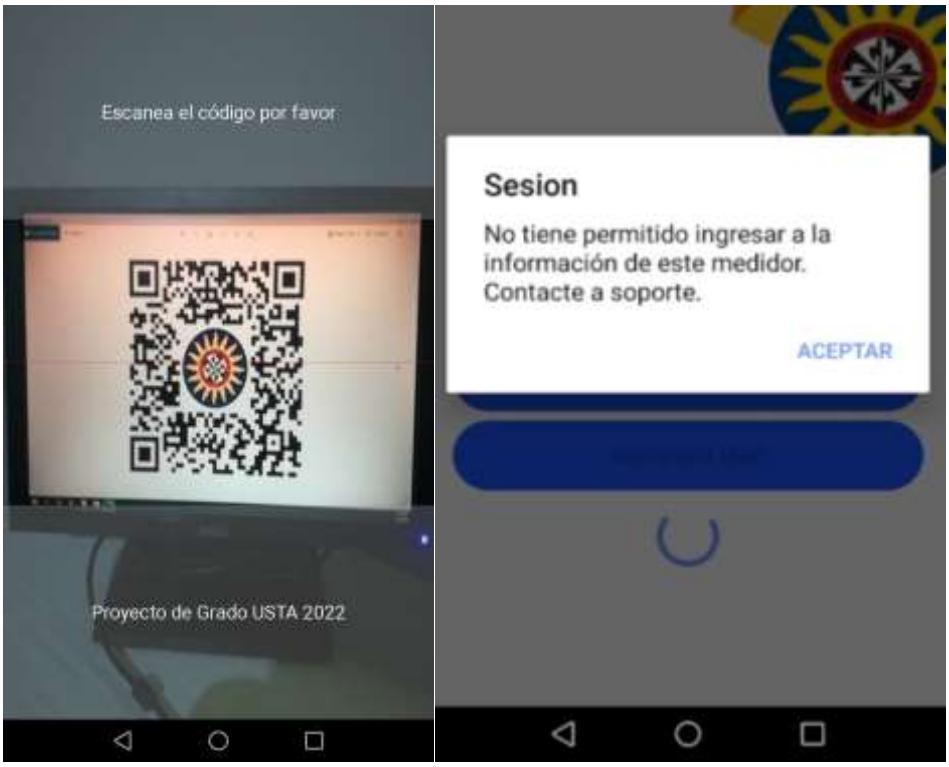
Figura 48 Ejemplo QR MAC de equipo [40]
Fuente: Crea códigos QR gratis con nuestro generador (qr-code-generator.com)

Como se mencionó anteriormente se debe conocer la dirección MAC del medidor para saber cuáles son los valores que se le mostrarán al usuario, y al ser esta identificación del medidor con un código QR se debe tener en la aplicación USTAPG una página en la que se pueda realizar esta validación (figura 49.a). Con el uso de la librería Zxing en la app se pueden leer códigos QR (figura 49. c) esta librería también necesita del permiso del dispositivo para utilizar la cámara (figura 49. b). Luego de escanear el código se obtendrá la dirección MAC del medidor por lo que ya se podrán observar los datos correspondientes a cada usuario. Si solo se tuviera esta validación para el ingreso del usuario a los datos del medidor implicaría que cualquier usuario inscrito en la base de datos podría acceder a los datos del medidor si éste logra escanear o conoce la dirección MAC de los otros medidores. Por esta razón es que existe la tabla *sesionTable* en donde se encuentran los correos autorizados para ingresar a la información del respectivo medidor y así evitar el ingreso de usuarios no deseados a dicha información.

Los códigos QR de los tres medidores se encuentran en el anexo D.


MAC
Dinos cual el la dirección de tu dispositivo
ESCANEAR QR
INGRESAR MAC
Escanea el código por favor
Allow
USTAPG.Android
to Direct access to
camera to take photos
or videos?
DENY
ALLOW
Proyecto de Grado USTA 2022

a) b)


Escanea el código por favor
Proyecto de Grado USTA 2022
Sesion
No tiene permitido ingresar a la
información de este medidor.
Contacte a soporte.
ACEPTAR

c) d)

Figura 49 Validación MAC medidor USTAPG app

a) página MAC,

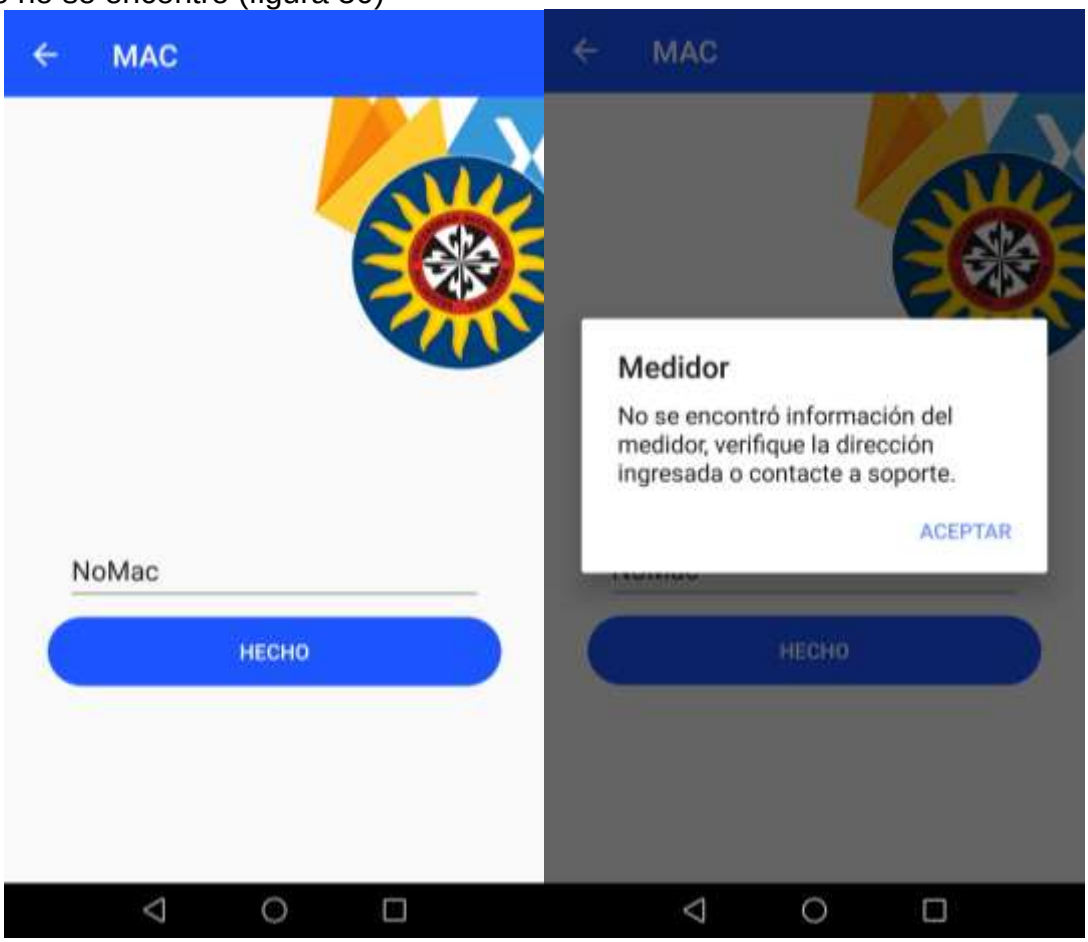
b) Mensaje de solicitud de permiso de cámara

c) Escáner de QR

d) Mensaje de error de inicio de sesión

Fuente: “El Autor”.

Sin embargo, se debe de tener en cuenta los dispositivos que no tienen cámara, por lo que la manera adicional con la cual se puede ingresar la MAC del equipo es oprimiendo el botón de Ingresar MAC en la cual también se tiene la validación si el espacio está correctamente diligenciado. La dirección MAC ingresada se valida con los registros existentes en Firebase por MAC, si no existe este dispositivo aparecerá que no se encontró (figura 50)



a) b)

Figura 50 Validación ingreso MAC medidor USTAPG app

a) página ingresar MAC,

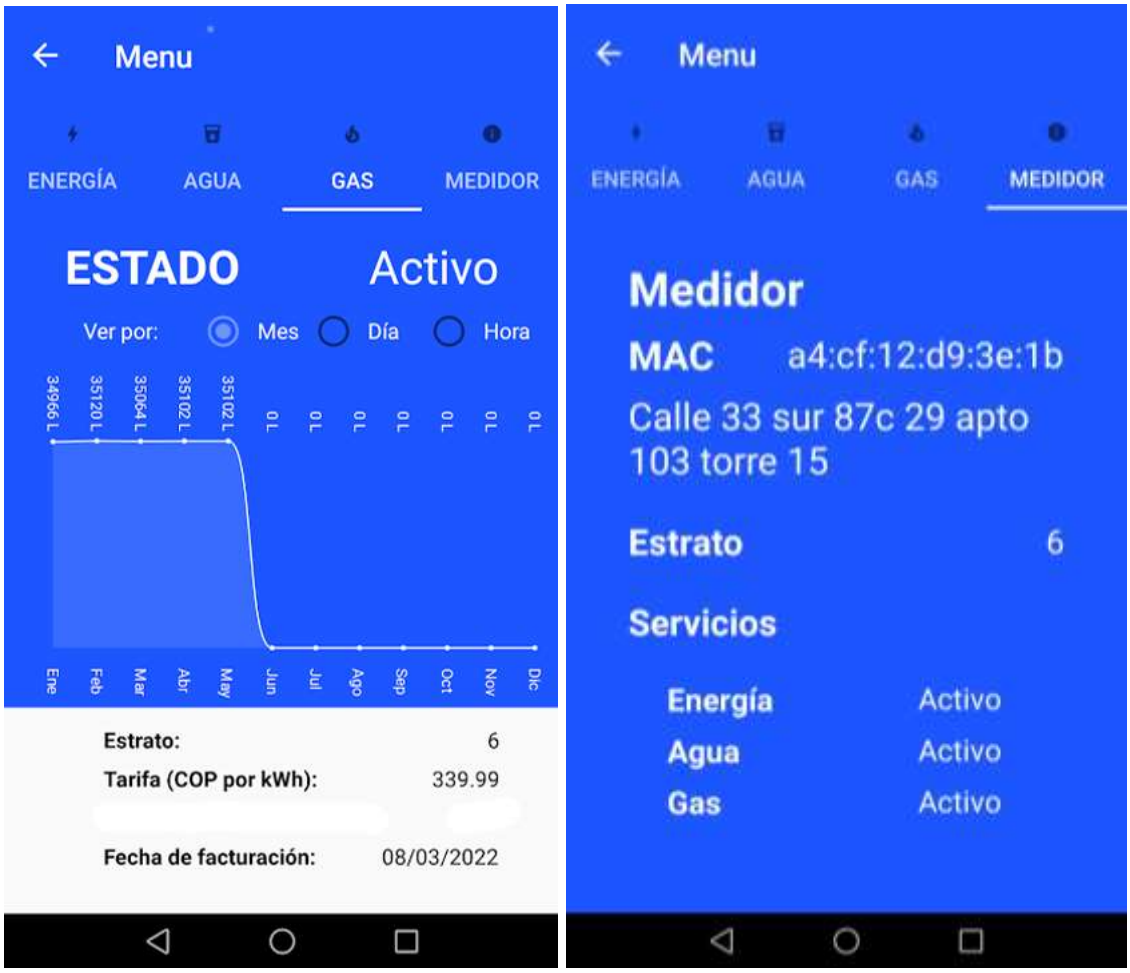
b) Mensaje de error medidor no encontrado

Fuente: “El Autor”.

Luego de escanear el código QR correspondiente o ingresar la dirección MAC manualmente se guardará este código en el almacenamiento local del dispositivo para que el usuario no tenga la necesidad de estar ingresando dicho código y las credenciales de inicio de sesión cada vez que se ingresa a la app. Posterior al escaneó del código se mostrará una página tabulada en la que podremos observar las gráficas de consumo de cada servicio (figura 51).

a)

b)



c) d)

Figura 51 Página tabulada USTAPG app
a) Consumo servicio de energía eléctrica
b) Consumo servicio de agua
c) Consumo servicio de gas
d) Información medidor
Fuente: “El Autor”.

Como se observa en la figura 51, existen cuatro páginas tabuladas que con el gesto de deslizar se puede acceder a ellas. En la primera página se encuentran las medidas de consumo en kilovatios por hora (kWh) del servicio de energía eléctrica (figura 51.a), en la segunda (figura 51.b) lo relacionado con el servicio del agua el cual está medido en Litros (L), seguida de esta se encuentra la página (figura 51.c) en la que se muestra el consumo del servicio del gas que también está medida en Litros (L) y por último, en la cuarta página se puede observar la información referente al medidor (figura 51.d). Además, en cada página se puede organizar la información por hora, día y mes. Es decir, si se coloca la escala en horas esta app identificará la fecha local y en base a eso realizará la búsqueda de los valores tomados dicho día y los clasificará por hora. También en la página de cada servicio,

en la parte inferior de la gráfica se encuentran dos variables, *tarifa (COP por kWh)* o el respectivo valor de medida con el cual se tarifica el servicio.

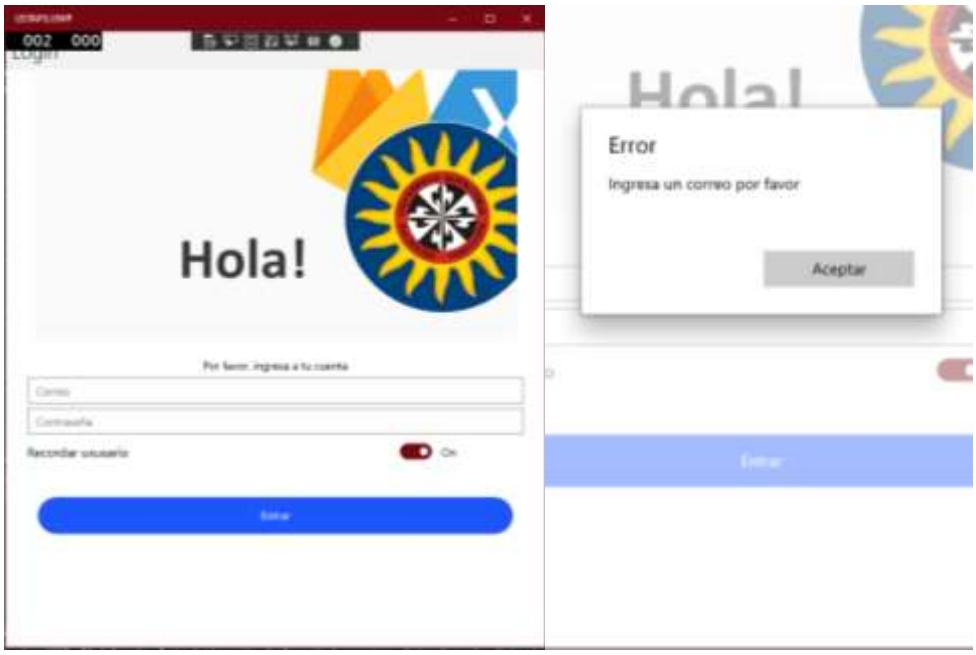
En la página de medidor se encontrará la información referente al medidor, se mostrarán variables como dirección MAC, dirección donde se encuentra el equipo, estrato del lugar donde se encuentra el equipo y el estado de cada uno de los servicios.

6.2.2.2 UWP

Como se mencionó anteriormente, uno de los beneficios de utilizar Xamarin form es que cuando se programa se hace para diferentes plataformas como UWP (Universal Windows Platform), Android y iOS, por lo que en lo que refiere a la programación back end no habría ningún tipo de problema, mientras que en la programación front end si habría problemas si no se desarrolla una solución adaptable ya que los tamaños de las pantallas de los dispositivos en los que se va a implementar cambian por lo que es un criterio que se debe tener muy en cuenta.

Sin embargo, Xamarin cuenta con la ventaja de que la programación front end puede ser específica para cada plataforma, solo se deben de cambiar los parámetros cuando se detecte una u otra plataforma para no tener inconvenientes y depurar en cada plataforma si es posible. Para este proyecto se depuró sobre las plataformas de UWP y Android (se sugiere al momento de depurar sobre android que se haga sobre un dispositivo físico y no sobre un emulador virtual de android ya que este consume bastantes recursos y podría retrasar el desarrollo).

En las siguientes imágenes (figura 52 a figura 57) se mostrará todas las páginas mostradas anteriormente, pero desde UWP.

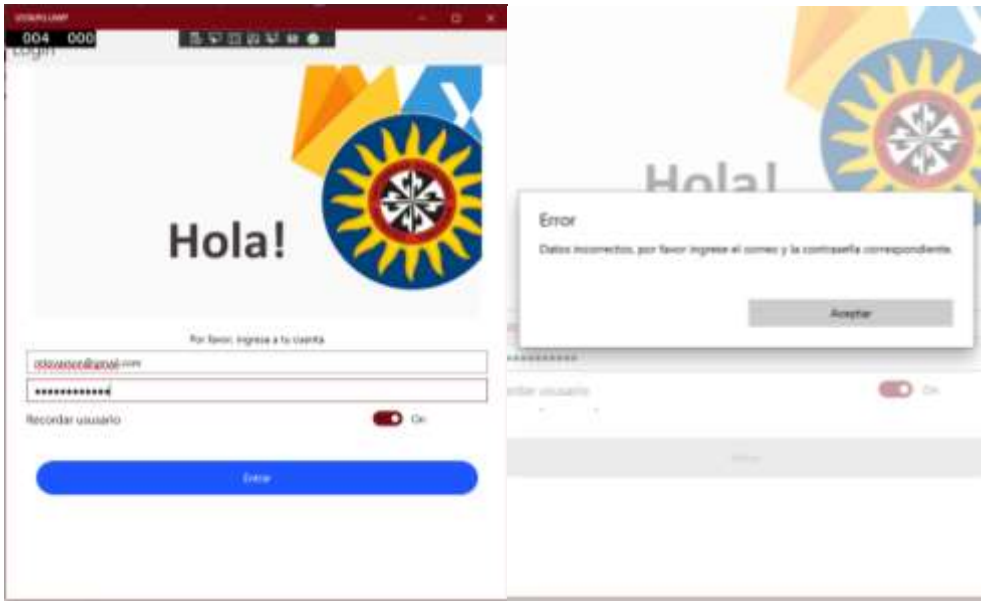


a) b)

Figura 52 Validación conectividad USTAPG UWP
a) página inicial, b) Mensaje de error conectividad
Fuente: “El Autor”.

a) b)

Figura 53 Validación clave USTAPG UWP
a) página inicial campo clave vacío,
b) Mensaje de error campo clave vacío
Fuente: “El Autor”.



a) b)

Figura 54 Validación correo USTAPG UWP
a) página inicial sin campos llenos,
b) Mensaje de error campo correo vacío
Fuente: “El Autor”.

Para UWP, en la página de ingresar MAC se deshabilito el botón de Escanear QR (figura 55) ya que si el dispositivo no tiene acceso a una cámara no podrá escanear dicho código (figura 56), por lo que solo se dejó habilitada la opción de ingresar MAC.

