

**GIEMA-IoT: Herramienta móvil para la gestión de información generada por estaciones meteorológicas autónomas en un modelo IoT**

**Pérez Martínez Jhon Alexander**

**Universidad de San Buenaventura, Sede Bogotá.  
Facultad de Ingeniería.  
Programas de Ingeniería de Sistemas  
Bogotá, Colombia  
2020**

**GIEMA-IoT: Herramienta móvil para la gestión de información generada por estaciones meteorológicas autónomas en un modelo IoT**

**Pérez Martínez Jhon Alexander**

**Ing. Sánchez Martín Andrés Armando**

**Universidad de San Buenaventura, Sede Bogotá.  
Facultad de Ingeniería.  
Programa de Ingeniería de Sistemas  
Bogotá, Colombia  
2020**

## **AGRADECIMIENTOS**

Por medio de este espacio quiero agradecer primeramente a mis padres por apoyarme en cada paso en mi carrera, seguido de la universidad San Buenaventura por abrirme sus puertas y formarme como profesional y por último al Ingeniero Andrés Sánchez por su apoyo y enseñanza en este proyecto, muchas gracias a todos.

## **Tabla De Contenido**

INTRODUCCION	8
Capítulo 1	10
Antecedentes	10
Planteamiento del problema	13
Justificación y pregunta de Investigación	15
Objetivo General	16
Objetivos Específicos	16
Alcances y Limitaciones	17
Marco Conceptual	17
Metodología	21
Capítulo 2	24
2.1 Modelo IoT	24
2.1.1 Evaluación De Modelos Existentes	25
2.1.2 Modelo GIEMA IoT Propuesto	27
2.2 Caso De Estudio	28
2.2.1 Requerimientos	29
2.2.2 Casos De Uso	34
2.3 Diseño GIEMA	35
2.3.1 Arquitectura	35
2.3.2 Modelo De Datos	39
2.3.3 Modelos UML	41
2.3.4 Interfaz Gráfica De Usuario	43
2.4 Desarrollo	45
2.4.1 Tecnologías Utilizadas	45
2.4.2 Configuración De Ambientes	47
2.4.3 Estándares Utilizados	50
2.4.4 Productos Resultantes	54
2.5 Pruebas	54
2.5.1. Pruebas Funcional E Integración	55
2.5.2 Pruebas De Aceptación GIEMA	59
Capítulo 3	63

	5
3.1 Resultado De Las Pruebas	63
3.2 Análisis De Resultados	68
CONCLUSIONES	71
RECOMENDACIONES	73
REFERENCIA	75
Anexos	77

**Tabla De Tablas**

Tabla 1 - Comparativa de modelos IoT.....	26
Tabla 2 - Capaz del Modelo GIEMA.....	27
Tabla 3 - Requerimientos funcionales APP.....	29
Tabla 4 - Requerimientos No Funcionales App .....	31
Tabla 5 - Requerimientos Funcionales WebService.....	33
Tabla 6 - Casos de uso GIEMA.....	34
Tabla 7 - Tecnologías de desarrollo.....	46
Tabla 8 - Casos de prueba .....	57
Tabla 9 - Criterio de Evaluación TAM Diseño .....	60
Tabla 10 - Criterio de Evaluación TAM Usabilidad.....	61
Tabla 11 - Criterio de Evaluación TAM Funcionalidad .....	62
Tabla 12 - Calificación, criterio de evaluación TAM .....	62
Tabla 13 - Diseño criterio TAM .....	64
Tabla 14 - Usabilidad Criterio TAM.....	66
Tabla 15 - Funcionalidad Criterio TAM .....	67
Tabla 16 - Análisis de resultados TAM .....	69

Figura 1 – Diagrama Metodológico .....	21
Figura 2 - Cronograma de Actividades .....	23
Figura 3 - Arquitectura GIEMA .....	36
Figura 4 - Componente Nodo GIEMA.....	36
Figura 5 - Componente Gateway GIEMA .....	37
Figura 6 – WebService y Aplicación GIEMA.....	38
Figura 7 - Server GIEMA.....	39
Figura 8 - Modelo de datos.....	41
Figura 9 - Diagrama de componentes .....	42
Figura 10 - Diagrama de casos de Uso APP.....	42
Figura 11 – Diagrama de casos de USO GIEMA .....	43
Figura 12 - Diagrama de Navegación .....	44
Figura 13 - Mockups Grafica y Generación de datos .....	44
Figura 14 - IDE AndroidStudio .....	48
Figura 15 - Configuración de permisos de RED.....	48
Figura 16 - Configuración de dependencias AndroidStudio .....	49
Figura 17 – Repositorios de propiedades AndroidStudio .....	49
Figura 18 – JSONarray Estructura del mensaje GIEMA .....	50
Figura 19 - Servicio Registro.....	51
Figura 20 - Ejemplo Consumo de servicio .....	53
Figura 21 – Datos generados a partir de la emulación .....	55
Figura 22 - Generador datos Mockaroo .....	56

## INTRODUCCION

El presente proyecto se enfocará en la gestión de la información de datos generados por medio de una herramienta de generación de datos en bloque dando como resultado datos meteorológicos procesados para que estén al alcance de usuario por medio de una aplicación. Se emulará el proceso final que realiza una estación meteorológica mediante un generador de datos ya que no se cuenta con la parte física, las variables se categorizan por medio de sensores que se encargarán de medir parámetros ambientales del aire, los cuales se escogerán para el presente proyecto solo cuatro: humedad, temperatura, monóxido de carbono y material particulado, siendo los dos últimos los de que tendrán mayor enfoque debido a su gran importancia en la relación calidad del aire y salud.

El proyecto denominado “Gestión de la Información en Estaciones Meteorológicas Autónomas”, o por sus siglas GIEMA-IoT, nace de la importancia de destacar la calidad con la cual se manejan los datos obtenidos del monitoreo de la calidad del aire en estaciones meteorológicas, con el agregado de la tecnología de vanguardia como es el internet de las cosas (IoT). A través del desarrollo de una aplicación móvil se buscan alternativas y soluciones que atribuyan a prevenir y reducir los altos índices de contaminación en la calidad del aire por medio de otorgar la información de lo que sucede en el ambiente de manera rápida y eficiente.

Hoy en día existen estaciones climáticas que se ubican en puntos estratégicos para la medición de variables meteorológicas importantes en un sector. En ese orden de ideas, la propuesta que se plantea es generar datos meteorológicos por medio de herramientas de generación de datos, con el fin de desarrollar una aplicación móvil en Android para la notificación de ciertas variables meteorológicas (Monóxido de carbono, Temperatura, Humedad, Material particulado) para que puedan ser monitoreadas y el usuario pueda determinar donde se considera nocivo para el ser humano por medio de la visualización de estos datos meteorológicos, y así posteriormente notificar mediante una gráfica, para que los usuarios de la aplicación se mantengan informados sobre el estado de la calidad del aire, que en este caso esa información

se generara por medio de herramientas de generación de datos, ya que no se cuenta con una parte física, pero su arquitectura será diseñada para conectarse a una estación si así se desea.

El presente documento está compuesto por tres capítulos en el primero se encontrara todo lo relacionado al planteamiento del problema objetivos, limitaciones, explicación de antecedentes y a donde quiere ir GIEMA, una vez concluido ese capítulo se entrara en el capítulo 2 en el cual se explica todo el proceso de análisis, evaluación, desarrollo, diseño de todos componentes y concluirá con una serie de pruebas aplicadas al producto final, lo que dará inicio al capítulo tres centrado en el análisis de resultados de estas pruebas para determinar el cumplimiento funcionalidad a nivel técnica cerrando con unas conclusiones para el cierre de este proyecto que inicia a continuación.

## Capítulo 1

En esta primera sección del documento se pondrá en evidencia temas como los antecedentes objetivos y metas planteadas para GIEMA, se presentará la metodología escogida y de qué manera se ejecutó durante el desarrollo del presente proyecto, para finalizar los conceptos y planteamiento del problema, todo en un orden ya predefinido para que se realice de forma organizada y entendible, dicho se esto se da inicio a este capítulo.

### Antecedentes

El uso y manejo de datos, en este caso variables meteorológicas, lleva años creciendo en su aplicabilidad debido a los altos índices de contaminación en el mundo y el impacto del cambio climático actual, por lo cual ya existen varias investigaciones sobre el tema y además, se han implementado estaciones meteorológicas que permiten tomar muestras, con las cuales se logran identificar problemáticas y plantear soluciones, constatadas en varios informes de relevancia e impacto de la Organización Mundial de la Salud, centros de investigación, universidades y empresas del sector público y privado, los cuales han sentado una base para el análisis y tratamiento de datos tanto para otras investigaciones, como para implementaciones de sistemas que lidian la contaminación. Es importante destacar que algunos informes fueron tenidos en cuenta, tales como:

#### **Informe del estado de la calidad del aire en Colombia 2016**

En este informe, se hace un análisis de los datos recolectados por estaciones meteorológicas que se permiten y están estipuladas en la resolución 610 de 2010 del Ministerio de Ambiente y Desarrollo Sostenible, las cuales recaen a: Partículas Suspendidas Totales (PST), Material Particulado menor a 10 micras ( $PM_{10}$ ), Material Particulado menor a 2,5 micras ( $PM_{2.5}$ ), Ozono ( $O_3$ ), Dióxido de Nitrógeno ( $NO_2$ ), Monóxido de Carbono (CO) y Dióxido de Azufre ( $SO_2$ ).

Dicho análisis se hizo a través de Sistemas de Vigilancia de Calidad de Aire ubicados en zonas de relevancia ambiental para el país, consta de 159 estaciones de monitoreo, siendo 142 fijas y 17 indicativas. Este informe recomienda darle importancia a los contaminantes que presentaron mayores índices PM<sub>2.5</sub> y el PM<sub>10</sub>. En varias estaciones de monitoreo sus concentraciones excedieron los límites máximos permisibles establecidos en la Resolución 610 de 2010. Las concentraciones más altas de PM<sub>2.5</sub> fueron observadas en las estaciones del Valle de Aburrá y en la ciudad de Bogotá. En cuanto al número de días que exceden el límite diario, las estaciones de estas jurisdicciones también se destacan por tener los valores más altos. Las tendencias anuales indican la mejoría de las concentraciones de este contaminante en algunas estaciones de monitoreo.

Esto permite concluir, que, debido a los resultados, es necesario continuar ampliando el monitoreo de este contaminante en el país, enfocándose en los puntos donde se presentan mayores índices y empezar a hacer mediciones en sitios donde aún no se hacen. Por otro lado, se mostrarán importantes investigaciones y proyectos actuales relacionados con la recolección de datos y estaciones meteorológicas congruentes con las energías renovables. Además, se hablará sobre el uso que le dieron a la información y su aplicación (IDEAM, 2016).

### **Diseño e implementación de un sistema de información de la calidad del aire en la universidad de San Buenaventura**

Los objetivos, alcances y conclusiones de este proyecto son relevantes destacarlas ya que sobre las bases de estas se propone el proyecto de este documento. Los autores diseñaron tarjetas con módulos de comunicación y transmisión, junto con sensores de temperatura, humedad, material particulado y concentración de monóxido de carbono. Estas variables fueron escogidas debido a la calidad del aire, puesto que presentaban índices altos. Sin embargo, se limitó en recolectar las variables meteorológicas y almacenarlas para una posterior visualización, se abstuvo de realizar

algún tipo de análisis mayor sobre la incidencia de las variables obtenidas en la base de datos adquirida (ALVARADO MORENO, y otros, 2017).

### **Análisis del estado de la calidad del aire en Bogotá**

En este proyecto se realizó un análisis de los registros contenidos en la Red de Monitoreo de la Calidad del Aire de Bogotá (RMCAB) se diseñó una base de datos que fue creada y constituida para proveer la validación y el estudio de la información, la cual fue utilizada para evaluar de forma cuantitativa el estado de la calidad del aire de la ciudad. Entre los resultados se encuentran que las concentraciones atmosféricas de material particulado superan los valores establecidos por la reglamentación ambiental de la ciudad. En particular, en la zona industrial de Bogotá las violaciones a la norma de calidad del aire se presentan de manera permanente desde hace varios años. Entre los años 1998 y 2005, siete estaciones de la red han reportado medidas anuales que superan la norma anual para PM<sub>10</sub>.

Los resultados de este proyecto fundamentan la base del análisis de datos recolectados e indican que rumbo tomar en la investigación y documentación de la información para el modelamiento de datos y un mejor entendimiento del impacto ambiental que se pueda establecer con las variables meteorológicas medidas (Gaitán, Cancino, & Behrentz, 2007)

### **Monitoring of the Florida Everglades: A Pilot Project**

Aunque en este proyecto las variables medidas son la temperatura de la superficie y la calidad del agua, de igual manera manejan sistema de medición autónomo al utilizar celdas solares lo que representa un monitoreo ecológico a largo plazo de datos ambientales. Es importante destacar que los autores de este proyecto lo definen como “un nuevo sistema capaz de combinar control remoto de audio y monitoreo ambiental”.

Cabe destacar que el enfoque ambiental que le dan a este proyecto es diferente y no hay un profundo análisis de datos. Sin embargo, la autonomía que le dan al sistema puede fundamentar el

suministro de alimentación fotovoltaica para el proyecto a realizar ( Leider, Mann, & Dickinson, 2010).

### **Fuzzy Energy Management of Autonomous Weather Station**

Este artículo contribuye a la administración de energía y la implementación de la recolección de potencia en un microcontrolador de baja potencia integrado dentro de un dispositivo autónomo alimentado por energía solar con supercondensador que reduce significativamente las necesidades de mantenimiento.

La importancia de tener en cuenta esta investigación es por el uso de energía renovable, la cual está aumentando rápidamente hoy en día. Los autores concluyen que la energía solar es uno de los recursos más poderosos que se puede utilizar para proporcionar suministro de energía para dispositivos eléctricos remotos. En su caso, hacia una estación meteorológica en pro de hacerla autónoma y tomar datos en un tiempo constante, es decir, sin que se apague al tener una batería que perseverantemente lo alimente (Prauzek, Konecny, Hamel, & Hlavica, 2015).

### **Planteamiento del problema**

En la actualidad, la contaminación del aire representa un importante riesgo medioambiental para la salud. De acuerdo con los últimos informes de la Organización Mundial de la Salud (OMS), se diagnosticó que más de una cuarta parte de las defunciones de niños menores de cinco años son consecuencia de la contaminación ambiental. Cada año, las condiciones insalubres del entorno elevan sus índices, tales como la contaminación del aire en espacios cerrados y en el exterior, la exposición al humo de tabaco ajeno, la insalubridad del agua, la falta de saneamiento y la higiene inadecuada, causan la muerte de 1,7 millones de niños menores de cinco años (OMS, 2017).

Los datos recolectados en este tipo de informes sobre la contaminación ambiental varían según el objetivo que se tenga. Por ejemplo, el caso de las estaciones RMCAB de la Secretaría Distrital de Bogotá, que es la entidad con las mejores estaciones de monitoreo de variables meteorológicas en la capital, sube

los datos recolectados a una página web cada hora, y aunque esto no siempre se cumple, las mediciones hechas no son las más importantes y no hay un proceso de validación de la información recolectada que demuestre una calidad en los datos obtenidos.

En la primera parte del año 2019 los índices de la calidad del aire se han elevado en Bogotá, debido a diferentes factores los niveles de material particulado y otros, como el monóxido de carbono presentan niveles perjudiciales para la salud. Estos casos, se pudieron haber prevenido con un mejor manejo y gestión eficiente de la información suministrada por las estaciones meteorológicas alrededor de la ciudad, también haber planteado con anterioridad soluciones fundamentadas en prevenir y mejorar la calidad del aire, diferente a las soluciones temporáneas de último momento planteadas en bajar los índices de contaminación en el aire, como lo es el pico y placa ambiental. Frente a estos casos, el distrito de la ciudad afirmó: “Estas mediciones no son reales y están basadas en aplicaciones chinas que no pueden validar datos reales, porque no tienen acceso a la red de monitoreo que es la encargada de manejar los datos oficiales avalados por el Ministerio de Ambiente” (Gómez, 2019). Por lo tanto, es evidente que la información suministrada por las principales estaciones meteorológicas en Bogotá no es de gran validez y no permiten una gestión de la información que podría derivar en soluciones para mejorar la calidad del aire.

También, no solo es importante destacar que las estaciones meteorológicas no sean fiables en cuanto la calidad de los datos, sino que, con las tecnologías de hoy en día haya un escaso o nulo análisis y filtrado de datos, los dispositivos no son confiables y no se utilicen energías autónomas para alimentar dichas estaciones y no adapten nuevas tecnologías que podrían optimizar estos procesos como lo es el internet de las cosas y la gestión de estos datos meteorológicos por medio de una aplicación móvil.

Estos inconvenientes que se presentan en el monitoreo de la calidad del aire en Bogotá no son solo evidenciables en las soluciones planteadas por el distrito, profundizando aún más se encuentra que debido a este proceso incorrecto no se detecten enfermedades causadas por la contaminación ambiental o se encuentren demasiado tarde. También, a partir de una información deficiente sobre datos meteorológicos

se crean políticas ambientales incorrectas que no hacen más que no dejar avanzar en la optimización del monitoreo de la calidad del aire.

Teniendo en cuenta lo anterior, el proyecto planteado justifica la necesidad de desarrollar un aplicativo móvil capaces de gestionar y optimizar la información de las variables meteorológicas correctas en la ciudad de Bogotá, validando la información recolectada para que los habitantes puedan observar estos datos de una manera clara, brindándoles información rápida y precisa sobre la calidad del aire.

### **Justificación y pregunta de Investigación**

En los últimos años en la ciudad se presentan altos índices de contaminación, los cuales traen consecuencias en la salud de los ciudadanos, estos índices se han determinado por una previa monitorización por parte de diferentes entidades publicadas y privadas, siendo la más destacada la Red de Monitoreo de la Calidad del Aire de Bogotá – RMCAB, donde se recopilan datos sobre la concentración de parámetros contaminantes a través de unas estaciones para un posterior análisis.

Dichos análisis de la RMCAB y otros receptores de datos, en la mayoría de los casos se evidencian a través de informes donde comparan las mediciones con indicadores de contaminación y así determinan los altos índices.

Para monitorear la calidad del aire hay que tener en cuenta muchos factores, como lo son las plataformas en las que se hace, los dispositivos (en el mejor de los casos, de bajo costo), la inversión y otros. Dichos elementos, plantean un modo de como tener una eficiencia en la medición de los parámetros ambientales, teniendo en cuenta lo anterior, es importante destacar que, a un monitoreo más eficiente, son mayores las capacidades de conocer las consecuencias de una mala calidad del aire, puesto que se puede advertir y tomar decisiones en pro de un bienestar tanto ambiental como de salud.

Mediante este proyecto se plantea lograr cumplir unos objetivos plasmados mediante una solución apuntando a la optimización de un sistema de información de la calidad del aire, además de desarrollar una aplicación móvil para la notificación de ciertas variables meteorológicas, y así posteriormente comunicar

mediante la misma para que usuarios de la aplicación se mantengan informados sobre el estado de la calidad del aire al que la aplicación esté integrado en una estación meteorológica, con el propósito de que si por ejemplo una persona presenta alguna enfermedad donde estas variables en picos muy altos puedan afectar su salud, se pueda mantener al margen y estar enterado de la situación. También, esta herramienta basada en el internet de las cosas podría ayudar a detectar las zonas más nocivas para los estudiantes, y posteriormente ser un medio para generar una solución.

Por lo tanto, se plantea la siguiente pregunta problema: ¿Cómo gestionar de forma eficiente la información recopilada en estaciones meteorológicas para el beneficio común?

## **Objetivo General**

Desarrollar una herramienta móvil para la gestión de la información generada por estaciones meteorológicas autónomas.

## **Objetivos Específicos**

1. Identificar las variables generadas por una estación meteorológica para generar los datos a través de un modelo de simulación, definiendo la estructura del mensaje.
2. Diseñar la plataforma IoT y aplicación móvil para la notificación de los datos que se generan en las estaciones meteorológicas simuladas.
3. Construir un WebService y aplicación móvil que hacen parte de la plataforma IoT
4. Validar la aplicación móvil y el WebService mediante un protocolo de pruebas funcionales y de integración.

## **Alcances y Limitaciones**

Para determinar hasta qué punto puede llegar el presente proyecto bajo las condiciones plateadas y estudiando las limitaciones, se plantearon los siguientes alcances y limitaciones.

### **Alcances**

1. Solo se medirán las variables meteorológicas de los datos generados mediante las ecuaciones generadoras de datos meteorológicos: MP, Temperatura, CO y Humedad.
2. Se entregará la aplicación móvil y el WebService.
3. Se entregará la estructura de datos y los datos generados.

### **Limitaciones**

1. La cantidad de datos dependerá de la plataforma donde se generarán los datos.
2. Las tecnologías estarán alineadas según lo que se tenga disponible en el servidor para desplegar el proyecto.
3. La prueba estará limitada a datos generados en bloque y la aplicación no graficará en tiempo real.

## **Marco Conceptual**

En este proyecto es necesario definir y concretar algunos aspectos, los cuales serán usados a lo largo de la investigación y que son esenciales al momento de su desarrollo, la propuesta de GIEMA y todo lo que conlleva su arquitectura se puede desglosar en diferentes términos para un mejor entendimiento. En un principio, una estación meteorológica se define como un lugar específico o móvil donde se realizan observaciones y mediciones de múltiples parámetros meteorológicos (Molina, 2013) .

El funcionamiento de los paneles se basa en el efecto fotovoltaico, este efecto se produce cuando sobre materiales semiconductores convenientemente tratados incide la radiación solar produciéndose electricidad. “La radiación solar es una variable menos compleja de modelar en comparación con el viento. Existen modelos matemáticos de poca complejidad utilizados para estimar la radiación en un lugar

específico de acuerdo a las características y la ubicación del mismo, lo que facilita su análisis” (Vergara, Rey, Osma, & Ordóñez, 2014).

La finalidad de las estaciones meteorológicas autónomas no solo es aportar una utilidad al uso de energías renovables, sino también la posibilidad de facilitar su ubicación en puntos de difícil acceso para el ser humano y así lograr una mayor expansión y cobertura de la medición de las variables meteorológicas. Lo anterior, con el fin de establecer los niveles en tiempo real del ambiente donde se encuentre la estación.

Teniendo en cuenta los conceptos anteriores, se debe también aclarar que la estación meteorológica pueden tener diferentes tipos de sensores, para el presente proyecto se seleccionaron cuatro los cuales son: el monóxido de carbono, el cual medido a través del sensor MQ-7 que se encarga de la recolección de información de esta variable y se tomara este sensor como ejemplo para integrarlo y emularlo en el presente proyecto, “es considerado uno de los mayores contaminantes de la atmósfera terrestre. Sus principales fuentes productoras responsables de aproximadamente 80% de las emisiones, son los vehículos automotores que utilizan como combustible gasolina o diésel y los procesos industriales que utilizan compuestos del carbono. Esta sustancia es bien conocida por su toxicidad para el ser humano. Sus efectos tóxicos agudos incluida la muerte han sido estudiados ampliamente; sin embargo, sus potenciales efectos adversos a largo plazo son poco conocidos” (Téllez, Rodríguez, & Fajardo, 2006). El segundo sensor es de temperatura que se mide con un sensor regular el cual se hace una estimación en grados centígrados del punto donde se encuentre la estación. El tercer sensor es de humedad un sensor genérico el cual se utiliza para detectar, como su nombre lo indica, la humedad en el aire. Por último, el cuarto sensor y quizás el más importante es del material particulado (PPM), el cual se define como la concentración intensiva de partículas suspendidas en el aire, de un tamaño de 2,5 micras ( $PM_{2.5}$ ) o 10 micras ( $PM_{10}$ ). La medición de  $PM_{10}$  está creciendo en importancia por su relación con datos de mortalidad y morbilidad de la población. “La evidencia epidemiológica indica que un aumento en  $10 \mu\text{g}/\text{m}^3$  en  $PM_{10}$  está asociado a un aumento en alrededor del 1% en la mortalidad por todas las causas. A pesar de diversos cuestionamientos sobre las

evidencias epidemiológicas, esta asociación continúa siendo aceptada por la mayor parte de las autoridades ambientales y de salud alrededor del mundo” (Galvis, Y, & Rojas, 2006).

Después de lo explicado sobre el uso de los sensores, se debe dar claridad de cómo van a ser usados y en qué condiciones, en este caso mediante un modelo de simulación de datos meteorológicos, pero primero se deben aclarar los conceptos modelo y simulación “Modelo es una representación de un objeto, sistema o idea de forma diferente a la de identidad misma. Por lo general el modelo nos ayuda a entender y mejorar un sistema, todos los modelos se valen del proceso de simulación en el sentido que imitan la realidad” (Fullana & Urquía , 2009). Para el presente proyecto se emulará el proceso final que realiza la parte física de una estación meteorológica autónoma tomada del modelo IoT GIEMA por medio de datos generados en bloque.

El modelo IoT GIEMA es uno de los pilares más importantes del presente proyecto ya que sobre este modelo se realiza toda la arquitectura y el flujo de como funcionara y que lo compone. La tecnología IoT nace de la iniciativa de dar solución a dos problemas puntuales “buscar una forma estándar de acceso al medio y a los dispositivos, y buscar la forma de integrar los dispositivos a la Internet” (Rodríguez D. , 2013).

Ahora, teniendo definidos los datos que se usarán mediante un modelo de simulación, se debe resaltar que se han hecho análisis de variables como el material particulado, donde abordaron resultados como “las concentraciones atmosféricas de material particulado superan los valores establecidos por la reglamentación ambiental de la ciudad. En particular, en la zona industrial de Bogotá las violaciones a la norma de calidad del aire se presentan de manera permanente desde hace varios años. Entre los años 1998 y 2005, siete estaciones de la red han reportado medias anuales que superan la norma anual para PM<sub>10</sub>” (Gaitán, Cancino, & Behrentz, 2007).

Además, en ese proyecto se deben entender algunos conceptos que abarcan la programación y diseño de aplicaciones móviles, para satisfacer el objetivo de crear una aplicación móvil que cumpla con los diferentes parámetros dados, para esto se debe entender que es la computación ubicua o previsiva.

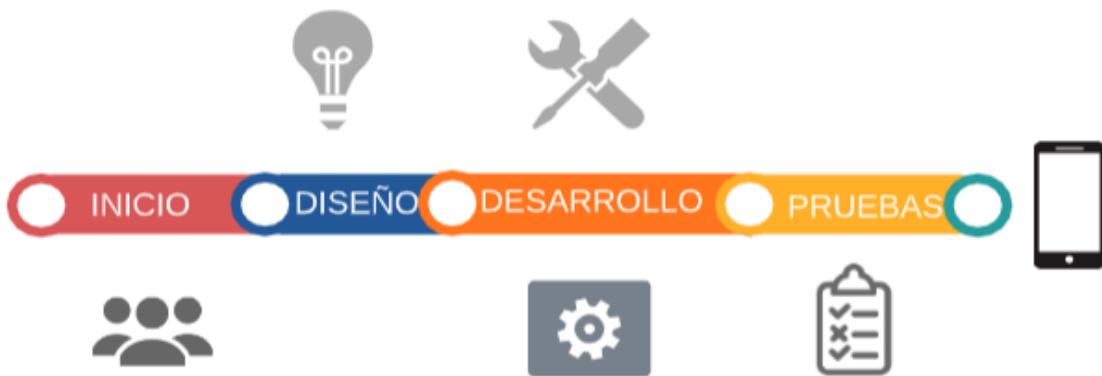
La computación ubicua o previsiva es la nueva era de la computación, su objetivo se centra en que el computador sea parte del entorno de una persona, se basa en que “la vida no debe pasar atrás de un escritorio” y que un individuo necesita moverse constantemente para desarrollar sus diferentes tareas, de tal manera, que la informática debe ser considerada parte del ser humano sin que éste se dé cuenta que está ahí para solucionar todas sus necesidades. La ubicuidad busca ampliar las capacidades de los navegadores web para dar paso a nuevos tipos de aplicaciones que se coordinan con otros dispositivos y se adaptan dinámicamente al usuario, a las características del dispositivo y a las condiciones del entorno. Las aplicaciones podrían utilizar los servicios de red para extender las capacidades de los dispositivos. Los usuarios se podrán concentrar en lo que hacen en lugar de en los dispositivos y se adaptan dinámicamente al usuario, las características del dispositivo y a las condiciones del entorno. Las aplicaciones podrán utilizar los servicios de la red para extender la capacidad de los dispositivos. La movilidad en las aplicaciones permitirá que los usuarios continúen trabajando mientras que, sin notarlo, pasaran de un dispositivo a otro (Rodríguez, y otros, 2016).

El presente proyecto se basa en los principios de la computación móvil, se debe hacer énfasis en la practicidad que conlleva esta disciplina, son muchos recursos agrupados a un gran concepto, “La comunicación móvil inicialmente consistió en la comunicación oral y actualmente posee la capacidad para el trabajo grupal a distancia como mensajería instantánea, compartir información y su modificación mediante diferentes aplicaciones, entre otras características” (Montiel, Rubio, & López, 2012). Se desea plasmar el concepto de computación móvil mediante un desarrollo de software que va orientada a una aplicación Móvil (App).

Es importante destacar que, como servicio final, se tendrá una aplicación móvil, a través de Android Studio, el cual es un entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan tu productividad durante la compilación de apps para Android y acompañado de MySQL el cual es sistema de gestión de base de datos relacional o SGBD hacen un gran equipo. MySQL un gestor de base de datos en multihilo y multiusuario, lo que le permite ser utilizado por varias personas al mismo tiempo, e incluso, realizar varias consultas a la vez, lo que lo hace sumamente versátil.

## Metodología

Dadas las características del problema que se está abordando se toma la decisión de converger un método de investigación para desarrollar la misma. Se toma el método experimental pues a través de este surgen los resultados de la primera y segunda fase del proyecto, además, se sigue un proceso de prueba en cada tarea a realizar como se muestra en la Figura 1.



*Figura 1 – Diagrama Metodológico*

La primera fase llamada Inicio, se comienza por una comparación y selección del modelo de Internet de las Cosas (IoT), con el fin de diseñar la arquitectura que se utilizará en el proyecto. Posterior a

este proceso, se planea identificar y seleccionar las tecnologías a utilizar en la arquitectura, como la arquitectura propuesta para el modelo IoT, generador de datos, comunicación, versión de Android, entre otros. Considerando aspectos principales a tener en cuenta, tales como el WebService, el servidor, la base de datos y la aplicación móvil que es donde se concentra el presente proyecto y se comenzara a trabajar más a fondo en la siguiente fase.

En la segunda fase denominada Desarrollo, se inicia con la construcción de la arquitectura de GIEMA a partir de las herramientas y componentes seleccionados en la primera fase del proyecto. Es importante destacar que en esta etapa se planea la construcción de las bases sólidas que darán vida a este proyecto por eso se manejarán temas como el modelo de datos, la construcción lógica GIEMA plasmado en planos UML y los mockups que darán el diseño y la forma para la interacción del usuario directamente con los datos por medio de un apartado gráfico, para una posterior construcción de la aplicación móvil e integración de los componentes utilizados.

La importancia de utilizar como método el método experimental en estas fases del proyecto es la validación y conclusión de cada proceso que se planea desarrollar. También, por medio de este método se permite que surjan los resultados que se quieren por medio del desarrollo de los conocimientos que se tengan, como consecuencia del esfuerzo que realizará en las actividades, controlando deliberadamente las variables y controlarlas dentro del modelo, ganando mejores resultados, tratando de simular los datos de manera real y probar el desempeño de la aplicación si está más adelante quiere ser usada y vaya evolucionando.

Para la tercera fase denominada desarrollo encontraremos todo lo relacionado con la construcción en código, despliegues, especificaciones técnicas, comunicación lógica, funcionamiento interno de los modelos, versiones, configuración de herramienta de desarrollo, configuración de la aplicación e instalación de la aplicación, en esta fase se ejecutara todo la información recopilada para dar como resultado nuestra

aplicación móvil, mediante el conjunto de habilidades y el uso de información para que sean aplicadas de manera eficiente y se dé un buen resultado.

Para la cuarta fase del proyecto denominada Pruebas, se utilizará el método de observación, a través del cual se buscará analizar la experiencia de usuario y el comportamiento de la aplicación procesando los datos en bloque generados a partir de herramientas de simulación de datos y comparar los diferentes resultados obtenidos realizando un ejercicio de observación y test en la aplicación. Esta fase inicia observando el flujo y el comportamiento de la aplicación mediante la evolución de backend (WebService, Modelo de datos y Arquitectura Lógica) con el apartado grafico entregando una experiencia al usuario con el fin de realizar una prueba final de la funcionalidad de la aplicación. Como producto final de este método, se pretende obtener las conclusiones del proyecto, siendo esta la fase más crítica en del presente proyecto ya que significa su terminación según lo planteado, todo este proceso se organizó mediante un cronograma que se puede observar en la Figura 2.

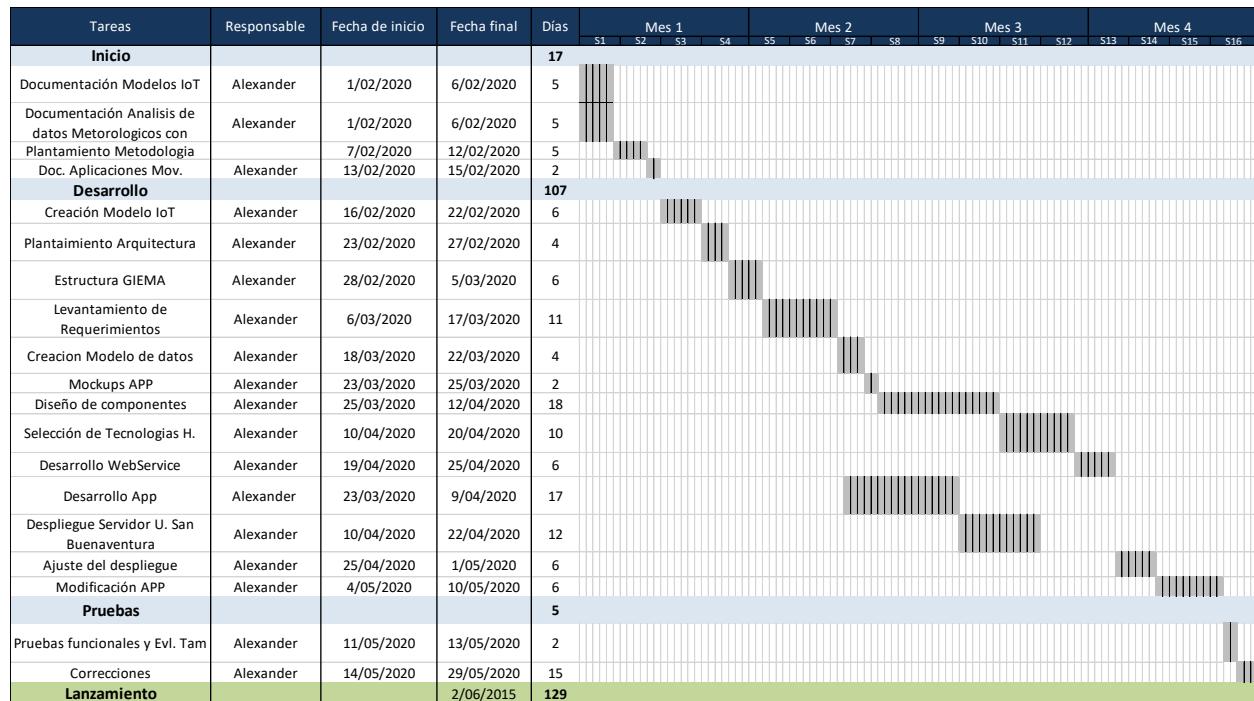


Figura 2 - Cronograma de Actividades

## Capítulo 2

En esta parte del documento se resaltara temas como el modelo IoT, planteamiento del diseño GIEMA y sus características, fundamentos del desarrollo y aplicación en la codificación la arquitectura, modelos UML planteados como pieza fundamental del presente proyecto ya que en base al modelo planteado se desprenderán todos los componentes que fueron necesarios para definir la arquitectura del proyecto y darle forma en base a los objetivos planteados, adaptándolo lo mejor posible en base al estudio, análisis y comparativas realizadas, para tener como resultado un modelo sólido y eficiente llamado GIEMA.