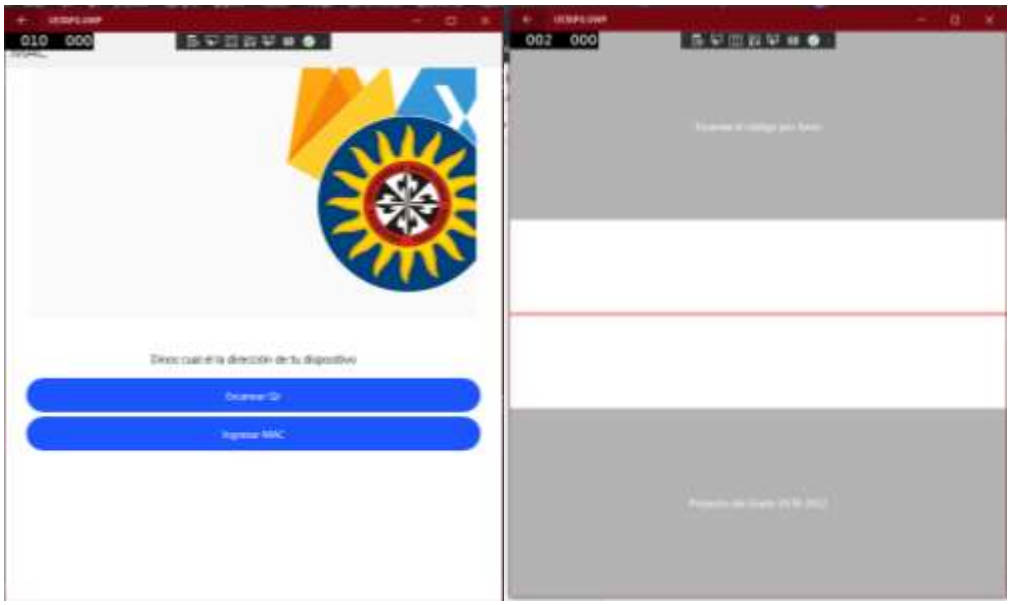
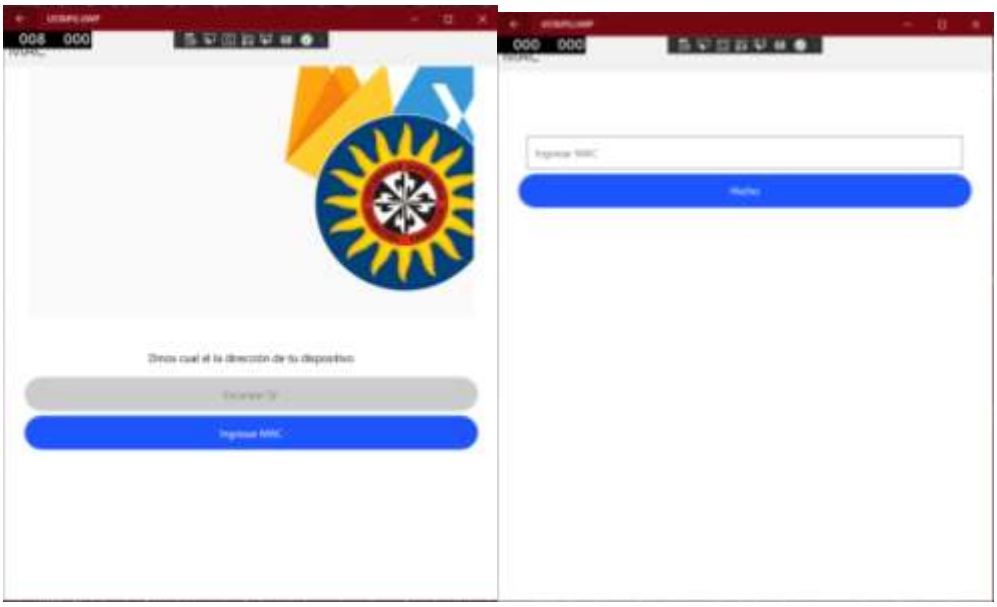


Figura 55 Validación MAC QR USTAPG UWP



a)

b)

c)

d)

e)

Figura 56 Validación MAC medidor USTAPG UWP

a) página MAC,
b) página ingresar MAC
c) MAC errónea
d) Mensaje de error al buscar medidor
e) Mensaje de error acceso a medidor no autorizado
Fuente: “El Autor”.

a)

b)

c)

d)

Figura 57 Página tabulada USTAPG UWP

a) Consumo servicio de energía eléctrica
b) Consumo servicio de agua
c) Consumo servicio de gas
d) Información medidor
Fuente: “El Autor”.

6.3 FIREBASE

En la siguiente sección se conocerá cual fue el método para la generación de los datos necesarios tanto para obtener las gráficas de consumo en la app multiplataforma y a su vez para tener datos de referencia para el centro de control.

6.3.1 Servicio energía eléctrica

La CREG o Comisión de Regulación de Energía y Gas presentó en diciembre del 2020 un estudio en el que se muestran las estrategias de señales de precios y cargos horarios para programas de respuesta a la demanda en donde se muestran además de ejemplos en otros países como Uruguay en donde se ha implementado la tarificación horaria; la demanda o carga que se presenta en Colombia según la hora (figura 58) y en la cual es en la que se basan los datos generados para este proyecto.[20]

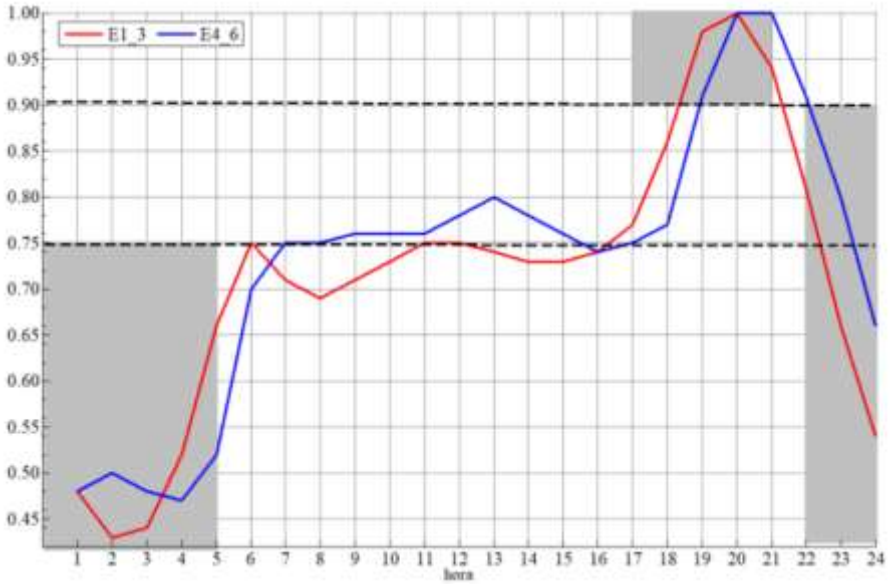


Figura 58 Curva de carga por estrato [20]
Fuente: CREG, Grupo energéticos consultore, “Estrategias para la implementación de esquemas de señales de precios y cargos horarios a los usuarios finales en el SIN, para ser utilizados en programas de respuesta de la demanda”, diciembre, 2020.

Además de los estudios de la CREG, se tomaron datos de una vivienda estrato 2 en promedio se consumen 6.6 kWh al día, es decir aproximadamente 0.22 kWh por

hora si se tuviera un consumo constante cada hora. En la generación de la Data se obtiene una medida cada 15 minutos que es el comportamiento que tendría el módulo de comunicaciones de los medidores (registro de información cada 15 minutos o cada vez que se requiera). Sin embargo, se debe tener en cuenta que no todas las horas del día tienen el mismo pico de consumo de energía eléctrica, por ejemplo, la hora de las 03:00 no tendría el mismo consumo que en la hora 19:00 según el estudio de la CREG, por lo que se debe de tener en cuenta la hora al momento de colocar un valor de medida. Más específicamente si se tuviera un consumo diario promedio de 5.5 kWh, los valores entre las 00:00 y 5:59 tendrán un valor máximo de 0.1 kWh, entre las 6:00 y 17:59 horas tendrán un valor máximo de 0.28 kWh y un valor mínimo de 0.2 kWh y en las demás horas del día (18:00 a 23:59) un máximo de 0.4 kWh y mínimo de 0.35 kWh.

6.3.2 Servicio de agua

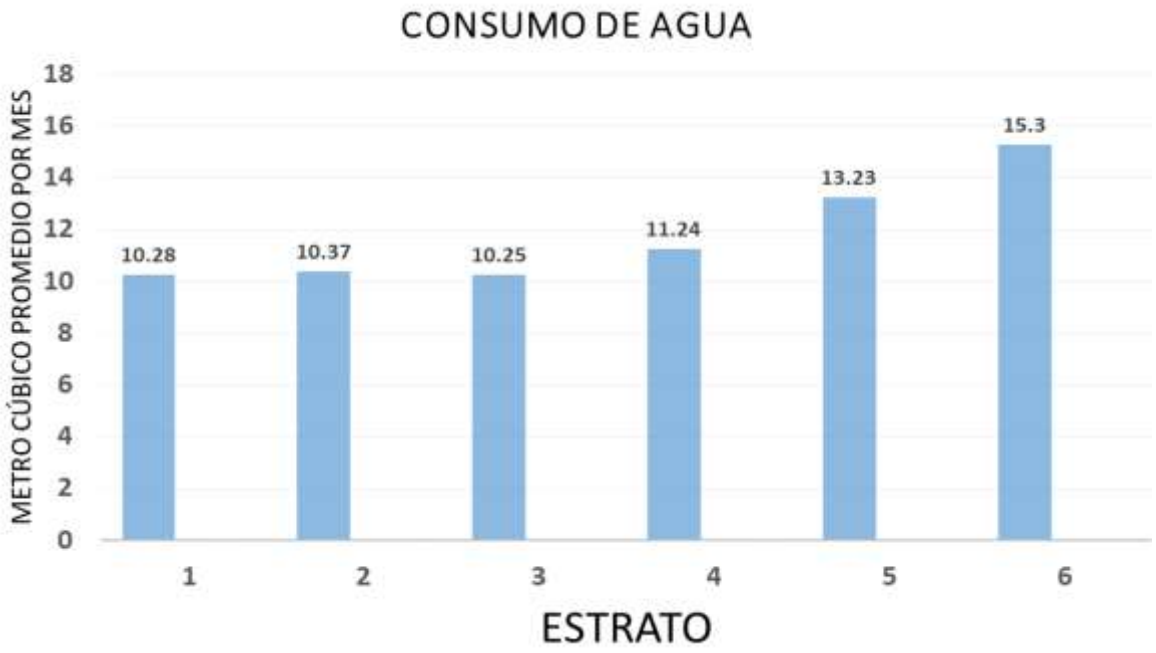


Figura 59 Consumo por estrato de servicio de agua potable Bogotá [21]
Fuente: Cortés, Ernesto; El Tiempo, CityTV, “4 años para salvar el agua de Bogotá”, Bogotá, 2020.

Según un estudio realizado por el periódico el Tiempo Y CityTV (figura 59) el promedio de consumo de agua mensual en Bogotá varía según el estrato socio-económico, sin embargo una media de consumo entre los estratos estaría alrededor de los 12.79 m3 (12790 litros) por lo que aproximadamente en una vivienda el consumo diario sería de 0.427 m3 (427 litros), en el caso de estrato 6 el valor diario sería de 0.520 m3 (520 litros) valor con el cual se generarán los datos con una variación máxima del 6% (± 31 litros diarios), es decir que tomará valores entre los 489 y 551 litros. En las horas de la madrugada y terminando el día se estima que el

consumo de agua es mucho menor (00:00 a 5:59 y 21:00 a 23:59) dado que en la mayoría de las viviendas no son horas en las que se realice algún tipo de actividad que involucre el uso de este servicio mientras que entre las 6:00 y las 20:59 el consumo debería ser considerablemente mayor.

6.3.3 Servicio de gas

Al igual que con el servicio de agua potable, no se encontró información de una fuente confiable sobre el consumo por horas de GNC (Gas Natural comprimido), sin embargo al igual que con el anterior servicio en las horas de la madrugada y terminando el día se estima que el consumo de agua es mucho menor (00:00 a 5:59 y 21:00 a 23:59) dado que en la mayoría de las viviendas no son horas en las que se realice algún tipo de actividad que involucre el uso de este servicio mientras que entre las 6:00 y las 20:59 el consumo debería ser considerablemente mayor. No obstante, se tiene el consumo mensual de metros cúbicos de gas es de 30 en promedio en una vivienda estrato 2 de alrededor de 8 personas (30000 litros). En las horas de mayor consumo se obtendrá un valor máximo de 102 L y un mínimo de 78 L mientras que en las horas de menor consumo se obtendrá un valor máximo de 24 L y un mínimo de 6 L.

6.4 SISTEMA INTELIGENTE

Unos de los algoritmos más utilizados de Machine Learning es el algoritmo de regresión el cual básicamente se obtiene haciendo una medida del error el cual lo ideal tendría que ser cero y para llegar a valores cercanos se debe de iterar el algoritmo para su disminución paulatina. En este algoritmo se encuentran dos que son Regresión lineal y Regresión Logística.

En la regresión lineal se busca la tendencia de un conjunto de datos a determinado valor (datos continuos) mientras que la regresión logística hace parte del aprendizaje supervisado ya que se tienen las salidas definidas por lo que la salida es discreta y no necesariamente continua por lo que sirve para la clasificación de los datos, es decir que cada dato va a hacer parte de un grupo ya predefinido.

El objetivo de implementar un sistema inteligente es predecir de forma individual, es decir por medidor, cual es el comportamiento que está teniendo, si es el adecuado o por el contrario hay alguna especie de falla en el suministro o en la red y este deba reportarse para su correcta gestión. Para esto se toman los datos de medida del medidor correspondiente para realizar las operaciones de aprendizaje. Se tendrá en cuenta cual es la hora en la que se toma la medida y el valor que debería tener normalmente. Para entrenar esta red se tienen inicialmente 4320 registros que corresponden a los valores de medida que normalmente debería tener el servicio, además de estos se les agregaron 8640 registros los cuales son registros errados, es decir que no son los valores de medida normal, la mitad de estos datos se les

sumó el 85% del valor y la otra mitad todo lo contrario, se le resto el 85% del valor normal dando así un total de 12960 registros para el entrenamiento que como se mencionó anteriormente tienen el dato de hora y medida.

Al tratarse de regresión logística se deben tener las salidas que deben tener según la hora y valor medido, en este caso se tienen tres grupos: grupo 1 que corresponde a un valor medido normal según la hora en la que se realizó la medición, grupo 2 que contienen un 85% más de valor medido según la hora y finalmente el grupo tres que comparado con el valor normal esta un 85% por debajo según la hora.

Una vez ya arreglada la data que se utilizará, que por cierto se encuentra en un archivo tipo CSV se carga dicha data a Colab (Colaboratory de Google Inc) para realizar el respectivo entrenamiento (figura 60).

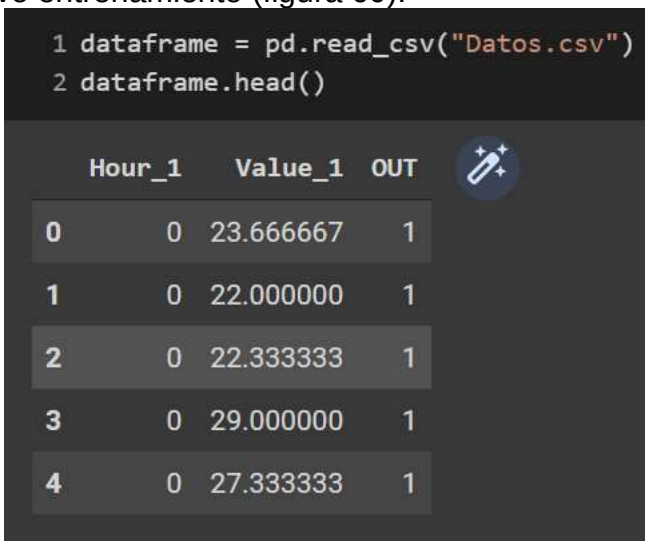


Figura 60 Data para entrenamiento [41]
Fuente: Michael Tovar, "Tesis.ipynb - Colaboratory (google.com)", 2022.

Si se entra a más detalle en la data que recién se subió se obtiene una tabla como la de la figura 61 en la que muestra el total de registros por cada columna (count), el valor máximo y mínimo (min y max respectivamente)

```
1 dataframe.describe()
```

	Hour_1	Value_1	OUT
count	12960.000000	12960.000000	12960.000000
mean	11.500000	76.103549	2.000000
std	6.922454	66.887925	0.816528
min	0.000000	3.250000	1.000000
25%	5.750000	13.650000	1.000000
50%	11.500000	63.858333	2.000000
75%	17.250000	128.266667	3.000000
max	23.000000	246.050000	3.000000

Figura 61 Detalle data [41]
Fuente: Michael Tovar, "Tesis.ipynb - Colaboratory (google.com)", 2022.

Al agrupar la data por las salidas posibles, la cual se encuentra en la columna 'OUT' se puede observar que existen los tres grupos anteriormente mencionados (figura 62)

```
[4]   1 print(dataframe.groupby('OUT').size())

    OUT
    1    4320
    2    4320
    3    4320
    dtype: int64
```

Figura 62 Posibles salidas [41]
Fuente: Michael Tovar, "Tesis.ipynb - Colaboratory (google.com)", 2022.

En la figura 63 se puede observar la implementación de la regresión logística y el score o puntación que se obtuvo al clasificar los valores en la cual se obtuvo un valor aproximadamente de 92.1836% que corresponde a la probabilidad de éxito al clasificar un registro según el valor medido y la hora.

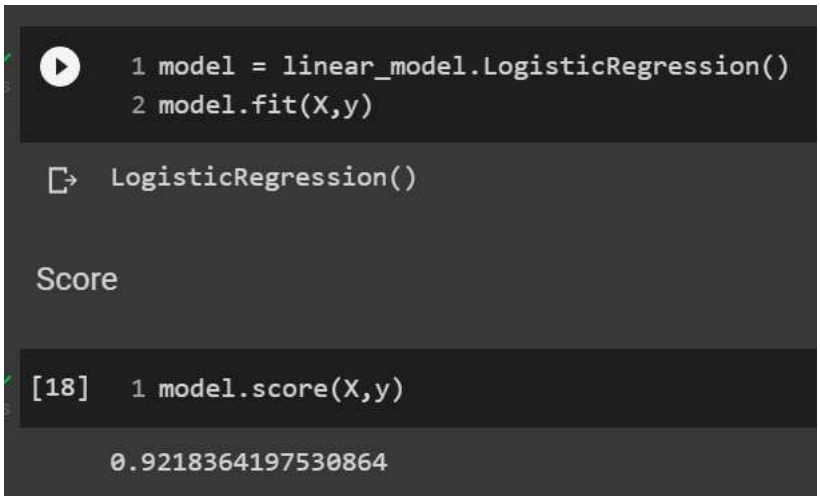


Figura 63 Regresión logística en Colab [41]
Fuente: Michael Tovar, "Tesis.ipynb - Colaboratory (google.com)", 2022.

Sin embargo, el anterior score obtenido se obtuvo utilizando todos los datos disponibles lo cual no es una práctica correcta ya que se cae en sobre entrenamiento (Overfitting) lo cual tendría como resultado una imposibilidad al tratar de comprender los datos de entrada. Por esto es recomendable utilizar entre un 70%-80% de los datos para realizar la regresión logística y el 20%-30% para realizar el testeo de la regresión con lo cual se obtiene un resultado de 91.9753% de score lo cual se aproxima bastante al obteniendo anteriormente, pero asegurando que no existe Overfitting (figura 64).

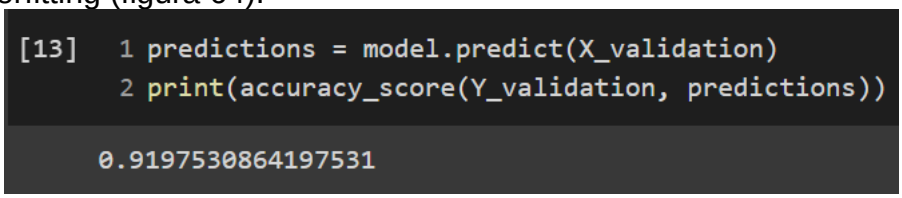


Figura 64 Entrenamiento sin caer en Overfitting [41]
Fuente: Michael Tovar, "Tesis.ipynb - Colaboratory (google.com)", 2022.

Si se observa la matriz de confusión obtenida (figura 65) la cual sirve como herramienta para visualizar el desempeño de la regresión se puede observar que sobre los datos que se separó para realizar el testeo de la regresión se obtuvieron 747 ciertos de 859 posibles, 799 aciertos de 895 posible y 838 de 838 posibles (Los valores correctos que se obtuvieron se encuentran en la diagonal principal de la matriz).

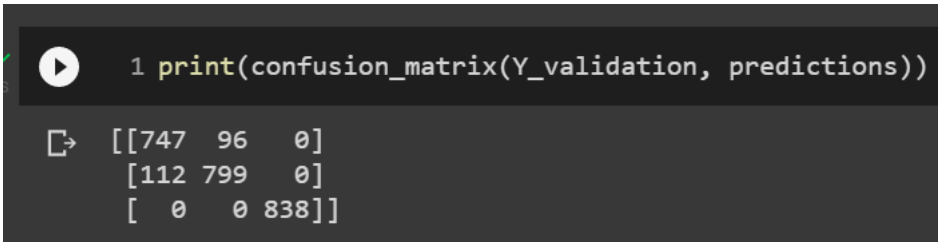
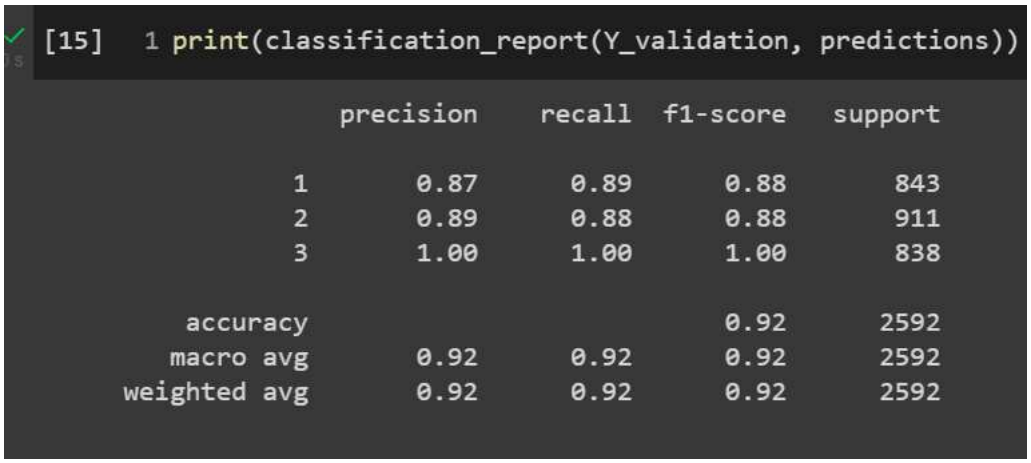


Figura 65 Matriz de confusión [41]

Fuente: Michael Tovar, “Tesis.ipynb - Colaboratory (google.com)”, 2022.

Además, si se quiere tener unas estadísticas más precisas por grupo se puede obtener con el reporte de clasificación donde se puede observar la precisión al momento de clasificar un registro en cualquiera de los 3 grupos (figura 66).



```
[15]  1 print(classification_report(Y_validation, predictions))
```

	precision	recall	f1-score	support
1	0.87	0.89	0.88	843
2	0.89	0.88	0.88	911
3	1.00	1.00	1.00	838
accuracy			0.92	2592
macro avg	0.92	0.92	0.92	2592
weighted avg	0.92	0.92	0.92	2592

Figura 66 Reporte de clasificación [41]
Fuente: Michael Tovar, “Tesis.ipynb - Colaboratory (google.com)”, 2022.

Luego de entrenar el sistema con los datos de prueba se conecta a Firebase para comprobar el funcionamiento. Para permitir la conexión del sistema inteligente con Realtime Database y realizar funciones de CRUD sin necesidad de tener correo y contraseña o alguno de los otros métodos de login se realiza mediante la obtención de una clave de API la cual se genera directamente desde la configuración de Firebase por lo que nadie aparte del administrador de la base de datos tendrá acceso a esta clave, de esta manera se tiene acceso a toda la información guardada por el Gateway.

El sistema inteligente validará la última medida obtenida y su respectiva fecha (el ultimo registro guardado por el medido) ya que estos son los datos que se necesitan para la validación y retornará en la tabla *InfoTable* en la variable *AproxMonth* un valor que en este caso oscilará entre tres valores. ‘1’ si la medida está dentro del rango aceptado con respecto al uso normal del consumidor, ‘2’ si el valor está muy por encima del valor normal y ‘3’ si el valor está muy por debajo del valor normal; por lo que teniendo este valor en la base de datos se puede acceder desde el aplicativo para que los usuarios puedan estar informados de la posible falla.

Todo el código relacionado al sistema inteligente se encuentra en el anexo E.

7 RESULTADOS

7.1 REDES HAN Y NAN

En primer lugar, se comprueba el funcionamiento de la red compuesta por las placas ESP que serán las encargadas de transmitir los datos obtenidos en el medidor y el Gateway. Se verifica el alcance máximo al cual podrían estar los medidores del Gateway utilizando el protocolo implementado ESP-Now y verificando que el tamaño del mensaje en bytes sea el mismo que el tamaño recibido, esto se verifica enviando junto con el mensaje el tamaño del mensaje a enviar y en la placa receptora (Gateway) se obtiene el tamaño enviado en el mensaje y el tamaño del mensaje entrante, con esto se obtiene como resultado que la mayor distancia que se puede lograr sin pérdida de paquetes sería de los 15 a 20 metros que a su vez corresponde a la distancia máxima a la que pueden estar los medidores del Gateway.

Una vez comprobado el correcto despliegue de los módulos de comunicación y su conexión con el Gateway se debe comprobar la conexión de este con la base de datos, para esto se debe tener en cuenta que el estado de los servicios en cada módulo de comunicación por defecto es inactivo, es decir que la variable *sService* de cada medidor está dispuesto como un array con todos sus valores en cero ({0,0,0}), por lo que a su vez la variable sMeasure está dispuesto como un array con todos sus valores en cero ({0,0,0})m es decir no se ha conectado al Gateway (figura 67).

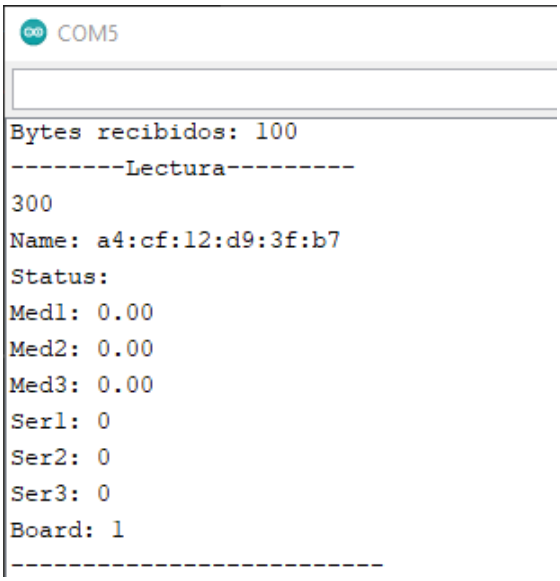


Figura 67 Valores por default de los medidores [41]
Fuente: Arduino IDE 1.8.19, Software | Arduino, 2022

Para cambiar los estados de estos servicios, en Realtime Database de Firebase, en la tabla *statusTable* se debe de colocar en el respectivo medidor, la orden "servicio" en la variable *Status* y el estado en el cual deberían estar cada servicio en la variable tipo array *Services* como se muestra en la figura 68.

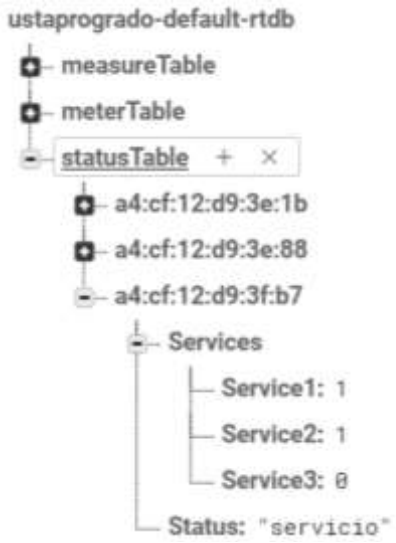


Figura 68 Valores de variables en Firebase [13]
Fuente: Google Inc, Firebase, 2022.

Cuando se coloca en la variable *Status* el valor de "servicio", el medidor tiene que cambiar el estado de los servicios a activos para empezar a medir dando como resultado él envió de las medidas según lo indicado en la base de datos, en este caso la variable *Status* es igual a "servicio" y en *Services* las dos primeras posiciones son '1' y la última posición es '0' por lo que se tendrá un valor de medida en el servicio de energía eléctrica y de agua, mientras que el de gas seguirá siendo '0' (figura 69). Cabe recalcar que el numero actual de bytes que utiliza el módulo de comunicaciones del medidor al momento de enviar los datos de los valores medidos es de 100 bytes de un máximo de 250 bytes por lo que aunque actualmente se tiene un correcto funcionamiento con la información proporcionada esta puede ser mayor dependiendo de las necesidades y requerimientos.

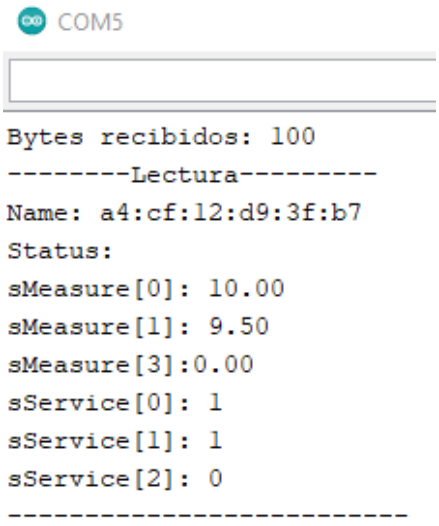


Figura 69 Valores de variables enviados desde el medidor [41]
Fuente: Arduino IDE 1.8.19, Software | Arduino, 2022

De esta manera se comprueba el funcionamiento tanto de la red HAN que comprende el envío de los datos de medición desde cada uno de los módulos de comunicación de los medidores al Gateway y la modificación por parte del Gateway al funcionamiento del respectivo medidor; y de la red NAN mediante la lectura de la información y escritura de los datos de los respectivos medidores en la Realtime Database de Firebase mediante el Gateway.

7.2 CURVAS DE CONSUMO DE SERVICIO

La solución multiplataforma desarrollada en Xamarin tiene como objetivo principal mostrar al usuario el consumo de los tres servicios en diferentes escalas, por mes, día y hora tanto en Android como en Universal Windows Platform. Teniendo en cuenta como se realizó la generación de los datos basados en la demanda en Colombia se obtienen las siguientes curvas de consumo.

En las siguientes figuras se puede observar el comportamiento anteriormente mencionado (6.3) en las gráficas en escala de hora (figura 70, 71 y 72) y en las figuras 73, 74 y 75 en escala de día de los servicios de energía eléctrica, agua y gas respectivamente.

Figura 70 Curva de carga por estrato energía eléctrica (escala de hora)

Fuente: “El Autor”.

Figura 71 Curva de consumo de agua potable (escala de hora)

Fuente: “El Autor”.

Figura 72 Curva de consumo de gas (escala de hora)
Fuente: “El Autor”.

Figura 73 Curva de carga por estrato energía eléctrica (escala de día)
Fuente: “El Autor”.

Figura 74 Curva de consumo de agua potable (escala de día)
Fuente: “El Autor”.

Figura 75 Curva de consumo de gas (escala de día)
Fuente: “El Autor”.

7.3 SISTEMA INTELIGENTE

El algoritmo de regresión logística que se implementó tiene un porcentaje de precisión de 91.9753% es decir que de diez registros que se prueben con este algoritmo aproximadamente ocho no van a estar clasificados correctamente. En esta prueba no se realizó la prueba con 100 registros si no con 10 todos pertenecientes a la data de testeo, es decir que ninguno de estos registros se tuvo en cuenta para el entrenamiento del algoritmo. De los 10 registros sobre

los cuales se realizó la prueba tan solo uno (número 10 el cual debía pertenecer al grupo 1 y se clasificó como 2) no se predijo de manera correcta, en la imagen se puede apreciar tanto la hora como el valor medido, además del grupo seleccionado (grupo que predijo el algoritmo de regresión) y el grupo real (grupo al que pertenece el registro). Todas estas pruebas se realizaron con los registros del servicio de energía eléctrica, de igual manera para los otros servicios se maneja el mismo procedimiento (figura 76).

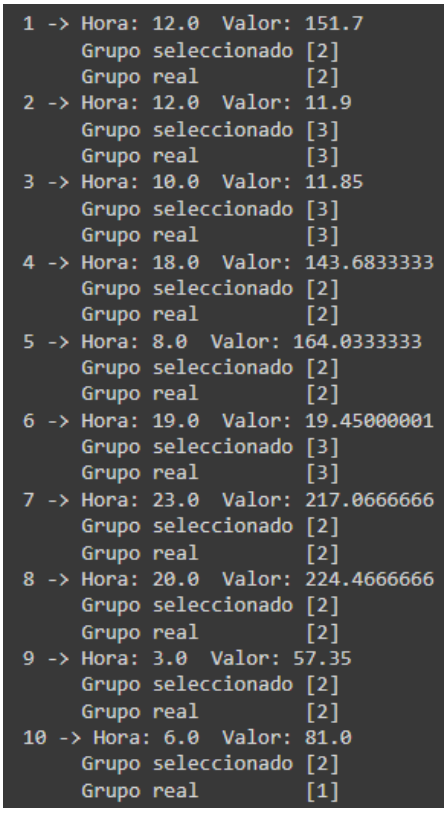


Figura 76 Test algoritmo de regresión logística [41]
Fuente: Michael Tovar, "Tesis.ipynb - Colaboratory (google.com)", 2022.

En la figura 77 se puede observar el resultado de la evaluación sobre la medida tomada a las 18 horas y la cual arroja como resultado que está dentro del valor estimado (valor de '1') para el respectivo medidor evaluado.

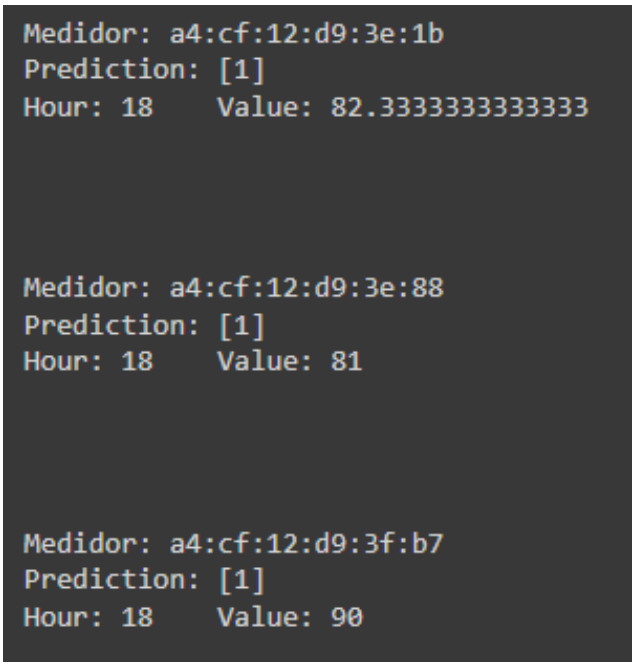


Figura 77 Evaluación sobre datos de Firebase [41]
Fuente: Michael Tovar, "Tesis.ipynb - Colaboratory (google.com)", 2022.

7.4 ANÁLISIS ECONÓMICO

7.4.1 Ahorro

La Infraestructura de Medición Avanzada permite a la empresa prestadora del servicio gestionar de manera remota todo lo que refiere al medidor, es decir que tanto la recolección de datos como en la conexión y desconexión del servicio son costos que se reducen. Además de reducir las pérdidas no técnicas por fraude y manipulación de medidores, y por pedidas técnicas al detectar fallas y comprobar el estado de la red de distribución. En temas de facturación del servicio también es posible que ya no existan las facturas físicas siempre y cuando el usuario acepte recibir dicha factura por medio digitales, incluso se podría implementar sobre la solución multiplataforma la opción de recibir la factura por este medio. [26]

7.4.2 Implementación del proyecto

7.4.2.1 Módulo de comunicaciones

El módulo de comunicaciones actual es la placa ESP 01 que en el mercado tiene un precio promedio de $ 10.000 COP ($ 2.5 US) sin embargo esta palca no cuenta con conversor análogo digital (ADC), solo cuenta con 2 pines GPIO (General Purpose Input/Output, Entrada/Salida de Propósito General) y dos pines TXD y RXD que si bien pueden funcionar como GPIO lo más recomendable seria utilizar estos dos pines para comunicación serial para la comunicación con el medidor ya que esta placa soporta protocolo I2C por lo que se puede comunicar con diversidad de sensores para posteriormente enviar al Gateway esta información.

De igual manera si se desea que el módulo de comunicaciones tenga entre otras funcionalidades un pin conversor análogo digital (ADC) se podría utilizar en vez de una placa ESP 01 el Dev Kit NodeMCU la cual sería programada con el mismo programa de la placa anterior y funcionaría de la misma manera, solo que tendría más almacenamiento ya que pasaría de 1MB a 4MB de memoria ROM, comunicación serial (I2C).

Si se compara el precio de los medidores inteligentes que están en un precio promedio que supera los 11 millones de pesos colombianos, específicamente el medidor MTR 5000 de Kele Inc. tiene un valor promedio de $11.862.903 COP ($2886 US) lo cual es mucho más alto que si le compara con un medidor digital actual, por ejemplo el medidor C2000 de la empresa Microstar tiene un costo de $405.000 COP ($ 99 US) el cual podría utilizarse para la implementación de la red propuesta ya que cuenta con opción de comunicación directa con Óptico/Infrarrojo, RS232, RS485 y comunicación remota GPRS/GSM, WiFi o Ethernet.

7.4.2.2 Gateway

El DevKit NodeMCU se utiliza como gateway y es el encargado de centralizar la información de los medidores y guardar vía WiFi la información de medición en la base de datos alojada en Firebase. Esta placa puede llegar a soportar un máximo de 8 conexiones simultaneas, es decir que podría llegar a gestionar la información de 8 módulos de comunicación, por ende 8 medidores. El costo de esta placa es de $25.000 COP ($7 US aproximadamente).

Cabe mencionar que una de las actualizaciones que puede tener el Gateway es ampliar su rango de comunicación permitiendo a este comunicarse a una red GSM mediante la conexión al Shield GSM 900 el cual permite este tipo de conexiones y permitiendo un alcance mucho mayor a la hora de transmitir información. Esta actualización sumaría al valor actual un aproximado de $40.000 COP ($10 US) [2]

7.4.2.3 Sistema inteligente

La utilización de Colab es totalmente gratis con algunas excepciones, que si bien no son obligatorias para su uso si presentan una eficiencia considerablemente mayor al utilizarlas. Para el alcance de este proyecto no hubo costos por el uso de Colaboratory, sin embargo, los planes de paga incluyen mayor velocidad de GPU y más tiempo de ejecución entre otras funcionalidades por un valor máximo de $181.509 COP ($45 US).

7.4.2.4 Almacenamiento en Base de datos

Firebase cuenta con un plan gratuito (Plan Spark) y un plan prepago (Plan Blaze) el cual se cobra de acuerdo a lo que se haya utilizado, sin embargo para el alcance de este proyecto no es necesario tener el plan prepago ya que el plan Spark permite que existan hasta 100 conexiones simultaneas de las cuales solo se utilizaría una conexión que corresponde al Gateway; también tiene un máximo de descarga de 10GB por mes, el mayor tráfico de descarga ocurre cuando se abre la aplicación USTAPG ya que utiliza 2.97MB y porque se deben de descargar todas las medidas por lo que si se supone que el usuario abre tres veces la aplicación cada día durante un mes (30 días) estaría descargando un total de 267.3MB por lo que para sobrepasar el límite de descarga mensual se necesitarían 37 usuarios con el mismo comportamiento de uso; y por último el almacenamiento máximo es de 1GB por mes de los cuales solo se han utilizado 6.82MB que corresponde al almacenamiento que se tienen de 3 medidores durante 6 meses (desde el 1 de enero hasta el 30 de mayo) por lo que para alcanzar el almacenamiento máximo se requerirían alrededor de 2000 medidores en el mes para superar el almacenamiento máximo. Por consiguiente, para efectos de alcance de este proyecto en tema económico no habría gastos en cuanto a base de datos se refiere.

7.4.2.5 Estimado

En la tabla 1 se muestra el valor estimado de la implementación de una Infraestructura de Medición Avanzada si se utilizan 8 medidores que corresponde al número máximo de conexiones que puede tener el Gateway simultáneamente y con este número de medidores no se tendrían costos por almacenamiento en la nube (Realtime Database) y con respecto a los costos que envuelven a al software multiplataforma se excluyen ya que no se necesitó de ningún aporte económico para ser desarrollado exceptuando el consumo energético en el cual se desarrolló dicho software. Además, se incluye el medidor para la parte de metrología del servicio de energía eléctrica que, aunque este fuera del alcance de este proyecto si hace parte fundamental de dicha infraestructura.

Elemento	Descripción	Unidades	Valor unidad (COP)	Valor unidad (US)	Valor total (COP)	Valor total (US)
ESP 01	Módulo de comunicaciones medidor	8	$10,000	$2.5	$80,000	$20
DevKit NodeMCU	Gateway	1	$25,000	$7	$25,000	$7
Medidor C2000	Medidor	8	$405,000	$99	$3,240,000	$792
TOTAL			$440,000	$109	$3,345,000	$819

Tabla 1 Presupuesto estimado para 8 medidores (solo con la medición del servicio de energía eléctrica)
Fuente: "El Autor".