Para iniciar la explicación de cómo se desarrolló y planeo cada componente hasta poder plasmarlo, se comenzará explicando la definición del modelo IoT con el que se trabajó en el presente proyecto presentado a continuación.

### 2.1 Modelo IoT

El modelo IoT planteado para GIEMA es un modelo netamente orientado a la administración de datos meteorológicos para su consulta y visualización, se trata de un modelo flexible que con su arquitectura puede ser modificado dinámicamente, ya sea quitando componentes o agregando sin afectar su fin mientras no se modifiquen componentes CORE que definen el modelo como IoT, si este principio se rompe ya no tendría las propiedades de un modelo IoT por eso los componentes Core no deben afectarse en base a los criterios que esta arquitectura planteada, estos componentes son todos aquellos que intervienen directamente en el proceso de que el modelo sea catalogado dentro del término “Internet de las cosas” el cual se define como una propiedad única que cumple un criterio de conexión a la nube sobre un nodo o una estación meteorológica autónoma en este caso.

Se definió la arquitectura de GIEMA realizando un estudio y análisis comparando algunas arquitecturas de modelos IoT más famosas y eficientes en el mercado que se verán a continuación.

### 2.1.1 Evaluación De Modelos Existentes

En base al estudio y análisis realizado a diferentes modelos IoT que en este momento lideran varios campos de la industria tecnológica se logró entender varios términos y componentes de sus modelos, aunque su orientación es distinta sus modelos IoT se plantean como un concepto general que deben cumplir todas las estructuras tecnológicas que crean una arquitectura IoT, en base a esto se planteó el modelo GIEMA examinando los componentes que toma cada arquitectura, su fin y como procesan esa información para plasmar su objetivo, para GIEMA el objetivo de esta investigación y análisis fue aprender cómo se maneja la tecnología IoT aplicándola a un modelo en este caso dedicado a la administración para la visualización y consulta de datos meteorológicos generados por estaciones meteorológicas, pero para este caso por medio bloques de datos generados por una herramienta generadora de datos ya que no se cuenta con la parte física, pero se evaluó y diseño una estructura física de integración en el modelo IoT para que sea integrada más adelante y tiene influencia de un proyecto desarrollado Previamente en la universidad San Buenaventura.

Para este proceso de análisis se evaluaron 6 modelos IoT en los cuales se analizó la composición de su arquitectura, su funcionamiento y fin al que está orientado el servicio que presta, el primero fue el modelo de IBM centrado en el manejo de conexiones para modelos híbridos, está orientada al control de esquemas y modelos de machine learning, seguidamente se evaluó el modelo IoT de Intel que está centrado en el proceso y control de data a gran velocidad aunque su modelo tiene varios componentes dedicados a la hibridación de control de procesos no pierde el principio de control de datos masivos que es lo que busca GIEMA, el siguiente modelo evaluado fue un modelo IoT genérico de Google donde se determinó al evaluarlo como uno de los modelos más adaptables a lo que busca GIEMA ya que su arquitectura se basa en el manejo de datos con un fin de consulta lo que lo convierte en un modelo óptimo para basarse y construir parte de la arquitectura de GIEMA, el siguiente modelo a evaluar fue el modelo Azure Cloud, construido bajo el concepto de administrar varias tareas, está construido para manejar grandes cantidades de datos y funciona cumpliendo tres servicios básicos de aplicaciones que son construcción, prueba y

despliegue. El siguiente modelo IoT evaluado fue el modelo IoT Cisco centrado en la seguridad de redes y análisis de datos está centrado a la automatización de procesos y redes, su gran aporte al proyecto fue explicar cómo estructuran una arquitectura más flexible a pesar de todos los componentes que pasan por cada una de las capaz planteadas y por último tenemos el modelo AWS IoT su composición se basa en los principios de poder mover grandes cantidades de datos por su arquitectura flexible lo convierten en una de las fuentes principales de las cuelas se recopilo bastante información de cómo se debe construir y repartir esta carga para que todo el modelo trabaje uniformemente, aunque todos los modelos fueron de gran ayuda para plantear GIEMA los que más resaltan son Google Cloud IoT y AWS IoT debido a su orientación al manejo de grandes cantidades de datos lo que convierte a estos dos modelos en un ejemplo a seguir por GIEMA para la manipulación y uso de los datos meteorológicos generados por estaciones meteorológicas autónomas de manera informativa, para más información sobre este proceso de análisis de los modelos IoT referirse al ANEXO A.

En esta sección se realizó la comparativa de todos los modelos para evaluar su tecnología y en base a la información evaluada arriba se pudo definir como estaban relacionadas entre sí en base a su arquitectura, ayudando así a la evaluación para determinar la arquitectura de GIEMA, como se muestra en la Tabla 1.

*Tabla 1 - Comparativa de modelos IoT*

Modelo/Característica	Dispositivo	Gateway	Network	Servicio	Análisis de Datos	Aplicación
<b>IBM</b>	Sí	Sí	Sí	Sí	Sí	Sí
<b>Intel</b>	Sí	Sí	Sí	Sí	Sí	Sí
<b>Plataforma de nube de Google</b>	Sí	Sí	Sí	Sí	Sí	Sí
<b>Azure Cloud</b>	Sí		Sí	Sí	Sí	Sí
<b>Cisco</b>	Sí	Sí	Sí		Sí	Sí
<b>AWS Amazon</b>	Sí	Sí	Sí	Sí	Sí	Sí

Después de realizar análisis se forjaron las bases para plantear un modelo para GIEMA.

### 2.1.2 Modelo GIEMA IoT Propuesto

El modelo IoT de GIEMA nace del resultado del proceso de análisis y estudio que se realizó anteriormente evaluando cuidadosamente los modelos IoT escogidos para que funcionaran de referencia y plantear algo mucho más sólido en un modelo que resalta su arquitectura, funcionalidad, tecnologías y su fin, dando como resultado el modelo de GIEMA que se plantea como se observa en la Tabla 2.

Para el presente proyecto se definieron cinco capaz en las cuales van a estar ubicadas cada uno de los componentes que van a hacer a GIEMA posible.

*Tabla 2 - Capaz del Modelo GIEMA*

<b>Modelo GIEMA - IoT</b>		<b>Funcionalidad</b>	<b>Condiciones</b>
<b>Aplicación:</b>	App Móvil	Su función es la notificación y visualización de datos, además de ser el medio la para realizar peticiones por medio de las diferentes funciones establecidas dentro la aplicación en su desarrollo	La aplicación funcionará desde Android 5.0 en adelante y requerirá una conexión a internet
<b>Servicio:</b>	WebService	Es la parte más dinámica de la arquitectura, encargado de la administración de datos, el WebService controlara el flujo de datos, recibiendo y procesando las peticiones y orientando los datos que se solicitan en el momento que se solicitan, además de cumplir órdenes y ejecutarlas según lo establecido en el desarrollo del código	El servidor escogido cumple con los parámetros establecidos de funcionalidad y soporte para el soportar el flujo de datos sin saturarse
	Servidor	Encargado de almacenar los datos y responder a las peticiones enviadas desde la app que pasan por el	El servidor se es propiedad de la universidad San Buenaventura por lo

<b>Modelo GIEMA - IoT</b>		<b>Funcionalidad</b>	<b>Condiciones</b>
		WebService, actúan según el modelo de datos diseñado	tanto todos los lineamientos de seguridad deben ser acatados, incluyendo las limitaciones a solo administrar la base de datos de GIEMA parcialmente
<b>Red:</b>	Conexión (IP)	Simulación	Simulación
<b>Gateway</b>	Protocolos de comunicación	Simulación	Simulación
<b>Dispositivos</b>	Tarjeta - Sensores	Simulación	Simulación

Para el presente proyecto se definieron cinco capaz en las cuales van a estar ubicadas cada uno de los componentes que van hacer a GIEMA posible la primera capa la capa de Dispositivos donde se encuentra la parte física compuesta por los sensores y la estación meteorológica donde se transmitirán los datos por medio de un módulo de conectividad inalámbrica que en este caso podría ser xBee una conexión por IP y pasar al Gateways para controlar todo el protocolo de comunicación entre la parte física y el WebService encargado de intercomunicar los servidores de GIEMA los datos que trae el Gateway enviados desde la estación meteorológica para ser almacenados o atender una petición de la APP móvil la cual es la interfaz de interacción con el usuario para que realice alguna de las acciones habilitadas, donde más adelante se va profundizar que da sección a medida que avance el documento, en este punto con la información más clara y el modelo definido se puede plantear de manera más completa sus componentes internos a continuación, para más información sobre este proceso consultar el Anexo A.

## 2.2 Caso De Estudio

Para un mejor desarrollo y planeación después de diseñar y definir un modelo IoT se procede a diseñar y evaluar los componentes internos y su funcionalidad lo que crea un requerimiento funcional o no funcional según su impacto y su tarea dentro del proyecto para posteriormente relacionarlos entre sí y

formar un caso de uso optimo que va a ser la guía y composición en nuestro diseño para el tratamiento de las variables meteorológicas a usar.

### **2.2.1 Requerimientos**

A partir del estudio que se realizó para solventar la problemática planteada y dar un inicio optimo al diseño del proyecto, se plantearon los casos de usos funcionales según el criterio de impacto y usabilidad en el proyecto.

Inicialmente se plantearon los requerimientos de la aplicación móvil ya que va a ser el medio en el que el usuario inicia una petición de esta manera comienza a construirse toda una funcionalidad y varios que componentes van a hacer fluir la parte lógica funcionan de la APP móvil representadas por peticiones que se enviarán desde la app y darán como resultado la acción requerida como se muestra en la Tabla 3.

*Tabla 3 - Requerimientos funcionales APP*

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RF-001	Entrada de Datos de Usuario	El sistema permitirá la entrada de datos para almacenar	ALTA
RF-002	Validación de datos de usuario	El sistema la autenticidad de los datos en la aplicación	ALTA
RF-003	Consulta de datos Meteorológicos	El sistema consultara los diferentes datos meteorológicos generando una respuesta	ALTA
RF-004	Filtración de datos por fecha	El sistema filtrara los datos por fecha indicada	MEDIA

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RF-005	Consumo de datos por consulta	El sistema ejecutará una consulta asignada al WebService	ALTA
RF-006	Encriptación de contraseña	La contraseña se encriptará	ALTA
RF-007	Clasificación de datos meteorológicos	Los datos meteorológicos se filtrarán por función y sensor	ALTA
RF-008	Llamada de datos por WebService	El sistema llamará los datos consumiendo el Webservice	ALTA
RF-009	Consumo de datos por clase	El sistema consumirá los datos según la clase en las que se ejecuten	ALTA
RF-010	Datos Meteorológicos formato JSON	Los datos meteorológicos se enviarán y se entregarán en formato JSON	ALTA
RF-011	Consumo de WebService por consulta	Se realizará una ruta de consumo para llegar al WebService	ALTA
RF-012	Consumo de datos por consulta Query	Se consumirá una consulta SQL para ejecutar el servicio solicitado	ALTA

También se catalogaron requerimientos no funcionales con el fin de organizar algunos componentes más gráficos y dependientes de la experiencia de usuario que si bien su implementación no está ligado a los procesos core como las funciones y métodos que se parametrizan en el desarrollo, su

implantación es muy importante para mejorar la experiencia de usuario y como está compuesta la app esto se puede observar a continuación en la Tabla 4.

*Tabla 4 - Requerimientos No Funcionales App*

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RNF-001	Ingreso de datos en cajas de Texto Login	La aplicación permite la digitalización de datos por medio de cajas	ALTO
RNF-002	Navegación entre pantallas por botones	La aplicación permitirá la navegación de pantalla por botones	ALTO
RNF-003	Visualización de un menú	La aplicación permitirá la visualización con las diferentes opciones de la app	ALTO
RNF-004	Visualización e interacción de un Spinner	Visualización de un Spinner para seleccionar el nodo	MEDIO
RNF-005	Selección de opción de datos por botones	El tipo de sensor que se desea consultar se debe hacer por medio de un botón	MEDIO
RNF-006	Grafica de datos	La aplicación mostrara una gráfica con los datos	ALTO
RNF-007	Imagen de Login Dinámica	La aplicación tendrá una imagen de presentación	BAJO

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RNF-008	Imágenes de fondo en todas las pantallas	Cada pantalla tendrá una imagen de fondo	BAJO
RNF-009	Almanaque de selección	La aplicación tendrá una forma de ingresar la fecha	ALTO
RNF-010	Ocultar contraseña	La aplicación ocultara la contraseña	ALTO
RNF-011	Visualización de datos meteorológicos	Los datos meteorológicos deben visualizarse	ALTO

Para el WebService se planteó el mismo proceso, pero su orientación y como se levantaron estos requerimientos iba más orientado al procesamiento de funciones y datos ya que el WebService es el encargado de administrar las funciones ya parametrizadas para cumplir las peticiones enviadas desde la app, para realizar la acción y notificar al servidor para que valide su operación dentro de la orden enviada, primero se plantearon los requerimientos funciones con el fin de organizar algunos componentes en ciertos formatos o parametrizar su envío y su respuesta, y las instrucciones de cómo se debe consumir el servicio todo esto evidenciado en la Tabla 5.

*Tabla 5 - Requerimientos Funcionales WebService*

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RF-001	Envío de datos al Servidor	El WebServices enviará los datos al Servidor	ALTA
RF-002	Parseo de datos formato JSON	Los datos serán parseados al formato JSON	ALTA
RF-003	Validación de datos	Se hará una validación de los datos ingresados	ALTA
RF-004	Almacenamiento de datos	Se almacenarán los datos	ALTA
RF-005	Consumo por servicio	Se consumirá el servicio requerido en el WebService	ALTA
RF-006	Ejecución de consulta Query	Se ejecutará el Query requerido	ALTA
RF-007	Repuesta de datos de las peticiones	Se generará una respuesta por petición	ALTA
RF-008	Esquema de datos con llaves foraneas	El esquema se relacionará con mediante llaves foraneas	ALTA
RF-009	Flujo de datos por petición	Según la petición se realizará un flujo de datos	ALTA
RF-010	Interconexión de los Servicios	Los servicios consumirán un medio de enlace	ALTA

Con esto definido se procedió a plantear como encajarían y que acción iban a formar en conjunto ya que al unir varios forman una secuencia de acciones que se verá a continuación para más información remitirse al ANEXO B.

## 2.2.2 Casos De Uso

A partir de los requerimientos planteados al unirlos y empezar a componer varias acciones dependientes de un flujo predefinido da como resultado un caso de uso, según lo planteado anteriormente en paralelo se diseñó una serie de casos de uso que componen la estructura y funcionamiento de la aplicación móvil y el WebService para esto se plantearon los siguientes casos de uso referenciados en la

Tabla 6.

*Tabla 6 - Casos de uso GIEMA*

Nº de Caso de Uso	Nombre del Caso de Uso	Descripción
CU-001	Logueo a la aplicación	El usuario podrá realizar loguearse a la aplicación realizando un proceso de validación
CU-002	Navegación entre pantallas	El usuario podrá navegar entre pantallas para ver su contenido
CU-003	Registro en la aplicación	El usuario podrá registrarse a la aplicación digitando sus datos
CU-004	Consulta datos recientes	El usuario podrá consultar sus datos básicos
CU-005	Consulta de nodo	El usuario podrá visualizar el nodo que va a consultar
CU-006	Consulta de datos por sensor	El usuario podrá seleccionar el sensor que desee visualizar
CU-007	Visualización de datos Meteorológicos	El usuario podrá visualizar los datos Meteorológicos
CU-008	Visualización de la gráfica de datos	El usuario podrá ver la gráfica de los datos consultados

Nº de Caso de Uso	Nombre del Caso de Uso	Descripción
CU-009	Selección de fecha de consulta de datos	El usuario podrá filtrar los datos que deseé consultar por fecha

Con esto se dio cierre al levantamiento de requerimientos para llevar un orden en lo que va a ser el inicio del diseño de GIEMA, en base a toda la información planteada anteriormente en conjunto con los casos de uso para más información sobre este proceso remitirse al ANEXO B.

## 2.3 Diseño GIEMA

Para el diseño de GIEMA se comenzó definiendo cuatro grandes pilares que serán la base del presente proyecto tomando como prioridad el diseño y la construcción de estos componentes para darle vida al proyecto, a continuación, se evidenciará como se estructuró cada componente y el resultado de cada uno, comenzando con la arquitectura seguido del modelo de datos, modelos UML y la interfaz gráfica del usuario.

### 2.3.1 Arquitectura

A partir del estudio previo y toda la información recogida mediante el análisis y comparativas de diferentes modelos dando como resultado el modelo propio de GIEMA, se determinó una arquitectura única separada en las capas del modelo planteados con los siguientes componentes, nodo o estación meteorológica seguido por un componente de red para conectarse con Gateway que está conectado a un webservice donde este componente actúa como mediador en el intercambio de datos que envía la aplicación por medio de una petición al servidor donde el webservice es el intermediador y transporta tanto el request como el response de la función o petición ejecutada recordemos que este flujo es bidireccional como se puede observar en la Figura 3 el modelo de arquitectura.

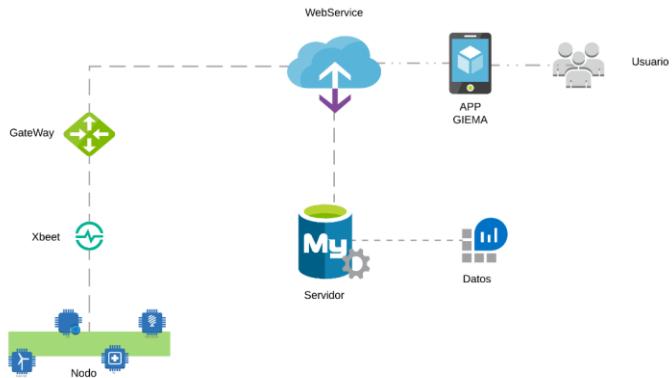


Figura 3 - Arquitectura GIEMA

Para el primer componente el cual es llamado nodo que se entiende por la parte física o estación meteorológica autónoma que está compuesta por sensores que son los encargados de recolectar los datos, para iniciar su flujo dentro de la estación meteorológica que provee la energía para el funcionamiento de los sensores y para ella misma ya que entendemos autónoma como la capacidad de poder obtener energía de manera autosuficiente en este caso por ejemplo por medio de energía fotovoltaica como se muestra en la Figura 4.

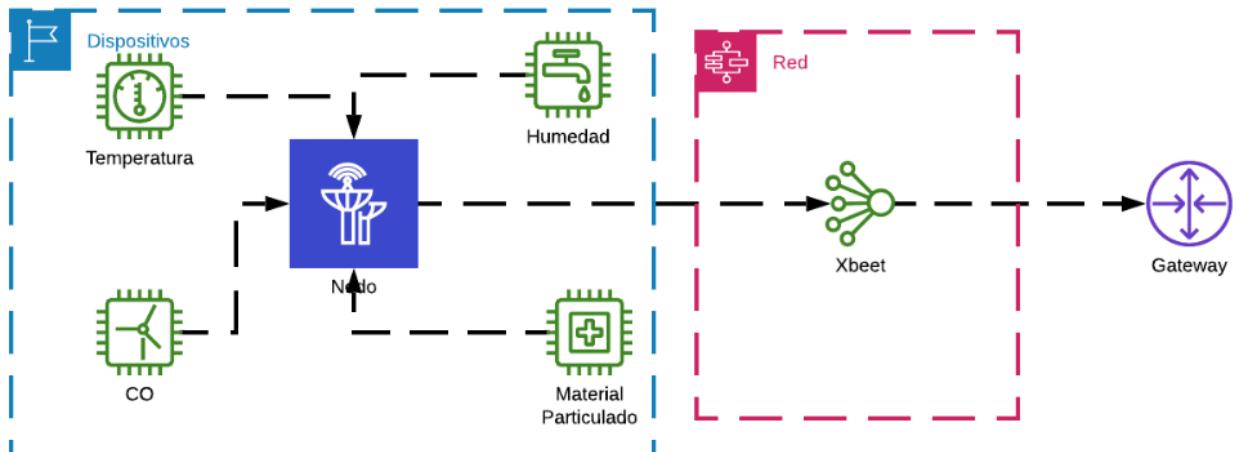


Figura 4 - Componente Nodo GIEMA

El siguiente componente es el medio de conexión, un dispositivo de comunicación de red por medio de una IP para iniciar un proceso de protocolos de red usada para la conexión entre el nodo y el gateway,

en este caso se plantea la tecnología de tarjetas Xbeet ya que son livianas y de fácil uso, además es un componente óptimo para nodos autónomos.

El siguiente componente es el Gateway o la puerta de enlace que se traduce como el dispositivo que actúa de interfaz entre la parte física que es la estación meteorológica y el WebService como se muestra en la Figura 5, cabe resaltar que esto puede ser de carácter lógico, a lo que se plantea que los datos serán procesados de manera limpia en esta sección hasta este punto estos componentes serán emulados por medio del resultado final que serán los datos normalizados ya que no se cuenta con la parte física, pero se plantea una arquitectura sólida y flexible para su integración.

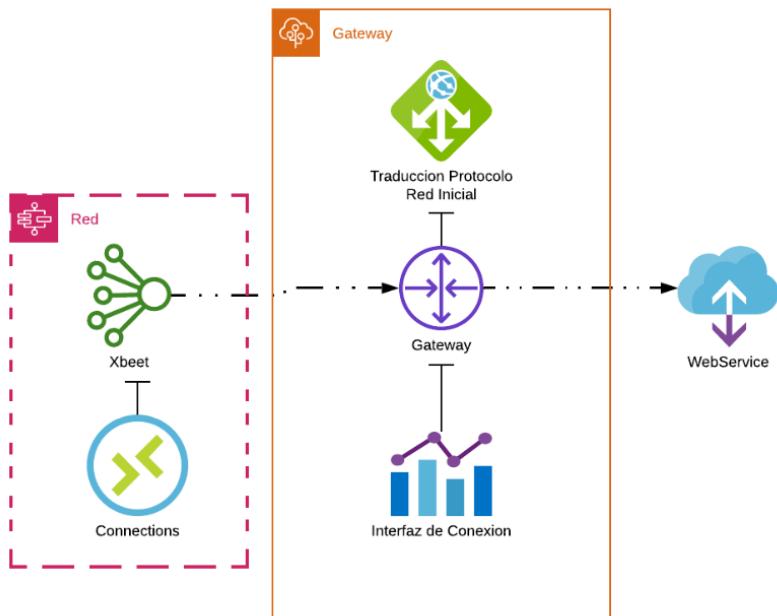


Figura 5 - Componente Gateway GIEMA

El siguiente componente es el WebService, este componente recibe los datos y los almacena en el servidor, en este punto como se comentó estos datos fueron creados por medio de un generador de datos para simular el proceso final de la parte física, adicionalmente el WebService es el puente entre la APP y el servidor, su función principal será procesar las funciones que solicite la app y enviarlas al servidor con

la parametrización indicada según la función que se necesite realizar, para posteriormente recibir la respuesta y remitirla a la aplicación como se muestra en la Figura 6.

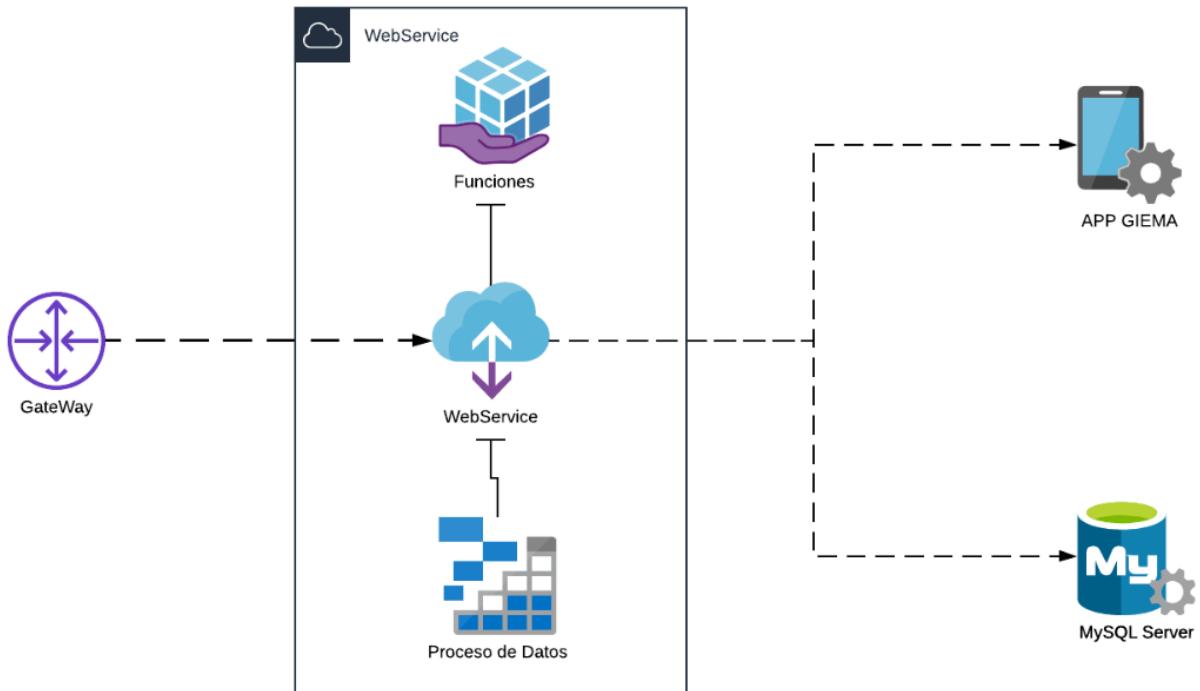


Figura 6 – WebService y Aplicación GIEMA

Y el ultimo componente que se muestra a continuación es el servidor el encargado de almacenar toda la información sobre el nodo, los datos meteorológicos sensores y usuario, su función también está en responder a las peticiones enviadas por medio de la app y usando como puente de procesamiento el WebService cumplir la orden que le da dicha función, ya sea consulta de datos de una petición parametrizada o la edición adición de algún dato al que el usuario quiera hacer uso como se muestra en la Figura 7.

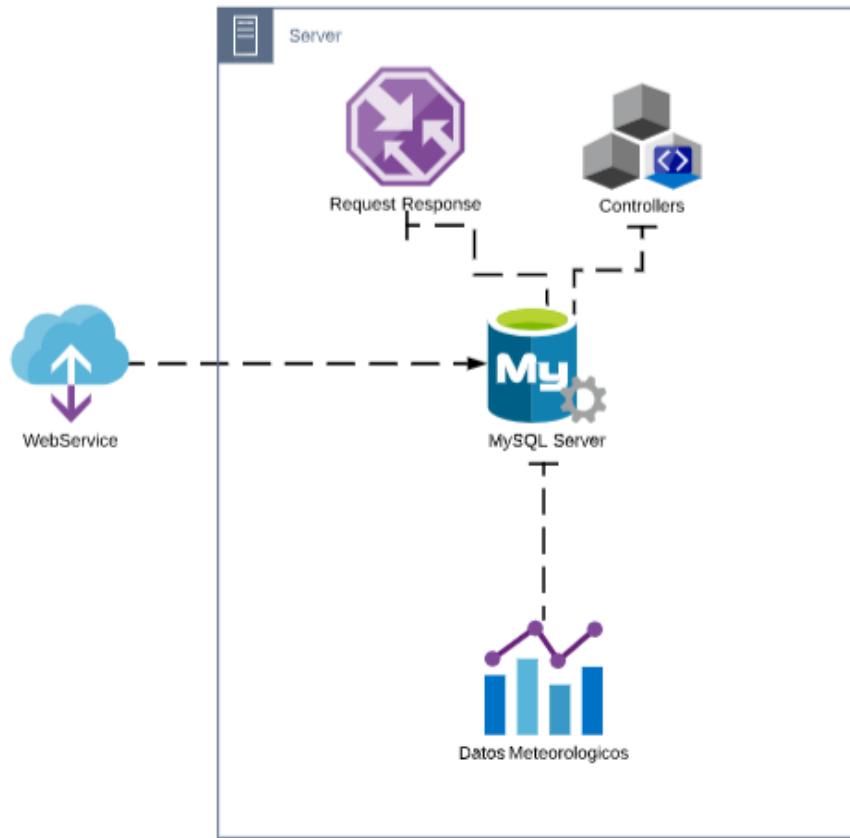


Figura 7 - Server GIEMA

Teniendo una arquitectura sólida definida y un modelo IoT por capaz se procede a plantear un modelo de datos.

### 2.3.2 Modelo De Datos

El modelo de datos como todos sus componentes es una pieza clave del proyecto debido que del buen flujo de las relaciones y limitaciones entre tablas hace que sea más óptimo los procesos o las peticiones a la base de datos, debido a que este proyecto maneja una cantidad considerable de datos producidos para posteriormente ser utilizados en diferentes funciones, como resultado el modelo de datos se diseña en base a las necesidades del sistema, ya que requiere un constante flujo de información en cuanto a los datos que recoge y su interacción con la aplicación, el flujo de modelo de datos se diseña de una manera que pueda darle flexibilidad y volver más liviana la carga para las peticiones y para la propia información que se

guarda en él servidor debido a que está constantemente activo durante el proceso de recolección de información de las estaciones meteorológicas que proveerán los datos para posteriormente ser procesados por el webService y trabajados juntos con el servidor mediante el modelo de datos cumple su función almacenando y manipulando datos entre tablas, debido al diseño de este modelo de datos.

Siguiendo los lineamientos usados y con el propósito de crear un buen desarrollo y flujo en el sistema, se pone especial atención al modelo de datos donde se determina la creación de siete tablas diferentes, donde dos son para la almacenar la información del usuario y estas dos están relacionadas, donde una guardara la información del usuario y la otra dará el atributo al cual pertenece al usuario, las otras cinco tablas manipulan la información del sistema IoT donde una tabla almacena la información del nodo que está relacionada a la tabla que almacena el Gateway, esto con el propósito de tener información de los datos que los interrelaciona, esto hace que el nodo se conecte con la información de la tabla sensor que se asocia a la tabla tipo\_sensor donde el tipo\_sensor almacena los datos del atributo principal que cumple dicho sensor y devuelve la información como llave foránea para ser asociado a la tabla sensor que almacena los datos y los atributos principales de un sensor junto con su funcionalidad principal, y finalmente llevado a la tabla mensaje que contiene los datos dados por la estación meteorológica y los atributos principales de los sensores dados por sus relaciones con las demás tablas como se muestra en la Figura 8, esto con el propósito de que todos los datos sean interrelacionados para crear procesos más rápidos y la respuesta sea más ágil y no sobre carguen al servidor, como se dijo anteriormente se necesita un modelo de datos eficiente ya que recibirá cantidades considerables de datos y esto se hace con el fin de no sobre cargar al servidor cuando el usuario haga una petición por la aplicación móvil o la estación meteorológica mande datos al servidor de manera continua para más información remitirse al ANEXO C.

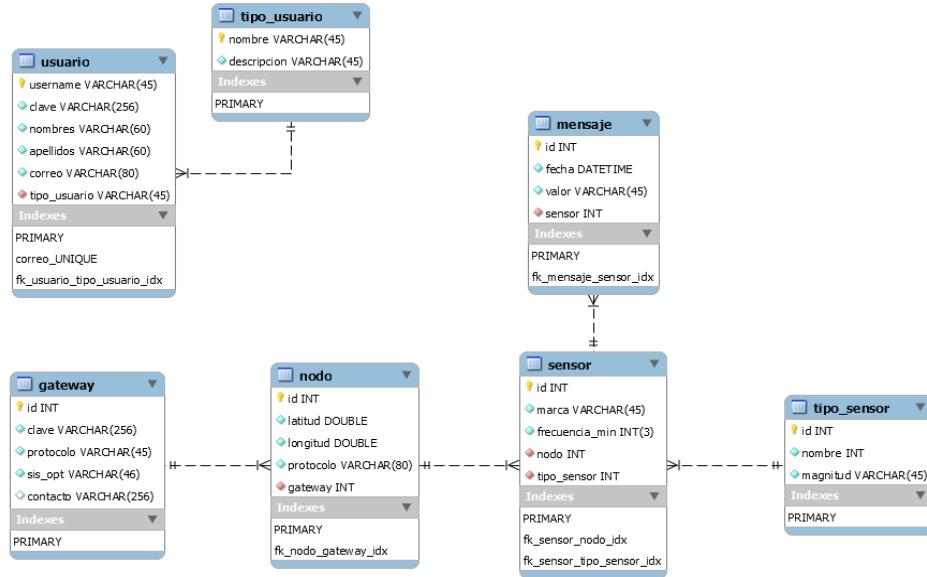


Figura 8 - Modelo de datos

Una vez se diseñó y estructuró el modelo de datos se procedió a darle un orden a todos los componentes, graficando toda la estructura de GIEMA mediante modelos UML.

### 2.3.3 Modelos UML

Parte de las funciones del sistema deben ser evaluadas según los requerimientos dados por los objetivos del presente proyecto, para determinar esos procesos y funcionalidades que se deben dar en el sistema para cumplir los objetivos dados se realizaron una serie de análisis para construir los diagramas necesarios y así pasar a la etapa de desarrollo del sistema IoT, con base a esto se facilitó mucho más la etapa de desarrollo que se verá más adelante.

Como primera medida se evaluó como va a estar compuesto el proyecto en su totalidad para construir de manera óptima los procesos con los que va a interactuar tanto los diferentes componentes del sistema y el usuario, como las funcionalidades que este va a prestar tanto los componentes como el usuario, referenciando en la Figura 9, donde se evidencia la estructura general de cada componente y como se

relacionan entre si convirtiéndose en un sistema armoniosos, para más información consultar el ANEXO C.