7.5 PROYECCIÓN

La Ley 142 de 1994 en sus artículos 34 y 133 resalta que no es obligación del usuario instalar los medidores inteligentes que ofrecen las empresas de energía, pueden adquirir estos medidores de proveedores ajenos a la empresa de energía siempre que este medidor cumpla con los requisitos técnicos. Por otra parte, desde mayo del 2019 el Ministerio de Minas y Energía (MME) expide la resolución 40483 en la que establece que como mínimo que el 75% de los usuarios conectados al Sistema Interconectado Nacional haga parte de una Infraestructura de Medición Inteligente y para los usuarios no interconectados no se tiene una meta aún establecida. [26]

Con respecto a la resolución expedida por el MME la Superintendencia de Industria y Comercio de Colombia en el documento "MEDICIÓN Y GESTIÓN INTELIGENTE DE CONSUMO ELÉCTRICO" muestra las tendencias a nivel nacional e internacional en cuanto a invención e impacto industrial se puede observar en la figura 78 que en materia de medidores inteligentes es en donde se invierten más recursos por el aumento global en la implementación de Infraestructuras de Medición Inteligente y la poca investigación que se presentó en años anteriores. Por el contrario, en el mercado de la arquitectura de la red de comunicación si se le compara con lo invertido en el campo de los medidores inteligentes es un área bastante olvidada. Las empresas que más invierten en este campo son Qualcomm Inc, Toshiba Corp y LG Electronics Inc. Que si bien son empresas con reconocimiento mundial no son tan influyentes en el ámbito industrial (figura 79) por lo que proyecta como una oportunidad ya que en Colombia hasta la fecha solo se han realizado la postulación de solo 9 patentes se las cuales solo se ha aprobado

una de origen sudafricano. De estas 9 patentes 4 son provenientes de Estados Unidos, 3 de Colombia, una de España y una de Sudáfrica. [28]

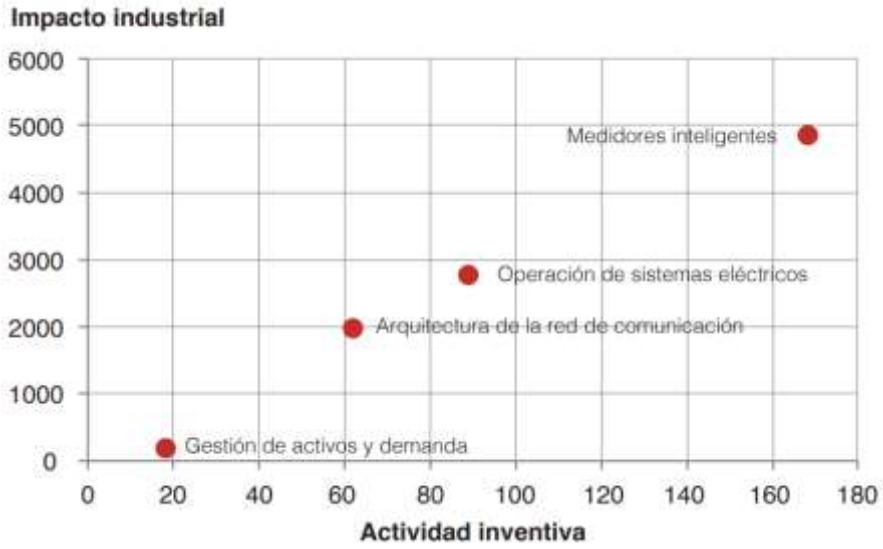


Figura 78 Impacto industrial vs Actividad inventiva en campos relacionados con la distribución y gestión de la energía eléctrica [28]

Fuente: Silva, Luis; Huertas, Andrea; "Medición Y Gestión Inteligente De Consumo Eléctrico", Superintendencia de industria y Comercio de Colombia, Boletín tecnológico, Centro de Información Tecnológica y Apoyo a la Gestión de la Propiedad Industrial (CIGEPI) junio, 2016.

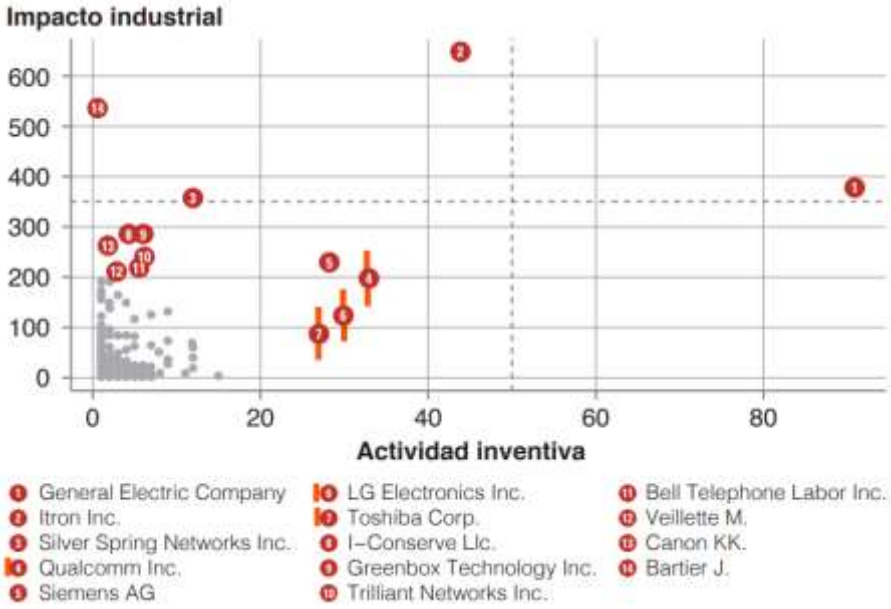


Figura 79 Impacto industrial vs Actividad inventiva de las empresas más representativas [28]

Fuente: Silva, Luis; Huertas, Andrea; "Medición Y Gestión Inteligente De Consumo Eléctrico", Superintendencia de industria y Comercio de Colombia, Boletín tecnológico, Centro de Información Tecnológica y Apoyo a la Gestión de la Propiedad Industrial (CIGEPI) junio, 2016.

En cuanto a la proyección que tiene el Departamento Nacional de Planeación se espera que a nivel energético 5 desarrollos tecnológicos contribuyan al cumplimiento de lo propuesto en Horizonte 2030 y a su vez ayuden con metas propuestas en la COP21. Un de estas tecnologías son las Smart Grids que más que una tecnología es un concepto que abarca muchas más tecnologías, que además de hacer un énfasis en la FNCER (Fuentes no convencionales de energía renovables) también involucra conceptos como gestión de la demanda, generación distribuida y vehículos eléctricos y claramente las infraestructuras de medición avanzada hacen parte de este desarrollo. En la figura 80 se muestran cuatro componentes fundamentales de las redes inteligentes y sus funciones principales a nivel general.

Figura 80 Aspectos fundamentales redes inteligentes. [27]

Fuente: DNP; World Bank Group; Enersinc, “Energy Demand Situation in Colombia”, Colombia, 2017.

En la actualidad, según el diario portafolio hasta el 7 de febrero de 2022 se encuentran 450.000 medidores inteligentes en las cuales las empresas con mayor porcentaje son Enel-Codensa, Grupo EPM, Surtigas, Air-e, Vanti, Gases del Caribe, Efigas, Afinia, Celsia, entre otras. [29]

8 CONCLUSIONES

La programación de plataforma cruzada (cross platform) que ofrece Xamarin Form es una ventaja al momento de desarrollar soluciones para diferentes plataformas ya que al estar desarrollado en lenguaje C# se tienen todas las ventajas del mismo como LinQ o la programación orientada a objetos, y en front end ofrece mucha versatilidad a la hora de realizar una solución responsive en cada plataforma con la ventaja de no tener la necesidad de programar en lenguaje nativo de cada plataforma.

Con el rápido crecimiento de la población que conlleva a una mayor demanda de los servicios básicos se hace sumamente necesario gestionar de manera adecuada los recursos existentes por lo que una infraestructura de medición avanzada pasa a ser una herramienta indispensable para cumplir con este fin, sin embargo el gran obstáculo a pasar a estas tecnologías es que resulta realmente costoso por lo que cualquier implementación que cumpla con las especificaciones mínimas requeridas y sea económica resulta acertada y oportuna.

La intercomunicación entre medidores y entre medidores y puerta de enlace o concentrador se ve limitada por el alcance que tiene la tecnología que se esté utilizando, en este caso ESP-NOW por lo que si este se actualizara a una tecnología que ya tenga una infraestructura construida como lo es una red GSM (Red celular) se amplían los limites en cuanto a distancia se refiere y no resulta una actualización que afecte negativamente el funcionamiento o planteamiento de las comunicaciones.

El sistema inteligente encargado de alertar si existe un consumo anormal en el cliente se entrenó con datos que, si bien está basada en cifras de un consumo real no lo son del todo, por lo que la respuesta de este está un poco sesgada. Sin embargo, el algoritmo de aprendizaje al ser un algoritmo supervisado ya se conocen

las salidas que debería tener cada registro lo que hace que para entrenar el sistema para que responda a condiciones reales este puede adaptarse a cualquier situación.

Es de suma importancia conocer el entorno en donde se implementará la Infraestructura de medición Avanzada ya que independientemente de que las conexiones Wireless sean las menos costosas y más fáciles de implementar no siempre son la mejor solución que se adapte al entorno.

Las funcionalidades que puede alcanzar el medidor se pueden ampliar si se utiliza para el módulo de comunicaciones el devKit NodeMCU ya que a pesar de que no se utilizó toda la capacidad de la placa actual, esta no cuenta con tanta versatilidad por lo que para futuras implementaciones no es recomendable utilizar la misma placa ya que se limita el funcionamiento a solo el envío o recepción de información.

9 BIBLIOGRÁFICAS

[1] A framework for intrusion detection system in advanced metering infrastructure, Article in Security and Communication Networks, January 2014

[2] INTEROPERABILIDAD ENTRE MEDIDORES INTELIGENTES DE ENERGÍA ELÉCTRICA RESIDENCIAL, Milton Gonzalo Ruiz Maldonado. Quito, Marzo del 2015.

[3] IEEE. 2030.2-2015 - IEEE Guide for the Interoperability of Energy Storage Systems Integrated with the Electric Power Infrastructure. 2015.

[4] Javier Bernardo Cabrera Mejía. Diseño de la red de telecomunicaciones para el sistema de medición avanzada (AMI) de energía eléctrica en el centro histórico de la ciudad de Cuenca. Quito,Ecuador, 2014.

[5] Javier Revelo-Fuelagán Álvaro José Cervelion-Bastidas Guefry L. Agredo-Méndez. Diseño y evaluación de desempeño de la infraestructura AMI para la microrred de la Universidad de Nariño. Pasto, Colombia, 2018.

[6] Santiago Xavier Guzmán Cabascango. Estudio Y Diseño De Un Sistema Domiciliario Para Control De Consumo De Energía Eléctrica Utilizando Redes Eléctricas Inteligentes. Quito, Ecuador, 2013.

[7] Edwin Marcelo García Torres Milton Gonzalo Ruiz Maldonado. Interoperabilidad entre medidores inteligentes de energía eléctrica reutilizando redes celulares. Quito, Ecuador, 2014.

[8] Lorena Fernanda Rodríguez Cortés. Arquitectura de red de la infraestructura de medición avanzada (AMI) para el desarrollo de redes inteligentes en Colombia. Bogotá, Colombia, 2013.

[9] Johan M. Alvarado B. Servicios de medición avanzada (AMI) para redes inteligentes y su adaptabilidad en el marco de la legislación ecuatoriana. Quito, Ecuador, 2011.

[10] P. Balakrishna; K. Rajagopal; K. S. Swarup. Analysis on AMI system requirements for effective convergence of distribution automation and AMI systems. Delhi, India, 2014.

[11] Sistemas Industriales Distribuidos: una filosofía de la automatización. Ingeniería Técnica Telecomunicación. 3er curso. Dpto. Ingeniería electrónica, Universidad de Valencia. Alfredo Rosado Muños

[12] IEEE, “IEC 61968-9 Application integration at electric utilities - System interfaces for distribution management - Part 9: Interfaces for meter reading and control”, International Standard, 2016.

[13] Google Inc, Firebase, 2022.

[14] Espressif Systems Ltda, “ESP8266 SDK Getting Started Guide (For SDK V2.X and earlier versions only)”, Octubre, 2020.

[15] Grokhotkov, Ivan, “ESP8266 Documentación Arduino Core”, 2017.

[16] Espressif Systems Ltda, “ESP-NOW User Guide”, Shanghai ,2022.

[17] Britch David; Schonning, Nick; Dunn, Craig; JohnCOsborne, “The Model-View-ViewModel Pattern”, Agosto, 2021.

[18] Aloïs Deniel, “Microcharts: Elegant Cross-Platform Charts for Every App”, Octubre, 2017.

[19] Linhart, John; J.Nma; Suwatchai, “Firebase Arduino Client Library for ESP8266 and ESP32”, 2022.

[20] CREG, Grupo energéticos consultore, “Estrategias para la implementación de esquemas de señales de precios y cargos horarios a los usuarios finales en el SIN, para ser utilizados en programas de respuesta de la demanda”, diciembre, 2020.

[21] Cortés, Ernesto; El Tiempo, CityTV, “4 años para salvar el agua de Bogotá”, Bogotá, 2020.

[22] Laurent Bugnion, “The MIT License (MIT)”, Copyright (c) 2009 - 2018

[23] MINISTERIO DE MINAS Y ENERGÍA, “RESOLUCIÓN 40072 DE 2018”, Diario Oficial No. 50.492, 30 enero, 2018.

[24] Comisión de Regulación de Energía y Gas (CREG), “Medición Avanzada en el servicio de energía eléctrica en Colombia “AMI””

[25] Universidad Nacional de Colombia. Facultad de Ingeniería, Departamento de Energía Eléctrica y Electrónica, “Definición de Funcionalidades Mínimas de un Medidor Inteligente en Colombia”, Bogotá, Universidad Nacional, 2016

[26] Ministerio de Minas y Energía de Colombia, “ABC sobre Infraestructura de Medición Avanzada (AMI)”, 2020.

[27] DNP; World Bank Group; Enersinc, “Energy Demand Situation in Colombia”, Colombia, 2017.

[28] Silva, Luis; Huertas, Andrea; “Medición Y Gestión Inteligente De Consumo Eléctrico”, Superintendencia de industria y Comercio de Colombia, Boletin tecnológico, Centro de Información Tecnológica y Apoyo a la Gestión de la Propiedad Industrial (CIGEPI) junio, 2016.

[29] Alfonso López Suárez, Portafolio, “Los medidores ‘inteligentes’ llegarían a 6 millones de usuarios”, febrero, 2022.

[30] https://www.kele.com/power-monitoring-and-protection/class-5000-smart-meter.aspx

[31] https://ineldec.com/producto/medidor-bidireccional-monofasico-c2000/

[32] Una Breve Historia del Machine Learning (timeline), Victor Gonzalez Pacheco, 18 enero, 2019.

[33] A Brief History of Machine Learning Keith D. Foote on December 3, 2021

[34] My First Deep Learning System of 1991 + Deep Learning Timeline 1962-2013, Jurgen Schmidhuber, The Swiss AI Lab IDSIA, Galleria 2, 6928 Manno-Lugano, University of Lugano & SUPSI, Switzerland, 20 December 2013

[35] In-Ho Choi ; Joung-Han Lee. Development of smart controller with demand response for AMI connection. Gyeonggi-do, South Korea, 2010.

[36] Nam-Joon Jung ; Il-Kwon Yang ; Seung-Woon Park ; Sei-Young Lee. A design of AMI protocols for two way communication in K-AMI. Gyeonggi-do, South Korea, 2011.1415

[37] Shi Chen ; Kenan Xu ; Zhenhong Li ; Fei Yin ; Haifeng Wang. A privacy-aware communication scheme in Advanced Metering Infrastructure (AMI) systems. Shanghai, China, 2013.

[38] David Britch, Craig Dunn, Hemant Arya, Justin Johnson, “What is Xamarin? - Xamarin | Microsoft Docs”, Diciembre, 2021.

[39] Arduino IDE 1.8.19, Software | Arduino, 2022

[40] Crea códigos QR gratis con nuestro generador (qr-code-generator.com)

[41] Michael Tovar, “Tesis.ipynb - Colaboratory (google.com)”, 2022

ANEXOS

9.1 ANEXO A

CÓDIGO MEDIDOR

Para la programación de los módulos de comunicación de los medidores se utiliza el mismo código con la excepción que se cambian las variables de dirección y de BOARD_ID.

Los módulos de comunicación de los medidores están configurados para que envíen datos cada 15 minutos (Variable “Interval”) por defecto o cada que el Gateway envié una solicitud de medida.

----------INICIO----------

```
//#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <espnow.h>
#define LED_BUILTIN 2
#define BOARD_ID 1
//#define BOARD_ID 2
//#define BOARD_ID 3
//#########################VARIABLES#################################
//MAC Gateway
uint8_t GatewayAddress[] = {0xE8, 0xDB, 0x84, 0x9D, 0x2D, 0x3E};

//Definición de variables a enviar
int Status;
bool Services[3];
float Measure[3];

char InOrder[10]; //Mensaje de entrada
int Message_lenGet; //Tamaño mensaje de entrada mensaje
int Message_lenGetReal; //Tamaño mensaje de entrada
//Intervalo de mediciones
const long Interval = 900000;//10000;
unsigned long PreviousMillis = 0;

char Success[10];//Estado del mensaje enviado
int Message_lenSend; //Tamaño mensaje de salida
```

```
//Estructuras de mensajes
typedef struct Message {
  char sName[18];
  char sAddress[50];
  int sStatus;
  int sServices[3];
  int sBoard_Id;
  float sMeasure[3];
} Message;

typedef struct Order_struct_message {
  char sInOrder[10];
  int sServices[3];
  int Message_len;
} Order_struct_message;

Message MeasureMessage;
Order_struct_message InOrder_struct;

//Callback cuando se envia mensaje
void Send_cb_Message(uint8_t *mac_addr, uint8_t sendStatus) {
  if (sendStatus == 0){
    strcpy(Success,"Enviado");
  }
  else{
    strcpy(Success,"No enviado");
  }
}

//Recibir datos
void Get_cb_Message(uint8_t * mac, uint8_t *incomingData, uint8_t len) {
  memcpy(&InOrder_struct, incomingData, sizeof(InOrder_struct));
  strcpy(InOrder, InOrder_struct.sInOrder);
  Serial.print("Orden: ");
  Serial.println(InOrder);
  Message_lenGet = InOrder_struct.Message_len;
  Message_lenGetReal = len;
  if(strcmp(InOrder, "estado") == 0){
    SendMessage();
```

```
  }
  else if(strcmp(InOrder, "inactivar") == 0){
    Services[0] = 0;
    Services[1] = 0;
    Services[2] = 0;
    MeasureMessage.sStatus = 0;
  }
  else if(strcmp(InOrder, "activar") == 0){
    Services[0] = InOrder_struct.sServices[0];
    Services[1] = InOrder_struct.sServices[1];
    Services[2] = InOrder_struct.sServices[2];
    MeasureMessage.sStatus = 1;
  }
  else if(strcmp(InOrder, "servicio") == 0){
    Services[0] = InOrder_struct.sServices[0];
    Services[1] = InOrder_struct.sServices[1];
    Services[2] = InOrder_struct.sServices[2];
    MeasureMessage.sStatus = 1;
  }
  else{}
}

void GetMeasure(){
  long Ram = random(10);
  float SMeasure[3] = {1.0+Ram, 0.5+Ram , 5.0+Ram};
  Measure[0] = SMeasure[0];
  Measure[1] = SMeasure[1];
  Measure[2] = SMeasure[2];
}

//Enviar mensaje
void SendMessage(){
  GetMeasure();
  //servicio 1
  if(Services[0] == 0){
    MeasureMessage.sMeasure[0] = 0.0;
  }
  else {
    MeasureMessage.sMeasure[0] = Measure[0];
```

```
  int m = Measure[0] == 0.0 ? Status = 2: Status;
 }
 //servicio 2
 if(Services[1] == 0){
  MeasureMessage.sMeasure[1] = 0.0;
 }
 else {
  MeasureMessage.sMeasure[1] = Measure[1];
  int m = Measure[0] == 0.0 ? Status = 3: Status;
 }
 //servicio 3
 if(Services[2] == 0){
  MeasureMessage.sMeasure[2] = 0.0;
 }
 else {
  MeasureMessage.sMeasure[2] = Measure[2];
  int m = Measure[0] == 0.0 ? Status = 4: Status;
 }
 if (Services[0]  == 1 && Services[1]  == 1) int m = Measure[0] == 0.0 && Measure[1]
== 0.0 ? Status = 5: Status;
 if (Services[0]  == 1 && Services[2]  == 1) int m = Measure[0] == 0.0 && Measure[2]
== 0.0 ? Status = 6: Status;
 if (Services[1]  == 1 && Services[2]  == 1) int m = Measure[1] == 0.0 && Measure[2]
== 0.0 ? Status = 7: Status;
 if (Services[0] == 1 && Services[1] == 1 && Services[2] == 1) int m = Measure[0]
== 0.0 && Measure[1] == 0.0 && Measure[2] == 0.0 ? Status = 8: Status;
 MeasureMessage.sStatus = Status;
 MeasureMessage.sBoard_Id = BOARD_ID;
 MeasureMessage.sServices[0] = Services[0];
 MeasureMessage.sServices[1] = Services[1];
 MeasureMessage.sServices[2] = Services[2];
 esp_now_send(GatewayAddress,        (uint8_t        *)        &MeasureMessage,
sizeof(MeasureMessage));
}

void printIncomingReadings(){
 Serial.println("--------Lectura---------");
 Serial.print("Status: ");
 Serial.println(MeasureMessage.sStatus);
```

```
  Serial.print("Address: ");
  Serial.println(MeasureMessage.sAddress);
  Serial.print("Med1: ");
  Serial.println(MeasureMessage.sMeasure[0]);
  Serial.print("Med2: ");
  Serial.println(MeasureMessage.sMeasure[1]);
  Serial.print("Med3: ");
  Serial.println(MeasureMessage.sMeasure[2]);
  Serial.print("Ser1: ");
  Serial.println(MeasureMessage.sServices[0]);
  Serial.print("Ser2: ");
  Serial.println(MeasureMessage.sServices[1]);
  Serial.print("Ser3: ");
  Serial.println(MeasureMessage.sServices[2]);
  Serial.print("Board Id: ");
  Serial.println(MeasureMessage.sBoard_Id);
  Serial.println("-------------------------");
}

void setup() {
  //Opcional
  Serial.begin(115200);
  pinMode(LED_BUILTIN,OUTPUT);
  randomSeed(5);
  //ESP en modo estación
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  //Iniciar ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  //Rol de ESP (enviar y recibir)
  esp_now_set_self_role(ESP_NOW_ROLE_COMBO);
  //Estado del mensaje enviado
  esp_now_register_send_cb(Send_cb_Message);
  //Conectar ESP
  esp_now_add_peer(GatewayAddress, ESP_NOW_ROLE_COMBO, 1, NULL, 0);
  //Estado del mensaje recibido
```

```
  esp_now_register_recv_cb(Get_cb_Message);
  Status = 1;
  strcpy(InOrder, "n");
  strcpy(MeasureMessage.sAddress, "Calle 33 sur 87c 29 apto 103 torre 15");
  //strcpy(MeasureMessage.sAddress, "Calle 33 sur 87c 29 apto 103 torre 16");
  //strcpy(MeasureMessage.sAddress, "Calle 33 sur 87c 29 apto 103 torre 17");
  MeasureMessage.sStatus = Status;
  for (int i = 0; i < 3; i++)
  {
    Services[i] = 0;
    Measure[i] = 0.0;
    MeasureMessage.sServices[i] = Services[i];
    MeasureMessage.sMeasure[i] = Measure[i];
  }
}

void loop() {
  unsigned long CurrentMillis = millis();
  if (CurrentMillis - PreviousMillis >= Interval) {
    PreviousMillis = CurrentMillis;
    SendMessage();
    printIncomingReadings();
  }else{}
  if (Success == "No enviado"){
    digitalWrite(LED_BUILTIN,LOW);
  }else{
    digitalWrite(LED_BUILTIN,LOW);
    delay(700);
    digitalWrite(LED_BUILTIN,HIGH);
    delay(700);
  }
}
```