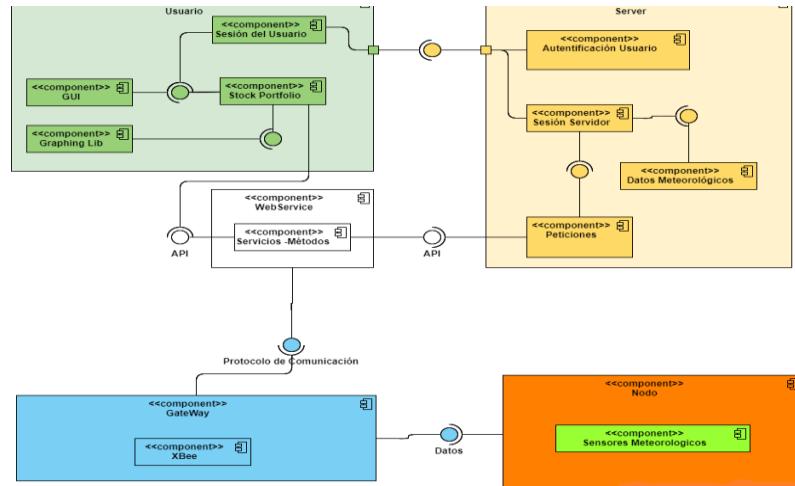


Figura 9 - Diagrama de componentes

El usuario podrá interactuar por medio de la aplicación móvil, para tener acceso a los datos meteorológicos proporcionados por el nodo autónomo, por medio de unos procesos internos que se estarán alcance de las opciones de la aplicación móvil para interactuar directamente con todo el sistema de manera rápida que están condicionados por unos casos de uso que darán lugar a las opciones de la aplicación que se pueden evidenciar en la Figura 10, se complementa en la Figura 11.

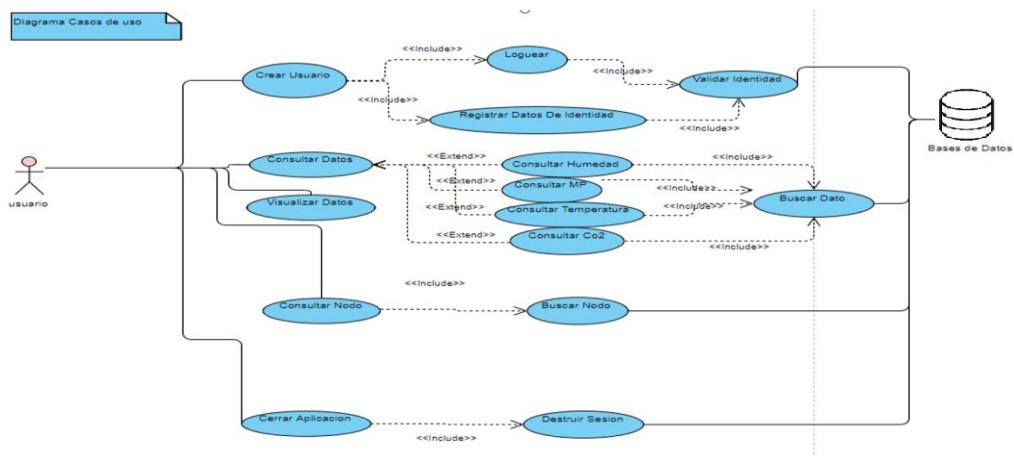


Figura 10 - Diagrama de casos de uso APP

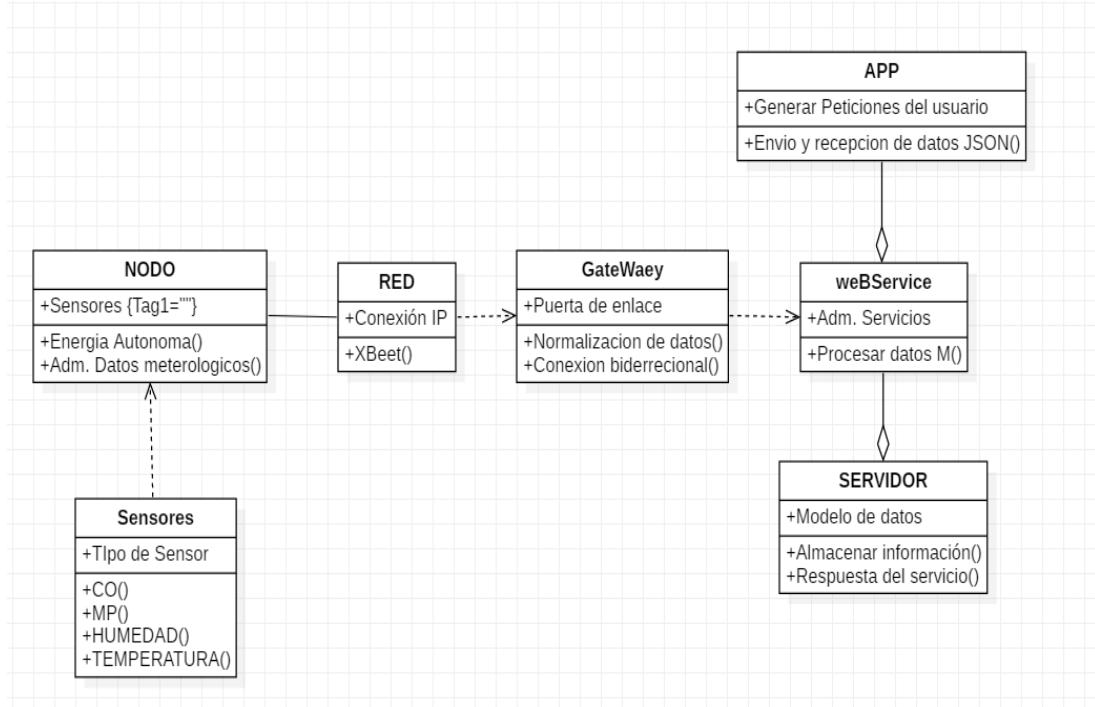


Figura 11 – Diagrama de casos de uso GIEMA

Con los diagramas UML estructurados se procedió realizar el diseño de la parte gráfica del proyecto, la interfaz de usuario.

### 2.3.4 Interfaz Gráfica De Usuario

La interfaz gráfica de usuario se diseñó en base a la orientación del proyecto, ya que de este componente depende la experiencia del usuario en cuanto a practicidad de las acciones y como la aplicación responde a la funcionalidad representándolo en la app y su navegación, en base a esto se diseñó un diagrama de navegación que es el que va a seguir GIEMA que vera a continuación en la Figura 12.

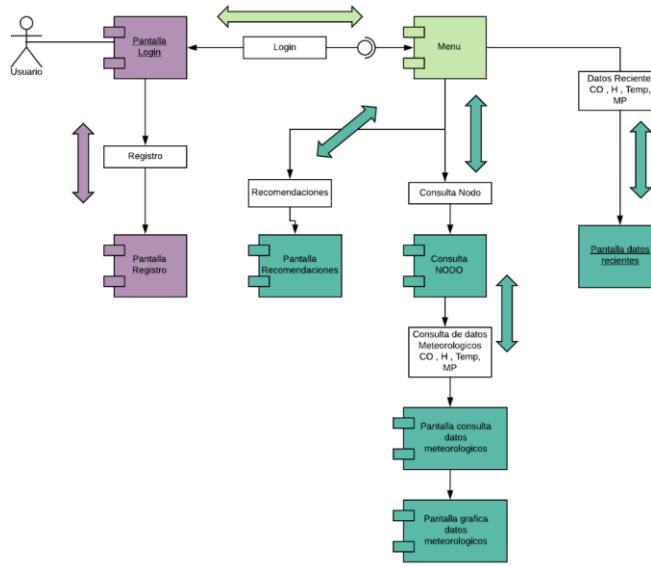


Figura 12 - Diagrama de Navegación

A partir de estos de este diagrama se diseñan los mockups planteando como se ve la aplicación donde el resultado será el siguiente, solo se mostrarán las pantallas más relevantes como se ve en la Figura 13, para más información sobre los Mockups consultar el ANEXO C.

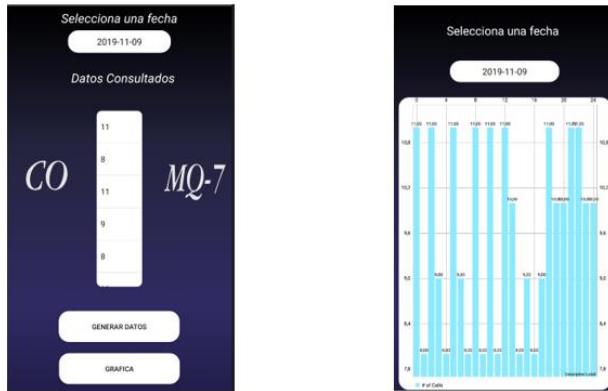


Figura 13 - Mockups Grafica y Generación de datos

GIEMA busca la practicidad a la hora de usar la aplicación, como los objetivos del proyecto están orientados netamente a la consulta de datos no tiene por qué ser una experiencia tediosa para el usuario, si no fluida y eficiente.

Con este segmento se concluye la fase de diseño, gracias al estudio previo y conocimiento adquirido se pudo organizar todas esas ideas e información aplicándolos a un modelo de diseño que fue la base para la construcción o desarrollo del presente proyecto, a continuación, se dará inicio a esa etapa de ejecución teniendo los lineamientos de desarrollo mucho más claros, gracias a un diseño sólido estructurado en todos los componentes.

## **2.4 Desarrollo**

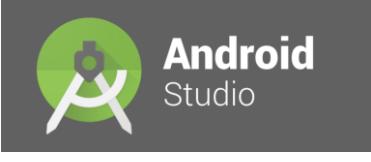
Para el desarrollo de la aplicación se tuvieron en cuenta muchos componentes y siguiendo los lineamientos previos planteados en el diseño fue cuestión de cómo se podría hacer realidad todas esas ideas planteadas de manera estructurada, en este punto se tuvo que investigar y seleccionar qué herramientas que se podrían usar para hacer realidad todo ese modelo y arquitectura planteado previamente, así que se separaron esas tareas en cuatro partes, Tecnologías usadas sección en la cual se explicará qué tecnologías se seleccionaron para la creación del proyecto, configuración de ambientes en este punto se explicará la configuración de los ambientes seleccionados, estándares y lineamientos de codificación usados y por último los productos resultantes del presente desarrollo.

### **2.4.1 Tecnologías Utilizadas**

El primer punto de partida para desarrollar el proyecto fue determinar con qué tecnologías se iban a desarrollar, en este punto se realizó un proceso de selección para determinar qué herramientas y tecnologías ayudarían mucho más en el desarrollo ayudando a su eficiencia y disponibilidad explotando lo que se tiene planteado aun con las limitaciones, para esto durante el estudio se seleccionaron las siguientes tecnologías para darle forma al proyecto, cada una seleccionada por su rendimiento o conveniencia en el desarrollo de la aplicación cada herramienta y tecnología aporta de manera positiva al proyecto lo que lo hace un aliado eficaz a largo plazo, las siguientes tecnologías fueron seleccionadas por su utilidad y eficiencia, fueron evaluadas previamente planteando como se iba realizar la aplicación móvil bajo qué

criterios y que entorno de desarrollo, por su eficiencia y renderización a la hora copilar el código se optó por seleccionar Android estudio y por su rendimiento y flexibilidad se decidió desarrollar el WebService en php para las demás tecnologías y herramientas usadas para la administración de bases de datos, archivos por ftp repositorios editores se optaron por los siguientes que pueden observan en la Tabla 7.

*Tabla 7 - Tecnologías de desarrollo*

Descripción	Aplicación
Android Studio 3.5.1 Build #AI-191.8026.42.35.5900203, built on September 25, 2019 JRE: 1.8.0_202-release-1483-b03 amd64 JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o Windows 10 10.0	
Version: MySQL Community 5.7 en Adelante Formato: MySQL Installer Community 5.7.15 MSI	
GitHub Desktop Versión 2.5.0	 GitHub Desktop

Descripción	Aplicación
PHP 7.0	
FileZilla 3.47 -Desktop	
Mysql Workbench Versión 8.0	

#### 2.4.2 Configuración De Ambientes

Para el desarrollo de la aplicación se realizaron algunas configuraciones en los entornos de desarrollo que se usaron para poder realizar de manera óptima la construcción del aplicativo móvil, como primera medida se configuro el entorno de desarrollo AndroidStudio en su versión 3.5.1.

Una vez iniciado el entorno de desarrollo comenzara a construir y certificar el correcto funcionamiento del IDE cuando el entorno de desarrollo se encuentre listo para poder construir un aplicativo notificara como se muestra en la Figura 14.

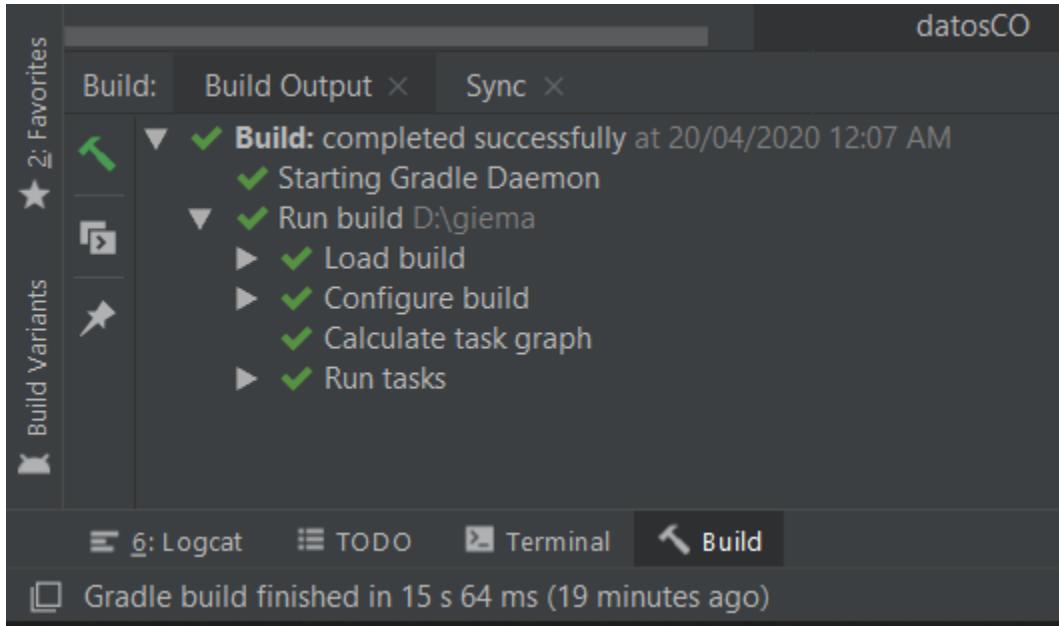


Figura 14 - IDE AndroidStudio

Después de verificar que se todo se encuentre en orden se realizaran las siguientes configuraciones para que la aplicación pida los permisos necesarios para ejecutar la aplicación de manera exitosa, en este caso serán los encargados de pedir acceso a las conexiones de internet del dispositivo y permisos completos de red, para que la aplicación pueda conectarse a internet de manera automática como se muestra en la Figura 15.

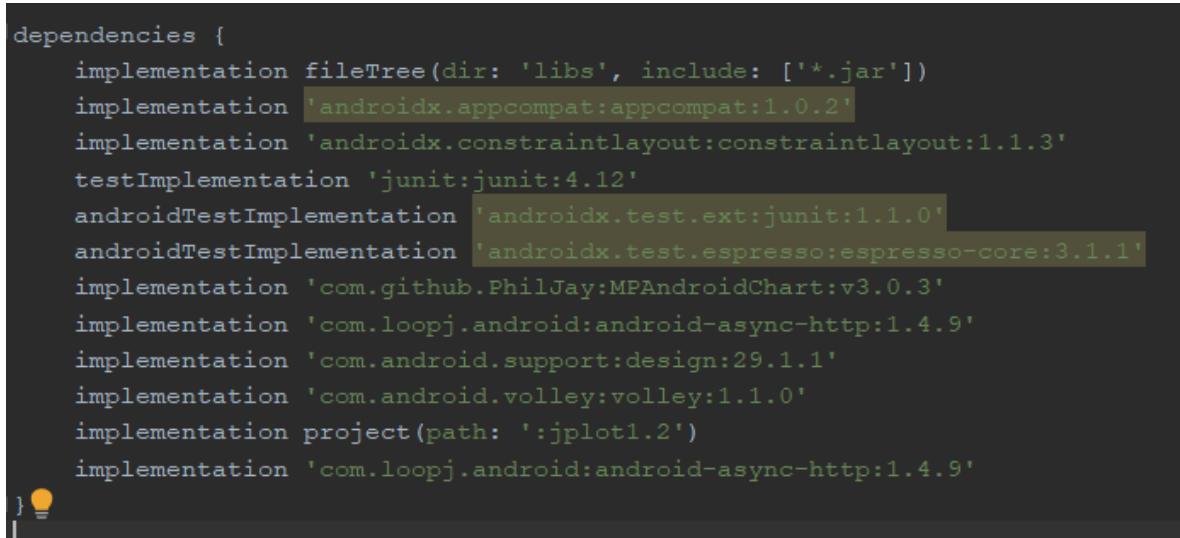
```

<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

Figura 15 - Configuración de permisos de RED

Para el siguiente paso se importaron las librerías necesarias para que el proyecto funcione de manera óptima en este caso se usó las librerías como Volley que es la encargada de administrar las redes de la aplicación, la librería jplot encargada de la parte gráfica y ayudas visuales de la aplicación, la librería

Loopj que se encarga de realizar peticiones HTTP de manera asíncrona, y la librería MPAndroidchart que es la encargada de realizar la graficas en la aplicación, entre otras como se muestra en la Figura 16.



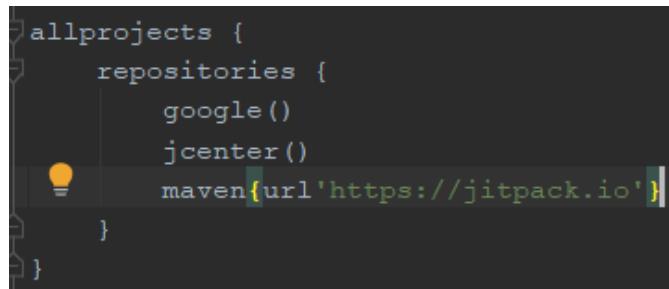
```

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'androidx.appcompat:appcompat:1.0.2'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.0'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
    implementation 'com.github.PhilJay:MPAndroidChart:v3.0.3'
    implementation 'com.loopj.android:android-async-http:1.4.9'
    implementation 'com.android.support:design:29.1.1'
    implementation 'com.android.volley:volley:1.1.0'
    implementation project(path: ':jplot1.2')
    implementation 'com.loopj.android:android-async-http:1.4.9'
}

```

*Figura 16 - Configuración de dependencias AndroidStudio*

Una vez importada las librerías necesarias y aplicando los cambios, se deberá refrescar la estructura de desarrollo y a continuación de deberá importar los componentes para que la librería MPAndroidchart funcione de manera correcta con la siguiente configuración como se muestra en la Figura 17.



```

allprojects {
    repositories {
        google()
        jcenter()
        maven{url'https://jitpack.io'}
    }
}

```

*Figura 17 – Repositorios de propiedades AndroidStudio*

Realizando esto, el entorno de desarrollo está listo para ser usado en cuanto Android, para el desarrollo del WebService no hubo que contar con ninguna configuración previa ya que usando el emulador de servidor XAMPP se puedo emular todo el trabajo de configuración del servicio a la hora de desarrollar

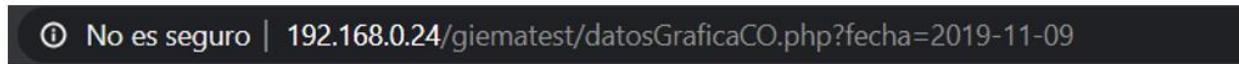
para posteriormente subir el WebService al servidor asignado por la universidad que también ya se encuentra previamente configurado, solo se realizó un trabajo de despliegue para más información consultar el ANEXO D Y E.

Con estas configuraciones y parametrizaciones usando las herramientas seleccionadas se configuraron las herramientas seleccionadas anteriormente lo que permite la aplicación a los estándares utilizados en el desarrollo.

#### 2.4.3 Estándares Utilizados

Para el desarrollo de la aplicación y el webservice se definieron algunos parámetros que se ven reflejados en la codificación en general como, por ejemplo: forma del mensaje, métodos de codificación y orden de las peticiones que se evidenciaran a continuación.

La parametrización de la estructura JSON está definida para formación del paquete de datos que va ser procesada ya sea en un request o un response o ambos, por ejemplo en una consulta el formato JSON se encuentra predefinido por nuestro IDE de desarrollo, cumpliendo esos mismos criterios se desarrolló un WebService flexible que tiene la capacidad de adaptarse a esa misma estructura manejando los datos con flexibilidad y cumpliendo con la petición ejecutada, recordemos es un formato de datos basado en texto que sigue la sintaxis de objeto de JavaScript, para la transición de datos en desarrollos web esto ejemplificado en un consumo directamente por el servicio por localhost como se observa en la Figura 18.



*Figura 18 – JSONarray Estructura del mensaje GIEMA*

En la actualidad podemos encontrar gran variedad de formas arquitectónicas para el desarrollo del servicio de los diferentes sistemas que se encuentran hoy en día en el mundo, después del análisis realizado se escogió la siguen arquitectura de aplicaciones general y creada a mano debido a que no se pueden

desplegar frameworks en el servidor asignado, así que esto fue previamente evaluado antes de iniciar un desarrollo, esta arquitectura se acopla perfectamente a lo que se busca para el presente proyecto y se basa en modelos generales de como parametrizar un servicio, ya que tiene las características para manipular los datos de las diferentes variables correctamente y conectarlos a una API de una manera muy dinámica sus características se centran permitir crear, leer información de manera ágil usando métodos como POST, GET, para las operaciones anteriores necesitan una URL y métodos HTTP para acceder, regresan los datos en formato JSON y pueden retornar códigos de respuesta como 200,400 etc. Lo que convierte a esta arquitectura en un aliado ideal para el desarrollo del presente proyecto, basándose en una estructura general y flexible que puede ser mejorada en la estructura del webService a continuación de un ejemplo de cómo se estructuro el servicio que consume en la función Registro en Figura 19.

```
<?php
if ($_SERVER['REQUEST_METHOD'] == 'POST'){

    $username = $_POST['username'];
    $correo = $_POST['correo'];
    $clave = $_POST['clave'];
    $nombres = $_POST['nombres'];
    $apellidos = $_POST['apellidos'];

    $clave = password_hash($clave, PASSWORD_DEFAULT);

    require_once 'connect.php';

    $sql = "INSERT INTO usuario (username, clave, nombres, apellidos,
    if ( mysqli_query($conn, $sql) ) {
        $result["success"] = "1";
        $result["message"] = "success";

        echo json_encode($result);
        mysqli_close($conn);
    } else {

        $result["success"] = "0";
        $result["message"] = "error";

        echo json_encode($result);
        mysqli_close($conn);
    }
}
?>
```

*Figura 19 - Servicio Registro*

Teniendo claro lo anterior ahora se continuará explicando el método de nomenclatura usado al codificar, para el presente proyecto se adoptó la nomenclatura Lower Camel Case muy usada en lenguajes

referente aplicaciones web, es una nomenclatura organizada y clara, lo que dio un gran manejo a la hora de codificar métodos y clases, teniendo claro las estructuras anteriormente desarrolladas, se continuara con la explicación de cómo se desarrolló la app y bajo qué criterios.

El desarrollo de aplicaciones móviles y su arquitectura es muy variada, en la actualidad existen ciertas arquitecturas que dominan su preferencia en uso, debido a que cuentan con una estructura más amigable de usar y cuidan el concepto de buenas prácticas en la programación, para el presente proyecto se determinó que esta arquitectura es la más adecuada debido a que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos donde el modelo accede a la capa de datos, el controlador recibe los eventos de la entrada y la vista recibe los datos del modelo y los muestra al usuario, es la más indicada, trabajando junto con el IDE de desarrollo que se seleccionó para la creación de la aplicación, Android Studio se adapta esta arquitectura de desarrollo para poder seguir con las buenas prácticas de la programación, siguiendo los lineamientos dados por el IDE de desarrollo se puede concluir que es la mejor opción para interactuar con los datos debido a que el IDE Android Studio 3.5.1 trabaja con Layouts para el apartado grafico que se codifican en XML y se trabaja su backen en JavaClass ofrece una herramienta completa para el desarrollo de la aplicación móvil con buenas prácticas, cada sensor se programó individualmente ya que cada sensor debe tener un hilo individual asignado, razón por la cual se programó de esta manera fue priorizando la estabilidad de la aplicación para optimizar recursos y gracias a esto para que los celulares de más baja gama tengan la oportunidad de acceder a la aplicación sin afectar el rendimiento de sus dispositivos.

Siguiendo la arquitectura de programación que ofrece el IDE Android Studio, se desarrolló el aplicativo separando cada sección en clases, debido a que en Android se maneja algo que se llama Activity que son las pantallas que deben ser programadas individualmente para su funcionamiento y navegación, se trató de aprovechar al máximo la arquitectura propuesta en el WebService para que todos los métodos y acciones quedarán dentro de los controller y los demás métodos que se crearon adicionalmente para ejecutar

la acción como validación registro y consulta, para el desarrollo de la aplicación se usó el concepto de llamar todo desde el webService para aliviar carga tanto en el servidor como en el aplicativo donde se evaluó y se desarrolló sobre un sistema Android.5.5 en adelante, es decir que los celulares con ese sistema operativo Android soportaran la carga de la aplicación, dicho esto la aplicación funciona por medio de clases que será la pantalla individual ligada a un Activity que es el apartado gráfico, dentro de cada clase se desarrolla el backen usando métodos para consumir el servicio desde el webservice, solicitamos el tipo de servicio que deseamos realizar y donde por ejemplo creando una clase stringRequest llamamos un método POST a una URL, este proceso es como ordenarle al servidor enviar una petición POST y el servicio que se solicita está en esa dirección lo que iniciara un proceso donde hay una respuesta y se crear un objeto JSON donde se almacena la cadena JSON que devuelve el servidor para almacenar los datos, ya en este punto los datos se encuentran en la cadena JSON en la aplicación solo por el momento de realizar la petición, ya después de finalizado este proceso se rompe lo que no le provoca carga a la aplicación ni al servidor, un ejemplo de cómo se consume el servicio es el siguiente, se codifica un método de comunicación que va enviar otro método con las instrucciones de cómo manejar los datos, esto se puede evidenciar en la

Figura 20.

```
private void llenarDatos() {
    String url = "http://software.ingusb.com/giema/giematest/datosGraficaTemperatura.php?fecha="+etFecha.getText().toString();
    client.get(url, new AsyncHttpResponseHandler() {
        //Creamos un metodo parametrizando el response
        @Override
        public void onSuccess(int statusCode, Header[] headers, byte[] responseBody) {
            if (statusCode == 200) {
                //Cargamos el metodo que contiene la logica que administra los datos meteorologicos
                cargarDatos(new String(responseBody));
            }
        }