----------FIN----------

9.2 ANEXO B

CÓDIGO GATEWAY

El Gateway lee información de la base de datos cada minuto y de ser necesario algún cambio se envía inmediatamente al medidor correspondiente y escribe en la base de datos cada vez que reciba información del medidor.

----------INICIO----------

```
#include <ESP8266WiFi.h>
#include <espnow.h>
#include <Firebase_ESP_Client.h>
#include <addons/TokenHelper.h>
#include <addons/RTDBHelper.h>
#include <ESP8266HTTPClient.h>
#include <WifiUDP.h>
#include <NTPClient.h>
#include <Time.h>
#include <TimeLib.h>
#include <Timezone.h>

#define WIFI_SSID "Claro_6B01F7"
#define WIFI_PASSWORD "M2Q9N9A6J4M9"
#define API_KEY "AIzaSyCczcx3VUDV58kVpOb6cUNdrdpCuJJKYK4"
#define USER_EMAIL "mtovarson@gmail.com"
#define USER_PASSWORD "Alfa199823"
#define DATABASE_URL "https://ustaprograd0-default-rtdb.firebaseio.com/"
#define DATABASE_SECRET "DATABASE_SECRET"
#define NTP_OFFSET   60 * 60
#define NTP_INTERVAL 60 * 1000
#define NTP_ADDRESS  "pool.ntp.org"

//#########################VARIABLES#################################
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, NTP_ADDRESS, NTP_OFFSET, NTP_INTERVAL);
TimeChangeRule CEST = {"EDT", Second, Sun, Mar, 2, -360};
TimeChangeRule CET = {"EST", First, Sun, Nov, 2, -420};
Timezone CE(CEST, CET);
FirebaseData fbdo;
FirebaseAuth auth;
FirebaseConfig config;
```

```
HTTPClient http;
String GetUrl;
String response;
time_t local, utc;
const char * days[] = {"Domingo", "Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado"};
const char * months[] = {"Ene", "Feb", "Mar", "Abr", "May", "Jun", "Jul", "Ago", "Sep", "Oct", "Nov", "Dic"};
uint8_t Medidor1_dir[] = {0xA4, 0xCF, 0x12, 0xD9, 0x3F, 0xB7};
uint8_t Medidor2_dir[] = {0xA4, 0xCF, 0x12, 0xD9, 0x3E, 0x88};
uint8_t Medidor3_dir[] = {0xA4, 0xCF, 0x12, 0xD9, 0x3E, 0x1B};
char GTaddress[] = "E8:DB:84:9D:2D:3E";
bool firstSend[3];
int Message_lenGetReal[3];
const long Interval = 4000;
unsigned long PreviousMillis = 0;
const long Interval2 = 2000;
unsigned long PreviousMillis2 = 0;
char Success[10];

//Estructuras de mensajes
typedef struct Message {
  char sName[18];
  char sAddress[50];
  int sStatus;
  int sServices[3];
  int sBoard_Id;
  float sMeasure[3];
} Message;

typedef struct Order_struct_message {
  char sInOrder[10];
  int sServices[3];
  int Message_len;
} Order_struct_message;

Message Med1;
Message Med2;
Message Med3;
```

```
Message Lectura;
Message MessageFirebase[3] = {Med1, Med2, Med3};
Order_struct_message InOrder_struct;

// Callback when data is sent
void IsConnected(uint8_t *mac, uint8_t sendStatus) {
  char dirmax[18];
  snprintf(dirmax,                  sizeof(dirmax),"%02x:%02x:%02x:%02x:%02x:%02x",
mac[0],mac[1],mac[2],mac[3],mac[4],mac[5]);
  if (sendStatus == 0){
    strcpy(Success,"GOOD");
  }
  else{
    strcpy(Success,"BAD");
  }
}

//Recibir datos
void GetDataMeasure(uint8_t * mac, uint8_t *incomingData, uint8_t len) {
  memcpy(&Lectura, incomingData, sizeof(Lectura));
  Serial.print("Bytes recibidos: ");
  Serial.print(len);
  if (len != 0){
    char dirmax[18];
    snprintf(dirmax,                  sizeof(dirmax),"%02x:%02x:%02x:%02x:%02x:%02x",
mac[0],mac[1],mac[2],mac[3],mac[4],mac[5]);
    MessageFirebase[Lectura.sBoard_Id - 1].sStatus = Lectura.sStatus;
    Serial.print(" - ");
    Serial.println(Lectura.sBoard_Id);
    strcpy(MessageFirebase[Lectura.sBoard_Id - 1].sName, dirmax);
    strcpy(MessageFirebase[Lectura.sBoard_Id - 1].sAddress, Lectura.sAddress);
    MessageFirebase[Lectura.sBoard_Id - 1].sServices[0] = Lectura.sServices[0];
    MessageFirebase[Lectura.sBoard_Id - 1].sServices[1] = Lectura.sServices[1];
    MessageFirebase[Lectura.sBoard_Id - 1].sServices[2] = Lectura.sServices[2];
    MessageFirebase[Lectura.sBoard_Id - 1].sMeasure[0] = Lectura.sMeasure[0];
    MessageFirebase[Lectura.sBoard_Id - 1].sMeasure[1] = Lectura.sMeasure[1];
    MessageFirebase[Lectura.sBoard_Id - 1].sMeasure[2] = Lectura.sMeasure[2];
    MessageFirebase[Lectura.sBoard_Id - 1].sBoard_Id = Lectura.sBoard_Id;
    Message_lenGetReal[Lectura.sBoard_Id - 1] = len;
```

```
  }
}

void printIncomingReadings(){
  for (int i = 0; i < 3; i++)
  {
    if (MessageFirebase[i].sBoard_Id != 0 || MessageFirebase[i].sBoard_Id != NULL)
    {
      Serial.println("--------Lectura---------");
      //Serial.println(sizeof(MessageFirebase));
      Serial.print("Name: ");
      Serial.println(MessageFirebase[i].sName);
      Serial.print("Status: ");
      Serial.println(MessageFirebase[i].sStatus);
      Serial.print("sMeasure[0]: ");
      Serial.println(MessageFirebase[i].sMeasure[0]);
      Serial.print("sMeasure[1]: ");
      Serial.println(MessageFirebase[i].sMeasure[1]);
      Serial.print("sMeasure[3]:");
      Serial.println(MessageFirebase[i].sMeasure[2]);
      Serial.print("sService[0]: ");
      Serial.println(MessageFirebase[i].sServices[0]);
      Serial.print("sService[1]: ");
      Serial.println(MessageFirebase[i].sServices[1]);
      Serial.print("sService[2]: ");
      Serial.println(MessageFirebase[i].sServices[2]);
      Serial.println("--------------------------");
    }
  }
}

void Write_meterTable_FB(bool IsMeterTable, float _value[3] ,char _name[17], char
_status, char _gt[17], char _address[50], int _services[3]){
  timeClient.update();
  unsigned long utc =  timeClient.getEpochTime();
  local = CE.toLocal(utc);
  if (Firebase.ready())
  {
    FirebaseJson json;
```

```
    FirebaseJson Mjson;
    FirebaseJson MjsonS;
    FirebaseJson jsonS1;
    FirebaseJson jsonS2;
    FirebaseJson jsonS3;
    String Mtable = "meterTable/";
    String table = "measureTable/";
    table += _name;
    Mtable += _name;
    json.setDoubleDigits(3);
    Mjson.setDoubleDigits(3);
    MjsonS.add("Service1", _services[0]);
    MjsonS.add("Service2", _services[1]);
    MjsonS.add("Service3", _services[3]);
    Mjson.add("Gateway", String(_gt));
    Mjson.add("Address", String(_address));
    Mjson.add("Status", String(_status));
    jsonS1.add("Status", _services[0]);
    jsonS1.add("Date", Fecha(local));
    jsonS1.add("Value", _value[0]);
    jsonS2.add("Status", _services[1]);
    jsonS2.add("Date", Fecha(local));
    jsonS2.add("Value", _value[1]);
    jsonS3.add("Status", _services[2]);
    jsonS3.add("Date", Fecha(local));
    jsonS3.add("Value", _value[2]);
    json.add("Servicio1", jsonS1);
    json.add("Servicio2", jsonS2);
    json.add("Servicio3", jsonS3);
    Mjson.add("Services", MjsonS);
    Serial.printf("JSON METER ... %s\n", Firebase.RTDB.setJSON(&fbdo, Mtable,
&Mjson) ? "ok" : fbdo.errorReason().c_str());
    delay(1000);
    Serial.printf("JSON MEASURE ... %s\n", Firebase.RTDB.pushJSON(&fbdo,
table, &json) ? "ok" : fbdo.errorReason().c_str());
  }
}

void Write_FB(int _index)
```

```
{
  if(Message_lenGetReal[_index] != 0){
    if (firstSend[_index]){
      Write_meterTable_FB(true,                    MessageFirebase[_index].sMeasure,
MessageFirebase[_index].sName,MessageFirebase[_index].sStatus,GTaddress,M
essageFirebase[_index].sAddress, MessageFirebase[_index].sServices);
      firstSend[_index] = false;
    }
    else{
      Write_meterTable_FB(false,                   MessageFirebase[_index].sMeasure,
MessageFirebase[_index].sName,MessageFirebase[_index].sStatus,GTaddress,M
essageFirebase[_index].sAddress,MessageFirebase[_index].sServices);
    }
  }
  Message_lenGetReal[_index] = 0;
}

Order_struct_message GetStatus(String _meter){
  Order_struct_message temp;
  if (Firebase.RTDB.getString(&fbdo,"statusTable/" + _meter + "/Status"))
  {
    if (fbdo.dataTypeEnum() == fb_esp_rtdb_data_type_string)
    {
      const char * Stat = fbdo.to<const char *>();
      strcpy(temp.sInOrder, Stat);
    }
    else
    {
      strcpy(temp.sInOrder, "nn");
    }
  }
  if        (Firebase.RTDB.getString(&fbdo,"statusTable/"        +        _meter        +
"/Services/Service1"))
  {
    if (fbdo.dataTypeEnum() == fb_esp_rtdb_data_type_integer)
    {
      temp.sServices[0] = fbdo.to<int>();
    }
    else
```

```
    {
      temp.sServices[0] = 2;
    }
  }
  if (Firebase.RTDB.getString(&fbdo,"statusTable/" + _meter +
"/Services/Service2"))
  {
    if (fbdo.dataTypeEnum() == fb_esp_rtdb_data_type_integer)
    {
      temp.sServices[1] = fbdo.to<int>();
    }
    else
    {
      temp.sServices[1] = 2;
    }
  }
  if (Firebase.RTDB.getString(&fbdo,"statusTable/" + _meter +
"/Services/Service3"))
  {
    if (fbdo.dataTypeEnum() == fb_esp_rtdb_data_type_integer)
    {
      temp.sServices[2] = fbdo.to<int>();
    }
    else
    {
      temp.sServices[2] = 2;
    }
  }
  //Serial.print("Temp: ");
  //Serial.println(temp.sInOrder);
  return temp;
}

String Fecha(time_t t){
  String date = "";if(hour(t) < 10) date += "0";
  date += hour(t);
  date += ":";
  if(minute(t) < 10) date += "0";
  date += minute(t);
```

```
  date += " ";
  date += day(t);
  date += "/";
  date += months[month(t)-1];
  date += "/";
  date += year(t);
  return date;
}

void setup() {
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  // Init ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  Serial.println("Puerta de enlace inicializada");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
      Serial.print(".");
      delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  Serial.printf("Firebase Client v%s\n\n", FIREBASE_CLIENT_VERSION);
  /* Assign the api key (required) */
  config.api_key = API_KEY;
  /* Assign the user sign in credentials */
  auth.user.email = USER_EMAIL;
  auth.user.password = USER_PASSWORD;
  /* Assign the RTDB URL */
  config.database_url = DATABASE_URL;
  Firebase.reconnectWiFi(true);
```

```
  fbdo.setResponseSize(4096);
  String base_path = "/UsersData/";
  /* Assign the callback function for the long running token generation task */
  config.token_status_callback = tokenStatusCallback; //see addons/TokenHelper.h
  /** Assign the maximum retry of token generation */
  config.max_token_generation_retry = 5;
  /* Initialize the library with the Firebase authen and config */
  Firebase.begin(&config, &auth);
  String var = "$userId";
  String val = "($userId === auth.uid && auth.token.premium_account === true &&
auth.token.admin === true)";
  Firebase.RTDB.setReadWriteRules(&fbdo, base_path, var, val, val,
DATABASE_SECRET);
  //Set ESP-NOW Role
  esp_now_set_self_role(ESP_NOW_ROLE_COMBO);
  // Once ESPNow is successfully Init, we will register for Send CB to
  // get the status of Trasnmitted packet
  esp_now_register_send_cb(IsConnected);
  // Register peer
  esp_now_add_peer(Medidor1_dir, ESP_NOW_ROLE_COMBO, 1, NULL, 0);
  esp_now_add_peer(Medidor2_dir, ESP_NOW_ROLE_COMBO, 1, NULL, 0);
  esp_now_add_peer(Medidor3_dir, ESP_NOW_ROLE_COMBO, 1, NULL, 0);
  // Register for a callback function that will be called when data is received
  esp_now_register_recv_cb(GetDataMeasure);

  Message_lenGetReal[0] = 0;
  Message_lenGetReal[1] = 0;
  Message_lenGetReal[2] = 0;
  firstSend[0] = true;
  firstSend[1] = true;
  firstSend[2] = true;
}

void loop() {

  unsigned long CurrentMillis = millis();
  unsigned long CurrentMillis2 = millis();
  if (CurrentMillis - PreviousMillis >= Interval) {
    PreviousMillis = CurrentMillis;
```

```
    Order_struct_message temp1 = GetStatus("a4:cf:12:d9:3f:b7");
    Order_struct_message temp2 = GetStatus("a4:cf:12:d9:3e:88");
    Order_struct_message temp3 = GetStatus("a4:cf:12:d9:3e:1b");
    esp_now_send(Medidor1_dir, (uint8_t *) &temp1, sizeof(InOrder_struct));
    esp_now_send(Medidor2_dir, (uint8_t *) &temp2, sizeof(InOrder_struct));
    esp_now_send(Medidor3_dir, (uint8_t *) &temp3, sizeof(InOrder_struct));
    printIncomingReadings();
  }
  if (CurrentMillis2 - PreviousMillis2 >= Interval2) {
    PreviousMillis2 = CurrentMillis2;
      //Write_FB(0);
      //Write_FB(1);
      //Write_FB(2);
    }
}
```

----------FIN----------

9.3 ANEXO C

CÓDIGO CROSS-PLATFORM

La solución multiplataforma esta construida con el patrón MVVM que será la manera en la que se mostrará el siguiente código el cual también esta disponible en un repositorio público el cual se puede acceder con este link Gestor de versiones: Repositorio https://github.com/MTovarM/USTAPG.

9.3.1 Model

Todos los modelos siguientes están en lenguaje C#.

9.3.1.1 Modelo de Grafica

----------INICIO----------

```
namespace USTAPG.Models
{
    using USTAPG.ViewModels;

    public class Grafica:BaseViewModel
    {
        public string x;
        public int y;

        public string X
        {
            get { return this.x; }
            set { SetValue(ref this.x, value); }
        }
        public int Y
        {
            get { return this.y; }
            set { SetValue(ref this.y, value); }
        }
    }
}
```

----------FIN----------

9.3.1.2 Modelo para leer la información de la tabla infoTable

----------INICIO----------

```
namespace USTAPG.Models
{
    using System;
```

```
    using System.Collections.Generic;
    using System.Text;

    public class InfoTable
    {
        #region Propiedades
        public string StatusClass { get; set; }
        public Services Service1 { get; set; }
        public Services Service2 { get; set; }
        public Services Service3 { get; set; }
        #endregion
    }

    public class Services
    {
        #region Propiedades
        public float AproxMonth { get; set; }
        public string BillDate { get; set; }
        public float Fare { get; set; }
        #endregion
    }
}
```

----------FIN----------

9.3.1.3 Modelo para leer la información de la tabla meterTable

----------INICIO----------

```
namespace USTAPG.Models
{
    using System;
    using System.Collections.Generic;
    using System.Text;

    public class MeterTable
    {
        public string Address { get; set; }
        public string Gateway { get; set; }
        public string Status { get; set; }
        public ServiciosMeter Services { get; set; }
    }
```

```
    public class ServiciosMeter
    {
        public int Service1 { get; set; }
        public int Service2 { get; set; }
        public int Service3 { get; set; }
    }
}
```

----------FIN----------

9.3.1.4 Modelo para leer la información de la tabla sesionTable

----------INICIO----------

```
namespace USTAPG.Models
{
    public class Sesion
    {
        public string Email { get; set; }
    }
}
```

----------FIN----------

9.3.1.5 Modelo para guardar credenciales en almacenamiento local

----------INICIO----------

```
namespace USTAPG.Models
{
    using SQLite;

    public class Usuario
    {
        [PrimaryKey, AutoIncrement]
        public int ID { get; set; }
        public string User { get; set; }
        public string Clave { get; set; }
        public string MAC { get; set; }
    }
}
```

----------FIN----------

9.3.1.6 Modelo para leer la información de measureTable

----------INICIO----------

```
namespace USTAPG.Models
{
    using System;
```

```
    using System.Collections.Generic;
    using System.Text;

    public class measureTable
    {
        public Servicio Servicio1 { get; set; }
        public Servicio Servicio2 { get; set; }
        public Servicio Servicio3 { get; set; }
    }

    public class Servicio
    {
        public string Date { get; set; }
        public int Status { get; set; }
        public float Value { get; set; }
    }
}
```

----------FIN----------

9.3.2 View

Todas las siguientes Views están programadas en el lenguaje de etiquetas XAML.

9.3.2.1 Pagina inicial (Login)

----------INICIO----------

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="USTAPG.Views.LoginPage"
             BindingContext="{Binding Main, Source={StaticResource Locator}}"
             Title="Login">
    <ContentPage.Content>
        <ScrollView
            BindingContext="{Binding Login}">
            <StackLayout>
                <Image
                    WidthRequest="600">
                    <Image.Source>
                        <OnPlatform x:TypeArguments="ImageSource">
                            <On Platform="Android" Value="Login"/>
                            <On Platform="UWP" Value="Login.png"/>
```

```
        </OnPlatform>
    </Image.Source>
</Image>
<StackLayout
    Padding="15">
    <Label
        Text="Por favor, ingresa a tu cuenta"
        HorizontalOptions="CenterAndExpand">
    </Label>
    <Entry
        Text="{Binding Correo, Mode=TwoWay}"
        Margin="12,0,12,0"
        Keyboard="Email"
        Placeholder="Correo">
    </Entry>
    <Entry
        Text="{Binding Clave, Mode=TwoWay}"
        Margin="12,0,12,0"
        IsPassword="True"
        Placeholder="Contraseña">
    </Entry>
    <StackLayout
        VerticalOptions="CenterAndExpand"
        Orientation="Horizontal"
        Margin="12,0,12,0">
        <Label
            FontSize="16"
            HorizontalOptions="StartAndExpand"
            VerticalOptions="Center"
            Text="Recordar ususario">
        </Label>
        <Switch
            IsToggled="{Binding Recordar, Mode=TwoWay}"
            HorizontalOptions="End">
        </Switch>
    </StackLayout>
    <!--<Label
        VerticalOptions="CenterAndExpand"
        HorizontalOptions="Center"
```

```
                    TextColor="#507CFF"
                    Text="¿Olvidaste tu contraseña?">
                </Label>-->
                <ActivityIndicator
                    IsRunning="{Binding Iniciado, Mode=TwoWay}"
                    VerticalOptions="Start">
                </ActivityIndicator>
                <StackLayout
                    VerticalOptions="Start"
                    Margin="25">
                    <Button
                        Command="{Binding EntrarCommand}"
                        IsEnabled="{Binding Habilitado, Mode=TwoWay}"
                        BackgroundColor="#1D56FF"
                        TextColor="#FFFFFF"
                        BorderRadius="23"
                        HeightRequest="46"
                        Text="Entrar">
                    </Button>
                </StackLayout>
            </StackLayout>
        </StackLayout>
    </ScrollView>
  </ContentPage.Content>
</ContentPage>
```

----------FIN----------

9.3.2.2 Pagina para elegir modo de ingresar MAC

----------INICIO----------

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             BindingContext="{Binding Main, Source={StaticResource Locator}}"
             x:Class="USTAPG.Views.MACPage"
             Title="MAC">
  <ContentPage.Content>
    <StackLayout
        BindingContext="{Binding MacMainPage}">
        <Image
            WidthRequest="600">
```

```
    <Image.Source>
        <OnPlatform x:TypeArguments="ImageSource">
            <On Platform="Android" Value="MAC"/>
            <On Platform="UWP" Value="MAC.png"/>
        </OnPlatform>
    </Image.Source>
</Image>
<StackLayout
    Padding="29">
    <Label
        Padding="10"
        FontSize="15"
        TextColor="Black"
        Text="Dinos cual el la dirección de tu dispositivo"
        VerticalOptions="CenterAndExpand"
        HorizontalOptions="CenterAndExpand">
    </Label>
    <Button
        Padding="10"
        Command="{Binding MacQr}"
        VerticalOptions="CenterAndExpand"
        IsEnabled="{Binding Habilitado1, Mode=TwoWay}"
        BackgroundColor="#1D56FF"
        BorderRadius="23"
        TextColor="#FFFFFF"
        HeightRequest="46"
        Text="Escanear Qr">
    </Button>
    <Button
        Padding="10"
        Command="{Binding MacText}"
        VerticalOptions="CenterAndExpand"
        IsEnabled="{Binding Habilitado, Mode=TwoWay}"
        BackgroundColor="#1D56FF"
        BorderRadius="23"
        TextColor="#FFFFFF"
        HeightRequest="46"
        Text="Ingresar MAC">
    </Button>
```

```
            <ActivityIndicator
                IsRunning="{Binding Iniciado, Mode=TwoWay}"
                VerticalOptions="CenterAndExpand">
            </ActivityIndicator>
          </StackLayout>
        </StackLayout>
    </ContentPage.Content>
</ContentPage>
```

----------FIN----------

9.3.2.3 Pagina para ingresar MAC de manera escrita

----------INICIO----------

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="USTAPG.Views.MACInTextPage"
             BindingContext="{Binding Main, Source={StaticResource Locator}}"
             Title="MAC">
    <ContentPage.Content>
        <StackLayout
            BindingContext="{Binding MacTextPage}">
            <Image
                WidthRequest="600"
                Source="MAC">
            </Image>
            <StackLayout
                Padding="29">
                <Grid VerticalOptions="CenterAndExpand">
                    <Grid.RowDefinitions>
                        <RowDefinition Height="*"></RowDefinition>
                        <RowDefinition Height="*"></RowDefinition>
                    </Grid.RowDefinitions>
                    <Entry
                        Grid.Row="1"
                        Grid.Column="0"
                        Text="{Binding MAC, Mode=TwoWay}"
                        Margin="12,0,12,0"
                        Keyboard="Email"
                        Placeholder="Ingresar MAC">
                    </Entry>
```

```
                <Button
                    Grid.Row="2"
                    Grid.Column="0"
                    Command="{Binding Hecho}"
                    VerticalOptions="CenterAndExpand"
                    IsEnabled="{Binding Habilitado, Mode=TwoWay}"
                    BackgroundColor="#1D56FF"
                    BorderRadius="23"
                    TextColor="#FFFFFF"
                    HeightRequest="46"
                    Text="Hecho">
                </Button>
                <ActivityIndicator
                    IsRunning="{Binding Iniciado, Mode=TwoWay}"
                    VerticalOptions="Start">
                </ActivityIndicator>
            </Grid>
        </StackLayout>
    </StackLayout>
  </ContentPage.Content>
</ContentPage>
```

----------FIN----------

9.3.2.4 Página tabulada

----------INICIO----------

```
<?xml version="1.0" encoding="utf-8" ?>
<TabbedPage
  xmlns="http://xamarin.com/schemas/2014/forms"
  xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
  xmlns:tabs="clr-namespace:USTAPG.Views"
  x:Class="USTAPG.Views.MenuTabbedPage"
  Title="Menu">
  <TabbedPage.Children>
    <tabs:ServicePage/>
    <tabs:Service2Page/>
    <tabs:Service3Page/>
    <tabs:MeterPage/>
  </TabbedPage.Children>
</TabbedPage>
```

----------FIN----------

9.3.2.5 Página que muestra servicio

Las páginas en donde se muestra cada servicio tienen la misma estructura, solo se cambian las variables a mostar.

----------INICIO----------

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="USTAPG.Views.ServicePage"
             BindingContext="{Binding Main, Source={StaticResource Locator}}"
             xmlns:Grafica="clr-
namespace:Microcharts.Forms;assembly=Microcharts.Forms"
             Icon="light"
             Title="Energía">
    <ContentPage.Content>
        <ScrollView
            BindingContext="{Binding Meter}">
            <StackLayout>
                <StackLayout
                    Padding="15,15,0,0"
                    BackgroundColor="#1D56FF">
                    <StackLayout
                    Orientation="Horizontal"
                    Padding="25,0,40,0">
                        <Label
                        TextColor="#FFF"
                        FontAttributes="Bold"
                        FontSize="30"
                        VerticalOptions="CenterAndExpand"
                        HorizontalOptions="StartAndExpand"
                        Text="ESTADO"></Label>
                        <Label
                        TextColor="#FFF"
                        VerticalOptions="CenterAndExpand"
                        HorizontalOptions="EndAndExpand"
                        FontSize="30"
                        Text="{Binding EstadoS1,Mode=TwoWay}"></Label>
                    </StackLayout>
                    <StackLayout
                    BackgroundColor="#1D56FF"
                    Orientation="Horizontal"
                    VerticalOptions="StartAndExpand"
                    HorizontalOptions="End"
                    Padding="0,0,20,0">
                        <Label
                        TextColor="#FFF"
```

```
            VerticalOptions="CenterAndExpand"
            Padding="0,0,20,0"
            Text="Ver por:"></Label>
            <RadioButton HorizontalOptions="End"

                    IsChecked="{Binding Mes,Mode=TwoWay}"
                    Command="{Binding RadioButtonCommand}" Text=""/>
            <Label
            TextColor="#FFF"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="End"
            Text="Mes"></Label>
            <RadioButton HorizontalOptions="End"
                    IsChecked="{Binding Dia,Mode=TwoWay}"
                    Command="{Binding RadioButtonCommand}"/>
            <Label
            TextColor="#FFF"
            Padding="0,0,10,0"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="End"
            Text="Día"></Label>
            <RadioButton HorizontalOptions="End"
                    IsChecked="{Binding Hora,Mode=TwoWay}"
                    Command="{Binding RadioButtonCommand}"/>
            <Label
            TextColor="#FFF"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="End"
            Text="Hora"></Label>
        </StackLayout>


        <Grafica:ChartView
            BackgroundColor="#1D56FF"
            VerticalOptions="StartAndExpand"
            x:Name="g1"
            HeightRequest="{Binding Height, Mode=TwoWay}"
            Chart="{Binding Gra1}">
        </Grafica:ChartView>


    </StackLayout>
    <StackLayout
        Padding="15,5,0,0">
        <StackLayout
```

```
Orientation="Horizontal"
Padding="50,0,40,0">
   <Label
      TextColor="#000"
      FontAttributes="Bold"
      VerticalOptions="CenterAndExpand"
      HorizontalOptions="StartAndExpand"
      Text="Estrato: "></Label>
   <Label
   TextColor="#000"
   VerticalOptions="CenterAndExpand"
   HorizontalOptions="EndAndExpand"
   Text="{Binding Estrato,Mode=TwoWay}"></Label>
</StackLayout>
<StackLayout
   Orientation="Horizontal"
   Padding="50,0,40,0">
   <Label
      TextColor="#000"
      FontAttributes="Bold"
      VerticalOptions="CenterAndExpand"
      HorizontalOptions="StartAndExpand"
      Text="Tarifa (COP por kWh): "></Label>
   <Label
      TextColor="#000"
      VerticalOptions="CenterAndExpand"
      HorizontalOptions="EndAndExpand"
      Text="{Binding TarifaS1,Mode=TwoWay}"></Label>
</StackLayout>
<StackLayout
   Orientation="Horizontal"
   Padding="50,0,40,0">
   <Label
   TextColor="#000"
   FontAttributes="Bold"
   VerticalOptions="CenterAndExpand"
   HorizontalOptions="StartAndExpand"
   Text="Aproximado siguiente mes: "></Label>
   <Label
   TextColor="#000"
   VerticalOptions="CenterAndExpand"
   HorizontalOptions="EndAndExpand"
   Text="{Binding AproximadoS1,Mode=TwoWay}"></Label>
</StackLayout>
<StackLayout
```

```
                Orientation="Horizontal"
                Padding="50,0,40,0">
                   <Label
                   TextColor="#000"
                   FontAttributes="Bold"
                   VerticalOptions="CenterAndExpand"
                   HorizontalOptions="StartAndExpand"
                   Text="Fecha de facturación: "></Label>
                   <Label
                   TextColor="#000"
                   VerticalOptions="CenterAndExpand"
                   HorizontalOptions="EndAndExpand"
                   Text="{Binding FechaFacS1,Mode=TwoWay}"></Label>
                </StackLayout>
             </StackLayout>
          </StackLayout>
       </ScrollView>
    </ContentPage.Content>
</ContentPage>
```

----------FIN----------

9.3.2.6 Página donde se muestran los datos del medidor

----------INICIO----------

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="USTAPG.Views.MeterPage"
             BindingContext="{Binding Main, Source={StaticResource Locator}}"
             Icon="meter"
             Title="Medidor">
    <ContentPage.Content>
       <ScrollView
          Padding="0,15,0,0"
          BindingContext="{Binding Meter}"
          BackgroundColor="#1D56FF">
          <StackLayout
             Padding="15,15,0,0">
             <StackLayout
                Orientation="Vertical"
                Padding="25,0,40,0">
                <StackLayout
                   Orientation="Horizontal">
```

```
            <Label
                TextColor="#FFF"
                FontAttributes="Bold"
                FontSize="30"
                VerticalOptions="CenterAndExpand"
                HorizontalOptions="StartAndExpand"
                Text="Medidor"></Label>
            <Label
                TextColor="#FFF"
                VerticalOptions="CenterAndExpand"
                HorizontalOptions="EndAndExpand"
                FontSize="28"
                Text="{Binding Estado,Mode=TwoWay}"></Label>
        </StackLayout>
        <StackLayout
            Orientation="Horizontal">
            <Label
                TextColor="#FFF"
                FontAttributes="Bold"
                FontSize="25"
                VerticalOptions="CenterAndExpand"
                HorizontalOptions="StartAndExpand"
                Text="MAC"></Label>
            <Label
                TextColor="#FFF"
                VerticalOptions="CenterAndExpand"
                HorizontalOptions="EndAndExpand"
                FontSize="23"
                Text="{Binding MAC,Mode=TwoWay}"></Label>
        </StackLayout>
        <Label
            TextColor="#FFF"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="EndAndExpand"
            FontSize="24"
            Text="{Binding Direccion,Mode=TwoWay}"></Label>
    </StackLayout>
    <!--<StackLayout
        Padding="0,0,20,0">
```

```
    <Image
        WidthRequest="600"
        Source="Login">
    </Image>
</StackLayout>-->
<StackLayout
    Padding="25,15,40,0"
    Orientation="Horizontal">
    <Label
        TextColor="#FFF"
        FontAttributes="Bold"
        FontSize="23"
        VerticalOptions="CenterAndExpand"
        HorizontalOptions="StartAndExpand"
        Text="Estrato"></Label>
    <Label
        TextColor="#FFF"
        FontSize="21"
        VerticalOptions="CenterAndExpand"
        HorizontalOptions="EndAndExpand"
        Text="{Binding Estrato,Mode=TwoWay}"></Label>
</StackLayout>
<StackLayout
    Padding="25,15,40,0">
    <Label
        TextColor="#FFF"
        FontAttributes="Bold"
        FontSize="23"
        VerticalOptions="CenterAndExpand"
        HorizontalOptions="StartAndExpand"
        Text="Servicios"></Label>
    <StackLayout
        Padding="25,15,40,0">
        <StackLayout
            Orientation="Horizontal">
            <Label
                TextColor="#FFF"
                FontAttributes="Bold"
                FontSize="20"
```

```
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="StartAndExpand"
            Text="Energía"></Label>
        <Label
            TextColor="#FFF"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="EndAndExpand"
            FontSize="19"
            Text="{Binding Servicio1,Mode=TwoWay}"></Label>
    </StackLayout>
    <StackLayout
        Orientation="Horizontal">
        <Label
            TextColor="#FFF"
            FontAttributes="Bold"
            FontSize="20"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="StartAndExpand"
            Text="Agua"></Label>
        <Label
            TextColor="#FFF"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="EndAndExpand"
            FontSize="19"
            Text="{Binding Servicio2,Mode=TwoWay}"></Label>
    </StackLayout>
    <StackLayout
        Orientation="Horizontal">
        <Label
            TextColor="#FFF"
            FontAttributes="Bold"
            FontSize="20"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="StartAndExpand"
            Text="Gas"></Label>
        <Label
            TextColor="#FFF"
            VerticalOptions="CenterAndExpand"
            HorizontalOptions="EndAndExpand"
```

```
                    FontSize="19"
                    Text="{Binding Servicio3,Mode=TwoWay}"></Label>
                </StackLayout>
            </StackLayout>
        </StackLayout>
    </StackLayout>
  </ScrollView>
 </ContentPage.Content>
</ContentPage>
```

----------FIN----------

9.3.3 View-Model

Todos los modelos siguientes están en lenguaje C#.

9.3.3.1 View Model que sirve como pueste para actualizar parámetros en las view mediante la ejecución del código

----------INICIO----------

```
namespace USTAPG.ViewModels
{
    using System.Collections.Generic;
    using System.ComponentModel;
    using System.Runtime.CompilerServices;

    public class BaseViewModel : INotifyPropertyChanged
    {
        public event PropertyChangedEventHandler PropertyChanged;

        protected void OnPropertyChanged([CallerMemberName] string
propertyName = null)
        {
            PropertyChanged?.Invoke(this, new
PropertyChangedEventArgs(propertyName));
        }

        protected void SetValue<T>(ref T backingField, T value, [CallerMemberName]
string propertyName = null)
        {
            if (EqualityComparer<T>.Default.Equals(backingField, value))
            {
                return;
            }
```

```
            backingField = value;
            OnPropertyChanged(propertyName);
        }
    }
}
```

----------FIN----------

9.3.3.2 View Model que se encarga de administrar todas las view model y que contiene el patrón singleton para asegurar solo una instancia de la view model correspondiente y en la que se pueden crear variables globales de toda la solución

----------INICIO----------

```
namespace USTAPG.ViewModels
{
    using System;
    using System.Collections.Generic;
    using System.Text;
    using USTAPG.Models;
    using USTAPG.Services;

    public class MainViewModel
    {
        #region
        public List<Usuario> U_Usuario { get; set; }
        public DataBase DB { get; set; }
        public Usuario UserDB { get; set; }
        public List<measureTable> MeasureTable { get; set; }
        public List<InfoTable> InfoTable { get; set; }
        public MeterTable Metertable { get; set; }
        public string SMAC { get; set; }
        public string SUsuario { get; set; }
        public string SClave { get; set; }
        #endregion

        #region ViewModels
        public bool YaExiste { get; set; }
        public LoginViewModel Login { get; set; }
        public MacViewModel MacMainPage { get; set; }
        public MeterViewModel Meter { get; set; }
        public MACInTextViewModel MacTextPage { get; set; }
```

```
        #endregion

        #region Constructor
        public MainViewModel()
        {
            Instance = this;
            this.Login = new LoginViewModel();
        }
        #endregion

        #region Singleton
        public static MainViewModel Instance;

        public static MainViewModel GetIntance()
        {
            if (Instance == null) return new MainViewModel();
            return Instance;
        }
        #endregion
    }
}
```

----------FIN----------

9.3.3.3 View model que tiene la lógica de la página de login

----------INICIO----------

```
namespace USTAPG.ViewModels
{
    using GalaSoft.MvvmLight.Command;
    using System.Windows.Input;
    using USTAPG.Services;
    using USTAPG.Views;
    using Xamarin.Forms;
    using Plugin.Connectivity;
    using System;
    using System.IO;
    using USTAPG.Models;
    using System.Collections.Generic;
    using System.Threading.Tasks;

    public class LoginViewModel : BaseViewModel
```

```
{
    #region Atributos
    private string _correo;
    private string _clave;
    private bool _iniciado;
    private bool _habilitado;
    #endregion

    #region Propiedades
    public bool IsUserCreated { get; set; }
    public SFirebase Firebase { get; set; }
    public bool Habilitado
    {
        get { return this._habilitado; }
        set { SetValue(ref this._habilitado, value); }
    }
    public string Correo
    {
        get { return this._correo; }
        set { SetValue(ref this._correo, value); }
    }
    public string Clave
    {
        get { return this._clave; }
        set { SetValue(ref this._clave, value); }
    }
    public bool Recordar { get; set; }
    public bool Iniciado
    {
        get { return this._iniciado; }
        set { SetValue(ref this._iniciado, value); }
    }
    #endregion

    #region Comandos
    public ICommand EntrarCommand
    {
        get
        {
```

```
            return new RelayCommand(Entrar);
        }
    }
    #endregion

    #region Constructor
    public LoginViewModel()
    {
        //this.Correo = "mtovarson@gmail.com";
        //this.Clave = "Alfa199823";
        this.Habilitado = true;
        this.Recordar = true;
        //GenerarData();
        //Encriptacion Codificacion = new Encriptacion();
        //Codificacion.Encriptar("a4:cf:12:d9:3f:b7");

        ValidarUsuario();
    }
#endregion

#region Métodos
public async void ValidarUsuario()
{
    MainViewModel.GetIntance().DB = new
DataBase(Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.
LocalApplicationData), "full.db3"));
    var u = await GetUser();
    MainViewModel.GetIntance().U_Usuario = new List<Usuario>();
    if (u.Count > 0)
    {
        MainViewModel.GetIntance().YaExiste = true;
        MainViewModel.GetIntance().U_Usuario.Add(u[0]);
        this.Correo = MainViewModel.GetIntance().U_Usuario[0].User;
        this.Clave = MainViewModel.GetIntance().U_Usuario[0].Clave;
    }
    else
    {
        MainViewModel.GetIntance().YaExiste = false;
        this.Correo = string.Empty;
```