        @Override
        public void onFailure(int statusCode, Header[] headers, byte[] responseBody, Throwable error) {
        }
    });
}
```

Figura 20 - Ejemplo Consumo de servicio

Con esto se concluyó el desarrollo de la aplicación y el webservice usando todos los componentes que se tenían a la mano, gracias a la organización de las tareas puntuales de desarrollo junto al asesor del proyecto el Ing. Andres Sanchez, se pudo realizar de manera organizada y optima este avance significativo, que dieron como resultado nuestros productos finales, para más información sobre el desarrollo consultar el ANEXO D.

#### **2.4.4 Productos Resultantes**

Una vez terminado el desarrollo se tuvo como resultado los siguientes componentes, por el lado de la aplicación GIEMA, que se encuentra disponible en el link a continuación: <http://software.ingusb.com/giema/descargaGiema.php>, se podrá descargar la aplicación haciendo clic en “Descargar App”.

Por su parte todo el proyecto GIEMA está disponible en el repositorio de github donde se encontrar el webservice y el modelo de datos, junto a la documentación de despliegue tanto del modelo de datos en el servidor como el uso de la app en el siguiente link: a continuación: <https://github.com/aledoc72/giema>, para más información consultar el ANEXO E.

Con esto concluye el segmento de desarrollo, ahora se procederá a evaluar el protocolo de pruebas realizado para certificar la funcionalidad del proyecto cumpliendo con las metas pactadas.

#### **2.5 Pruebas**

Para esta sección se explicaran como se generaron los datos meteorológicos ya que no se cuenta con la parte física y los métodos de criterio de evaluación para el análisis de pruebas de GIEMA el primero será un protocolo de pruebas funcionales y de integración que medirán la funcionalidad y fluidez de GIEMA certificando su correcto funcionamiento dentro de sus capacidades y el segundo será una evaluación TAM realizada por Ingenieros los cuales probaron la app y dieron a GIEMA su criterio en cuanto al diseño

funcionalidad y usabilidad en base a las pruebas que ellos mismos aplicaron a la app en base a la evaluación TAM, teniendo claro se iniciara los criterios funcionales y de integración.

### 2.5.1. Pruebas Funcional E Integración

Debido a la ausencia de la parte física del proyecto se realizó un proceso de simulación en los datos para procesar datos reales y poder realizar diferentes procesos con ellos, tomando el concepto de que se tienen 4 sensores donde cada uno mide una variable distinta, se debe aplicar diferentes fórmulas para cada sensor, ya que cada uno toma un dato y una nomenclatura diferente, usando el generador de datos Mockaroo y usando las siguientes formulas se logró generar datos válidos para las pruebas que pueden sacar los mismos sensores.

Comenzando por el sensor de CO, se tomó la siguiente ecuación:

$$\text{partes por millon (ppm)} = \frac{\text{volumen de la sustancia analizada}}{\text{volumen total}} \cdot 10^6 \rightarrow (1)$$

Es una unidad empleada para la medición de presencia de elementos en pequeñas cantidades (trazas), debido a que el sensor MQ-7 toma la medida final ppm se aplicó este proceso para emular datos para poder usarlos dentro del aplicativo móvil evidenciadas en la Figura 21.

	+ Opciones	← →		id	fecha	valor	sensor
	Editar	Copiar	Borrar				
				1	2019-11-10	11	10
				2	2019-11-09	11	10
				3	2019-11-09	8	10
				4	2019-11-10	9	10
				5	2019-11-09	11	10
				7	2019-11-09	9	10
				8	2019-11-09	8	10
				9	2019-11-09	11	10
				10	2019-11-09	9	10
				11	2019-11-09	8	10
				12	2019-11-09	11	10
				13	2019-11-09	8	10
				15	2019-11-10	10	10
				16	2019-11-09	11	10
				17	2019-11-09	8	10
				18	2019-11-10	10	10

Figura 21 – Datos generados a partir de la emulación

Siguiendo este mismo proceso se calculó la humedad:

$$w = \left( \frac{Ww}{Ws} \right) \times 100 \rightarrow (2)$$

donde:

$w$  = es el contenido de humedad expresado en %

$Ww$  = peso del agua existente en la masa del suelo

$Ws$  = peso de las partículas sólidas

Material particulado:

$$\text{partes por millón (ppm)} = \frac{\text{peso de la sustancia analizada}}{\text{peso total}} \cdot 10^6 \rightarrow (3)$$

Y finalmente temperatura la cual se tomó de la media de Bogotá ofrecida por los diferentes canales de información que ofrecen estos datos.

Con este proceso se sacaron diferentes datos de las diferentes variables filtrando y haciendo una media de los datos donde se siguió el proceso por medio de fórmulas para usar datos limpios cuando el aplicativo los llame por medio del usuario por medio de la herramienta Mockaroo que me permite colocar ciertas funciones para crear una media de los datos sacados.

$$\text{Media}(X) = \bar{x} = \frac{\sum_{i=1}^N X_i}{N} \rightarrow (4)$$

y posteriormente exportados a un archivo SQL demostrado en la Figura 22.

Nombre del campo	Tipo	Opciones
id	Row Number	blanco: 0 % <input type="button" value="fx"/> <input type="button" value="x"/>
fecha	Date	11/09/2019 a 11/11/2019 en aaaa-mm-dd blanco: 0 % <input type="button" value="fx"/> <input type="button" value="x"/>
valor	Number	min: 0 max: 1 decimales: 0 blanco: 0 % <input type="button" value="fx"/> <input type="button" value="x"/>
sensor	Number	min: 12 max: 18 decimales: 0 blanco: 0 % <input type="button" value="fx"/> <input type="button" value="x"/>
<input type="button" value="Agregar otro campo"/>		
# Filas:	1000	Formato: SQL
Nombre de la tabla:		mensaje
<input type="checkbox"/> incluir crear tabla		

Figura 22 - Generador datos Mockaroo

Teniendo claro cómo se generaron los datos a partir de un generador en los cuales se parametrizaron las ecuaciones anteriores según la parametrización de los sensores a medir, se continuará explicando cómo se realizaron las pruebas funcionales y de integración a partir de un set de pruebas el cual se planteó para probar la funcionalidad y el rendimiento del aplicación, donde cada uno de los ítems a evaluar dará respuesta al comportamiento y si la aplicación en su nivel funcional es óptimo para ser usada, los factores a evaluar se podrán observar en la Tabla 8.

*Tabla 8 - Casos de prueba*

Nº Caso de Prueba	Nombre	Descripción
CP-001	Login	El usuario debe loguearse validando los datos que se encuentran en la base de datos Usuario y contraseña
CP-002	Registro	El usuario debe poder Registrarse en la App para que sus datos puedan ser usados en el login
CP-003	Consulta de nodo	El nodo debe el cual se encuentra el usuario actualmente debe ser llamado de la base de datos para tener trazabilidad del proceso
CP-004	Consulta de datos Sensor CO	Los datos meteorológicos CO deben ser llamados de la base de datos y mostrase en la sección indicada
CP-005	Consulta de datos Sensor Humedad	Los datos meteorológicos Humedad deben ser llamados de la base de datos y mostrase en la sección indicada
CP-006	Consulta de datos Sensor Temperatura	Los datos meteorológicos Temperatura deben ser llamados de la base de datos y mostrase en la sección indicada

Nº Caso de Prueba	Nombre	Descripción
CP-007	Consulta de datos Sensor Material Particulado	Los datos meteorológicos MP deben ser llamados de la base de datos y mostrarse en la sección indicada
CP-008	Generación de grafica CO	Se debe generar una gráfica de los datos CO
CP-009	Generación de grafica Humedad	Se debe generar una gráfica de los datos Humedad
CP-010	Generación de grafica Temperatura	Se debe generar una gráfica de los datos Temperatura
CP-011	Generación de grafica Material Particulado	Se debe generar una gráfica de los datos MP
CP-012	Los datos deben filtrarse por fechas	Los datos consultados se podrán consultar por fecha

Bajo estos criterios de evaluación se realizaron las pruebas correspondientes donde todas salieron con un resultado exitoso lo cual nos arrojó una correcta funcionalidad en cuanto a lo que se esperaba de la aplicación, donde cada caso de prueba planteado funciona de manera correcta, se priorizo las pruebas de manejo consulta y filtrado de datos por sensores y tráfico de datos meteorológicos ya que esto es el centro del proyecto y el manejo de estos datos es esencial para cumplir con los objetivos planteados para más información consultar el ANEXO F.

Cada caso de prueba se ejecutó con normalidad siguiendo los parámetros e indicaciones señalados para hacer un buen proceso de evaluación en cada ítem de prueba, con el objetivo de probar su rendimiento y funcionalidad, el resultado de estas pruebas fueron satisfactorios lo que arrojo datos para concluir que la aplicación a nivel de login realiza un proceso de validación normal enviando los parámetros y validando la identidades del usuario, el registro enviando los datos del usuario e insertándolos en la base de datos encryptando la contraseña, la correcta visualización de los datos meteorológicos separados por sensores para mostrar la información correspondiente, donde estos datos en una consulta por fecha también tienen la posibilidad de ser graficados por medio de la app teniendo en cuenta eso se concluye que las pruebas funcionales ejercidas sobre la app se realizaron de manera exitosa lo que da un gran campo a la continuación y mejora de la app propia para más información consultar el ANEXO F, para mayor veracidad en esta evaluación sobre GIEMA como tal se realizaron una serie de pruebas mediante una evaluación TAM a ingenieros que se verá a continuación.

### **2.5.2 Pruebas De Aceptación GIEMA**

Una opinión por parte de un experto es un punto aparte para tener una perspectiva más amplia de cómo se debe comportar o como debe funcionar la aplicación, en este punto junto a la ayuda de 4 ingenieros aplicamos la evaluación TAM a GIEMA con el propósito de tener una visión más amplia de lo que es y puede llegar a ser la aplicación, a continuación, se explicara como se aplicó la prueba y que criterios fueron evaluados junto a la ayuda de cuatro ingenieros los cuales presentare a continuación:

- Ingeniero de sistemas con experiencia de 5 años en gerencia de proyectos, aplicaciones móviles, Alejandro Poveda Ingeniero de sistemas – especialista en gerencia de proyectos – Evl 1.
- Ingeniera de Telecomunicaciones con dos años de experiencia en sistemas de conexiones inalámbricas, Natalia Torres Ingeniero Telecomunicaciones – Evl 2.

- Ingeniero de Telecomunicaciones con 3 años de experiencias en modelos IoT y conexiones vía satélite, Pablo Nonsoque Ingeniero de Telecomunicaciones – Evl 3.
- Ingeniero Mecánico con 2 años de experiencia en la automatización de máquinas y e ingeniería de materiales, Sebastián Florez Ingeniero Mecánico – Maestría en ingeniería materiales – Evl 4.

Con la ayuda de estos cuatro ingenieros se evaluaron tres temas relevantes el diseño, la funcionalidad y la usabilidad, estos componentes se ingresaron en una evaluación en la cual los ingenieros probaron la app y dieron su concepto profesional según los ítems estructurados en la prueba.

En base a estos resultados donde se evaluaron diferentes aspectos, se toma como referencia de modelo de mejora de la aplicación para una última versión antes de ser liberada.

Para esta sección se estructuro una evaluación TAM en base a la aplicación en la cual se viera reflejada los diferentes componentes y funcionalidades de la aplicación, así como su estructura su diseño, la usabilidad y funcionalidades, esta sección los ingenieros que ayudaron con la retroalimentación evaluación y observaciones dadas al probar esta aplicación para crecimiento técnico del proyecto y retro alimentación y crecimiento personal como profesional a su autor para más información consultar el ANEXO F donde se encuentra toda la información detallada de esta prueba.

Para la esta evaluación técnica realizada por profesionales el primer paso a evaluar fue el diseño, para esta sección se plantearon los siguientes ítems como criterio de evaluación de app para que puedan ser calificados como se puede observar en la Tabla 9.

*Tabla 9 - Criterio de Evaluación TAM Diseño*

<b>Diseño</b>	El texto usado en la aplicación es difícil de leer
	La paleta de colores usada en el prototipo tiene un contraste inadecuado
	Las imágenes de fondo no opacan la visibilidad de las pantallas
	Los datos se visualizan de manera armoniosa
	El color de las gráficas diferencia cada dato Señalado
	El diseño del prototipo no se ajusta a la funcionalidad de este
	La información transmitida con código de color es congruente

En esta sección se observa todo lo referente al diseño de la aplicación como modelo de evaluación en este caso resaltando como se estructuro su diseño y la experiencia de usuario al ser manejada.

El segundo aspecto para evaluar como criterio de evaluación fue la usabilidad, que tal iba al usuario usando la aplicación y su comodidad por medio de la experiencia de usuario manejo de excepciones y comportamiento de app a nivel técnico como se puede observar en la Tabla 10.

*Tabla 10 - Criterio de Evaluación TAM Usabilidad*

<b>Usabilidad</b>	Visibilidad del estado del sistema (el prototipo no da retroalimentación de las acciones realizadas por el usuario)
	Relación entre el sistema y el mundo real (el prototipo no presenta la información de manera sencilla y entendible)
	Control y libertad del usuario (el prototipo no presenta las opciones de salida de funciones o no se puede regresar)
	Consistencia y estándares (los iconos y frases del prototipo realizan diferentes acciones)
	Prevención de errores (el prototipo no es claro en los pasos al realizar una acción)
	Reconocimiento antes que recuerdo (el prototipo no presenta un acceso fácil a las acciones y funcionalidades)
	Flexibilidad y eficiencia de uso (el prototipo no presenta funciones específicas para cada tipo de usuario)
	Estética y diseño minimalista (el prototipo esta sobrecargado de información irrelevante y componentes innecesario o redundantes, no es minimalista)
	Ayudar a los usuarios a reconocer (el prototipo no informa los errores y colapsa)
	La generación de graficas es rápida (el tiempo respuesta de la app para este proceso no tiene contratiempos)
	El tiempo de respuesta de la aplicación es rápida
	La aplicación realiza peticiones indicadas en las opciones establecidas

Siguiente y último aspecto fue la funcionalidad, donde se evaluó que ofrecía cada componente y como podía ser integrado a nivel funcional creando una utilidad al usuario, para este paso los ingenieros probaron las opciones que tenían disponibles en la aplicación y realizaron sus observaciones como se puede observar en la Tabla 11.

*Tabla 11 - Criterio de Evaluación TAM Funcionalidad*

Funcionalidad	La aplicación realiza un proceso de validación (Login) para ingresar
	La aplicación filtra los datos por fecha
	La aplicación genera un reporte por fecha de los datos consultados
	La aplicación grafica los datos de los sensores meteorológicos
	La aplicación genera una consulta individual por cada sensor disponible
	La aplicación muestra información sobre el nodo consultado

Donde los criterios de evaluación se basan en la siguiente el siguiente rango como se puede ver en la Tabla 12.

*Tabla 12 - Calificación, criterio de evaluación TAM*

5. Si considera que el aspecto es excelente
4. Si considera que el aspecto es bueno
3. Si considera que el aspecto es aceptable
2. Si considera que el aspecto regular
1. Si considera que el aspecto es malo

Arrojando datos positivos sobre la evaluación de la mano de un experto, GIEMA se puntuó con una calificación sobresaliente en cuanto a su funcionalidad diseño y usabilidad, cumpliendo sus objetivos y cumpliendo las funciones por a cuál fue creado este proyecto, certificando la app gracias a la opinión, prueba y criterio de ingenieros expertos.

Con esto concluye el segmento de pruebas y se cierra arrojando resultados positivos donde GIEMA fue probado para realizar su labor y fue calificado bajo diferentes pruebas, pruebas que supero satisfactoriamente, aunque todavía puede crecer bastante según las recomendaciones de los evaluadores, la aplicación cumple con sus funciones de manera correcta y funciona de manera fluida, por lo cual se resaltó el potencial de esta app.

Después de realizar todo este proceso de investigación y análisis, para la construcción de un modelo de la cual se basó una arquitectura, donde sus componentes se definieron a partir de un estudio de requerimientos que uniéndolos formaron casos de uso, para realizar un diseño a profundidad donde cada

componente fue demarcado de manera cuidadosa, para que el proceso de desarrollo fuera rápido y eficiente y este desarrollo pudiera ser probado y certificado así se cierra este capítulo con resultados satisfactorios donde GIEMA supero cada una de las fases propuestas con resultados exitosos y con una visión amplia y profunda del futuro pensando en un mejor manejo de datos meteorológicos para un beneficio común, el de informar.

## **Capítulo 3**

A partir de todo el desarrollo del proyecto GIEMA en el cual se realizaron varias pruebas y ajustes, un proceso de investigación exhaustivo que concluyo en un desarrollo fluido y eficiente al final de todo este proceso GIEMA arrojo unos datos que deben ser analizados de manera cuidadosa, datos que arrojo gracias a las diferentes pruebas y procesos de análisis a los cuales se sometió el proyecto a continuación se presentara esos resultados que arrojo todo este proceso que llevo a GIEMA hasta este punto

### **3.1 Resultado De Las Pruebas**

Para evaluar los resultados de las pruebas se debe entender a qué punto iban las pruebas realizadas, dicho en el capítulo anterior se explicó cómo y con qué criterios se realizaron las pruebas en este punto se resaltaran los resultados obtenidos de las pruebas anteriores, primero se comenzara analizando las pruebas funcionales, como se había resaltado en el capítulo anterior se plantearon doce pruebas funcionales con el propósito de medir el desempeño de la aplicación cada segmento se evaluó individualmente y se le realizo seguimiento al proceso por las capaz que involucraba, resaltando la eficiencia de la aplicación, de estas pruebas doce fueron probadas con resultados satisfactorios, resaltando primero la funcionalidades más importantes que el tratamiento de datos para un fin informativo, se evaluó cada sensor de manera individual CO, HUMEDAD, TEMPERATURA, MP, recordemos que cada sensor arroja datos diferentes así que estos componentes en el sistemas involucran todo el proyecto, al evaluar el comportamiento y proceso de cómo se manejaban los datos meteorológicos, arrojo resultados satisfactorios donde cada sensor respondió al

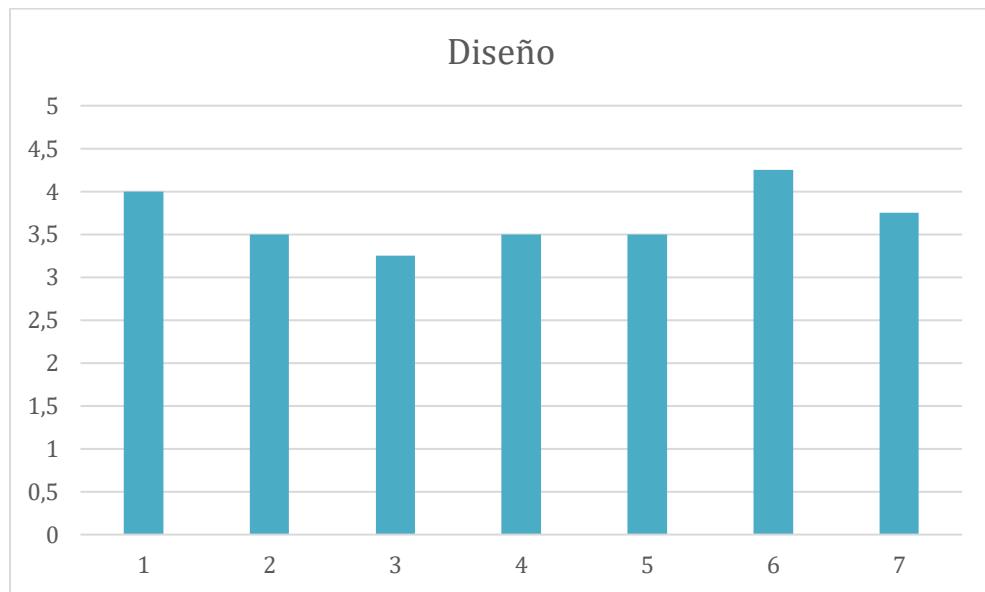
100% con la petición realizada, debido a que desde el momento en el que se realiza la petición esta pasa por los diferentes componentes y gracias a la arquitectura los datos son procesados de manera correcta al evaluar el sensor respondió de manera correcta ejecutando varias consultas, la aplicación no fuerza al cierre ya que se administra de manera correcta estos componentes dándole camino fluido a los datos meteorológicos y dando como conclusión un resultado exitoso en las pruebas de manejo de datos, después de realizar este proceso se realizó el mismo método de evaluación individualmente a las gráficas por sensor meteorológico donde cada sensor fue evaluado por separado ya que por cada uno pasa un dato meteorológico distinto en la cual respondió de manera correcta, la aplicación responde normalizando con un dato que se encuentra normalizado y alojado en la base de datos por cada hora y cada dato es graficado para tener una certeza de cómo se encuentra el ambiente en ese momento bajo el criterio de ese sensor, ya que las pruebas se realizaron con datos en bloque aun así la prueba pudo ser ejecutada de manera correcta arrojando resultados satisfactorios respondiendo en un 100% sobre el requerimiento planteado.

Luego del proceso anterior se reforzaron estos criterios con la prueba TAM que se menciona y se explica en el capítulo anterior, bajo estos los criterios de evaluación planteados ahí arrojaron los siguientes datos separados en tres secciones diseño, usabilidad y funcionalidad.

Comenzando por el diseño, según los criterios de evaluación se saca una media en base a las respuestas de los ingenieros que aplicaron la evaluación, para evaluar en este caso la parte de diseño, cada criterio está marcado con un número para identificar la media de cada dato en la evaluación TAM como se muestra en la Tabla 13.

*Tabla 13 - Diseño criterio TAM*

1	El texto usado en la aplicación es difícil de leer
2	La paleta de colores usada en el prototipo tiene un contraste inadecuado
3	Las imágenes de fondo no opacan la visibilidad de las pantallas
4	Los datos se visualizan de manera armoniosa
5	El color de las gráficas diferencia cada dato Señalado
6	El diseño del prototipo no se ajusta a la funcionalidad del mismo
7	La información transmitida con código de color es congruente



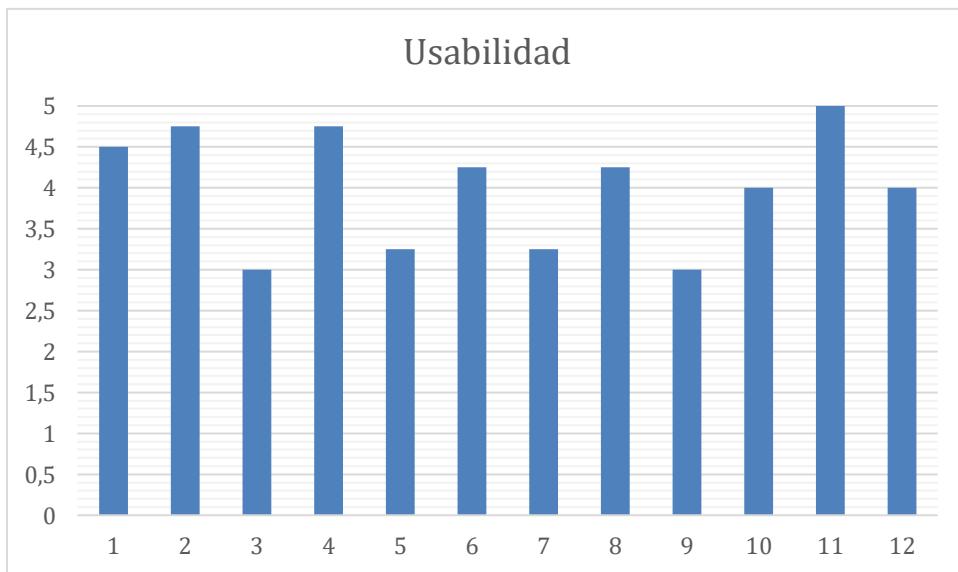
*Grafica 1 - Criterio de Evaluación Diseño*

Como se muestra en la Grafica 1, los resultados para diseño fueron satisfactorios en la gran mayoría donde la distribución de la aplicación y la forma en la que se transmite la aplicación fueron gratamente evaluadas y aceptadas por los ingenieros evaluadores, de parte de los aspectos recomendados y observaciones para futuros cambios.

Para la usabilidad los criterios de evaluación planteados aquí son a todas esas características a la que el usuario estará conectado de manera continua durante toda su experiencia en la app, ya que la app de GIEMA es una app netamente concentrada en la información de datos meteorológicos, fue un gran enfoque concentrarnos mucho en la usabilidad y los eventos que se pueden presentar al usar la app de manera continua, manejo de errores, en si es la experiencia de usuario dentro de app, para que sea lo más cómoda posible, se evaluaron los siguientes criterios de evaluación como se puede observar en la Tabla 14.

Tabla 14 - Usabilidad Criterio TAM

1	Visibilidad del estado del sistema (el prototipo no da retroalimentación de las acciones realizadas por el usuario)
2	Relación entre el sistema y el mundo real (el prototipo no presenta la información de manera sencilla y entendible)
3	Control y libertad del usuario (el prototipo no presenta las opciones de salida de funciones o no se puede regresar)
4	Consistencia y estándares (los iconos y frases del prototipo realizan diferentes acciones)
5	Prevención de errores (el prototipo no es claro en los pasos al realizar una acción)
6	Reconocimiento antes que recuerdo (el prototipo no presenta un acceso fácil a las acciones y funcionalidades)
7	Flexibilidad y eficiencia de uso (el prototipo no presenta funciones específicas para cada tipo de usuario)
8	Estética y diseño minimalista (el prototipo esta sobrecargado de información irrelevante y componentes innecesario o redundantes, no es minimalista)
9	Ayudar a los usuarios a reconocer (el prototipo no informa los errores y colapsa)
10	La generación de graficas es rápida (el tiempo respuesta de la app para este proceso no tiene contratiempos)
11	El tiempo de respuesta de la aplicación es rápida
12	La aplicación realiza peticiones indicadas en las opciones establecidas



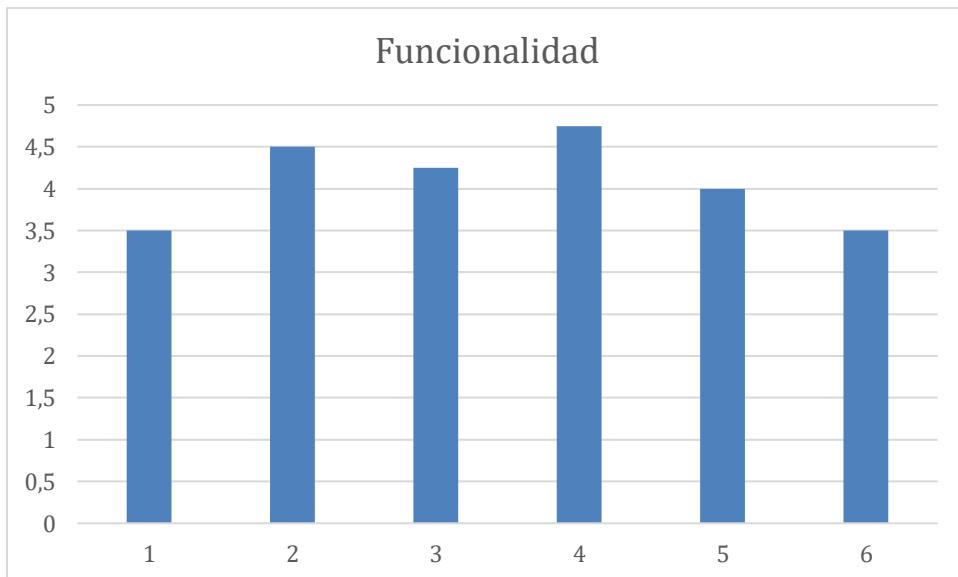
Grafica 2 - Usabilidad Criterio TAM

Como se muestra en la Grafica 2, aunque el desempeño de la app fue evaluado de manera sobresaliente y se resaltó de gran manera su rapidez y facilidad al usar, algunos aspectos necesitan un especial trabajo dentro de los criterios de evaluación.

En esta sección llamada funcionalidad se evaluó la parte funcional de app como respondía a cada función y como se desempeñó la app bajo estas pruebas aplicadas por nuestros evaluadores como se puede ver en la Tabla 15.

*Tabla 15 - Funcionalidad Criterio TAM*

1	La aplicación realiza un proceso de validación (Login) para ingresar
2	La aplicación filtra los datos por fecha
3	La aplicación genera un reporte por fecha de los datos consultados
4	La aplicación grafica los datos de los sensores meteorológicos
5	La aplicación genera una consulta individual por cada sensor disponible
6	La aplicación muestra información sobre el nodo consultado



*Grafica 3 - Funcionalidad Criterio TAM*

Según la Grafica 3, los criterios funcionales fueron muy bien acogidos ya que, aunque tuvimos dos aspectos regulares fueron los aspectos que implementaron más por regulación de app que por pedido de requerimiento con el propósito esta app siga creciendo en el futuro.

Con esto concluye el resultado de las pruebas en las siguientes secciones se analizará los resultados recogidos durante todo el proceso de evaluación.

### 3.2 Análisis De Resultados

Para esta segmento llamado análisis de resultados se pondrá en evidencia la conclusión de resultados arrojados gracias a la investigación pruebas y análisis de las diferentes pruebas a las que fue sometida GIEMA, con el propósito de dar un análisis correcto y entendible, todo este proceso se va basar en la pruebas realizadas dando la reflexión de cada proceso de prueba, iniciando primero con las pruebas de funcionalidad e integración, según las pruebas aplicadas y bajo las condiciones que se aplicaron se tiene los siguientes resultados para estas pruebas.

- App se comporta de manera fluida y responde bien a los tiempos estándares aun cuando depende del servidor.
- Los datos generados por filtro se generan solo día consultado y cada consulta se ejecuta por separado para no saturar la app ni el servidor.
- La aplicación grafica los datos de manera correcta para su visualización, para la generación de la gráfica también se debe filtrar por fecha los datos que se desea consultar ya que como son bastantes datos se necesita un parámetro para filtrar toda esta información.
- El login realiza su funcionalidad correctamente, aunque es un proceso sencillo la administración de datos no estaba contemplada en esta app por lo que solo se probó que se realizara la validación correctamente.
- El registro es un proceso normal dentro de los estándares de lo que debe ser, ya que encripta la contraseña y pide datos básicos, nada sensible.

Se evidencia un buen comportamiento, realiza las funciones permitidas en la aplicación y cumple con los requerimientos planteados de esta manera y bajo estos criterios de análisis dependiendo de los resultados de pruebas se puede decir con seguridad que GIEMA cumplió en su totalidad con el 100% de las pruebas realizadas, y para apoyar este análisis bajo los resultados que arrojo las pruebas TAM de la mano de las

opiniones de ingenieros expertos donde evaluaron cada componente y el resultado del análisis es presentado a continuación en la Tabla 16.

*Tabla 16 - Análisis de resultados TAM*

Criterios	Evl 1	Evl 2	Evl 3	Evl 4	Promedio del Criterio
<b>Diseño</b>	3,86	3,71	3,86	3,71	3,79
<b>Usabilidad</b>	4,00	3,92	4,08	4,00	4,00
<b>Funcionalidad</b>	4,17	3,83	4,00	4,33	4,08
<b>Promedio evaluador</b>	4,01	3,82	3,98	4,02	4,01

En base al análisis anterior como se muestra en la tabla 14 se concluye que la aplicación cumple con los requerimientos planteados y aunque hay elementos agregados que se integraron para fines de que la experiencia de usuario sea más grata se espera que en el futuro esta aplicación siga creciendo, el objetivo de aplicar esta evaluación a Ingenieros reales fue el dar un espacio de opinión técnica y dar un panorama más amplio de lo que es GIEMA y cómo puede mejorar en un futuro, dicho esto también se desea concluir a nivel general sobre su funcionamiento diseño y usabilidad en base a las observaciones que dieron los evaluadores los siguientes aspectos.

- Mejora en el apartado gráfico, ya que el objetivo de GIEMA es la practicidad para la lectura y manejo de datos de manera informativa un punto que se puede mejorar a futuro es la reinención de un nuevo apartado grafico esto con el fin de hacer la aplicación más agradable y con esto aumentar la experiencia de usuario notablemente, entre estas están las cajas de texto forma de la fuente, imágenes de fondo botones y graficas.
- Para el criterio de usabilidad solo se tocaron puntos como más orientación en la información en la aplicación y mejora e integración en botones de return dentro de la aplicación, además de manejo de errores y excepciones más claras, pueden ayudar mucho a pulir la aplicación en el futuro.
- En cuanto a la funcionalidad, aunque fue uno de los elementos mejor valorados, las observaciones estuvieron más orientadas a temas como el login o visualización del nodo, temas que se

desarrollaron como un adicional y fueron bien recibidas esto con el propósito de que en el futuro se siga integrando más funcionalidades respecto a estos componentes adicionales como manejo de información personal y del nodo, dicho esto podemos concluir que la funcionalidad en lo que reafirma sus objetivos están completos y las observaciones solo van dirigidas a futuros ítems de mejora en el futuro.

Se concluye que GIEMA cumple con las funcionalidades y requerimientos planteados durante su diseño y gracias a las observaciones de los evaluadores que dan un panorama más amplio respecto a lo técnico de cómo puede crecer esta app en el futuro y ser integrada en temas más grandes, ya que fue bien recibida su diseño y su propósito para que en el futuro sea una app en constante mejora y pueda ser integrada a trabajos de datos ambientales más pesados.

Con este análisis concluye el capítulo 3 en el cual se evidencia los resultados arrojados durante todo el proyecto gracias al cumplimiento y superación de cada fase lo que concluye en resultados.

## CONCLUSIONES

Para concluir con el presente proyecto se resaltarán las conclusiones como resultado del desarrollo y trabajo de análisis durante la realización de este, donde durante todo la ejecución y desarrollo de GIEMA fue un proceso de aprendizaje en sí y un reto como tal, ya que GIEMA nace de la integración de diferentes conceptos y tecnologías llevados a un proyecto de tesis para su desarrollo con unos objetivos claros, fomentando al buen uso de las variables meteorológicas de manera práctica y sencilla con un objetivo informativo, dicho esto se concluye que se cumplió con todos los objetivos planteados de manera eficiente aun con las limitaciones planteadas, se pudo desarrollar el proyecto de manera óptima dando como resultado GIEMA, el proyecto en si fue un buen punto de aprendizaje ya que al construir de cero un modelo se fortalecieron muchos aspectos a nivel técnico y se aprendiendo muchos más como por ejemplo el fortalecimiento de levantamiento de requerimientos y diseño de arquitecturas tecnológicas, metodologías de desarrollo y como aplicarlas, modelo de datos aplicado a gran escala y pensado en un manejo masivo dentro de las limitaciones, este proyecto enseño al autor el manejo de una arquitectura real aplicada a un modelo y como se compone dando como resultado un proyecto, una visión y un punto aparte para que ese sueño siga creciendo llamado GIEMA, a continuación se resaltaron las conclusiones más relevantes del proyecto.

1. Con esta capacidad de reutilizar los datos meteorológicos y darles nuevos propósitos, podemos continuar con su análisis y transformarlos en nuevas formas de producir una información valiosa que permitan ahorrar tiempo y dinero e, incluso, salvar vidas.
2. Por medio de una aplicación móvil es mucho más sencillo poder gestionar los datos meteorológicos, debido a que su practicidad, ya que ofrece mayor comodidad a los usuarios acompañado de la disponibilidad de tener al alcance una consulta la información meteorológica al alcance de su mano.

3. Un modelo IoT planteado de manera correcta junto con una arquitectura sólida y nivelada es la clave para un flujo de datos organizado en este caso, las variables meteorológicas, bajo una buena administración técnica el resultado será un rendimiento más alto.
4. Un modelo de datos optimo permite un mejor procesamiento ya que no genera tanta carga para las maquinas que procesan los datos meteorológicos, lo que permite una mejor fluidez en los procesos y peticiones dando como resultado un proceso más ágil que se pueden hacer con los datos a nivel.

## RECOMENDACIONES

Para darle continuación a este proyecto ya que su potencial es muy alto, el manejo de datos es muy grande y se cuenta con una arquitectura flexible por la cual se podrían modificar componentes para otros fines, se realizan las siguientes recomendaciones, también de la mano de los resultados de evaluación TAM en la cual ingenieros expertos dieron su opinión profesional.

- Las imágenes en su dimensión deberían ser un poco menos pesadas y esto ayudaría a aliviar la carga en la app cuando se carga cada pantalla
- El apartado grafico puede mejorar haciendo énfasis en una buena paleta de colores lo que mejorara notablemente el aspecto de la aplicación y la experiencia de usuario
- Las gráficas pueden ser más coloridas y con mayor amplitud, un cambio en el aspecto que podría darle un aire grafico a la app
- El manejo de los datos puede ser de forma dinámica conectados a una estación meteorológica funcional.
- Los datos podrían ser tratados de forma clínica, por ejemplo, previniendo enfermedades respiratorias.
- La aplicación podría administrar datos dinámicamente para un uso analítico en tiempo real.
- La aplicación posee el potencial para gestionar la información por usuario lo que le daría mejor de experticia de usuario y funcionalidad.
- La aplicación podría notificar picos de información debido al manejo de datos practico que posee por su arquitectura

GIEMA es proyecto que posee mucho potencial y ventajas debido a su arquitectura IoT que se adapta fácilmente a cambios sin afectar los componentes principales, es un proyecto practico que puede ser integrado a estaciones neurológicas para diferentes fines donde su objetivo sea el manejo y análisis de datos meteorológicos autónomas, su visión es amplia y el proyecto puede involucrarse en cualquier tema referente

a variables meteorológicas, la estructura estará disponible en el repositorio, con el propósito de que GIEMA siga creciendo para un beneficio para todos.

## REFERENCIA

- Leider, C., Mann, D., & Dickinson, D. (2010). Wireless multisensor monitoring of the Florida Everglades: A pilot project. *A pilot project. In 129th Audio Engineering Society Convention 2010*, pp. 1089-1098.
- ALVARADO MORENO, J., C, C., KEVIN, D, ESTEBAN, F, DAVID, G, DAVID, M, . . . CASTELLANOS, C. (2017). Diseño e Implementación de un sistema de información de la calidad del aire en la Universidad de San Buenaventura. *Editorial Bonaventuriana*.
- Aprendiendo Arduino. (23 de Noviembre de 2018). *Aprendiendo Arduino*. Obtenido de <https://aprendiendoarduino.wordpress.com/2018/11/23/plataformas-cloud-publicas/>
- Asc for IOT. (s.f.). *microsoft*. Obtenido de <https://docs.microsoft.com/es-es/azure/asc-for-iot/overview>
- AwS Anuncia IoT Analytics. (s.f.). *siliconweek*. Obtenido de <https://www.siliconweek.com/e-innovation/aws-anuncia-iot-analytics-96356?print=print>
- Cisco IoT System. (2017). *infopl*c. Obtenido de <http://www.infopl.net/noticias/item/102888-cisco-iot-system>
- Fullana, C., & Urquía , E. (2009). *LOS MODELOS DE SIMULACIÓN: UNA HERRAMIENTA MULTIDISCIPLINAR DE INVESTIGACIÓN*. Madrid: Universidad Pontificia de Comillas.
- Gaitán, M., Cancino, J., & Behrentz, E. (2007). *Análisis del estado de la calidad del aire en Bogotá*. Bogota: Revista de Ingeniería - Universidad de Los Andes.
- Galvis, B., Y, N., & Rojas. (2006). Relación entre PM2,5 y PM10 en la ciudad de Bogotá. *RevActaNova. vol.3 no.2 Cochabamba*.
- Gómez, J. D. (16 de Marzo de 2019). *RCN Radio* . Obtenido de <https://www.rcnradio.com/estilo-de-vida/medio-ambiente/que-esta-pasando-con-las-estaciones-que-miden-la-calidad-del-aire-en>
- Google Cloud IoT Core. (s.f.). *Mas que negocio*. Obtenido de <https://www.masquenegocio.com/2017/05/17/google-cloud-iot-core/>
- IDEAM. (2016). *Informe del Estado de la Calidad del Aire en Colombia*. Bogotá, D.C.,: <http://www.ideam.gov.co/documents/51310/68521396/3.+Informe+del+Estado+de+la+Calidad+del+Aire+en+Colombia+2016.pdf/fb3eee92-6bcf-4979-9ea2-de0101496a2f?version=1.0>.
- Intel. (2020). *Nueva Arquitectura de referencia para IoT de Intel*. Obtenido de <https://www.lomasnuevo.net/iot/nuev-arquitectura-de-referencia-para-iot-de-intel/>
- José, G., Santiago, G., & Maialen, M. (2018). *Un estudio sobre aplicaciones móviles meteorológicas en países iberoamericanos, España y Portugal*. Asociación Meteorológica Española; Agencia Estatal de Meteorología.
- Kanaroglou, P, Jerrett, M, Morrison, J., Beckerman, B.,, Arain, M., Gilbert, N, & Brook, J. (2005). *Establishing an air pollution monitoring network for intra-urban population*. Atmospheric Environment.

- Lina , C., Escobar, C., & Javier, Á. (2012). ANÁLISIS CLÚSTER COMO TÉCNICA DE ANÁLISIS EXPLORATORIO DE REGISTROS MÚLTIPLES EN DATOS METEOROLÓGICOS. *Ingeniería de Recursos Naturales y del Ambiente*, núm. 11, pp. 11-20.
- Mauro, D., & Jose, H. (2018). *Aplicación móvil para el cálculo de factores meteorológicos y ayudas a la navegación*. 2018: Universidad de la Laguna.
- MINISTERIO DE EDUCACIÓN DE MADRID. (2011). La gestion estrategica de la eduacion electronica. *Revista Educación*.
- Molina, O. J. (2013). *Desarrollo de una metodología para evaluar la cobertura espacial de la Red de Monitoreo de la Calidad del Aire de Bogotá*. Bogota: Universidad Nacional.
- Montiel, J., Rubio, M., & López, J. (2012). Computación móvil. *Revista Chilena de Ingenieria*.
- OMS. (2017). *repasso de la salud mundial de la OMS*. OMS.
- Prauzek, M., Konecny, J., Hamel, A., & Hlavica, J. (2015). Fuzzy Energy Management of Autonomous. *IFAC-PapersOnLine*, 226–229.
- Rodríguez , D. (2013). Arquitectura y Gestión de la IoT. *Revista Telem@tica*. Vol. 12. No. 3, p. 49-60.
- Rodríguez, G., Quevedo, A., Castro, M., Arteaga, R., M. Alberto, M., & Zamora, B. (2016). *Predicción de variables meteorológicas por medio de modelos ARIMA*. Mexico: Agrociencia vol.50.
- Téllez, J., Rodríguez, A., & Fajardo, Á. (2006). Contaminación por Monóxido de Carbono: un Problema de Salud Ambiental. *Revista Salud Publica*, 108 - 117.
- Tobajas, A. G. (09/06/2016). *Diseño e implementación de una estación*. Catalunia: Universidad de Catalunia.
- Vergara, P., Rey, J., Osma, G., & Ordóñez, G. (2014). *Evaluación del potencial solar y eólico del campus central de la Universidad Industrial de Santander y la ciudad de Bucaramanga, Colombia*. Bucaramanga: Revista UIS Ingenierías.
- Verónica, T. (Agosto 2003). *Selección y aplicación de una metodología para la estimación de los factores de emisión de las fuentes móviles vehiculares de la ciudad de Bogotá*. Bogota D.C.: Universidad de los Andes.
- Vincenti, S., Zuleta, D., Moscoso, V., Jácome, P., Palacios, E., & Villacís, M. ((July-December 2012)). *Ánalysis estadístico de datos meteorológicos mensuales y diarios para la determinación de variabilidad climática y cambio climático en el Distrito Metropolitano de Quito*. Quito: Universidad Politécnica Salesiana of Ecuador.

## Anexos

Los siguientes anexos son el resultado de cada una de las capaz por la cual paso GIEMA donde se encuentra de manera más técnica y a profundidad la información que puede ser útil si desea consultar indicar más en base a al proyecto.

- ANEXO A Informe Técnico Tecnología IoT
- ANEXO B Documento Requerimientos
- ANEXO C Documento de diseño
- ANEXO D Documento de desarrollo
- ANEXO E Configuración de despliegue
- ANEXO F Documento Pruebas

**GIEMA-IoT: Herramienta móvil para la gestión de información generada por estaciones meteorológicas autónomas en un modelo IoT**

**ANEXO - A  
Pérez Martínez Jhon Alexander**

**Universidad de San Buenaventura, Sede Bogotá.  
Facultad de Ingeniería.  
Programas de Ingeniería de Sistemas  
Bogotá, Colombia  
2020**

## Tabla de contenido

1. Informe Técnico IoT.....	3
1.1. Dispositivos IoT .....	3
1.1.1. Sensores.....	3
1.1.2. Nodo.....	3
1.1.3. Servidor .....	4
1.1.4. Celular.....	4
1.1.5 WebService.....	4
1.2. Gateway IoT .....	4
1.2.1 Gateway Giema.....	4
1.2.3. Plataformas CLOUD IoT.....	5
1.3 Modelos IoT .....	6
1.3.1. Modelo IoT IBM.....	6
1.3.2. Modelo IoT de Intel .....	7
1.3.3. Modelo IoT con Plataforma de nube de Google:.....	8
1.3.4. Modelo IoT de Azure Cloud.....	8
1.3.5. Modelo IoT Cisco .....	9
1.3.6. Modelo AWS IoT.....	9
1.4. Comparación Modelos y Estructura final.....	10
1.4.1. Comparativa de los modelos IoT .....	10
1.4.2. Capaz del Modelo GIEMA.....	10
REFERENCIAS .....	13

## **1. Informe Técnico IoT**

“La Internet de las cosas es un tema emergente de importancia técnica, social económica. En este momento se están combinando productos de consumo, bienes duraderos, automóviles y camiones, componentes industriales y de servicios públicos, sensores y otros objetos de uso cotidiano con conectividad a Internet y potentes capacidades de análisis de datos que prometen transformar el modo en que trabajamos, vivimos y jugamos” (Rose, Eldridge, & Chapin, 2015).

Durante la investigación se analizó y se investigó diferentes dispositivos que tienen integrada tecnologías IoT esto con el objetivo de comprender la utilidad y la practicidad que esta tecnología aplicada a diferentes dispositivos pueden hacer diferentes actividades y funciones más prácticas y posteriormente poner en evidencia la importancia de como esta integrada esta tecnología al proyecto GIEMA para marcar una diferencia en el proyecto y es un aspecto muy resaltante.

### **1.1. Dispositivos IoT**

Aun con la ausencia física del proyecto se investigó en su totalidad cuales serían los componentes tanto físicos como lógicos que crean una arquitectura y hacen que posibles la conectividad, es decir la interacción de todos los componentes que crean

#### **1.1.1. Sensores**

Los sensores encargados de captar los datos y transformarlas con un transductor en variables eléctricas para que el nodo los procese y puedan ser enviados por conexión Xbeet a un Gateway, en este caso solo se encargarían de la recolección de datos para que la estación meteorológica pueda captar esta información así que, aunque su función es una sola es el comienzo de la recolección de datos para que sean procesados posteriormente.

#### **1.1.2. Nodo**

Nodo o la estación Meteorológica se plantea que sea de carácter autónoma usando energías renovables, esto con el fin de que sea ubicada en lugares estratégicos y no se tenga que usar

### **1.1.3. Servidor**

El servidor para este caso estará alojado en la nube será el dispositivo IoT encargado de guardar la información recolectada por los nodos para posteriormente ser usada según el caso de uso o lo que se requiera para consumir datos por medio del WebService

### **1.1.4. Celular**

El principal dispositivo IoT, se encargará de ser el medio para que el usuario y GIEMA se comuniquen con cada uno de sus componentes y servicios, siendo un medio de navegación y ejecución para los diferentes casos de uso que se tienen planteados

### **1.1.5 WebService**

El WebService definido en GIEMA es una tecnología que utiliza una serie de protocolos y estándares con la utilidad de intercambiar datos entre los componentes de la estructura ya definida, cada parte del webService se programa de manera separa ya que no se hará uso de ningún framework para la administración de estos componentes lógicos.

## **1.2. Gateway IoT**

En esta sección se explicará que es el Gateway a mayor profundidad su importancia y que rol cumple dentro de un modelo IoT lo que lo hace un elemento relevante que necesita ser explicado como se desempeña en la arquitectura.

### **1.2.1 Gateway Giema**

La puerta de enlace en la arquitectura es un punto de conexión entre la nube y los sensores, controladores y los dispositivos Smart, también es un punto de seguridad ya que como se había mencionado es un punto de comunicación para el modelo planteado en GIEMA, se planteó como un programa de software, debido a su practicidad y la cantidad de datos que en un punto son públicos, no van a manejar datos de los usuarios, ya que los datos del usuario no van a pasar por aquí si no que van hacer administrados por el WebService (información ingresada por la aplicación), por este medio pasara todo el flujo de datos referente a la información recopilada por el nodo correspondiente, esta parte del proyecto va ser simulada

mediante el resultado final que serían los datos generados y normalizados en bloques de datos que serán almacenados directamente en el servidor debido a la ausencia de la parte física.

### **1.2.3. Plataformas CLOUD IoT**

Las plataformas de IoT son plataformas SW que reciben los datos recogidos por nuestros sensores y luego son enviados por los microcontroladores y que se almacenan, además de dar otros servicios. Pueden ser plataformas de terceros o plataformas propias desarrolladas por nosotros.

Previamente para hacer una tomar guía de la diferente composición de algunas de las plataformas Cloud IoT más famosas y eficientes en el mercado, existen varias diferencias en su composición y su propósito así que se debe realizar una comparación previa de cual se adapta al propósito de GIEMA

- Virtualización, las plataformas SW pueden estar virtualizadas generalmente por motivos de escalabilidad.
- Bases de Datos, principalmente nosql como Mongo DB, RavenDB, cassandra y otras muchas. Big Data.
- Tratamiento de big data con Hadoop o spark.
- Plataformas propias como Amazon AWS que tiene para IoT <https://aws.amazon.com/es/iot/>.
- Google cloud platform: <https://cloud.google.com/solutions/iot/>.
- Google Weave, es una plataforma de comunicaciones para IoT.
- IBM blue Mix <http://www.ibm.com/cloud-computing/bluemix/>.
- Desarrollo con Microsoft Azure.
- Desarrollo de software de plataformas webs con .NET, php, javascript, python, django, node.js.
- Scadas industriales, scadas en la nube o scada as a service.

AWS Amazon IoT: Orientado a la manipulación de datos a gran volumen, AWS ofrece un enfoque en los dispositivos físicos que quieren ser conectados a internet para tener un amplio acceso, AWS IoT es la única plataforma en la nube que combina administración de datos y análisis ricos en servicios fáciles de usar diseñados para datos IoT.

Google Cloud IoT: Google Cloud IoT es un completo conjunto de herramientas para conectar, procesar, almacenar y analizar datos tanto en el perímetro como en la nube. La plataforma se compone de

servicios en la nube escalables y totalmente gestionados: una pila de software integrada con funciones de aprendizaje automático para los recursos de computación on-premise o del perímetro.

**IBM Blue Mix:** La nube híbrida es una plataforma para aplicaciones e infraestructura, construida sobre dos o más componentes de la nube pública, la nube privada y la TI local. En todas sus formas, la nube híbrida facilita la flexibilidad y la portabilidad de las aplicaciones y los datos (aprendiendoarduino.wordpress, 2017).

### **1.3 Modelos IoT**

Para el presente proyecto GIEMA se realizó una comparativa de diferentes modelos IoT, evaluando cada uno de sus componentes por separado y realizar una comparativa para saber cuál modelo se adapta más a los objetivos de GIEMA para cumplir los objetivos planteados del proyecto.

Para poder realizar un mejor proceso de análisis, se consulto individualmente cada modelo seleccionado, estudiando cuidadosamente su arquitectura y composición con el fin de aprender como este compuesto, su funcionamiento y como se ejecuta para cumplir su propósito, de esta manera se podrá utilizar esta información

El modelo Blue Mix a pesar de ser uno de los más versátiles y ofrecer servicios agregados no es lo que busca GIEMA en el momento, aunque cumple con el modelo en general pero su orientación y propósito es diferente, ya que se centra en infraestructuras hibridas con diferentes componentes, Amazon AWS ofrece una arquitectura flexible para mover gran volumen de datos así que es un modelo del cual se puede aprender mucho.

#### **1.3.1. Modelo IoT IBM**

El modelo IBM está centrado en el manejo de y conexión de modelos híbridos por lo que no comparte el ideal de GIEMA de como se procesan estos datos, aunque su arquitectura es flexible y cómoda esta orientada al control de esquemas y modelos de aprendizaje Machine Learning como se ve en la figura.

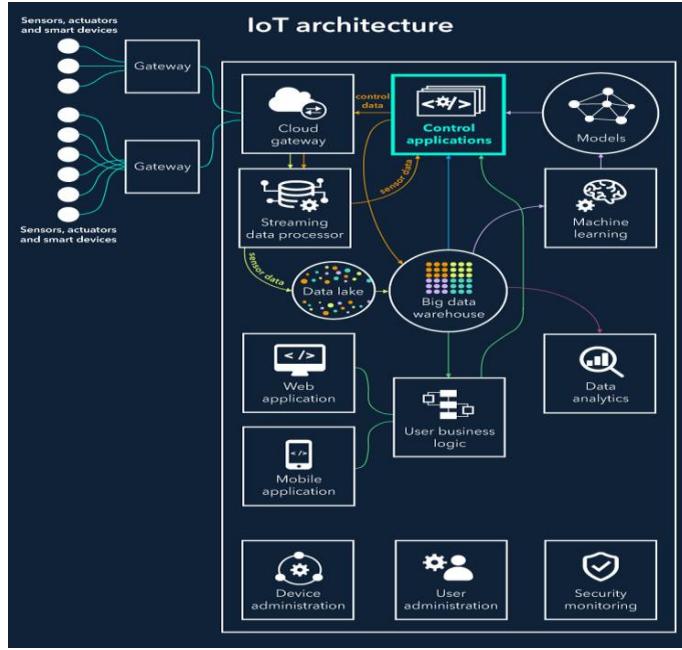


Figura 1 – Modelo IoT IBM

### 1.3.2. Modelo IoT de Intel

El modelo de Intel está basado en el proceso y control de datos a gran velocidad. Esta plataforma se basa más en lo que busca GIEMA, pero tiene varios componentes dedicados a la hibridación de control de otros procesos por lo que se debería afectar mucho su arquitectura para conseguir el verdadero propósito de GIEMA.

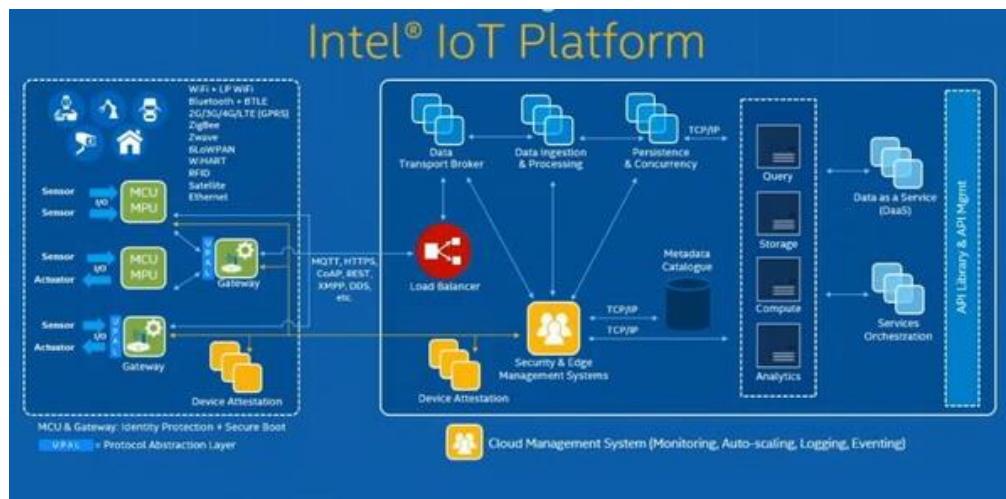


Figura 2 - Modelo IoT de Intel

### 1.3.3. Modelo IoT con Plataforma de nube de Google:

Los modelos IoT basados en la tecnología de Google son los más adaptables a lo que se busca ya que gracias su enfoque basado en el manejo masivo de datos como fin de consulta, es el modelo más cómodo para GIEMA ya que al manejar el flujo de datos de manera correcta permite hacer mejor uso de los recursos.

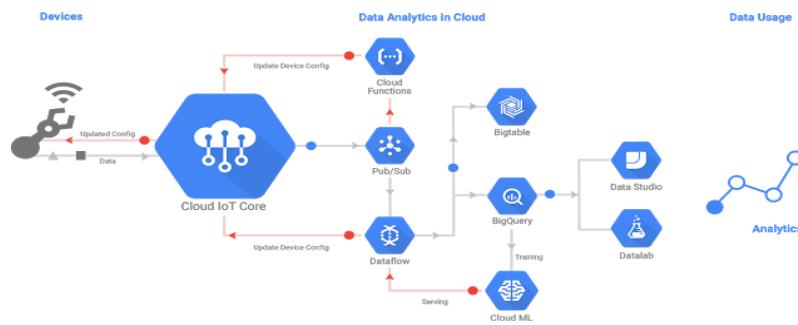


Figura 3 Modelo IoT con Plataforma Google Cloud

### 1.3.4. Modelo IoT de Azure Cloud

Es un servicio centrado en la construcción, prueba, despliegue de aplicaciones su objetivo es el de administrar varias tareas y aunque maneja grandes cantidades de datos su objetivo esta basado en los cuatro servicios que presta nombrados anteriormente

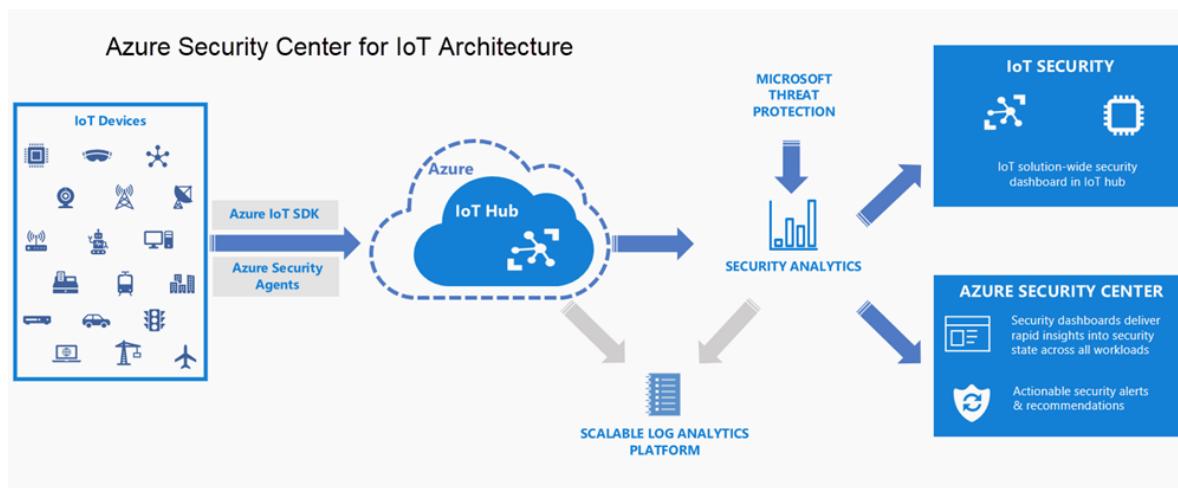


Figura 4 Modelo IoT de Azure Cloud

### 1.3.5. Modelo IoT Cisco

Cisco esta centrado en la seguridad de redes y el análisis de datos, por cual podría estar afín con GIEMA, pero solo se rescata esto para nuestro proyecto ya que esta orientada a la automaticen de procesos.

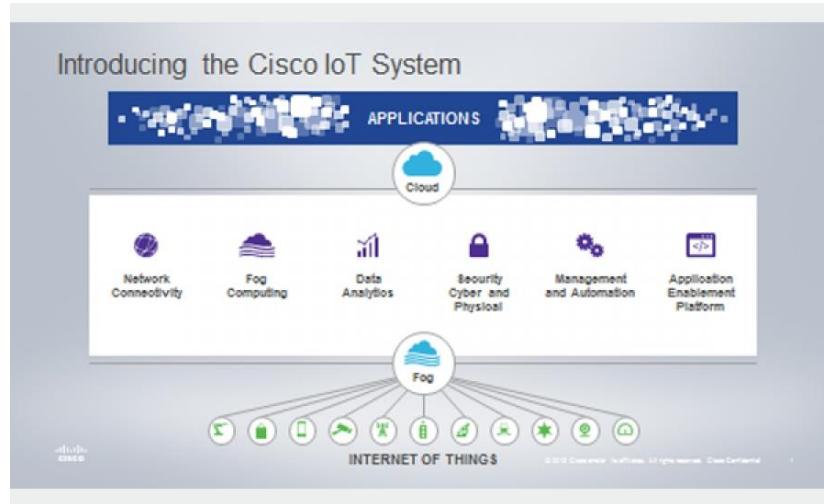


Figura 5 Modelo IoT Cisco

### 1.3.6. Modelo AWS IoT

Amazon combina una arquitectura flexible para mover gran flujo de datos debido a los diferentes servicios que presta esto lo hace un modelo flexible y adaptable que junto con Google cumple con los principios de GIEMA.

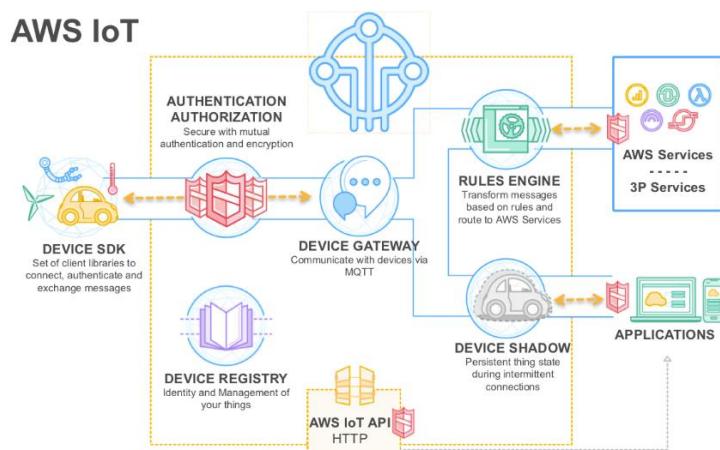


Figura 6 Modelo AWS IoT

## 1.4. Comparación Modelos y Estructura final

### 1.4.1. Comparativa de los modelos IoT

En esta sección se realizó la comparativa de todos los modelos para evaluar su tecnología y en base a la información evaluada arriba se pudo definir como estaban relacionadas entre sí en base a su arquitectura ayudan así a la evaluación para determinar la arquitectura de GIEMA.

Tabla 1- Comparativa de modelos IoT

Modelo/Característica	Dispositivo	Gateway	Network	Servicio	Análisis de Datos	Aplicación
<b>IBM</b>	Sí	Sí	Sí	Sí	Sí	Sí
<b>Intel</b>	Sí	Sí	Sí	Sí	Sí	Sí
<b>Plataforma de nube de Google</b>	Sí	Sí	SÍ	SÍ	Sí	Sí
<b>Azure Cloud</b>	Sí	Sí		Sí	Sí	Sí
<b>Cisco</b>	Sí	Sí	Sí		Sí	Sí
<b>AWS Amazon</b>	Sí	Sí	Sí	Sí	Sí	Sí

### 1.4.2. Capaz del Modelo GIEMA

Para el presente proyecto se definieron cinco capaz en las cuales van a estar ubicadas cada uno de los componentes que van hacer a GIEMA posible la primera capa la capa de Dispositivos donde se encuentra la parte física compuesta por los sensores y la estación meteorológica donde se transmitirán los datos por medio de un módulo de conectividad inalámbrica que en este caso podría ser xBee una conexión por IP y pasar al Gateways para controlar todo el protocolo de comunicación entre la parte física y el WebService encargado de intercomunicar los servidores de GIEMA los datos que trae el Gateway enviados desde la estación meteorológica para ser almacenados o atender una petición de la APP móvil la cual es la interfaz de interacción con el usuario para que realice alguna de las acciones habilitadas.

Tabla 2 Capaz del Modelo GIEMA

Modelo GIEMA - IoT		Funcionalidad	Condiciones
<b>Aplicación:</b>	App Móvil	Su función es la notificación y visualización de datos, además de ser el medio la para realizar peticiones por medio de las	La aplicación funcionará desde Android 5.0 en adelante y requerirá una conexión a internet

Modelo GIEMA - IoT		Funcionalidad	Condiciones
		diferentes funciones establecidas dentro la aplicación en su desarrollo	
<b>Servicio:</b>	WebService	Es la parte más dinámica de la arquitectura, encargado de la administración de datos, el WebService controlará el flujo de datos, recibiendo y procesando las peticiones y orientando los datos que se solicitan en el momento que se solicitan, además de cumplir órdenes y ejecutarlas según lo establecido en el desarrollo del código	El servidor escogido cumple con los parámetros establecidos de funcionalidad y soporte para el soportar el flujo de datos sin saturarse
	Servidor	Encargado de almacenar los datos y responder a las peticiones enviadas desde la app que pasan por el WebService, actúan según el modelo de datos diseñado	El servidor se es propiedad de la universidad San Buenaventura por lo tanto todos los lineamientos de seguridad deben ser acatados, incluyendo las limitaciones a solo administrar la base de datos de GIEMA parcialmente
<b>Red:</b>	Conexión (IP)	Simulación	Simulación
<b>Gateway</b>	Protocolos de comunicación	Simulación	Simulación
<b>Dispositivos</b>	Tarjeta - Sensores	Simulación	Simulación

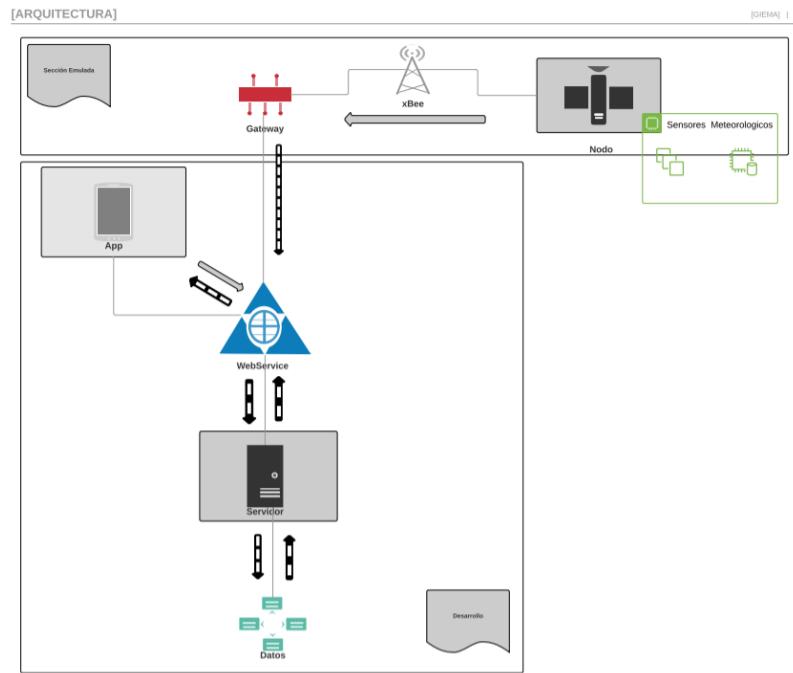


Figura 7 Modelo IoT GIEMA

## REFERENCIAS

- Aprendiendo Arduino. (23 de Noviembre de 2018). *Aprendiendo Arduino*. Obtenido de <https://aprendiendoarduino.wordpress.com/2018/11/23/plataformas-cloud-publicas/>
- aprendiendoarduino.wordpress. (31 de Marzo de 2017). *aprendiendoarduino*. Obtenido de <https://aprendiendoarduino.wordpress.com/2017/03/31/plataformas-iot/>
- Asc for IOT. (s.f.). *microsoft*. Obtenido de <https://docs.microsoft.com/es-es/azure/asc-for-iot/overview>
- AwS Anuncia IoT Analytics. (s.f.). *siliconweek*. Obtenido de <https://www.siliconweek.com/e-innovation/aws-anuncia-iot-analytics-96356?print=print>
- Cisco IoT System. (2017). *infopl*c. Obtenido de <http://www.infopl.net/noticias/item/102888-cisco-iot-system>
- Google Cloud IoT Core. (s.f.). *Mas que negocio*. Obtenido de <https://www.masquenegocio.com/2017/05/17/google-cloud-iot-core/>
- IBM. (2020). *IBM*. Obtenido de <https://www.ibm.com/co-es/internet-of-things>
- Intel. (2020). *Nueva Arquitectura de referencia para IoT de Intel*. Obtenido de <https://www.lomasnuevo.net/iot/nuev-arquitectura-de-referencia-para-iot-de-intel/>
- Rose, K., Eldridge, S., & Chapin, L. (2015). *LA INTERNET DE LAS COSAS - UNA BREVE RESEÑA*. Internet Society.

**GIEMA-IoT: Herramienta móvil para la gestión de información  
generada por estaciones meteorológicas autónomas en un modelo IoT**

**ANEXO - B  
Pérez Martínez Jhon Alexander**

**Universidad de San Buenaventura, Sede Bogotá.  
Facultad de Ingeniería.  
Programas de Ingeniería de Sistemas  
Bogotá, Colombia  
2020**

## **Tabla de contenido**

2. Levantamiento Requerimientos	3
2.1 Lista de Requerimientos Funcionales APP	3
2.2 Lista de requerimientos no Funcionales – APP	8
2.3. Requerimientos Funcionales – WebService	10
2.4. Casos de Uso	14
2.5. Trazabilidad	20
2.5.1 App	20
2.5.2 WebService	20
2.6 Conclusión del Análisis de trazabilidad	21

## 2. Levantamiento Requerimientos

Para el presente proyecto se realizaron se levantaron una serie de requerimientos para dar cumplimiento a los objetivos planteados, por medio de estos requerimientos se plantearon lo que se debe dar cumplimiento en el desarrollo del sistema.

### 2.1 Lista de Requerimientos Funcionales APP

A partir del estudio realizado y analizando los objetivos a cumplir se plantearon lo siguientes requerimientos funcionales que suplieran las necesidades del proyecto planteado para garantizar su funcionalidad y estabilidad.

*Tabla 1 - Requerimientos funcionales APP*

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RF-001	Entrada de Datos de Usuario	El sistema permitirá la entrada de datos para almacenar	ALTA
RF-002	Validación de datos de usuario	El sistema la autenticidad de los datos en la aplicación	ALTA
RF-003	Consulta de datos Meteorológicos	El sistema consultara los diferentes datos meteorológicos generando una respuesta	ALTA
RF-004	Filtración de datos por fecha	El sistema filtrara los datos por fecha indicada	MEDIA
RF-005	Consumo de datos por consulta	El sistema ejecutar una consulta asignada al WebService	ALTA
RF-006	Encriptación de contraseña	La contraseña se encriptará	ALTA

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RF-007	Clasificación de datos meteorológicos	Los datos meteorológicos se filtrarán por función y sensor	ALTA
RF-008	Llamada de datos por WebService	El sistema llamará los datos consumiendo el Webservice	ALTA
RF-009	Consumo de datos por clase	El sistema consumirá los datos según la clase en las que se ejecuten	ALTA
RF-010	Datos Meteorológicos formato JSON	Los datos meteorológicos se enviarán y se entregarán en formato JSON	ALTA
RF-011	Consumo de WebService por consulta	Se realizará una ruta de consumo para llegar al WebService	ALTA
RF-012	Consumo de datos por consulta Query	Se consumirá una consulta SQL para ejecutar el servicio solicitado	ALTA

Nº del Requerimiento	RF-001
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Entrada de Datos de Usuario
Precondición	El usuario debe Loguearse
	Acción
Secuencia	El sistema permitirá la entrada de datos para ser enviada al servidor y se almacenén
Pos condición	Ingresar a la aplicación
Excepciones	Solo se pueden digitar datos Validos
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-002
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Validación de datos de usuario
Precondición	El usuario debe digitar datos verídicos
	Acción
Secuencia	El sistema validara la autenticidad de los datos en la aplicación
Pos condición	El sistema entregara y validara datos verídicos
Excepciones	Si no hay datos se enviará un mensaje de aviso
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-003
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Consulta de datos Meteorológicos
Precondición	El usuario debe Loguearse
	Acción
Secuencia	El sistema consultara los diferentes datos meteorológicos generando una respuesta en la aplicación para ser procesada y mostrada
Excepciones	Solo se limitara a la consulta realizada
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-004
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Filtración de datos por fecha
Precondición	El usuario debe Loguearse
	Acción
Secuencia	El sistema filtrara los datos por fecha indicada por el usuario
Excepciones	Se mostrarán todas las fechas
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-005
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Consumo de datos por consulta
Precondición	El usuario debe Loguearse
	Acción
Secuencia	El sistema ejecutara una consulta asignada al WebService
Excepciones	Solo se ejecutarán consultas validas
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-006
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Encriptación de contraseña
Precondición	El usuario debe Loguearse
	Acción
Secuencia	La contraseña se encriptara
Excepciones	La contraseña debe ser digitada por el Usuario
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-007
Versión	1.0
Autor	Jhon Alexander Perez Martínez
Descripción	Clasificación de datos meteorológicos
Precondición	El usuario debe Loguearse
	Acción
Secuencia	Los datos meteorológicos se filtraran por función y sensor dando datos exactos
Excepciones	Solo un sensor por clasificación
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-008
Versión	1.0
Autor	Jhon Alexander Perez Martínez
Descripción	Llamada de datos por WebService
Precondición	El usuario debe Loguearse
Secuencia	Acción
	El sistema llamara los datos consumoiendo el Webservice
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-009
Vesión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Consumo de datos por clase
Precondición	El usuario debe Loguearse
Secuencia	Acción
	El sistema consumira los datos según la clase en las que se ejecuten
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-010
Versión	1.0
Autor	Jhon Alexander Perez Martínez
Descripción	Datos Meteorologicos formato JSON
Precondición	El usuario debe Loguearse
Secuencia	Acción
	Los datos meteorologicos se enviaran y se entraran en formato JSON
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-011
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Consumo de WebService por consulta
Precondición	El usuario debe Loguearse
Secuencia	Acción
	Se realizara una ruta de consumo para llegar al WebService
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-012
Vesión	1.0
Autor	Jhon Alexander Perez Martínez
Descripción	Consumo de datos por consulta Query
Precondición	El usuario debe Loguearse
Secuencia	Acción
	Se consumira una consulta sql para ejecutar el servicio solicitado
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

## 2.2 Lista de requerimientos no Funcionales – APP

Según lo dicho anteriormente se plantearon los siguientes requerimientos no funcionales.

*Tabla 2 - Requerimientos No Funcionales*

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RNF-001	Ingreso de datos en cajas de Texto Login	La aplicación permite la digitalización de datos por medio de cajas	ALTO
RNF-002	Navegación entre pantallas por botones	La aplicación permitirá la navegación de pantalla por botones	ALTO

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RNF-003	Visualización de un menú	La aplicación permitirá la visualización con las diferentes opciones de la app	ALTO
RNF-004	Visualización e interacción de un Spinner	Visualización de un Spinner para seleccionar el nodo	MEDIO
RNF-005	Selección de opción de datos por botones	El tipo de sensor que se desea consultar se debe hacer por medio de un botón	MEDIO
RNF-006	Grafica de datos	La aplicación mostrara una gráfica con los datos	ALTO
RNF-007	Imagen de Login Dinámica	La aplicación tendrá una imagen de presentación	BAJO
RNF-008	Imágenes de fondo en todas las pantallas	Cada pantalla tendrá una imagen de fondo	BAJO
RNF-009	Almanaque de selección	La aplicación tendrá una forma de ingresar la fecha	ALTO
RNF-010	Ocultar contraseña	La aplicación ocultara la contraseña	ALTO
RNF-011	Visualización de datos meteorológicos	Los datos meteorológicos deben visualizarse	ALTO

## 2.3. Requerimientos Funcionales – WebService

Para el WebService se plantearon los siguientes requerimientos para dar cumplimiento a los objetivos trazados.

*Tabla 3 - Requerimientos Funcionales WebService*

Nº de Requerimiento	Nombre del Requerimiento	Descripción	Prioridad
RF-001	Envío de datos al Servidor	El WebServices enviará los datos al Servidor	ALTA
RF-002	Parseo de datos formato JSON	Los datos serán parseados al formato JSON	ALTA
RF-003	Validación de datos	Se hará una validación de los datos ingresados	ALTA
RF-004	Almacenamiento de datos	Se almacenarán los datos	ALTA
RF-005	Consumo por servicio	Se consumirá el servicio requerido en el WebService	ALTA
RF-006	Ejecución de consulta Query	Se ejecutará el Query requerido	ALTA
RF-007	Repuesta de datos de las peticiones	Se generará una respuesta por petición	ALTA
RF-008	Esquema de datos con llaves foráneas	El esquema se relacionará con mediante llaves foráneas	ALTA
RF-009	Flujo de datos por petición	Según la petición se realizará un flujo de datos	ALTA
RF-010	Interconexión de los Servicios	Los servicios consumirían un medio de enlace	ALTA

Nº del Requerimiento	RF-001
Vesión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Envio de datos al Servidor
Precondición	Debe haber trafico de datos
	Acción
Secuencia	Los datos seran enviados por medio de la aplicación, atravez del WebService
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-02
Vesión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Parseo de datos formato JSON
Precondición	Debe haber trafico de datos
	Acción
Secuencia	Los datos seran entregados y recibidos en formator JSON
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-003
Vesión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Validación de datos
Precondición	Debe haber trafico de datos
	Acción
Secuencia	Se deben validar los datos enviados con los que estan alojados en el servidor
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-04
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Almacenamiento de datos
Precondición	Debe haber trafico de datos
	Acción
Secuencia	Los datos enviados al servidor deberan ser almacenados cuando es requerido
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-05
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Consumo por servicio
Precondición	Debe haber trafico de datos
	Acción
Secuencia	El sistema deberá consumir el servicio indicado según la petición que se le de
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-06
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Ejecución de consulta Query
Precondición	Debe haber trafico de datos
	Acción
Secuencia	La petición al servicio esta acompañada con una instrucción de ejecución que debe ser entregada al servidor para ejecutar la consulta
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-07
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Repuesta de datos de las peticiones
Precondición	Debe haber trafico de datos
Secuencia	Acción
	Se generara una respuesta a la petición realizada
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-08
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Esquema de datos con llaves foraneas
Precondición	Debe haber trafico de datos
Secuencia	Acción
	El esquema de la base de datos debe contener llaves foraneas para realazcionar las vairables que se deseen indicar
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-09
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Flujo de datos por petición
Precondición	Debe haber trafico de datos
Secuencia	Acción
	El de datos dependera de la petición recibida para que sea orientada al servicio
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

Nº del Requerimiento	RF-010
Vesión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Interconexión de los Servicios
Precondición	Debe haber trafico de datos
Secuencia	Acción La interconexión de los servicios debe hacerse por protolo TCP
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta

## 2.4. Casos de Uso

Para el presente proyecto se plantearon lo siguiente casos de Uso.

*Tabla 4 - Casos de Uso GIEMA*

Nº de Caso de Uso	Nombre del Caso de Uso	Descripción
CU-001	Logueo a la aplicación	El usuario podrá realizar loguearse a la aplicación realizando un proceso de validación
CU-002	Navegación entre pantallas	El usuario podrá navegar entre pantallas para ver su contenido
CU-003	Registro en la aplicación	El usuario podrá registrarse a la aplicación digitando sus datos
CU-004	Consulta datos recientes	El usuario podrá consultar sus datos básicos
CU-005	Consulta de nodo	El usuario podrá visualizar el nodo que va a consultar
CU-006	Consulta de datos por sensor	El usuario podrá seleccionar el sensor que desee visualizar
CU-007	Visualización de datos Meteorológicos	El usuario podrá visualizar los datos Meteorológicos

Nº de Caso de Uso	Nombre del Caso de Uso	Descripción
CU-008	Visualización de la gráfica de datos	El usuario podrá ver la gráfica de los datos consultados
CU-009	Selección de fecha de consulta de datos	El usuario podrá filtrar los datos que deseé consultar por fecha

Nº Casos de Uso	CU-001
Versión	1.0
Autor	Jhon Alexander Perez Martínez
Descripción	Logueo a la aplicación
Secuencia	<p>Acción</p> <p>El usuario al ingresar los datos registrados en la base de datos &lt;Read&gt;</p> <p>Los datos viajan y se procesan por medio de una función y se almacenan en el servidor &lt;Function&gt;</p> <p>Al realizar el login se dispara una validación, los datos son comparados a que sean los mismos almacenados y si es correcto se ingresa a la aplicación &lt;load&gt;</p>
Excepciones	La aplicación debe estar conectada a internet
Importancia	Vital
Atributos	Add/ Edit / Delete
Prioridad	Alta

Nº Casos de Uso	CU-002
Versión	1.0
Autor	Jhon Alexander Perez Martínez
Descripción	Navegación entre pantallas
Secuencia	Acción
	El usuario podrá navegar entre pantallas para ver su contenido <Load>
	Al seleccionar un boton el usuario podra navegar entre las diferentes pantallas según el flujo <Load>
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta
Atributos	Add/ Edit / Delete

Nº Casos de Uso	CU-003
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Registro en la aplicación
Secuencia	Acción
	El usuario podrá registrarse a la aplicación digitando sus datos <Load>
	Los datos registrados seran almacenados en la base de datos <Function>
La contraseña digitada en ese proceso sera encriptada <Function>	
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta
Atributos	Add/ Edit / Delete

Nº Casos de Uso	CU-004
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Consulta de datos recientes
Secuencia	Acción
	El usuario podrá consultar los ultimos datos disponibles <Read>
	La aplicación ejecutara la consulta con los ultimos datos disponibles <Function>
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta
Atributos	Add/ Edit / Delete
Nº Casos de Uso	CU-005
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Consulta de nodo
Secuencia	Acción
	El usuario podrá visualizar el nodo que va a consultar <Read>
	El usuario tendra acceso al nodo que se encuentra actualmente activo visualizado en un spinner <Read>
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta
Atributos	Add/ Edit / Delete

Nº Casos de Uso	CU-006
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Consulta de datos por sensor
Secuencia	Acción
	El usuario podrá seleccionar el sensor que desee visualizar <Load>
	La aplicación dara acceso a la opcionde consulta para iniciar las peticiones<Read>
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta
Atributos	Add/ Edit / Delete

Nº Casos de Uso	CU-007
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Visualización de datos Meteorológicos
Secuencia	Acción
	El usuario podrá visualizar los datos Meteorológicos <Read>
	La app llamara los datos según sus condiciones <Funtion>
	El servidor procesara los datos y los entregara a la app para que sean vizualizados <Load>
Excepciones	La app debe estar conectada a Internet
Importancia	Vital
Prioridad	Alta
Atributos	Add

Nº Casos de Uso	CU-008
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Visualización de la grafica de datos
Secuencia	Acción El usuario podrá ver la grafica de los datos consultados<Read>
	Según los datos parametrizados en la consulta la app enviara una peticion para buscar y tomar los datos <funtion>
	Los datos seran procesados y graficados <Load>
Excepciones	Ninguna
Importancia	Vital
Prioridad	Alta
Atributos	Add

Nº Casos de Uso	CU-009
Versión	1.0
Autor	Jhon Alexander Perez Martinez
Descripción	Selección de fecha de consulta de datos
Secuencia	Acción El usuario podra filtrar los datos que desee consultar por fecha <Load>
	Según la consulta enviada , se filtrara la consulta y generara los datos seleccionados<Read>
	El servidor respondera con los datos y se enviaran atravez del WebService<funtion>
Excepciones	La fecha debe coincidir con los datos filtrados
Importancia	Vital
Prioridad	Alta
Atributos	Add/ Edit / Delete

## 2.5. Trazabilidad

Según la siguiente tabla se muestra la relación entre los requerimientos y los casos de uso.

### 2.5.1 App

Para asegurar la trazabilidad de la aplicación se realizó un seguimiento de como los componentes encajados en este caso la comparativa de en qué cada requerimiento podría estar presente en uno o más casos de uso.

	CU-001	CU-002	CU-003	CU-004	CU-005	CU-006	CU-007	CU-008	CU-009	CU-010
RF-001	X	X			X		X		X	
RF-002						X				
RF-003		X				X		X		
RF-004										X
RF-005		X						X		
RF-006				X	X		X		X	
RF-007	X		X							
RF-008		X			X		X		X	
RF-009		X		X		X				X
RF-010			X			X		X		X
RF-011	X		X	X				X		
RF-012		X			X				X	

### 2.5.2 WebService

La comparativa y la relación bilateral de los requerimientos con los casos de uso fue también aplicada en el WebService con el propósito de lograr un equilibrio en sus componentes.

	CU-001	CU-002	CU-003	CU-004	CU-005	CU-006	CU-007	CU-008	CU-009	CU-010
RF-001	x		x							x
RF-002	x	x				x		x	x	
RF-003		x	x		x	x	x			
RF-004				x	x		x	x	x	x
RF-005	x	x	x					x		
RF-006		x		x		x	x		x	x
RF-007	x	x				x	x	x		
RF-008				x	x		x	x		x
RF-009	x	x	x			x			x	
RF-010	x		x					x		x

## 2.6 Conclusión del Análisis de trazabilidad

- Realizar un proceso de trazabilidad hace que el proyecto sea mas organizado para estructurar mejor sus componentes.
- El proceso de análisis ayuda a fortalecer el flujo de la aplicación afianzando los requerimientos con los casos de uso de cada componente.
- El proceso de análisis ayuda a la estructuración de los componentes de mayor prioridad dando una vista mas amplia de lo que se quiere lograr.

**GIEMA-IoT: Herramienta móvil para la gestión de información  
generada por estaciones meteorológicas autónomas en un modelo IoT**

**ANEXO - C**

**Pérez Martínez Jhon Alexander**

**Universidad de San Buenaventura, Sede Bogotá.  
Facultad de Ingeniería.  
Programas de Ingeniería de Sistemas  
Bogotá, Colombia  
2020**

## **Tabla de Contenido**

3. Diseño.....	3
3.1 Modelo de datos .....	3
3.1.1. Diccionario de datos .....	5
3.2. Modelado del sistema UML.....	7
3.3. Esquema del mensaje.....	9
3.4. Arquitectura GIEMA.....	11
3.5. Mockups y Navegación .....	15

### **3. Diseño**

En esta sección se desarrollará todo lo relacionado al diseño de GIEMA de manera que pueda ser plasmado y representado mediante gráficos y sea más fácil su explicación para iniciar la etapa de desarrollo.

#### **3.1 Modelo de datos**

El modelo de datos es una pieza clave para el proyecto ya que se el buen flujo de las relaciones y limitaciones entre tablas hace que sea más optimo los procesos o las peticiones a la base de datos, debido a que este proyecto maneja una cantidad considerable de datos producidos para posteriormente ser utilizados en diferentes funciones, como resultado el modelo de datos se diseña en base a las necesidades del sistema, ya que requiere un constante flujo de información en cuanto a los datos que recoge y su interacción con la aplicación, el flujo de modelo de datos se diseña de una manera que pueda darle flexibilidad y volver más liviana la carga para las peticiones y para la propia información que se guarda en el servidor, debido a que está constantemente activo durante el proceso de recolección de información de las estaciones meteorológicas que proveerán los datos para posteriormente ser procesados por el webService y trabajados juntos con el servidor mediante el modelo de datos cumple su función almacenando y manipulando datos entre tablas, debido al diseño de este modelo de datos.

El modelo de datos será integrado usando MySql en versión 5.7 en adelante debido a su gran capacidad para manejar modelo de datos su estabilidad, a su gran rendimiento para procesar datos, su facilidad de conexión y su licencia OpenSource.

Siguiendo los lineamientos usados y con el propósito de crear un buen desarrollo y flujo en el sistema, se pone especial atención al modelo de datos donde se determina la creación de siete tablas diferentes donde dos son para la almacenar la información del usuario y estas dos están relacionadas, donde una guardara la información del usuario y la otra dará el atributo al cual pertenece al usuario, las otras cinco tablas manipulan la información del sistema IoT donde una tabla almacena la información del nodo que está relacionada a la tabla que almacena el Gateway , esto con el propósito de tener información de los datos que los interrelaciona, esto hace que el nodo se conecte con la información de la tabla sensor que se

asocia a la tabla tipo\_sensor donde el tipo\_sensor almacena los datos del atributo principal que cumple dicho sensor y devuelve la información como llave foránea para ser asociado a la tabla sensor que almacena los datos y los atributos principales de un sensor junto con su funcionalidad principal, y finalmente llevado a la tabla mensaje que contiene los datos dados por la estación meteorológica y los atributos principales de los sensores dados por sus relaciones con las demás tablas , esto con el propósito de que todos los datos sean interrelacionados para crear procesos más rápidos y que los request sean más agiles y no sobre carguen al servidor , como se dijo anteriormente se necesita un modelo de datos eficiente ya que recibirá cantidades considerables de datos y esto se hace con el fin de no sobre cargar al servidor cuando el usuario haga una petición por la aplicación móvil o la estación meteorológica mande datos al servidor de manera continua.

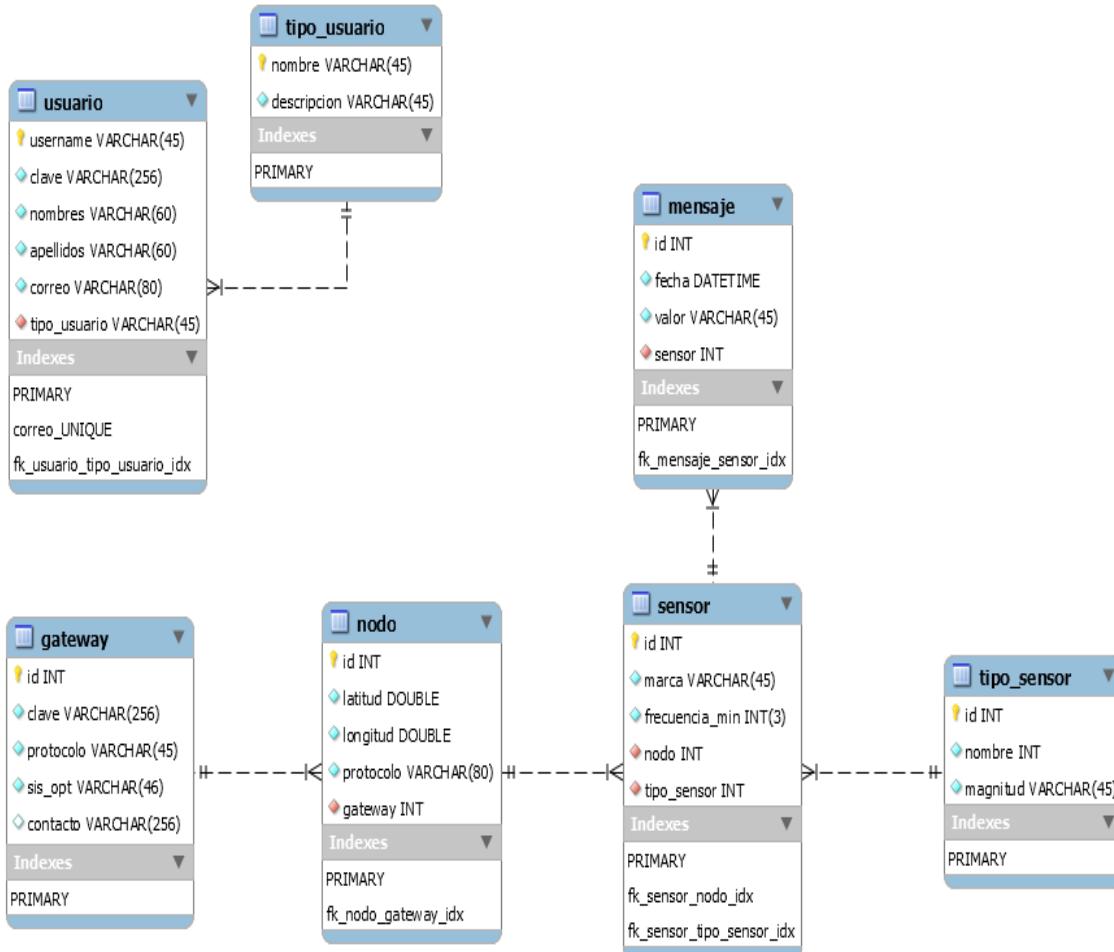


Figura 1 - Modelo de Datos

### 3.1.1. Diccionario de datos

Usuario		
Nombre Dato	Tipo	Descripcion
Username PK	VARCHAR	Nick del Usuario
clave	VARCHAR	Clave del usuario
nombres	VARCHAR	Nombre del Usuario
apellidos	VARCHAR	Apelli del Usuario
correo	VARCHAR	Correo del Usuario
tipo_usuario	VARCHAR	Tipo de roll del Usuario_Llave Foranea

Tipo_Usuario		
Nombre Dato	Tipo	Descripcion
Nombre PK	VARCHAR	Nombre del tipo Usuario
descripcion	VARCHAR	descripcion del roll

gateway		
Nombre Dato	Tipo	Descripcion
id PK	INT	id del Gateway
clave	VARCHAR	clave del gateway
protocolo	VARCHAR	protocolo del gateway
sis_opt	VARCHAR	sisema operativo del gateway
contacto	VARCHAR	contacto del gateway

nodo		
Nombre Dato	Tipo	Descripcion
id PK	INT	id del nodo
latitud	DOUBLE	Latitud del nodo
longitud	DOUBLE	Longitud del nodo
protocolo	VARCHAR	Tipo de protocolo de comunicación
Gateway	INT	ID del Gateway Llave Foranea

sensor		
Nombre Dato	Tipo	Descripcion
id PK	INT	id Sensor
marca	VARCHAR	Marca del sensor
frecuencia	INT	Frecuencia del sensor
nodo	INT	id nodo llave foranea
tipo sensor	INT	id tipo_sensor llave foranea

tipo_sensor		
Nombre Dato	Tipo	Descripcion
id PK	INT	id tipo sensor
Nombre	VARCHAR	nombre del tipo sensor
magnitud	VARCHAR	Frecuencia del sensor

### 3.2. Modelado del sistema UML

Parte de las funciones del sistema deben ser evaluadas según los requerimientos dados por los objetivos del presente proyecto, para determinar esos procesos y funcionalidades que se deben dar en el sistema para cumplir los objetivos dados se realizaron una serie de análisis construir los diagramas necesarios para pasar a la siguiente etapa, la etapa de desarrollo del sistema IoT.

Como primera medida se evaluó como va a estar compuesto el proyecto en su totalidad para construir de manera óptima los procesos con los que va a interactuar tanto los diferentes componentes del sistema y el usuario, como las funcionalidades que este va a prestar tanto los componentes como el usuario, referenciando en la figura 2, donde se evidencia la estructura general de cada componente y como se relacionan entre si convirtiéndose en un sistema armoniosos.

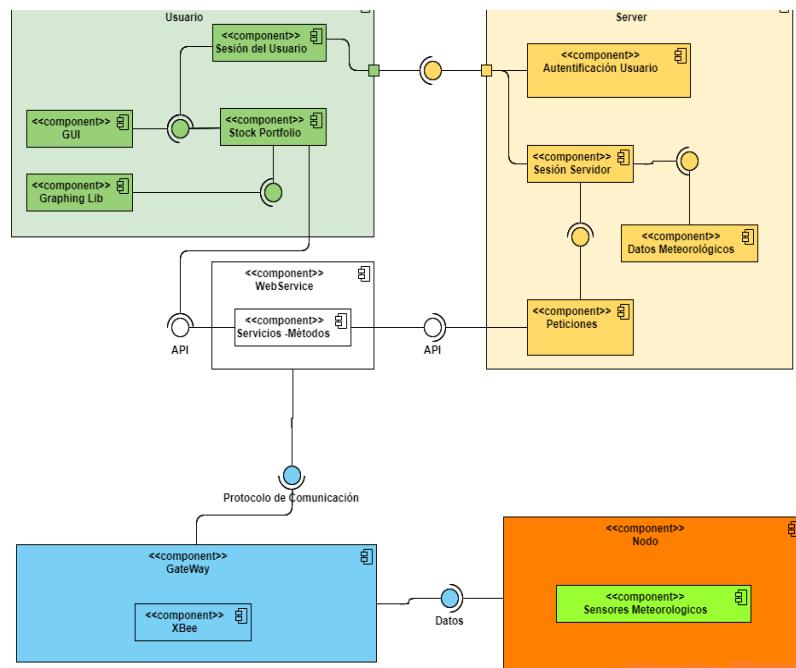


Figura 2 - Diagrama de componentes

Donde el usuario podrá interactuar por medio de la aplicación móvil, para tener acceso a los datos meteorológicos proporcionados por el nodo autónomo, por medio de unos procesos internos que se estarán alcance de las opciones de la aplicación móvil para interactuar directamente con todo el sistema de manera

rápida que están condicionados por unos casos de uso que darán lugar a las opciones de la aplicación que se pueden evidenciar en la Figura 3.

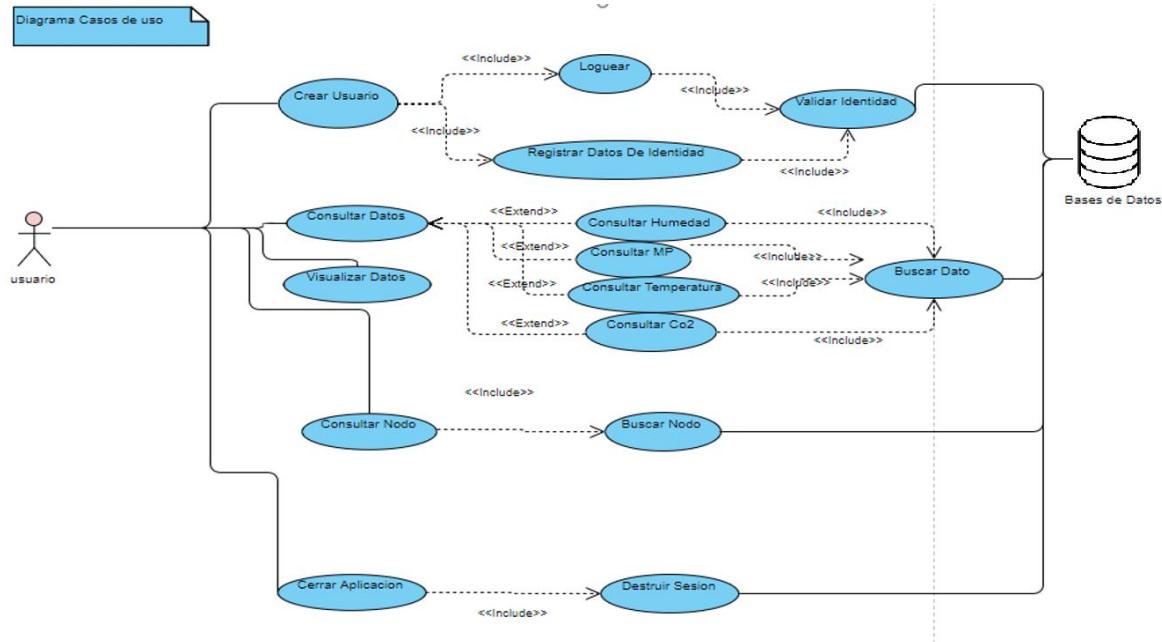


Figura 3- Diagrama de casos de uso

Donde el usuario podrá interactuar con el aplicativo siguiendo un flujo para las diferentes funciones y procesos que llama la aplicación por medio del WebService, siguiendo esta serie de pasos el usuario se encontrar con diferentes opciones, garantizando una experiencia óptima para acceder a los datos meteorológicos del sistema esto se puede evidenciar en la figura 4.

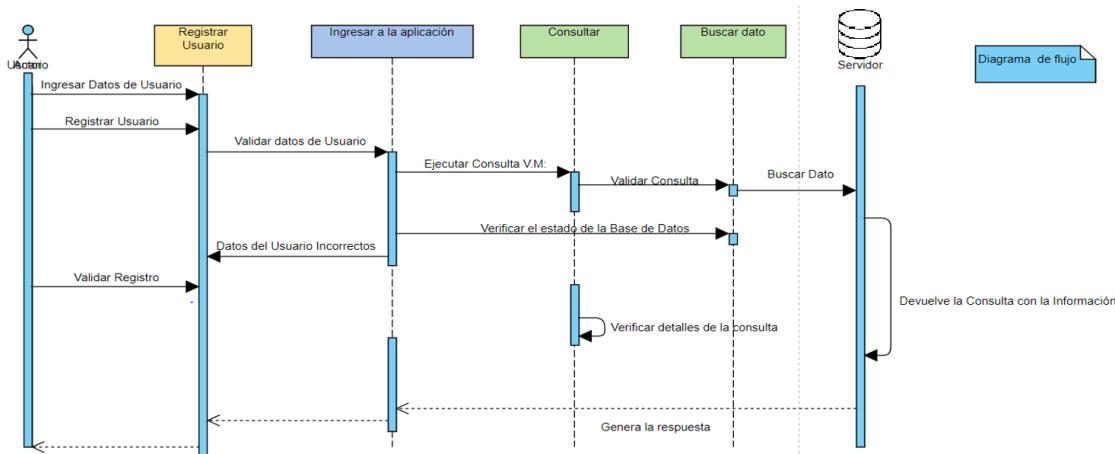


Figura 4 - Diagrama de flujo

Para que el usuario no rompa esta cadena se diseña una secuencia de comunicación que el sistema debe seguir para que cumpla con los procesos requeridos e integrados que se puede evidenciar en la figura 5, estos procesos se representan en funciones que siguen un camino para que su ciclo no sea roto, si no que el aplicativo se mueva en base a lo que el usuario desea dentro de las opciones dadas para la consulta de los datos meteorológicos.

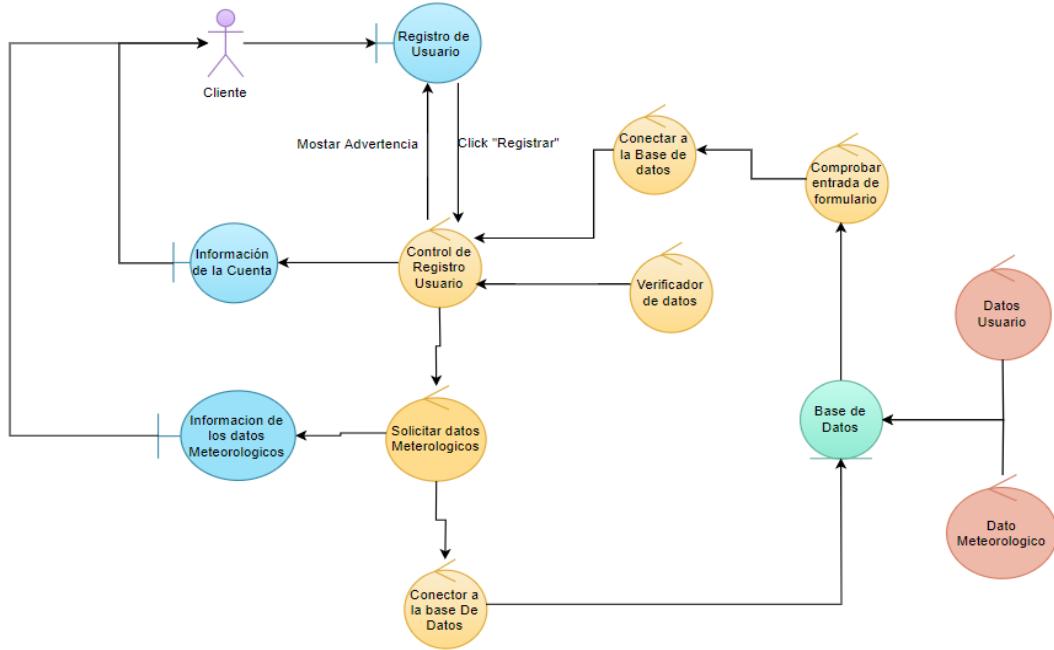


Figura 5 - Diagrama de comunicación

### 3.3. Esquema del mensaje

Tomando los principios de la arquitectura RESTful se logra crear un ambiente propicio para la manipulación de datos sin saturar el servidor usando procesos individuales que generan un utilidad parecida a RESTful pero se basa más en la utilidad ya que no es generada por un framework si no que se diseña para que la aplicación responda de manera rápida, adicionalmente bajando la carga en la aplicación ya que el webService al almacenar todos los métodos que manipulan los datos, lo demás son peticiones al mismo WebService para poder ejecutar la acción requerida dentro de los márgenes de la arquitectura ya montada, se desarrolló el webService usando los diferentes métodos GET, POST, PUT, DELETE creando una clase

controller la cual contiene todos estos métodos y pueden usarse solo llenando el método requerido y llamándolo a la aplicación.