```
            this.Clave = string.Empty;
        }
    }

    private async void Entrar()
    {
        if (!CrossConnectivity.Current.IsConnected)
        {
            await Application.Current.MainPage.DisplayAlert(
                "Error de conexión",
                "Por favor habilita tu conexión a internet.",
                "Aceptar");
            return;
        }
        var inter = await CrossConnectivity.Current.IsRemoteReachable("google.com");
        if (!inter)
        {
            await Application.Current.MainPage.DisplayAlert(
                "Error de conexión",
                "Verifica tu conexión a internet.",
                "Aceptar");
            return;
        }
        if (string.IsNullOrEmpty(this.Correo))
        {
            await Application.Current.MainPage.DisplayAlert(
                "Error",
                "Ingresa un correo por favor",
                "Aceptar");
            return;
        }
        if (string.IsNullOrEmpty(this.Clave))
        {
            await Application.Current.MainPage.DisplayAlert(
                "Error",
                "Ingresa una clave por favor",
                "Aceptar");
            this.Clave = string.Empty;
```

```
            return;
        }
        this.Habilitado = false;
        this.Iniciado = true;

        Firebase = new SFirebase() { Email = this.Correo, Clave = this.Clave };
        var a = await Firebase.LoginWithEmail(false);
        //var a = await Firebase.GetMeasure("a4:cf:12:d9:3f:b7");
        if (string.IsNullOrEmpty(a))
        {
            await Application.Current.MainPage.DisplayAlert(
            "Error",
            "Datos incorrectos, por favor ingrese el correo y la contraseña correspondiente.",
            "Aceptar");
            this.Correo = string.Empty;
            this.Clave = string.Empty;
            this.Habilitado = true;
            this.Iniciado = false;
            return;
        }

        if (this.Recordar)
        {
            if (MainViewModel.GetIntance().U_Usuario.Count > 0)
            {
                MainViewModel.GetIntance().U_Usuario[0].Clave = this.Clave;
                MainViewModel.GetIntance().U_Usuario[0].User = this.Correo;
            }
            else
            {
                MainViewModel.GetIntance().U_Usuario.Add(new Usuario()
                {
                    User = this.Correo,
                    Clave = this.Clave
                });
            }
        }
        else
```

```
        {
            if (MainViewModel.GetIntance().U_Usuario.Count > 0)
            {
                MainViewModel.GetIntance().U_Usuario[0].Clave = string.Empty;
                MainViewModel.GetIntance().U_Usuario[0].User = string.Empty;
            }
            else
            {
                MainViewModel.GetIntance().U_Usuario.Add(new Usuario()
                {
                    User = string.Empty,
                    Clave = string.Empty
                });
            }
        }
        this.Habilitado = true;
        this.Iniciado = false;
        MainViewModel.GetIntance().SUsuario = this.Correo;
        MainViewModel.GetIntance().SClave = this.Clave;
        MainViewModel.GetIntance().MacMainPage = new
MacViewModel(this.Correo, this.Clave);
        await Application.Current.MainPage.Navigation.PushAsync(new MACPage());
        this.Correo = string.Empty;
        this.Clave = string.Empty;
    }

    public async Task<List<Usuario>> GetUser()
    {
        var a = await MainViewModel.GetIntance().DB.GetPeopleAsync<Usuario>();
        return a;
    }
    #endregion
}
}
```

----------FIN----------

9.3.3.4 View model que se encarga de la lógica de la view de MAC

----------INICIO----------

```
namespace USTAPG.ViewModels
{
```

```
using GalaSoft.MvvmLight.Command;
using System;
using System.Windows.Input;
using ZXing.Net.Mobile.Forms;
using Xamarin.Forms;
using USTAPG.Views;
using USTAPG.Services;
using System.Collections.Generic;
using USTAPG.Models;
using System.Linq;
using System.IO;
using System.Threading.Tasks;

public class MacViewModel:BaseViewModel
{
    #region Atributos
    private bool _iniciado;
    private bool _habilitado;
    private bool _habilitado1;
    #endregion

    #region Propiedaes

    public SFirebase Firebase { get; set; }
    public string Correo { get; set; }
    public string Clave { get; set;}
    public bool Iniciado
    {
        get { return this._iniciado; }
        set { SetValue(ref this._iniciado, value); }
    }
    public bool Habilitado
    {
        get { return this._habilitado; }
        set { SetValue(ref this._habilitado, value); }
    }
    public bool Habilitado1
    {
        get { return this._habilitado1; }
```

```
    set { SetValue(ref this._habilitado1, value); }
}
#endregion

#region Comandos
public ICommand MacQr
{
    get
    {
        return new RelayCommand(MacQrM);
    }
}
public ICommand MacText
{
    get
    {
        return new RelayCommand(MacTextM);
    }
}
#endregion

#region Constructor
public MacViewModel(string _correo, string _clave)
{
    this.Correo = _correo;
    this.Clave = _clave;
    this.Iniciado = false;
    ValidarpriVez();
}
#endregion

#region Metodos
public void Botones(bool _value)
{
    if (Device.RuntimePlatform == Device.UWP)
    {
        this.Habilitado = _value;
        this.Habilitado1 = false;
    }
```

```
            else
            {
                this.Habilitado = _value;
                this.Habilitado1 = _value;
            }
        }

        private void ValidarpriVez()
        {
            if (MainViewModel.GetIntance().YaExiste)
            {
                SiguientePaso(MainViewModel.GetIntance().U_Usuario[0].MAC, true);
            }
            else
            {
                Botones(true);
                return;
            }
        }

        private async void MacTextM()
        {
            MainViewModel.GetIntance().MacTextPage = new
MACInTextViewModel(this.Correo, this.Clave);
            await Application.Current.MainPage.Navigation.PushAsync(new
MACInTextPage());
        }

        private async void MacQrM()
        {
            try
            {
                var qr = new ZXing.Mobile.MobileBarcodeScanner();
                qr.TopText = "Escanea el código por favor";
                qr.BottomText = "Proyecto de Grado USTA 2022";
                var lectura = await qr.Scan();
                if (lectura != null) SiguientePaso(lectura.Text, false);
                else { }
            }
```

```
            catch (Exception e)
            {
                await Application.Current.MainPage.DisplayAlert(
                    "Error",
                    "Error escaneando el código, intente de nuevo.",
                    "Aceptar");
                this.Iniciado = false;
            }
            this.Iniciado = false;
        }

        public async void GuardarDB(bool f)
        {
            if (f)
            {
                await
MainViewModel.GetIntance().DB.UpdatePersonAsync(MainViewModel.GetIntance
().U_Usuario[0]);
            }
            else
            {
                await MainViewModel.GetIntance().DB.SavePersonAsync(new Usuario
                {
                    User = MainViewModel.GetIntance().SUsuario,
                    Clave = MainViewModel.GetIntance().SClave,
                    MAC = MainViewModel.GetIntance().SMAC
                });
            }
        }

        public async void SiguientePaso(string lectura, bool _first)
        {
            Botones(false);
            List<measureTable> Datos = new List<measureTable>();
            List<InfoTable> Info = new List<InfoTable>();
            MeterTable Meter = new MeterTable();
            Encriptacion Codificacion = new Encriptacion();
            //var _text = Codificacion.Encriptar(lectura.Text);
            string UN_MAC = "";
```

```
            if (_first) UN_MAC = lectura;
            else UN_MAC = Codificacion.DesEncriptar(lectura);
            this.Iniciado = true;
            try
            {
                Firebase = new SFirebase() { Email = this.Correo, Clave = this.Clave };
            }
            catch (System.Exception e)
            {
                await Application.Current.MainPage.DisplayAlert(
                    "Error",
                    "Hubo un problema al conectar la base de datos, revise su conexión a
internet.",
                    "Aceptar");
                Botones(true);
                this.Iniciado = false;
                return;
            }
            try
            {
                Datos = await Firebase.GetMeasure(UN_MAC);//"a4:cf:12:d9:3f:b7");
                Info = await Firebase.GetInfo(UN_MAC);
                Meter = await Firebase.GetMeter(UN_MAC);
                if (string.IsNullOrEmpty(Meter.Gateway))
                {
                    await Application.Current.MainPage.DisplayAlert(
                    "Medidor",
                    "No se encontró información del medidor, por favor contacte a
soporte.",
                    "Aceptar");
                    Botones(true);
                    this.Iniciado = false;
                    return;
                }
                if(!await VerificarUsuario(Firebase, UN_MAC))
                {
                    await Application.Current.MainPage.DisplayAlert(
                    "Sesion",
```

```
                "No tiene permitido ingresar a la información de este medidor.  Contacte
a soporte.",
                "Aceptar");
                Botones(true);
                this.Iniciado = false;
                return;
            }
        }
        catch (Exception)
        {
            await Application.Current.MainPage.DisplayAlert(
                "Error",
                "No se encontró el medidor, intente de nuevo.",
                "Aceptar");
            Botones(true);
            this.Iniciado = false;
            return;
        }

        MainViewModel.GetIntance().SMAC = UN_MAC;
        MainViewModel.GetIntance().MeasureTable = Datos;
        MainViewModel.GetIntance().InfoTable = Info;
        MainViewModel.GetIntance().Metertable = Meter;
        GuardarDB(MainViewModel.GetIntance().YaExiste);
        MainViewModel.GetIntance().Meter = new MeterViewModel();
        await                Application.Current.MainPage.Navigation.PushAsync(new
MenuTabbedPage());
        Botones(true);
        this.Iniciado = false;
    }

    public async Task<bool> VerificarUsuario(SFirebase _fb, string _medidor)
    {
        bool val = false;
        var usuarios = await _fb.GetSesion(_medidor);
        foreach (var u in usuarios) if (u.Email == _fb.Email) return true;
        return val;
    }
    #endregion
```

```
    }
}
```
----------FIN----------

9.3.3.5 View model que contiene la lógica de ingresar MAC manualmente

----------INICIO----------

```
namespace USTAPG.ViewModels
{
    using GalaSoft.MvvmLight.Command;
    using System.Windows.Input;
    using Xamarin.Forms;
    using System;
    using USTAPG.Views;
    using USTAPG.Services;
    using System.Collections.Generic;
    using USTAPG.Models;
    using System.Linq;
    using System.IO;
    using System.Threading.Tasks;

    public class MACInTextViewModel:BaseViewModel
    {
        #region Atributos
        public string mac;
        public bool _habilitado;
        private bool _iniciado;
        #endregion

        #region Propíedades
        public string Correo { get; set; }
        public string Clave { get; set; }
        public SFirebase Firebase { get; set; }
        public bool Iniciado
        {
            get { return this._iniciado; }
            set { SetValue(ref this._iniciado, value); }
        }
        public bool Habilitado
        {
            get { return this._habilitado; }
```

```
    set { SetValue(ref this._habilitado, value); }
}
public string MAC
{
    get { return this.mac; }
    set { SetValue(ref this.mac, value);}
}
#endregion

#region Comandos
public ICommand Hecho
{
    get
    {
        return new RelayCommand(HechoM);
    }
}
#endregion

#region Constructor
public MACInTextViewModel(string _correo, string _clave)
{
    this.Clave = _clave;
    this.Correo = _correo;
    this.MAC = string.Empty;
    this.Habilitado = true;
}
#endregion

#region Metodos
private async void HechoM()
{
    if (string.IsNullOrEmpty(MAC))
    {
        await Application.Current.MainPage.DisplayAlert(
              "Error",
              "Ingrese la dirección MAC del equipo.",
              "Aceptar");
        return;
```

```
            }
            MainViewModel.GetIntance().SMAC = this.MAC;
            SiguientePaso(this.MAC, true);
        }

        public async void SiguientePaso(string lectura, bool _first)
        {
            List<measureTable> Datos = new List<measureTable>();
            List<InfoTable> Info = new List<InfoTable>();
            MeterTable Meter = new MeterTable();
            Encriptacion Codificacion = new Encriptacion();
            //var _text = Codificacion.Encriptar(lectura.Text);
            string UN_MAC = lectura;
            this.Iniciado = true;
            try
            {
                Firebase = new SFirebase() { Email = this.Correo, Clave = this.Clave };
            }
            catch (System.Exception e)
            {
                await Application.Current.MainPage.DisplayAlert(
                    "Error",
                    "Hubo un problema al conectar la base de datos, revise su conexión a
internet.",
                    "Aceptar");
                this.Iniciado = false;
                return;
            }
            try
            {
                Datos = await Firebase.GetMeasure(UN_MAC);//"a4:cf:12:d9:3f:b7");
                Info = await Firebase.GetInfo(UN_MAC);
                Meter = await Firebase.GetMeter(UN_MAC);
                if (string.IsNullOrEmpty(Meter.Gateway))
                {
                    await Application.Current.MainPage.DisplayAlert(
                    "Medidor",
                    "No se encontró información del medidor, verifique la dirección
ingresada o contacte a soporte.",
```

```
                "Aceptar");
                this.Iniciado = false;
                return;
            }
            if (!await VerificarUsuario(Firebase, UN_MAC))
            {
                await Application.Current.MainPage.DisplayAlert(
                "Sesion",
                "No tiene permitido ingresar a la información de este medidor. Contacte
a soporte.",
                "Aceptar");
                this.Iniciado = false;
                return;
            }
        }
        catch (Exception)
        {
            await Application.Current.MainPage.DisplayAlert(
                "Error",
                "No se encontró el medidor, intente de nuevo.",
                "Aceptar");
            this.Iniciado = false;
            return;
        }

        MainViewModel.GetIntance().SMAC = UN_MAC;
        MainViewModel.GetIntance().MeasureTable = Datos;
        MainViewModel.GetIntance().InfoTable = Info;
        MainViewModel.GetIntance().Metertable = Meter;
        await MainViewModel.GetIntance().DB.SavePersonAsync(new Usuario
        {
            User = MainViewModel.GetIntance().SUsuario,
            Clave = MainViewModel.GetIntance().SClave,
            MAC = MainViewModel.GetIntance().SMAC
        });
        MainViewModel.GetIntance().Meter = new MeterViewModel();
        await                Application.Current.MainPage.Navigation.PushAsync(new
MenuTabbedPage());
        this.Iniciado = false;
```

```
        }

        public async Task<bool> VerificarUsuario(SFirebase _fb, string _medidor)
        {
            bool val = false;
            var usuarios = await _fb.GetSesion(_medidor);
            foreach (var u in usuarios) if (u.Email == _fb.Email) return true;
            return val;
        }
        #endregion
    }
}
```

----------FIN----------

9.3.3.6 View model que contiene la lógica de las paginas de los tres servicios y de la página de meter.

----------INICIO----------

```
namespace USTAPG.ViewModels
{
    using GalaSoft.MvvmLight.Command;
    using Microcharts;
    using SkiaSharp;
    using System;
    using System.Collections.Generic;
    using System.Globalization;
    using System.Linq;
    using System.Windows.Input;
    using USTAPG.Models;
    using Xamarin.Forms;

    public class MeterViewModel : BaseViewModel
    {
        #region Atributos
        public int height;
        public LineChart gra1;
        public LineChart gra2;
        public LineChart gra3;
        public bool dia;
        public bool hora;
        public bool mes;
        public string mac;
        public string estrato;
        public string estadoS1;
```

```
public string estadoS2;
public string estadoS3;
public string fechaFacS1;
public string fechaFacS2;
public string fechaFacS3;
public string tarifaS3;
public string tarifaS1;
public string tarifaS2;
public string aproximadoS1;
public string aproximadoS2;
public string aproximadoS3;
public string servicio1;
public string servicio2;
public string servicio3;
public string direccion;
#endregion

#region Propiedades
public int FontSize { get; set; }
public int Height
{
    get { return this.height; }
    set { SetValue(ref this.height, value); }
}
public string Servicio1
{
    get { return this.servicio1; }
    set { SetValue(ref this.servicio1, value); }
}
public string Servicio2
{
    get { return this.servicio2; }
    set { SetValue(ref this.servicio2, value); }
}
public string Servicio3
{
    get { return this.servicio3; }
    set { SetValue(ref this.servicio3, value); }
}
public string Direccion
{
    get { return this.direccion; }
    set { SetValue(ref this.direccion, value); }
}
public MeterTable MeterT { get; set; }
```

```
public string[] Meses { get; set; }
public List<Servicio> S1 { get; set; }
public List<Servicio> S2 { get; set; }
public List<Servicio> S3 { get; set; }
public string AcMes { get; set; }
public string AcAnio { get; set; }
public string AcDia { get; set; }
public List<measureTable> Measure { get; set; }
public List<InfoTable> Info { get; set; }
public List<float> Gra1Y { get; set; }
public List<float> Gra2Y { get; set; }
public List<float> Gra3Y { get; set; }
public List<string> Gra1X { get; set; }
public List<string> Gra2X { get; set; }
public List<string> Gra3X { get; set; }
public LineChart Gra1
{
    get { return this.gra1; }
    set { SetValue(ref this.gra1, value); }
}
public LineChart Gra2
{
    get { return this.gra2; }
    set { SetValue(ref this.gra2, value); }
}
public LineChart Gra3
{
    get { return this.gra3; }
    set { SetValue(ref this.gra3, value); }
}
public bool Dia
{
    get { return this.dia; }
    set { SetValue(ref this.dia, value); }
}
public bool Hora
{
    get { return this.hora; }
    set { SetValue(ref this.hora, value); }
}
public bool Mes
{
    get { return this.mes; }
    set { SetValue(ref this.mes, value); }
}
```

```
public string Estrato
{
    get { return this.estrato; }
    set { SetValue(ref this.estrato, value); }
}
public string FechaFacS1
{
    get { return this.fechaFacS1; }
    set { SetValue(ref this.fechaFacS1, value); }
}
public string FechaFacS2
{
    get { return this.fechaFacS2; }
    set { SetValue(ref this.fechaFacS2, value); }
}
public string FechaFacS3
{
    get { return this.fechaFacS3; }
    set { SetValue(ref this.fechaFacS3, value); }
}
public string EstadoS1
{
    get { return this.estadoS1; }
    set { SetValue(ref this.estadoS1, value); }
}
public string EstadoS2
{
    get { return this.estadoS2; }
    set { SetValue(ref this.estadoS2, value); }
}
public string EstadoS3
{
    get { return this.estadoS3; }
    set { SetValue(ref this.estadoS3, value); }
}
public string AproximadoS1
{
    get { return this.aproximadoS1; }
    set { SetValue(ref this.aproximadoS1, value); }
}
public string AproximadoS3
{
    get { return this.aproximadoS3; }
    set { SetValue(ref this.aproximadoS3, value); }
}
```

```
        public string AproximadoS2
        {
            get { return this.aproximadoS2; }
            set { SetValue(ref this.aproximadoS2, value); }
        }
        public string TarifaS1
        {
            get { return this.tarifaS1; }
            set { SetValue(ref this.tarifaS1, value); }
        }
        public string TarifaS2
        {
            get { return this.tarifaS2; }
            set { SetValue(ref this.tarifaS2, value); }
        }
        public string TarifaS3
        {
            get { return this.tarifaS3; }
            set { SetValue(ref this.tarifaS3, value); }
        }
        public string MAC
        {
            get { return this.mac; }
            set { SetValue(ref this.mac, value); }
        }
        #endregion

        #region Comandos
        public ICommand RadioButtonCommand
        {
            get
            {
                return new RelayCommand(RevisarRadioButton);
            }
        }
        #endregion

        #region Constructor
        public MeterViewModel()
        {
            this.MAC = MainViewModel.GetIntance().SMAC;
            this.Servicio1                                                    =
ObtenerEstado(MainViewModel.GetIntance().Metertable.Services.Service1);
            this.Servicio2                                                    =
ObtenerEstado(MainViewModel.GetIntance().Metertable.Services.Service2);
```