```
if ($_SERVER['REQUEST_METHOD'] == 'GET')
{
    if (isset($_GET['username']))
    {
        //Mostrar un post
        $sql = $dbConn->prepare("SELECT * FROM usuario where username=:username");
        $sql->bindValue(':username', $_GET['username']);
        $sql->execute();
        header("HTTP/1.1 200 OK");

        echo json_encode( $sql->fetch(PDO::FETCH_ASSOC) );
        exit();
    }
}
```

Figura 6 - Esquema de mensaje

La arquitectura del WEB Servicie notifica al servidor el tipo de método http que va usar para realizar la acción cuando esta es validada , toma la variable que va ser en este caso tomada para una consulta por el método GET o POST , donde una variable toma la consulta no sin antes llamar una variable que se encarga de almacenar los datos de conexión, cuando realiza la consulta valida que el proceso sea exitoso, si el proceso tiene éxito el servidor devolverá un estado HTTP 200 que significa que el proceso fue exitoso y procede a volver la respuesta de la consulta en un formato JSON.

Esta arquitectura se aplica a todas las tablas del modelo de datos, el WebService junto con el modelo de datos nos darán un gran manejo para optimizar recursos y manejar datos de manera más eficiente y segura ya, que todas las tablas tienen la misma estructura el manejo se vuelve más fácil a medida que se le van agregando más métodos y acciones que tiene que realizar el sistema IoT en su totalidad, esta estructura es el WebService como tal aplicado a las 7 tablas con los diferentes métodos, dándole a la arquitectura gran ventaja en cuanto a cómo manipula los datos que envía la estación meteorológica .

### 3.4. Arquitectura GIEMA

En base a todo lo anterior analizado y estudiado se desarrolla un diagrama para representar la arquitectura de GIEMA.

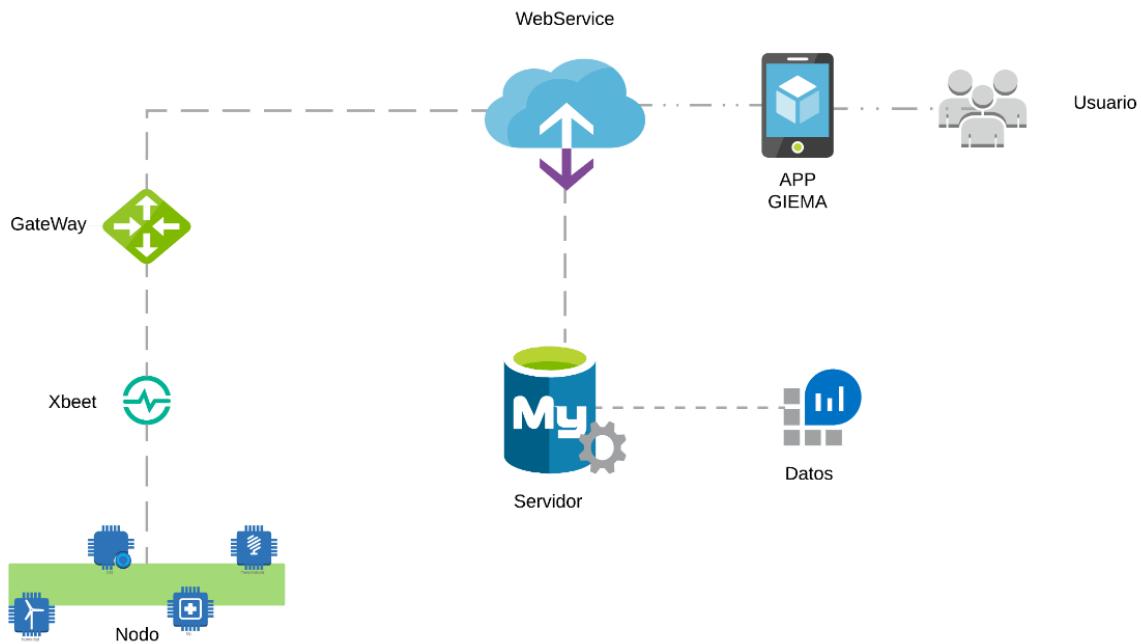


Figura 7 - Diagrama De Arquitectura GIEMA

Para el primer componente el cual es llamado nodo que se entiende por la parte física o estación meteorológica autónoma que está compuesta por sensores que son los encargados de recolectar los datos , para iniciar su flujo dentro de la estación meteorológica que provee la energía para el funcionamiento de los sensores y para ella misma ya que entendemos autónoma como la capacidad de poder obtener energía de manera autosuficiente en este caso por ejemplo por medio de energía fotovoltaica como se muestra en la figura 8.

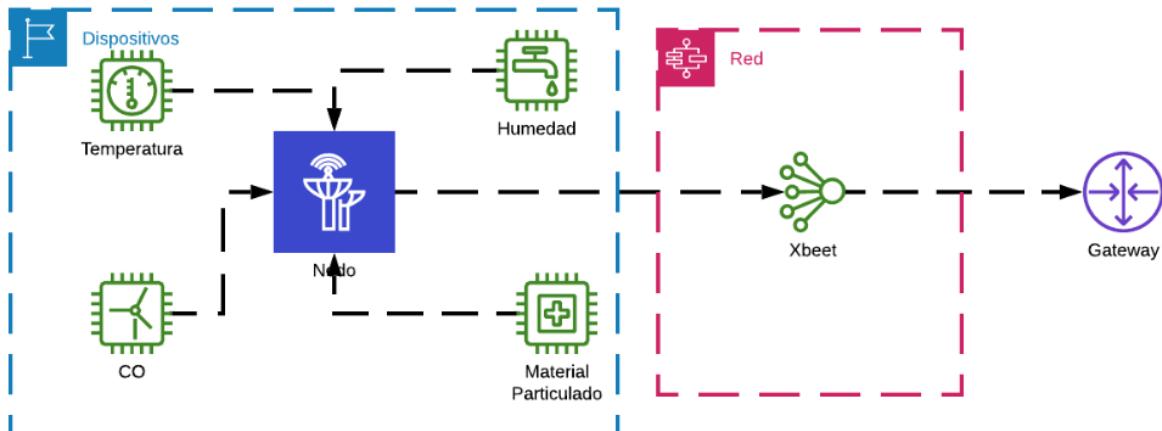


Figura 8 - Diagrama dispositivos GIEMA

El siguiente componente es el medio de conexión, un dispositivo de comunicación de red por medio de una IP para iniciar un proceso de protocolos de red usada para la conexión entre el nodo y el WebLogic, en este caso se plantea la tecnología de tarjetas Xbeet ya que son livianas y de fácil uso, además es un componente óptimo para nodos autónomos como se muestra en la figura 9.

El siguiente componente es el Gateway o la puerta de enlace que se traduce como el dispositivo que actúa de interfaz entre la parte física que es la estación meteorológica y el WebService como se muestra en la figura 9 , cabe resaltar que esto puede ser de carácter lógico , a lo que se plantea que los datos serán procesados de manera limpia en esta sección hasta este punto estos componentes serán emulados por medio del resultado final que serán los datos normalizados ya que no se cuenta con la parte física, pero se plantea una arquitectura sólida y flexible para su integración.

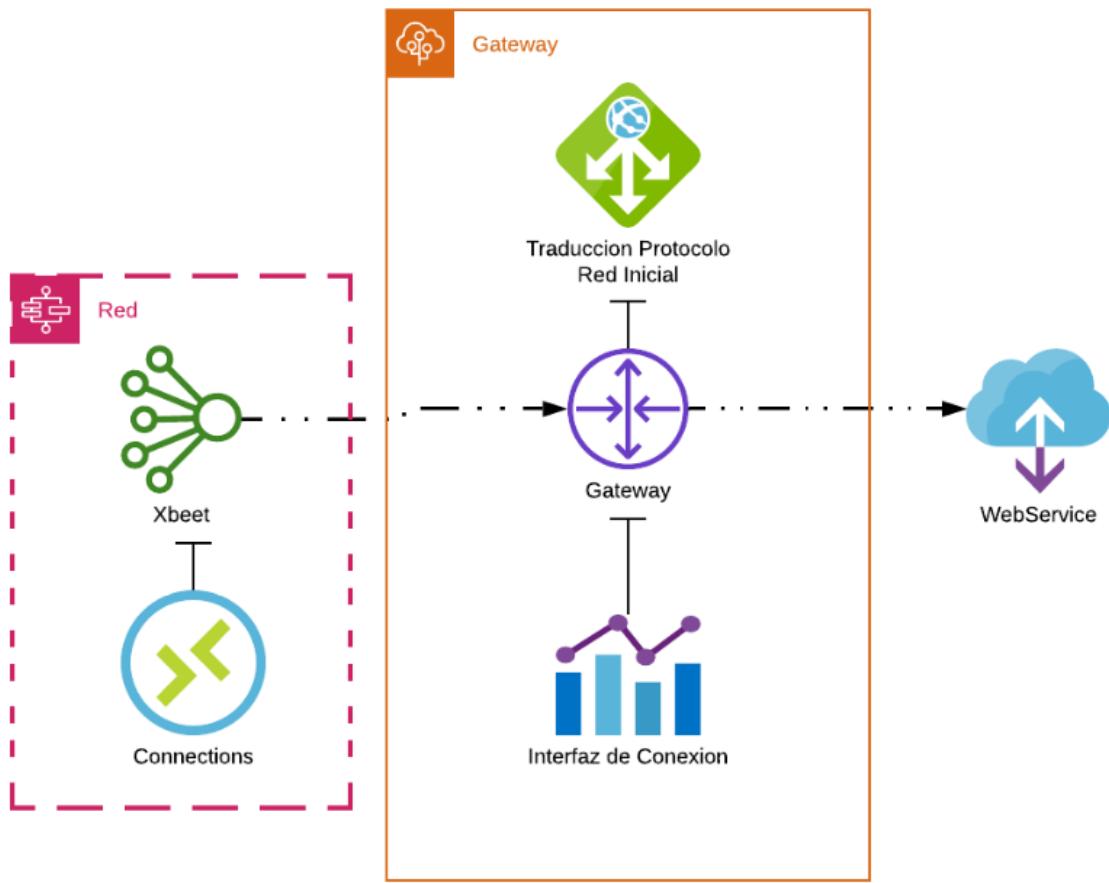


Figura 9 - Diagrama WebLogic GIEMA

El siguiente componente es el WebService, este componente recibe los datos y los almacena en el servidor , a este punto como se comentó estos datos fueron creados por medio de un generador de datos para simular el proceso final de la parte física como se muestra en la figura 10, adicionalmente el WebService es el puente entre la APP y el servidor , su función principal será procesar las funciones que solicite la app y enviarlas al servidor con la parametrización indicada según la función que se necesite realizar , para posteriormente recibir la respuesta y remitirla a la aplicación.

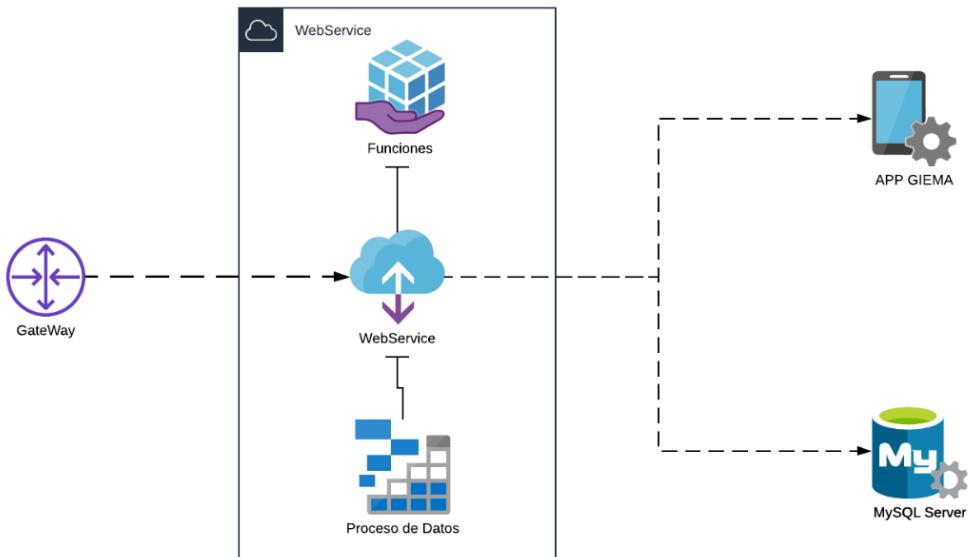


Figura 10 - Diagrama GateWay GIEMA

Y el ultimo componente que se muestra a continuación es el servidor el encargado de almacenar toda la información sobre el nodo , los datos meteorológicos sensores y usuario , su función también esta en responder a las peticiones enviadas por medio de la app y usando como puente de procesamiento el WebService cumplir la orden que le da dicha función , ya sea consulta de datos de una petición parametrizada o la edición adición de algún dato al que el usuario quiera hacer uso como se muestra en la figura 11.

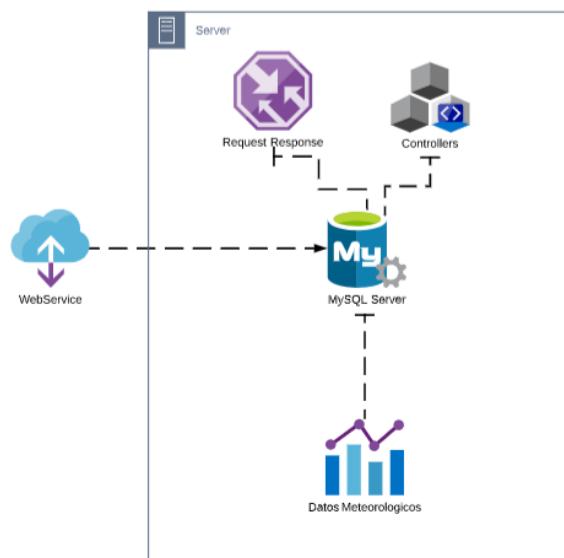


Figura 11 Server GIEMA

### 3.5. Mockups y Navegación

En base a la arquitectura planteada, se diseño el flujo de la aplicación, ya que este es el método por el cual el usuario se comunicará, el diseño en base en la simplicidad de buscar una consulta dato meteorológico sin mucho protocolo, y que toda la información se mueva por el Backend, esto con el propósito de ofrecer una experiencia fluida y rápida al usuario.

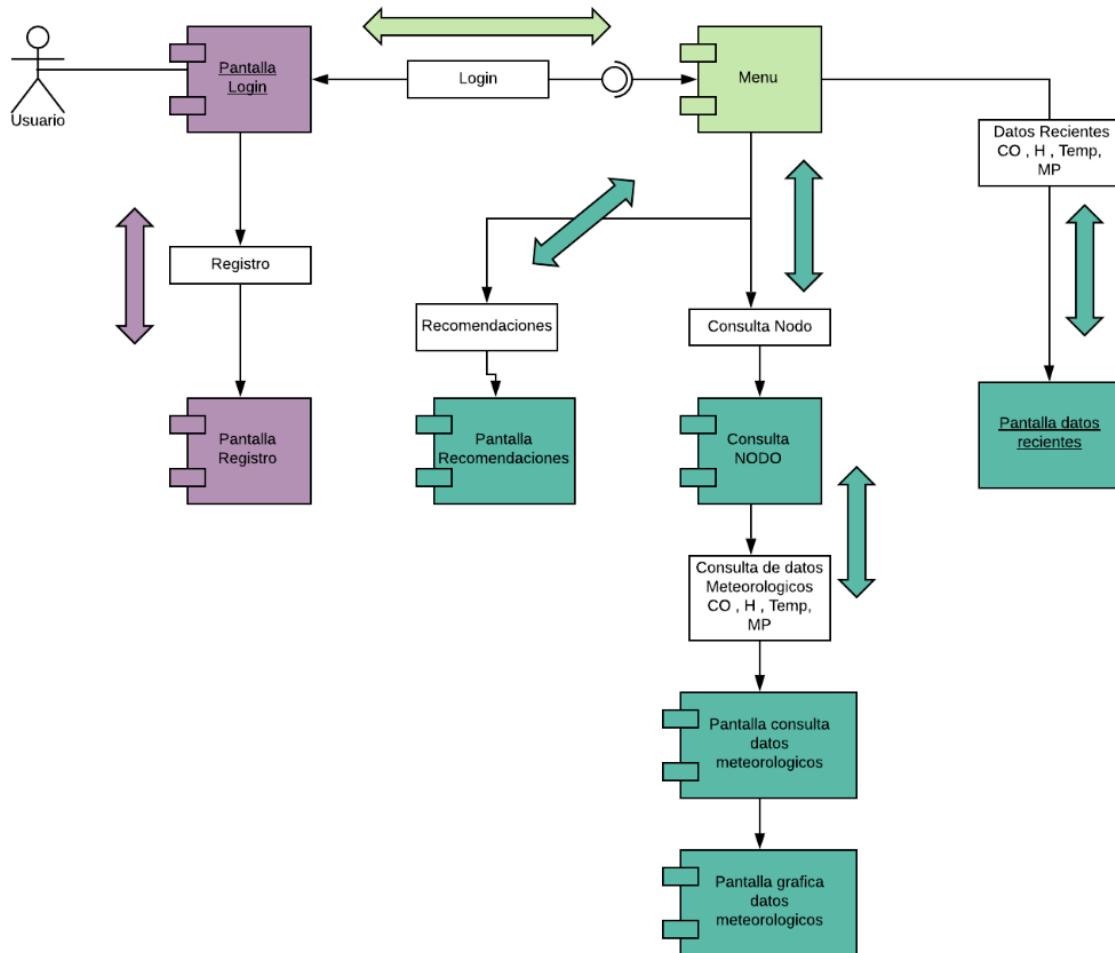
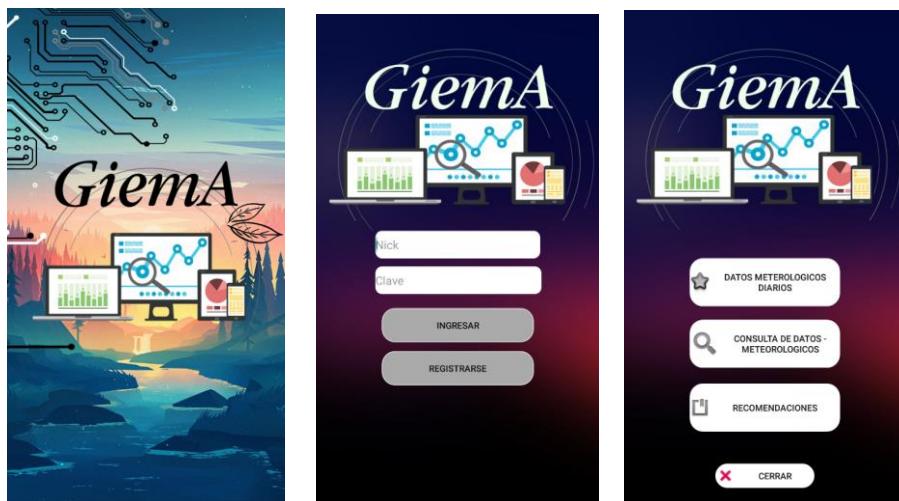


Figura 12 Diagrama de navegación

El flujo de la aplicación es el siguiente según lo planteado en el diagrama de navegación en la figura



**GIEMA-IoT: Herramienta móvil para la gestión de información  
generada por estaciones meteorológicas autónomas en un modelo IoT**

**ANEXO - D**  
**Pérez Martínez Jhon Alexander**

**Universidad de San Buenaventura, Sede Bogotá.  
Facultad de Ingeniería.  
Programas de Ingeniería de Sistemas  
Bogotá, Colombia  
2020**

## **Tabla de contenido**

4. Fase de desarrollo .....	3
4.1 Configuración del Entorno de Desarrollo Y Versiones.....	3
4.2. Desarrollo del WebService .....	5
4.3 Desarrollo de la aplicación. ....	7
4.4. Repositorio del código fuente .....	10

## 4. Fase de desarrollo

### 4.1 Configuración del Entorno de Desarrollo Y Versiones.

Para el desarrollo de la aplicación se realizaron algunas configuraciones en los entornos de desarrollo que se usaron para poder realizar de manera óptima la construcción del aplicativo móvil, como primera medida se configuro el entorno de desarrollo AndroidStudio en su versión:

Android Studio 3.5.1  
Build #AI-191.8026.42.35.5900203, built on September 25, 2019  
JRE: 1.8.0\_202-release-1483-b03 amd64  
JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o  
Windows 10 10.0

Cuando el entorno de desarrollo se encuentre listo para poder construir un aplicativo saldrá

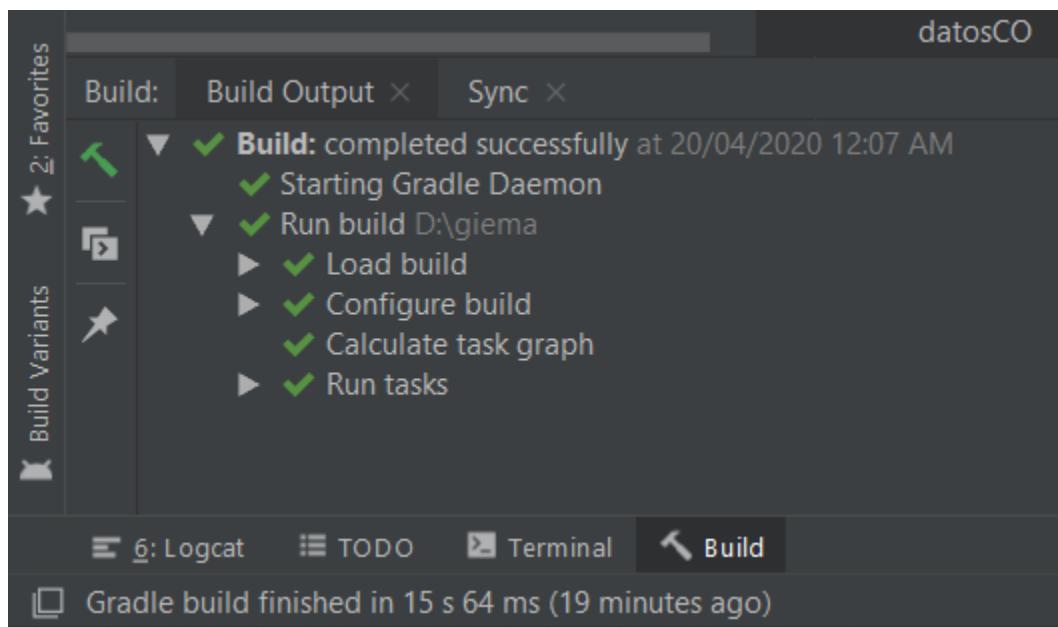


Figura 1 Entorno de desarrollo

Después de verificar que se todo se encuentre en orden se realizaran las siguientes configuraciones para que la aplicación pida los permisos necesarios para ejecutar la aplicación de manera exitosa, en este caso serán los encargados de pedir acceso a las conexiones de internet del dispositivo y permisos completos de red, para que la aplicación pueda conectarse a internet de manera automática.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

Figura 2 Permisos en la aplicación

Para el siguiente paso se importaron las librerías necesarias para que el proyecto funcione de manera óptima en este caso se usó las librerías como Volley que es la encargada de administrar las redes de la aplicación, la librería jplot encargada de la parte gráfica y ayudas visuales de la aplicación, la librería Loopj que se encarga de realizar peticiones HTTP de manera asíncrona, y la librería MPAndroidchart que es la encargada de realizar la graficas en la aplicación, entre otras.

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'androidx.appcompat:appcompat:1.0.2'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.0'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.1.1'
    implementation 'com.github.PhilJay:MPAndroidChart:v3.0.3'
    implementation 'com.loopj.android:android-async-http:1.4.9'
    implementation 'com.android.support:design:29.1.1'
    implementation 'com.android.volley:volley:1.1.0'
    implementation project(path: ':jplot1.2')
    implementation 'com.loopj.android:android-async-http:1.4.9'
}
```

Figura 3 Librerías y complementos

Una vez importada las librerías necesarias y aplicando los cambios, se deberá refrescar la estructura de desarrollo y a continuación de deberá importar los componentes para que la librería MPAndroidchart funcione de manera correcta con la siguiente configuración.



```
allprojects {
    repositories {
        google()
        jcenter()
        maven{url 'https://jitpack.io'}
```

Figura 4 Repositorios

Realizando esto el entorno de desarrollo está listo para ser usado.

## 4.2. Desarrollo del WebService

En la actualidad podemos encontrar gran variedad de formas arquitectónicas para el desarrollo del servicio de los diferentes sistemas que se encuentran hoy en día en el mundo, después del análisis realizado se esta arquitectura ya que se acopla perfectamente a lo que se busca para el presente proyecto ya que tiene las características para manipular los datos de las diferentes variables correctamente y conectarlos a una API de una manera muy dinámica sus características se centran permitir crear, leer, actualizar y borrar información de manera ágil usando métodos como POST , GET, para las operaciones anteriores necesitan una URL y métodos HTTP para accederlas, regresan los datos en formato JSON y pueden retornar códigos de respuesta como 200,201,404 etc. Lo que convierte a esta arquitectura en un aliado ideal para el desarrollo del presente proyecto

El desarrollo del WebService será realizado de manera manual en PHP 7.0 debido a que por temas de permisos donde se alojará parte de los archivos del proyecto del WebService en el Servidor de la Universidad San Buenaventura no se puede usar un Framework para realización de este proceso ya que se deben instalar ciertos componentes para que este funcione y no se cuenta con los permisos, supliendo este requerimiento se hace una análisis y se determina que PHP en su línea 7.0 en adelante puede ofrecernos las herramientas en cuanto a seguridad y viabilidad para el desarrollo del WebService.

Tomando los principios de la arquitectura RESTful se logra crear un ambiente propicio para la manipulación de datos sin saturar el servidor y adicionalmente bajando la carga en la aplicación ya que el webService al almacenar todos los métodos que manipulan los datos, lo demás son peticiones

al mismo WebService para poder ejecutar la acción requerida dentro de los márgenes de la arquitectura ya montada, se desarrolló el webService usando los diferentes métodos GET , POST, creando una clase controller la cual contiene todos estos métodos y pueden usarse solo llenando el método requerido y llamándolo a la aplicación como se ve en la figura 7.

```
if ($_SERVER['REQUEST_METHOD'] == 'GET')
{
    if (isset($_GET['username']))
    {
        //Mostrar un post
        $sql = $dbConn->prepare("SELECT * FROM usuario WHERE username=:username");
        $sql->bindValue(':username', $_GET['username']);
        $sql->execute();
        header("HTTP/1.1 200 OK");

        echo json_encode( $sql->fetch(PDO::FETCH_ASSOC) );
        exit();
    }
}
```

Figura 5 Método GET - Controller

La arquitectura del WEB Service notifica al servidor el tipo de método http que va usar para realizar la acción cuando esta es validada , toma la variable que va ser en este caso tomada para una consulta por el método GET , donde una variable toma la consulta no sin antes llamar una variable que se encarga de almacenar los datos de conexión, cuando realiza la consulta valida que el proceso sea exitoso, si el proceso tiene éxito el servidor devolverá un estado HTTP 200 que significa que el proceso fue exitoso y procede a volver la respuesta de la consulta en un formato JSON, más adelante se explicara más para que sirve este formato.

Esta arquitectura se aplica a todas las tablas del modelo de datos, el WebService junto con el modelo de datos nos darán un gran manejo para optimizar recursos y manejar datos de manera más eficiente y segura ya, que todas las tablas tienen la misma estructura el manejo se vuelve más fácil a medida que se le van agregando más métodos y acciones que tiene que realizar el sistema IoT en su totalidad, esta estructura es el WebService como tal aplicado a las 7 tablas con los diferentes métodos,

dándole a la arquitectura gran ventaja en cuanto a cómo manipula los datos que envía la estación meteorológica .

#### **4.3 Desarrollo de la aplicación.**

El desarrollo de aplicaciones móviles y su arquitectura es muy variada, en la actualidad existen ciertas arquitecturas que dominan su preferencia en uso, debido a que cuentan con una estructura más amigable de usar y cuidan el concepto de buenas prácticas en la programación, para el presente proyecto se determinó que esta arquitectura es la más adecuada debido a que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos donde el modelo accede a la capa de datos , el controlador recibe los eventos de la entrada y la vista recibe los datos del modelo y los muestra al usuario, es la más indicada, trabajando junto con el IDE de desarrollo que se seleccionó para la creación de la aplicación ,Android Studio se adapta esta arquitectura de desarrollo para poder seguir con las buenas prácticas de la programación, siguiendo los lineamientos dados por el IDE de desarrollo se puede concluir que es la mejor opción para interactuar con los datos debido a que el IDE Android Studio 3.5.1 trabaja con Layouts para el apartado grafico que se codifican en XML y se trabaja su backen en JavaClass ofrece una herramienta completa para el desarrollo de la aplicación móvil con buenas prácticas.

Se programó el aplicativo separando cada sección en clases , debido a que en Android se maneja algo que se llama Activity que son las pantallas que deben ser programadas individualmente para su funcionamiento y navegación se trató de aprovechar al máximo la arquitectura propuesta en el WebService para que todos métodos y acciones quedaran dentro de los controller y los demás métodos que se crearon adicionalmente para ejecutar acción como validación registro y consulta, para el desarrollo de la aplicación se usó el concepto de llamar todo desde el webService para aliviar carga tanto en el servidor como en el aplicativo donde se evaluó y se desarrolló sobre un sistema Android.5.5 en adelante, es decir que los celulares con ese sistema operativo Android soportaran la carga de la aplicación, dicho esto la aplicación funciona por medio de clases que será la pantalla

individual ligada a un Activity que es el apartado gráfico , dentro de cada clase se desarrolla el backend usando métodos para consumir el servicio desde el webservice , solicitamos el tipo de servicio que deseamos realizar y donde por ejemplo creando una clase stringRequest llamamos un método POST a una URL , este proceso es como ordenarle al servidor enviar una petición POST y el servicio que se solicita está en esa dirección lo que iniciara un proceso donde hay una respuesta y se crear un objeto JSON donde se almacena la cadena JSON que devuelve el servidor para almacenar los datos , ya en este punto los datos se encuentran en nuestra cadena JSON en nuestra aplicación solo por el momento de realizar la petición , ya después de finalizado este proceso se rompe lo que no le provoca carga a la aplicación ni al servidor, si la aplicación tiene éxito en el proceso devuelve un estado success 1, si se llega a fallar envía un error con el motivo todo esto se puede evidenciar.

```
private void llenarDatos() {
    String url = "http://software.ingusb.com/giema/giematest/datosGraficaTemperatura.php?fecha="+etFecha.getText().toString();
    client.get(url, new AsyncHttpResponseHandler() {
        //Creamos un metodo parametrizando el response
        @Override
        public void onSuccess(int statusCode, Header[] headers, byte[] responseBody) {
            if (statusCode == 200) {
                //Creamos el metodo que contiene la logica que administra los datos meteorologicos
                cargarDatos(new String(responseBody));
            }
        }

        @Override
        public void onFailure(int statusCode, Header[] headers, byte[] responseBody, Throwable error) {
        }
    });
}
```

*Figura 6 Ejemplo de petición*

Para la visualización de los datos se tuvo especial cuidado para tratar de forma individual cada sensor en su petición donde hay una clase para cada uno y poder visualizar los datos, pero hay una javaClass independiente que realiza la llamada del objeto para ser filtrado por la consulta SQL en el webService , recordando que la estructura de datos la tabla mensaje contiene los datos que envía la estación meteorológica de los diferentes sensores , es un constante flujo de información en un punto, pero en su almacenamiento está debidamente filtrado lo que realiza una petición que se mantiene en la aplicación solo durante su consulta como se evidencia..

```
V/AsyncHttpRH: Progress 38 from 38 (100%)
I/tagconvertstr: [{"nombreNodo":"U. San Buenaventura"}]
V/AsyncHttpRH: Progress 699 from 699 (100%)
I/tagconvertstr: [{"valor":"11"}, {"valor":"11"}, {"valor":"8"}, {"valor":"9"}, {"valor":"11"}, {"valor":"9"}, {"valor":"8"}, {"valor":"11"}, {"valor":"9"}, {"valor":"11"}, {"valor":"30"}, {"valor":"30"}, {"valor":"30"}, {"valor":"29"}, {"valor":"30"}, {"valor":"29"}, {"valor":"29"}, {"valor":"28"}, {"D/OpenGLRenderer: endAllStagingAnimators on 0xb3d70e00 (RippleDrawable) with handle 0xb3fdac0
D/OpenGLRenderer: endAllStagingAnimators on 0xa0cd5b80 (RippleDrawable) with handle 0xb3f18660
V/AsyncHttpRH: Progress 361 from 361 (100%)
I/tagconvertstr: [{"valor":"14"}, {"valor":"15"}, {"valor":"11"}, {"valor":"13"}, {"valor":"13"}, {"valor":"17"}, {"valor":"16"}, {"valor":"14"}, {"valor":"17"}, {"D/OpenGLRenderer: endAllStagingAnimators on 0xb3d71500 (RippleDrawable) with handle 0xb3f49f90
V/AsyncHttpRH: Progress 337 from 337 (100%)
I/tagconvertstr: [{"valor":"0"}, {"valor":"0"}, {"valor":"1"}, {"valor":"1"}, {"valor":"1"}, {"valor":"1"}, {"valor":"0"}, {"valor":"0"}, {"D/OpenGLRenderer: endAllStagingAnimators on 0xb3d71880 (RippleDrawable) with handle 0xaec29ba0
```

*Figura 7 Log de datos*

Y muestra estos datos solo durante la petición y de consulta, ya sea para su visualización o para ser graficados como se muestra en figura 8.



*Figura 8 Visualización de datos en la aplicación.*

#### **4.4. Repositorio del código fuente**

Una vez terminado el desarrollo se tuvo como resultado los siguientes componentes, por el lado de la aplicación GIEMA, que se encuentra disponible en el link a continuación: <http://software.ingusb.com/giema/descargaGiema.php>, se podrá descargar la aplicación haciendo clic en “Descargar App”.

Por su parte todo el proyecto GIEMA está disponible en el repositorio de github donde se encontrar el webservice y el modelo de datos, junto a la documentación de despliegue tanto del modelo de datos en el servidor como el uso de la app en el siguiente link: a continuación: <https://github.com/aledoc72/giema>.

Con esto concluye el segmento de desarrollo, ahora se procederá a evaluar el protocolo de pruebas realizado para certificar la funcionalidad del proyecto cumpliendo con las metas pactadas.

**GIEMA-IoT: Herramienta móvil para la gestión de información  
generada por estaciones meteorológicas autónomas en un modelo IoT**

**ANEXO – E**  
**Pérez Martínez Jhon Alexander**

**Universidad de San Buenaventura, Sede Bogotá.  
Facultad de Ingeniería.  
Programas de Ingeniería de Sistemas  
Bogotá, Colombia  
2020**

## **Tabla de Contenido**

5. Configuración y despliegue .....	3
5.1. Servidor.....	3
5.1.1. Configuración del ambiente al servidor .....	3
5.2. Configuración de la aplicación.....	6

## 5. Configuración y despliegue

Manual de configuración y despliegue-> proceso de acceso - acceso despliegue y de usuario como se instala a como se usa

Para el presente proyecto se tienen que realizar algunas configuraciones óptimas para que el proyecto funcione de manera correcta, a continuación, se dará las especificaciones de cómo funciona el proyecto

### 5.1. Servidor

Configuración Esquema de base de datos

Nota: Esta primera medida solo se aplicará si se desea realizar algún procedimiento dentro de base de datos para que siga creciente este proyecto

GIEMA tiene su propio esquema como primera medida será el acceso a este esquema, que en este caso se encuentra alojado en el servidor con los siguientes datos y debe ser configurado previamente de cualquier actividad que se requiera realizar en el proyecto

#### 5.1.1. Configuración del ambiente al servidor

- Para este proceso se usará el programa MySQL Workbench que se muestra en la Figura 1.

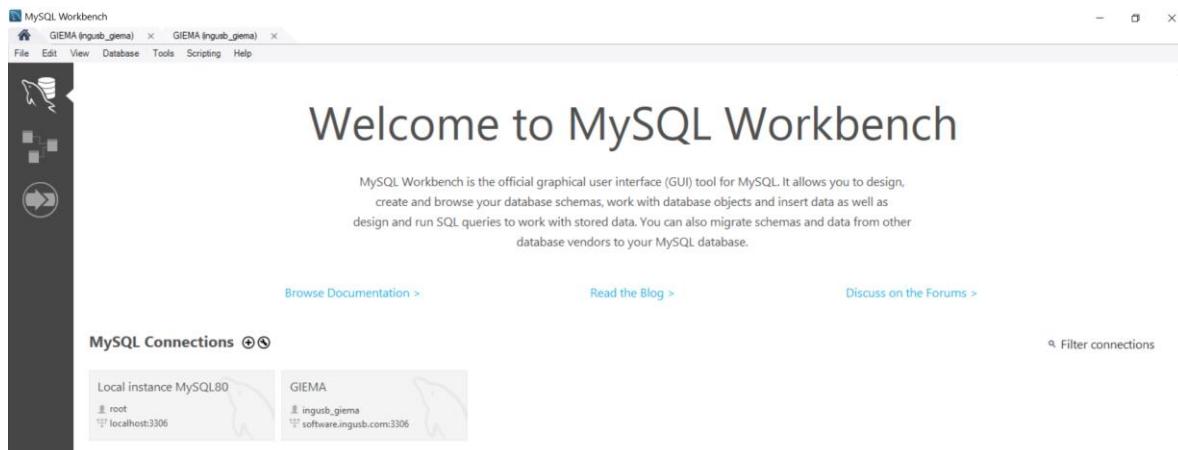


Figura 1 - IDE MySQL Workbench

- Para el siguiente paso configuraremos el acceso al dominio donde va a estar alojada nuestra información como se ve en la figura 2

## DDL

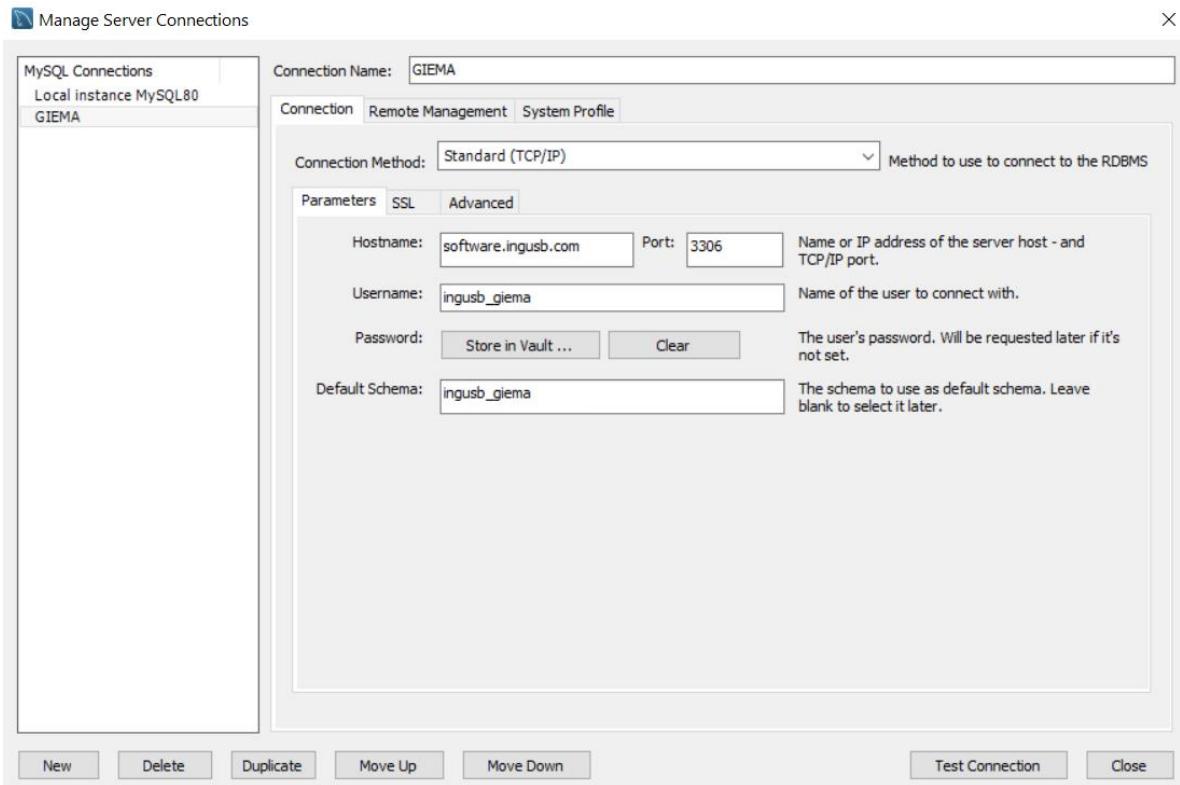


Figura 2 - Configuración acceso al dominio

Una vez adentro se creará el esquema y se importaran los datos como se muestra en la figura 3 y 4.