```
        this.Servicio3                                                    =
ObtenerEstado(MainViewModel.GetIntance().Metertable.Services.Service3);
        this.Direccion = MainViewModel.GetIntance().Metertable.Address;
        this.MeterT = MainViewModel.GetIntance().Metertable;
        this.Measure = MainViewModel.GetIntance().MeasureTable;
        this.Info = MainViewModel.GetIntance().InfoTable;
        this.Estrato = this.Info[0].StatusClass;
        this.AproximadoS1 = this.Info[0].Service1.AproxMonth.ToString();
        this.AproximadoS2 = this.Info[0].Service2.AproxMonth.ToString();
        this.AproximadoS3 = this.Info[0].Service3.AproxMonth.ToString();
        this.FechaFacS1 = this.Info[0].Service1.BillDate;
        this.FechaFacS2 = this.Info[0].Service2.BillDate;
        this.FechaFacS3 = this.Info[0].Service3.BillDate;
        this.TarifaS1 = this.Info[0].Service1.Fare.ToString();
        this.TarifaS2 = this.Info[0].Service2.Fare.ToString();
        this.TarifaS3 = this.Info[0].Service3.Fare.ToString();
        this.EstadoS1     =     ObtenerEstado(this.Measure[Measure.Count     -
1].Servicio1.Status);
        this.EstadoS2     =     ObtenerEstado(this.Measure[Measure.Count     -
1].Servicio2.Status);
        this.EstadoS3     =     ObtenerEstado(this.Measure[Measure.Count     -
1].Servicio3.Status);
        this.S1 = new List<Servicio>();
        this.S2 = new List<Servicio>();
        this.S3 = new List<Servicio>();
        this.Gra1X = new List<string>();
        this.Gra2X = new List<string>();
        this.Gra3X = new List<string>();
        this.Gra1Y = new List<float>();
        this.Gra2Y = new List<float>();
        this.Gra3Y = new List<float>();
        DateTime Today = DateTime.Today;
        var split = Today.ToString("d").Split('/');
        Meses = new string[12] {"Ene", "Feb", "Mar", "Abr", "May", "Jun", "Jul", "Ago",
"Sep",
        "Oct", "Nov", "Dic"};
        try
        {
            this.AcDia = split[1];
            this.AcMes = Meses[Convert.ToInt32(split[0]) - 1];
        }
        catch (Exception)
        {
            this.Height = 400;
            this.AcDia = split[0];
```

```
    this.AcMes = Meses[Convert.ToInt32(split[1]) - 1];
}
AcAnio = split[2];
switch (Device.RuntimePlatform)
{
    case Device.UWP:
        this.FontSize = 19;
        this.Height = 400;
        break;
    default:
        this.FontSize = 30;
        this.Height = 220;
        break;
}
foreach (var M in this.Measure)
{
    S1.Add(M.Servicio1);
    S2.Add(M.Servicio2);
    S3.Add(M.Servicio3);
}
this.Hora = true;
this.Dia = false;
this.Mes = false;
var GHora1 = OrganizarHora(1);
var GHora2 = OrganizarHora(2);
var GHora3 = OrganizarHora(3);
GHora1.Reverse();
GHora3.Reverse();
GHora2.Reverse();
this.Gra1 = new LineChart()
{
    Entries = GHora1,
    LabelColor = SKColor.Parse("#FFFF"),
    LabelTextSize = this.FontSize,
    BackgroundColor = SKColor.Parse("#1D56FF")
};
this.Gra2 = new LineChart()
{
    Entries = GHora2,
    LabelColor = SKColor.Parse("#FFFF"),
    LabelTextSize = this.FontSize,
    BackgroundColor = SKColor.Parse("#1D56FF")
};
this.Gra3 = new LineChart()
{
```

```
            Entries = GHora3,
            LabelColor = SKColor.Parse("#FFFF"),
            LabelTextSize = this.FontSize,
            BackgroundColor = SKColor.Parse("#1D56FF")
        };
    }
    #endregion

    #region Metodos
    public List<ChartEntry> OrganizarHora(int _NumService)
    {
        //Necesito los tres: Dia, mes y año
        List<Servicio> _temp = new List<Servicio>();
        if (_NumService == 1) _temp = S1;
        else if (_NumService == 2) _temp = S2;
        else _temp = S3;
        string AcDiaTemp = "";
        var f = Convert.ToInt32(AcDia) < 10 ? AcDiaTemp += " " + AcDia :
AcDiaTemp += AcDia;
        var _diaMes = _temp.Where(m => m.Date.ToLower().Contains(AcDiaTemp
+ "/" + AcMes.ToLower() + "/" + AcAnio)).ToList();
        List<ChartEntry> Gra1entry = new List<ChartEntry>();
        int _hours = 23;
        int _count = 0;
        float _sum = 0;
        float _average = 0;
        for(int i = _diaMes.Count; i > 0; i--)
        {
            var li = _diaMes.Where(h => h.Date.Contains(_hours + ":")).ToList();
            if (li.Count != 0)
            {
                foreach (var it in li)
                {
                    _sum += it.Value;
                    _count++;
                }
                try
                {
                    _average = _sum;
                    string GraLabel = "";
                    if (_NumService == 1) GraLabel = (_average / 1000).ToString("N1")
+ " kWh";
                    else if (_NumService == 2) GraLabel = Convert.ToInt32(_average) +
" L";
                    else GraLabel = Convert.ToInt32(_average) + " L";
```

```
                    Gra1entry.Add(new ChartEntry(Convert.ToInt32(_average))
                    {
                        Label = _hours.ToString() + ":00",
                        ValueLabel = GraLabel,
                        Color = SKColor.Parse("#FFFFFF"),
                        ValueLabelColor = SKColor.Parse("#FFFFFF")
                    });
                    _diaMes.RemoveAll(a => a.Date.Contains(_hours + ":"));
                }
                catch (Exception)
                { }
                _hours--;
                _sum = 0;
                _count = 0;
                _average = 0;
            }
        }
        return Gra1entry;
    }

    public List<ChartEntry> OrganizarDia(int _NumService)
    {
        //Necesito dos: mes y año
        List<Servicio> _temp = new List<Servicio>();
        if (_NumService == 1) _temp = S1;
        else if (_NumService == 2) _temp = S2;
        else _temp = S3;
        List<ChartEntry> Gra1entry = new List<ChartEntry>();
        var Mes = _temp.Where(m => m.Date.ToLower().Contains(AcMes.ToLower() + "/" + AcAnio)).ToList();
        var Dias = Mes.GroupBy(g => {
            int In = g.Date.IndexOf(" ");
            return g.Date.Substring(In);
        }).ToList();
        int _count = 0;
        float _sum = 0;
        float _average = 0;
        foreach (var D in Dias)
        {
            foreach (var H in D)
            {
                _sum += H.Value;
                _count++;
            }
            try
```

```
            {
                _average = _sum;
                string GraLabel = "";
                if (_NumService == 1) GraLabel = (_average / 1000).ToString("N1") + "
kWh";
                else if (_NumService == 2) GraLabel = Convert.ToInt32(_average) + "
L";
                else GraLabel = Convert.ToInt32(_average) + " L";
                Gra1entry.Add(new ChartEntry(Convert.ToInt32(_average))
                {
                    Label = D.Key.Remove(D.Key.Length - 5, 5),
                    ValueLabel = GraLabel,
                    Color = SKColor.Parse("#FFFFFF"),
                    ValueLabelColor = SKColor.Parse("#FFFFFF")
                });
            }
            catch (Exception)
            { }
            _sum = 0;
            _count = 0;
            _average = 0;
        }
        return Gra1entry;
    }

    public List<ChartEntry> OrganizarMes(int _NumService)
    {
        //Necesito solo el año
        List<Servicio> _temp = new List<Servicio>();
        if (_NumService == 1) _temp = S1;
        else if (_NumService == 2) _temp = S2;
        else _temp = S3;
        var _meses = _temp.Where(m =>
m.Date.ToLower().Contains(AcAnio)).ToList();
        List<ChartEntry> Gra1entry = new List<ChartEntry>();
        int _count = 0;
        float _sum = 0;
        float _average = 0;
        foreach (var M in Meses)
        {
            var Mes = _meses.Where(h => h.Date.Contains(M)).ToList();
            foreach (var MM in Mes)
            {
                _sum += MM.Value;
                _count++;
```

```
            }
            try
            {
                _average = _sum;
                string GraLabel = "";
                if (_NumService == 1) GraLabel = (_average / 1000).ToString("N1") + "
kWh";
                else if (_NumService == 2) GraLabel = Convert.ToInt32(_average) + "
L";
                else GraLabel = Convert.ToInt32(_average) + " L";
                Gra1entry.Add(new ChartEntry(Convert.ToInt32(_average))
                {
                    Label = M,
                    ValueLabel = GraLabel,
                    Color = SKColor.Parse("#FFFFFF"),
                    ValueLabelColor = SKColor.Parse("#FFFFFF")
                });
            }
            catch (Exception)
            { }
            _sum = 0;
            _count = 0;
            _average = 0;
        }
        return Gra1entry;
    }

    private void RevisarRadioButton()
    {
        if (this.Mes)
        {
            var GMes1 = OrganizarMes(1).ToList();
            var GMes2 = OrganizarMes(2).ToList();
            var GMes3 = OrganizarMes(3).ToList();
            this.Gra1 = new LineChart() { Entries = GMes1,
                LabelColor = SKColor.Parse("#FFFF"),
                LabelTextSize = this.FontSize,
                ValueLabelOrientation = Orientation.Vertical,
                BackgroundColor = SKColor.Parse("#1D56FF")};
            this.Gra2 = new LineChart() {
                Entries = GMes2,
                LabelColor = SKColor.Parse("#FFFF"),
                LabelTextSize = this.FontSize,
                ValueLabelOrientation = Orientation.Vertical,
                BackgroundColor = SKColor.Parse("#1D56FF")
```

```
    };
    this.Gra3 = new LineChart() {
        Entries = GMes3,
        LabelColor = SKColor.Parse("#FFFF"),
        LabelTextSize = this.FontSize,
        ValueLabelOrientation = Orientation.Vertical,
        BackgroundColor = SKColor.Parse("#1D56FF")
    };
}
else if (this.Hora)
{
    //var GHora1 = OrganizarHora(1).OrderBy(o => o.Label).ToList();
    //var GHora2 = OrganizarHora(2).OrderBy(o => o.Label).ToList();
    //var GHora3 = OrganizarHora(3).OrderBy(o => o.Label).ToList();
    var GHora1 = OrganizarHora(1);
    var GHora2 = OrganizarHora(2);
    var GHora3 = OrganizarHora(3);
    GHora1.Reverse();
    GHora3.Reverse();
    GHora2.Reverse();
    this.Gra1 = new LineChart() { Entries = GHora1,
        LabelColor = SKColor.Parse("#FFFF"),
        LabelTextSize = this.FontSize,
        ValueLabelOrientation = Orientation.Vertical,
        BackgroundColor = SKColor.Parse("#1D56FF")
    };
    this.Gra2 = new LineChart() { Entries = GHora2,
        LabelColor = SKColor.Parse("#FFFF"),
        LabelTextSize = this.FontSize,
        ValueLabelOrientation = Orientation.Vertical,
        BackgroundColor = SKColor.Parse("#1D56FF")
    };
    this.Gra3 = new LineChart() { Entries = GHora3,
        LabelColor = SKColor.Parse("#FFFF"),
        LabelTextSize = this.FontSize,
        ValueLabelOrientation = Orientation.Vertical,
        BackgroundColor = SKColor.Parse("#1D56FF")
    };
}
else if(this.Dia)
{
    var GDia1 = OrganizarDia(1).ToList();
    var GDia2 = OrganizarDia(2).ToList();
    var GDia3 = OrganizarDia(3).ToList();
    this.Gra1 = new LineChart() { Entries = GDia1,
```

```
                LabelColor = SKColor.Parse("#FFFF"),
                LabelTextSize = this.FontSize,
                ValueLabelOrientation = Orientation.Vertical,
                BackgroundColor = SKColor.Parse("#1D56FF")
            };
            this.Gra2 = new LineChart() { Entries = GDia2,
                LabelColor = SKColor.Parse("#FFFF"),
                LabelTextSize = this.FontSize,
                ValueLabelOrientation = Orientation.Vertical,
                BackgroundColor = SKColor.Parse("#1D56FF")
            };
            this.Gra3 = new LineChart() { Entries = GDia3,
                LabelColor = SKColor.Parse("#FFFF"),
                LabelTextSize = this.FontSize,
                ValueLabelOrientation = Orientation.Vertical,
                BackgroundColor = SKColor.Parse("#1D56FF")
            };
        }
    }

    public string ObtenerEstado(int _status)
    {
        if (_status == 1) return "Activo";
        else if (_status == 0) return "Inactivo";
        else if (_status == 2) return "Activo (P)";
        else if (_status == 3) return "Activo (P)";
        else if (_status == 4) return "Activo (P)";
        else if (_status == 5) return "Activo (P)";
        else if (_status == 6) return "Activo (P)";
        else if (_status == 7) return "Activo (P)";
        else return "Activo (P)";
    }
    #endregion
  }
}
```

----------FIN----------

*NOTA: Normalmente en un patrón MVVM cada View tiene su ViewModel correspondiente, sin embargo, cuando se tiene una página tabulada se recomienda que la lógica de todas las paginas que contine la tabpage se encuentre solo en una ViewModel por facilidad al momento de programar.

9.4 ANEXO D

CÓDIGOS QR

En el presente anexo se muestran los códigos generados para cada módulo de comunicación de los medidores

9.4.1 Código QR de medidor con MAC: **a4:cf:12:d9::3f:b7**

9.4.2 Código QR de medidor con MAC: **a4:cf:12:d9:3e:88**

9.4.3 Código QR de medidor con MAC: **a4:cf:12:d9:3f:1b**

9.5 ANEXO E

CÓDIGO SISTEMA INTELIGENTE

El siguiente código se encuentra publico en el siguiente enlace: https://colab.research.google.com/drive/1q50o-XlnTCNHQ5SsozcibPVeVmHNPkD4?usp=sharing

A continuación, se muestra el código implementado y su salida

9.5.1 Importar librerias

```
"""
# Commented out IPython magic to ensure Python compatibility.
import pandas as pd
import numpy as np
from sklearn import linear_model
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sb
# %matplotlib inline
```

9.5.2 Importar Data

```
dataframe = pd.read_csv("Datos.csv")
dataframe.head()
"""
```

	Hour_1	Value_1	OUT
0	0	23.666667	1
1	0	22.000000	1
2	0	22.333333	1
3	0	29.000000	1
4	0	27.333333	1

```
"""
```

9.5.3 Información básica de la data

```
dataframe.describe()
"""

Hour_1          Value_1         OUT
count  12960.000000         12960.000000         12960.000000
mean   11.500000     76.103549     2.000000
std      6.922454      66.887925     0.816528
min      0.000000      3.250000      1.000000
25%    5.750000      13.650000     1.000000
50%    11.500000     63.858333     2.000000
75%    17.250000     128.266667   3.000000
max    23.000000     246.050000   3.000000
"""
```

9.5.4 Creacion modelo

```
X = np.array(dataframe.drop(columns=['OUT']))
y = np.array(dataframe['OUT'])
```

9.5.5 Ajuste de modelo

```
model = linear_model.LogisticRegression()
model.fit(X,y)
```

9.5.6 Score

```
model.score(X,y)
"""
0.9218364197530864
"""
```

9.5.7 Validación

```
Separar data testing del data traning
"""

validation_size = 0.20
seed = 7
```

```
X_train, X_validation, Y_train, Y_validation = model_selection.train_test_split(X, y,
test_size=validation_size, random_state=seed)

name='Logistic Regression'
kfold = model_selection.KFold(n_splits=10, random_state=seed,shuffle=True)
cv_results = model_selection.cross_val_score(model, X_train, Y_train, cv=kfold,
scoring='accuracy')
msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
print(msg)
"""Logistic Regression: 0.922260 (0.006536)"""

predictions = model.predict(X_validation)
print(accuracy_score(Y_validation, predictions))
"""0.9197530864197531"""
```

9.5.8 Matriz de confusión

```
print(confusion_matrix(Y_validation, predictions))
"""
[[747  96   0]
 [112 799   0]
 [  0   0 838]]
"""
```

9.5.9 Reporte de clasificación

```
print(classification_report(Y_validation, predictions))
"""
  precision    recall  f1-score   support

           1       0.87      0.89      0.88       843
           2       0.89      0.88      0.88       911
           3       1.00      1.00      1.00       838

    accuracy                           0.92      2592
   macro avg       0.92      0.92      0.92      2592
weighted avg       0.92      0.92      0.92      2592
"""
```

9.5.10 Entrenando solo con data de entrenamiento

```
X_new = pd.DataFrame({'Hour_1': [4],'Value_1': [300]})
model.predict(X_new)

for n in range(810,820):
  Xnew = pd.DataFrame({'Hour_1': [X_validation[n][0]],'Value_1': [X_validation[n][1]]})
  numa = (n-810) + 1;
  print(str(numa) + " -> Hora: " + str(X_validation[n][0]) + "  Valor: " + str(X_validation[n][1]))
  print("  Grupo seleccionado " + str(model.predict(Xnew)))
  print("  Grupo real  [" + str(Y_validation[n]) + "]")
```

9.5.11 Resultados

"""

```
1 -> Hora: 12.0  Valor: 151.7
    Grupo seleccionado [2]
    Grupo real  [2]
2 -> Hora: 12.0  Valor: 11.9
    Grupo seleccionado [3]
    Grupo real  [3]
3 -> Hora: 10.0  Valor: 11.85
    Grupo seleccionado [3]
    Grupo real  [3]
4 -> Hora: 18.0  Valor: 143.6833333
    Grupo seleccionado [2]
    Grupo real  [2]
5 -> Hora: 8.0  Valor: 164.0333333
    Grupo seleccionado [2]
    Grupo real  [2]
6 -> Hora: 19.0  Valor: 19.45000001
    Grupo seleccionado [3]
    Grupo real  [3]
7 -> Hora: 23.0  Valor: 217.0666666
    Grupo seleccionado [2]
    Grupo real  [2]
8 -> Hora: 20.0  Valor: 224.4666666
    Grupo seleccionado [2]
    Grupo real  [2]
```

```
9 -> Hora: 3.0  Valor: 57.35
    Grupo seleccionado [2]
    Grupo real         [2]
10 -> Hora: 6.0  Valor: 81.0
    Grupo seleccionado [2]
    Grupo real         [1]
"""
```

9.5.12 Conexión Firebase

```
!pip install firebase_admin
import firebase_admin

cred_obj = firebase_admin.credentials.Certificate('/content/ClavePrivada.json')
default_app = firebase_admin.initialize_app(cred_obj, {
        'databaseURL':"https://ustaprograd o-default-rtdb.firebaseio.com/"
        })
```

9.5.13 Prueba Firebase

```
from firebase_admin import db
Hor = str(8)
Medidor1 = "a4:cf:12:d9:3e:1b"
Medidor2 = "a4:cf:12:d9:3e:88"
Medidor3 = "a4:cf:12:d9:3f:b7"

refMedida1 = db.reference("/measureTable/" + Medidor1 + "/MaymEd1D41" + Hor +
"4520mEd1D4/Servicio1")
Medida1 = refMedida1.get()
Valor1 = Medida1["Value"]
Hora1 = Medida1["Date"].split(sep=':')
refMedida2 = db.reference("/measureTable/" + Medidor2 + "/MaymEd1D41" + Hor +
"4520mEd1D4/Servicio1")
Medida2 = refMedida2.get()
Valor2 = Medida2["Value"]
Hora2 = Medida2["Date"].split(sep=':')
refMedida3 = db.reference("/measureTable/" + Medidor3 + "/MaymEd1D41" + Hor +
"4520mEd1D4/Servicio1")
Medida3 = refMedida3.get()
```

```
Valor3 = Medida3["Value"]
Hora3 = Medida3["Date"].split(sep=':')

Pr1 = pd.DataFrame({'Hour_1': [Hora1[0]],'Value_1': [Valor1]})
ResulatadoPre1 = model.predict(Pr1)
ref1 = db.reference("/InfoTable/" + Medidor1 + "/Val/Service1")
ref1.update({"AproxMonth":ResulatadoPre1.item()})
Pr2 = pd.DataFrame({'Hour_1': [Hora2[0]],'Value_1': [Valor2]})
ResulatadoPre2 = model.predict(Pr2)
ref2 = db.reference("/InfoTable/" + Medidor2 + "/Val/Service1")
ref2.update({"AproxMonth":ResulatadoPre2.item()})
Pr3 = pd.DataFrame({'Hour_1': [Hora3[0]],'Value_1': [Valor3]})
ResulatadoPre3 = model.predict(Pr3)
ref3 = db.reference("/InfoTable/" + Medidor3 + "/Val/Service1")
ref3.update({"AproxMonth":ResulatadoPre3.item()})

print("Medidor: " + Medidor1)
print("Prediction: " + str(model.predict(Pr1)))
print("Hour: " + str(Hora1[0]) + "    Value: " + str(Valor1) )
print("\n\n\n")
print("Medidor: " + Medidor2)
print("Prediction: " + str(model.predict(Pr2)))
print("Hour: " + str(Hora2[0]) + "    Value: " + str(Valor2) )
print("\n\n\n")
print("Medidor: " + Medidor3)
print("Prediction: " + str(model.predict(Pr3)))
print("Hour: " + str(Hora3[0]) + "    Value: " + str(Valor3) )
print("\n\n\n")

Medidor: a4:cf:12:d9:3e:1b
Prediction: [1]
Hour: 18    Value: 82.3333333333333
```

9.5.14 Resultado

```
Medidor: a4:cf:12:d9:3e:88
Prediction: [1]
Hour: 18    Value: 81
```

Medidor: a4:cf:12:d9:3f:b7
Prediction: [1]
Hour: 18 Value: 901