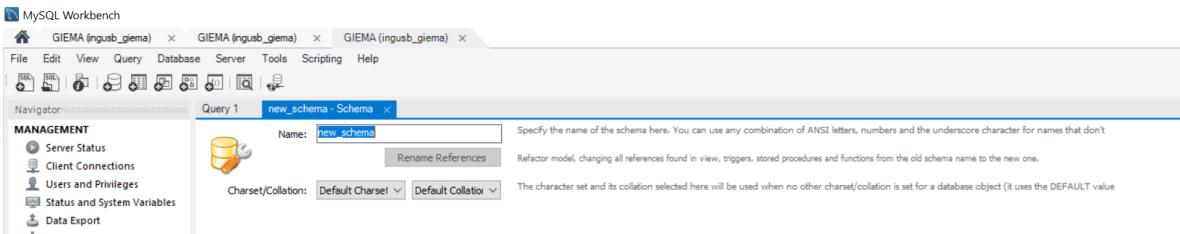


Figura 3 - Creación de esquema

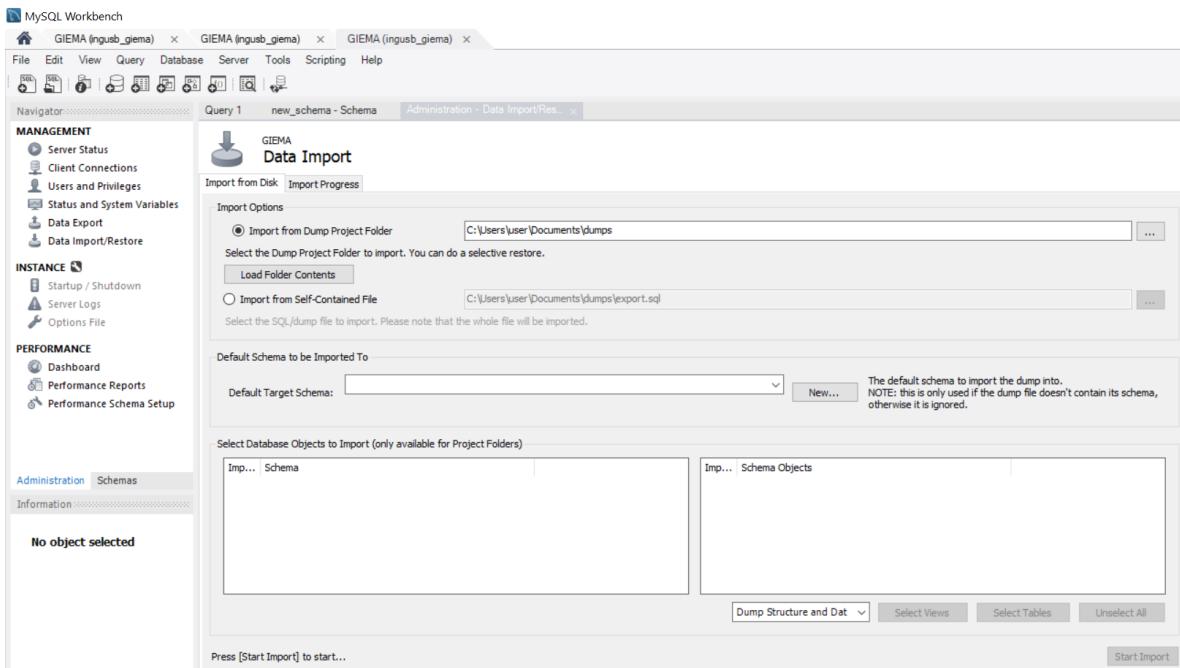


Figura 4 - Creación de esquema

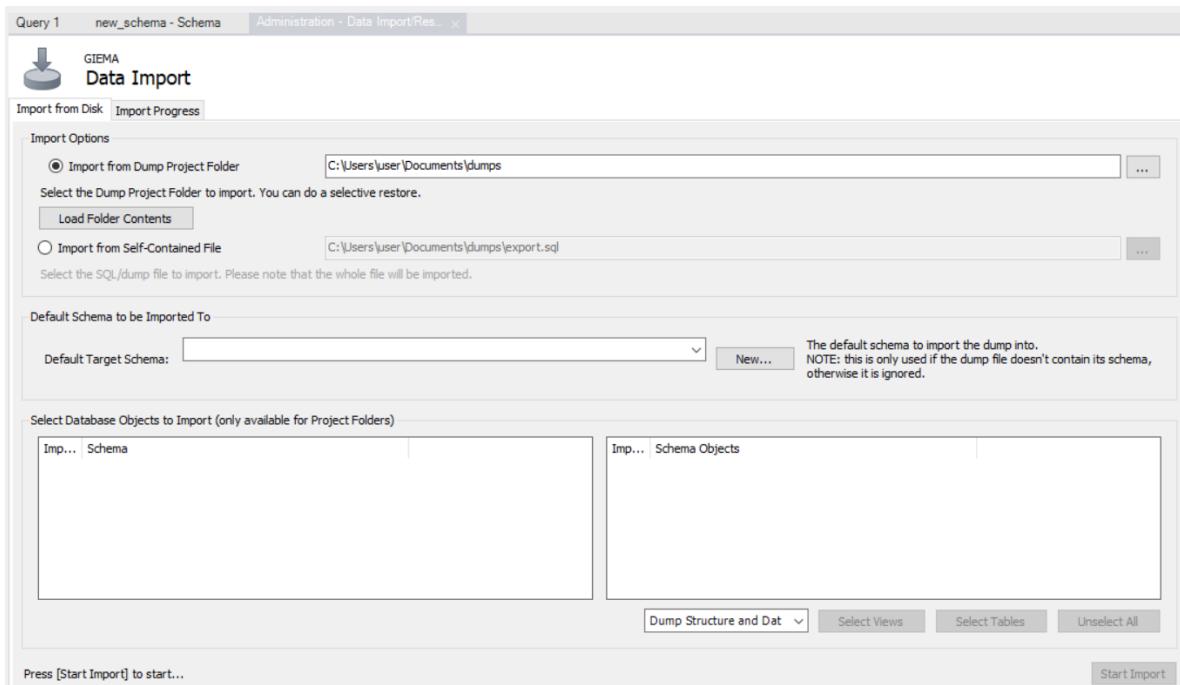


Figura 5 - Importación de esquema

- Una vez importado ya podremos tener acceso al esquema de la base de datos de GIEMA

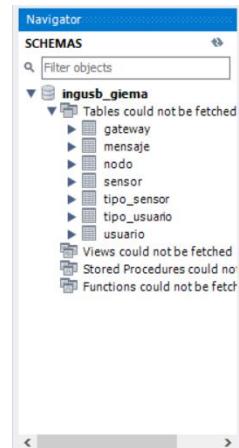


Figura 6 - Vista de tablas

## 5.2. Configuración de la aplicación

Para usar la siguiente app se deberán seguir esta lista de pasos

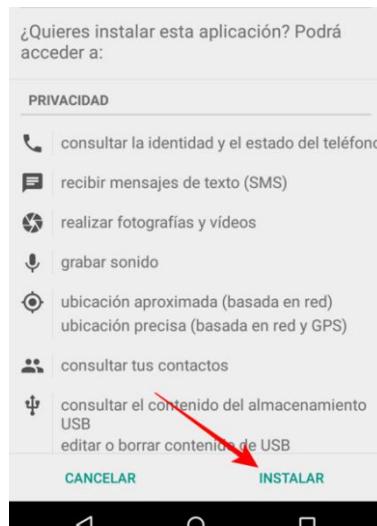
1. Ingresar a la página de GIEMA  
<http://software.ingusb.com/GIEMA/DescargaGiema.php>



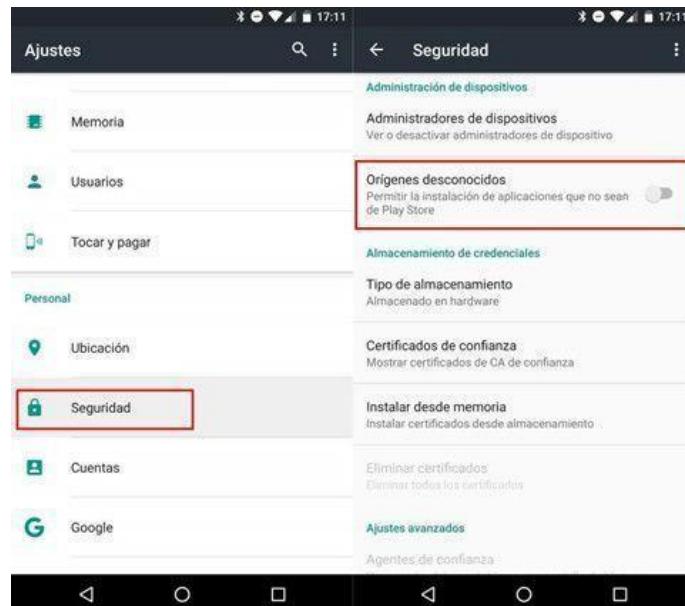
2. Al entra en la página descargaremos el APK de la aplicación.



3. Ejecutamos la aplicación y la instalamos

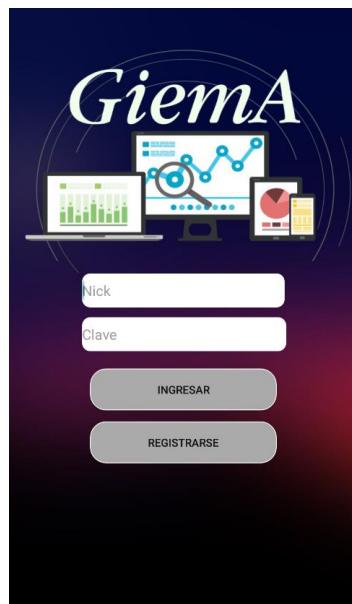


En algunos dispositivos se tendrá que otorgarle el siguiente permiso a la app

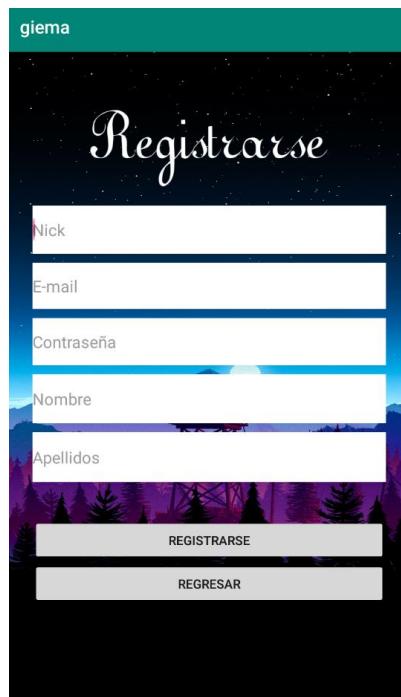


Provisional – acceso origenes desconocidos

4. Despues de instalar la aplicación el usuario podrá realizar el logueo.



5. Previamente deberá registrarse el usuario que desee usar la aplicación



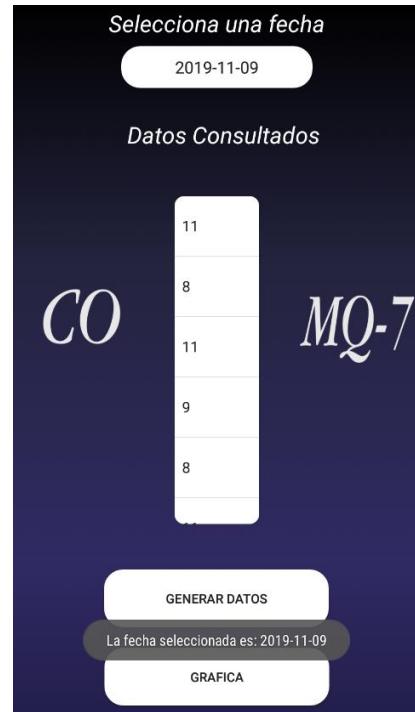
6. Al loguear el usuario tendrá acceso al menú y podrá accederá la consulta de datos



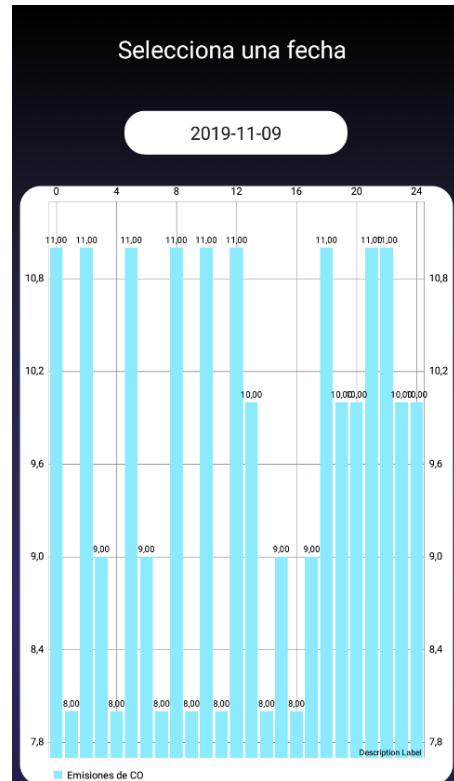
7. En la selección de datos el usuario podrá seleccionar el nodo que desee usar y consultar los datos meteorológicos según el sensor.



8. Al seleccionar un nodo podrá ver la información del estado de esa variable.



9. Y Al seleccionar graficar los datos, se graficarán y serán mostrados al usuario del valor del sensor contra la hora del dato normalizado.



**GIEMA-IoT: Herramienta móvil para la gestión de información  
generada por estaciones meteorológicas autónomas en un modelo IoT**

**ANEXO – F**  
**Pérez Martínez Jhon Alexander**

**Universidad de San Buenaventura, Sede Bogotá.  
Facultad de Ingeniería.  
Programas de Ingeniería de Sistemas  
Bogotá, Colombia  
2020**

## **Tabla de Contenido**

6. Pruebas .....	3
6.1. Pruebas App .....	3
6.1.3. Conclusiones De Las Pruebas Funcionales .....	19
6.2 Evaluación TAM .....	19

## 6. Pruebas

Para certificar el cumplimiento y desempeño de la app se generará una serie de pruebas para cerciorarse que la app es funcional y cumple con lo requerido.

### 6.1. Pruebas App

Se generó el siguiente set de pruebas para realizar esta evaluación

*Tabla 1 - Casos de prueba*

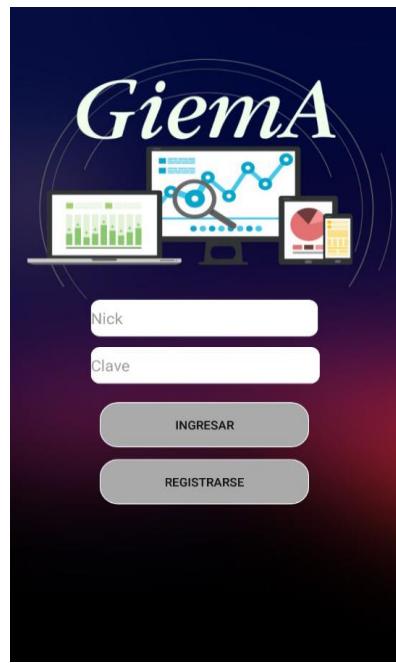
Nº Caso de Prueba	Nombre	Descripción
CP-001	Login	El usuario debe loguearse validando los datos que se encuentran en la base de datos Usuario y contraseña
CP-002	Registro	El usuario debe poder Registrarse en la App para que sus datos puedan ser usados en el login
CP-003	Consulta de nodo	El nombre del nodo el cual se encuentra el usuario actualmente debe ser llamado de la base de datos para tener trazabilidad del proceso
CP-004	Consulta de datos Sensor CO	Los datos meteorológicos CO deben ser llamados de la base de datos y mostrarse en la sección indicada
CP-005	Consulta de datos Sensor Humedad	Los datos meteorológicos Humedad deben ser llamados de la base de datos y mostrarse en la sección indicada
CP-006	Consulta de datos Sensor Temperatura	Los datos meteorológicos Temperatura deben ser llamados de la base de datos y mostrarse en la sección indicada
CP-007	Consulta de datos Sensor Material Particulado	Los datos meteorológicos MP deben ser llamados de la base de datos y mostrarse en la sección indicada

Nº Caso de Prueba	Nombre	Descripción
CP-008	Generación de grafica CO	Se debe generar una gráfica de los datos CO
CP-009	Generación de grafica Humedad	Se debe generar una gráfica de los datos Humedad
CP-010	Generación de grafica Temperatura	Se debe generar una gráfica de los datos Temperatura
CP-011	Generación de grafica Material Particulado	Se debe generar una gráfica de los datos MP
CP-012	Los datos deben filtrarse por fechas	Los datos consultados se podrán consultar por fecha

CP-001	Login	El usuario debe loguearse validando los datos que se encuentran en la base de datos Usuario y contraseña
--------	-------	--

Para este proceso se digitó un usuario y contraseña que estuviera registrado en la base de datos y se validó que se ejecutara la validación correspondiente, un proceso que dispara en WebService cuando el usuario le da ingresar.

+ Opciones		username	clave	nombres	apellidos	correo	tipo_usuario
<input type="checkbox"/>		andres	\$2y\$10\$Seew5tpFeg5qD7Eethl82uVByw5U9zwXeQhwMnsR5x...	andres	sanchez	andres@gmail.com	regular
<input type="checkbox"/>		cesar	\$2y\$10\$rln7c68YSbf69HBwV3BpweoYokZjMneGijbb31sXnqB...	cesar	perez	cesar@gmail.com	regular
<input type="checkbox"/>		hola	\$2y\$10\$Lcauv68JtmGUxyJJj7ffheKNlrozxjrKCsW/Zgh25...	jhon	hola	hola@gmail.com	regular
<input type="checkbox"/>		jhon	\$2y\$10\$WVvdw7j8wJRvZmGf0bj5uyJzlKF5L0aXXBCoB8SuUU...	jhon	perez	jhon@gmail.com	regular
<input type="checkbox"/>		kirara	\$2y\$10\$qqcQB7jr3Bg/yxYMkd5AneCYbAgO.50PapkLJowaT1...	Kirara	perez	kirara@gmail.com	regular
<input type="checkbox"/>		Lara7	\$2y\$10\$PMAnDWXojM6DP1E0gKeYujHO5b/CFMoVmZGr/Rj5p...	Laura	Lopez	hjkl@gmail.com	regular
<input type="checkbox"/>		Prueba1	\$2y\$10\$/DxQRIRnomtg3Qa2wMt7HOx5X3IxKqTfEE3Si/LH3...	prueba	test	prueba1@gmail.com	regular
<input type="checkbox"/>		yuki	\$2y\$10\$Tl6q4cmAAPvs8nUTVPOaG0sG0V3ZJhvIWUKKZI...	yuki	lopez	yuki.michi@gmail.com	regular



Validamos la información en la base de datos

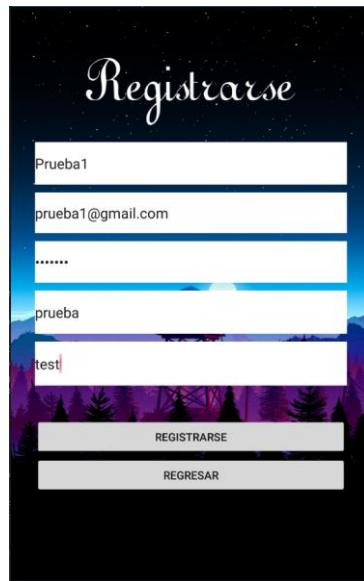
Posteriormente validamos que nos allá enviado a menú donde nos muestra un mensaje enviado por la app donde se realizó el proceso satisfactoria mente, el proceso no distingue mayúsculas o minúsculas.



Resultado: **EXITOSO**

CP-002	Registro	El usuario debe poder Registrarse en la App para que sus datos puedan ser usados en el login
--------	----------	--

Para esta prueba se digitaron datos que cumplan con los campos requeridos y posteriormente se enviaron al servidor para que queden almacenados y puedan usarse.



Evidencia de cómo se almacenan los datos

+ Opciones		username	clave	nombres	apellidos	correo	tipo_usuario
<input type="checkbox"/>		andres	\$2y\$10\$seew5tppFeg5qD7Ethl82uVBywU9zwXeEQhwMnsR5x...	andres	sanchez	andres@gmail.com	regular
<input type="checkbox"/>		cesar	\$2y\$10\$rln7c68YSbf69HBwV3BpweoYokZjMneGlibb31sXnqB...	cesar	perez	cesar@gmail.com	regular
<input type="checkbox"/>		hola	\$2y\$10\$Lcauv68JtmGUxYJj7ffheKNlrhozxjrKCsw/Zgh25...	jhon	hola	hola@gmail.com	regular
<input type="checkbox"/>		jhon	\$2y\$10\$Wv/dw7j8wJiRVzM6t0bj5uyJzlKF5L0aXXBCoB8SuUU...	jhon	perez	jhon@gmail.com	regular
<input type="checkbox"/>		kirara	\$2y\$10\$qqcQB7jr3Bg/yxYmkd5AneCybAgO.50PapkLJowaT1...	kirara	perez	kirara@gmail.com	regular
<input type="checkbox"/>		Lara7	\$2y\$10\$PMAnDWXojM6DP1E10gKeYujHO5b/CFMoVmzGpr/Rj5p...	Laura	Lopez	hjkl@gmail.com	regular
<input type="checkbox"/>		Prueba1	\$2y\$10\$/DxQRIRnomtg3Qa2wmf7Hox5X3fxKqTfEE3Si./LH3...	prueba	test	prueba1@gmail.com	regular
<input type="checkbox"/>		yuki	\$2y\$10\$Tol6q4vcAPVs8nUTVPOaG0sG0Vi3Z2JhvHWUKKZI...	yuki	lopez	yuki.michi@gmail.com	regular

Resultado: EXITOSO

CP-003	Consulta de nodo	El nodo debe el cual se encuentra el usuario actualmente debe ser llamado de la base de datos para tener trazabilidad del proceso
--------	------------------	---

Esta es la información del nodo en la base de datos.

+ Opciones		id	latitud	longitud	protocolo	gateway	nombreNodo
<input type="checkbox"/>		1	4°45'04"N	74°01'47"	xBee	1	U. San Buenaventura

Cuando ejecutamos seleccionar nodo, ejecuta la función para llamar este dato desde el servidor, como se ve el log

```
04-30 10:19:02.801 1747-1770/? I/ActivityManager: Displayed com.example.giema/.menu.seleccionarNodo: +666ms
04-30 10:19:02.814 3619-3619/com.example.giema V/AsyncHttpRH: Progress 38 from 38 (100%)
04-30 10:19:02.814 3619-3619/com.example.giema I/tagconvertstr: [{"nombreNodo":"U. San Buenaventura"}]]
```

Y por último se valida que muestre el dato.



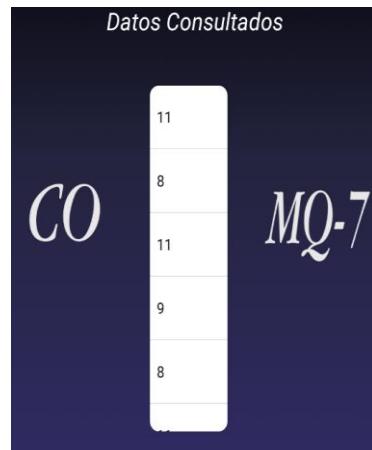
Resultado: EXITOSO

CP-004	Consulta de datos Sensor CO	Los datos meteorológicos CO deben ser llamados de la base de datos y mostrarse en la sección indicada
--------	-----------------------------	---

Evidencia de la información de los sensores

← T →	id	nombre	magnitud
<input type="checkbox"/>	1	CO	PPM
<input type="checkbox"/>	2	HUMEDAD	%
<input type="checkbox"/>	3	TEMPERATURA	°C
<input type="checkbox"/>	4	MP	µm-PPM

La aplicación genera datos según los parámetros que se envíe y se generan correctamente en este caso los datos pertenecen a CO



Miramos el log de datos que genera la aplicación.

```
04-30 10:21:19.007 3619-3619/com.example.giema V/AsyncHttpRH: Progress 659 from 659 (100%)
04-30 10:21:19.007 3619-3619/com.example.giema I/tagconvertstr: [[[{"valor": "11"}, {"valor": "11"}, {"valor": "8"}, {"valor": "9"}, {"valor": "8"}, {"valor": "11"}, {"valor": "9"}, {"valor": "8"}, {"valor": "11"}, {"valor": "9"}, {"valor": "8"}, {"valor": "11"}, {"valor": "10"}, {"valor": "10"}, {"valor": "11"}, {"valor": "11"}, {"valor": "10"}, {"valor": "8"}, {"valor": "9"}, {"valor": "11"}, {"valor": "8"}, {"valor": "11"}, {"valor": "10"}, {"valor": "8"}, {"valor": "9"}, {"valor": "11"}, {"valor": "10"}, {"valor": "9"}, {"valor": "8"}, {"valor": "10"}, {"valor": "10"}, {"valor": "9"}, {"valor": "11"}, {"valor": "10"}, {"valor": "10"}, {"valor": "9"}, {"valor": "11"}, {"valor": "10"}, {"valor": "11"}, {"valor": "10"}]]]
```

Y por último se comparó esta información con los datos almacenados en la base de datos.

+ Opciones			id	fecha	valor	sensor
		↔				
<input type="checkbox"/>		Editar		Copiar		Borrar
		1	2019-11-10	11	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		2	2019-11-09	11	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		3	2019-11-09	8	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		4	2019-11-10	9	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		5	2019-11-09	11	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		7	2019-11-09	9	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		8	2019-11-09	8	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		9	2019-11-09	11	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		10	2019-11-09	9	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		11	2019-11-09	8	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		12	2019-11-09	11	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		13	2019-11-09	8	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		15	2019-11-10	10	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		16	2019-11-09	11	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		17	2019-11-09	8	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		18	2019-11-10	10	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		19	2019-11-09	11	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		20	2019-11-10	11	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		21	2019-11-09	10	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		22	2019-11-10	8	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		23	2019-11-10	8	10	
<input type="checkbox"/>		Editar		Copiar		Borrar
		24	2019-11-10	9	10	

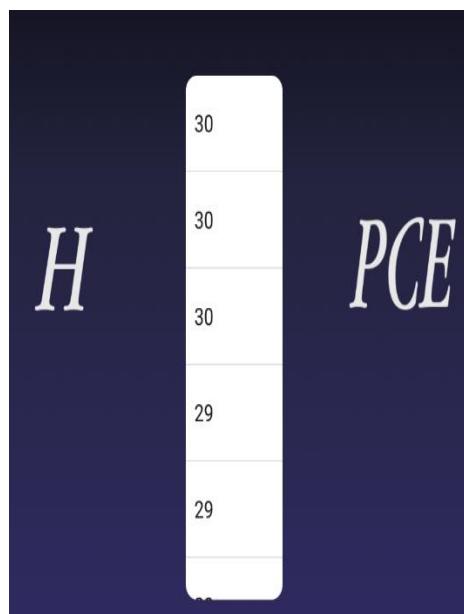
Resultado: EXITOSO

CP-005	Consulta de datos Sensor Humedad	Los datos meteorológicos Humedad deben ser llamados de la base de datos y mostrarse en la sección indicada
--------	-------------------------------------	--

## Evidencia de la información de los sensores

		id	nombre	magnitud
<input type="checkbox"/>	Editar	1	CO	PPM
<input type="checkbox"/>	Editar	2	HUMEDAD	%
<input type="checkbox"/>	Editar	3	TEMPERATURA	°C
<input type="checkbox"/>	Editar	4	MP	µm-PPM

La aplicación genera datos según los parámetros que se envíe y se generan correctamente en este caso los datos pertenecen a Humedad



Miramos el log de datos que genera la aplicación.

```
04-30 10:24:10.042 3619-3619/com.example.giema I/tagconvertstr: [[[{"valor":"30"}, {"valor":"30"}, {"valor":"30"}, {"valor":"29"}, {"valor":"30"}, {"valor":"25"}, {"valor":"28"}, {"valor":"29"}, {"valor":"30"}, {"valor":"25"}, {"valor":"29"}, {"valor":"28"}], [{"valor":"29"}, {"valor":"29"}, {"valor":"28"}, {"valor":"30"}, {"valor":"28"}, {"valor":"25"}, {"valor":"29"}, {"valor":"29"}, {"valor":"28"}, {"valor":"25"}, {"valor":"29"}, {"valor":"28"}]]
```

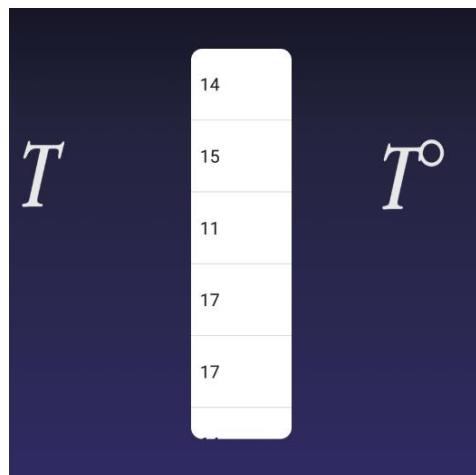
Y por último se comparó esta información con los datos almacenados en la base de datos.

	+ Opciones				<b>id</b>	<b>fecha</b>	<b>valor</b>	<b>sensor</b>			
	<input type="checkbox"/>		Editar		Copiar		Borrar	53	2019-11-09	11	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	54	2019-11-10	13	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	55	2019-11-10	13	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	56	2019-11-09	17	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	57	2019-11-10	16	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	58	2019-11-10	14	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	59	2019-11-09	17	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	60	2019-11-09	14	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	61	2019-11-09	12	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	62	2019-11-09	17	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	63	2019-11-09	12	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	64	2019-11-09	12	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	65	2019-11-09	12	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	66	2019-11-09	14	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	67	2019-11-10	11	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	68	2019-11-10	11	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	69	2019-11-09	13	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	70	2019-11-09	11	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	71	2019-11-10	12	20
	<input type="checkbox"/>		Editar		Copiar		Borrar	72	2019-11-10	12	20
	<input checked="" type="checkbox"/>		Editar		Copiar		Borrar	Consola			

Resultado: EXITOSO

CP-006	Consulta de datos Sensor Temperatura	Los datos meteorológicos Temperatura deben ser llamados de la base de datos y mostrarse en la sección indicada
--------	--------------------------------------	--

La aplicación genera datos según los parámetros que se envíe y se generan correctamente en este caso los datos pertenecen a CO



Miramos el log de datos que genera la aplicación.

```
04-30 10:21:19.007 3619-3619/com.example.giema V/AsyncHttpRH: Progress 699 from 699 (100%)
04-30 10:21:19.007 3619-3619/com.example.giema I/tagconvertstr: [[[{"valor":"11"}, {"valor":"11"}, {"valor":"8"}, {"valor":"9"}, {"valor":"11"}, {"valor":"8"}, {"valor":"11"}, {"valor":"9"}, {"valor":"11"}, {"valor":"11"}, {"valor":"8"}, {"valor":"10"}, {"valor":"11"}, {"valor":"8"}, {"valor":"10"}, {"valor":"11"}, {"valor":"11"}, {"valor":"8"}, {"valor":"10"}, {"valor":"9"}, {"valor":"11"}, {"valor":"8"}, {"valor":"10"}, {"valor":"11"}, {"valor":"9"}, {"valor":"10"}, {"valor":"8"}, {"valor":"11"}, {"valor":"10"}, {"valor":"9"}, {"valor":"11"}, {"valor":"10"}, {"valor":"11"}, {"valor":"11"}, {"valor":"10"}]]
```

Y por último se comparó esta información con los datos almacenados en la base de datos.

	id	fecha	valor	sensor
<input type="checkbox"/>   	78	2019-11-10	29	30
<input type="checkbox"/>   	79	2019-11-10	29	30
<input type="checkbox"/>   	80	2019-11-09	30	30
<input type="checkbox"/>   	81	2019-11-09	29	30
<input type="checkbox"/>   	82	2019-11-09	29	30
<input type="checkbox"/>   	83	2019-11-09	28	30
<input type="checkbox"/>   	84	2019-11-09	29	30
<input type="checkbox"/>   	85	2019-11-09	29	30
<input type="checkbox"/>   	86	2019-11-10	28	30
<input type="checkbox"/>   	87	2019-11-09	30	30
<input type="checkbox"/>   	88	2019-11-09	28	30
<input type="checkbox"/>   	89	2019-11-10	28	30
<input type="checkbox"/>   	90	2019-11-10	29	30
<input type="checkbox"/>   	91	2019-11-09	29	30
<input type="checkbox"/>   	92	2019-11-09	29	30
<input type="checkbox"/>   	93	2019-11-09	28	30

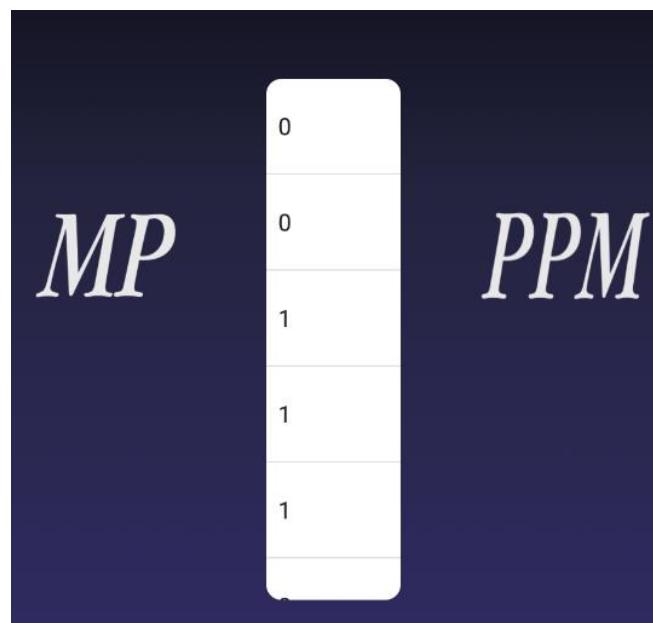
Resultado: EXITOSO

CP-007	Consulta de datos Sensor Material Particulado	Los datos meteorológicos MP deben ser llamados de la base de datos y mostrarse en la sección indicada
--------	---	---

Evidencia de la información de los sensores

	id	nombre	magnitud
<input type="checkbox"/>   	1	CO	PPM
<input type="checkbox"/>   	2	HUMEDAD	%
<input type="checkbox"/>   	3	TEMPERATURA	°C
<input type="checkbox"/>   	4	MP	µm-PPM

La aplicación genera datos según los parámetros que se envíe y se generan correctamente en este caso los datos pertenecen a Material Particulado



Miramos el log de datos que genera la aplicación.

	<input type="button" value="←"/>	<input type="button" value="→"/>		<input type="button" value="▼"/>	<b>id</b>	<b>fecha</b>	<b>valor</b>	<b>sensor</b>
<input type="checkbox"/>				Borrar	103	2019-11-10	1	40
<input type="checkbox"/>				Borrar	104	2019-11-10	1	40
<input type="checkbox"/>				Borrar	105	2019-11-10	1	40
<input type="checkbox"/>				Borrar	106	2019-11-10	0	40
<input type="checkbox"/>				Borrar	107	2019-11-10	0	40
<input type="checkbox"/>				Borrar	108	2019-11-09	1	40
<input type="checkbox"/>				Borrar	109	2019-11-10	1	40
<input type="checkbox"/>				Borrar	110	2019-11-09	1	40
<input type="checkbox"/>				Borrar	111	2019-11-10	0	40
<input type="checkbox"/>				Borrar	112	2019-11-10	1	40
<input type="checkbox"/>				Borrar	113	2019-11-10	0	40
<input type="checkbox"/>				Borrar	114	2019-11-10	0	40
<input type="checkbox"/>				Borrar	115	2019-11-09	1	40
<input type="checkbox"/>				Borrar	116	2019-11-09	1	40
<input type="checkbox"/>				Borrar	117	2019-11-10	1	40
<input type="checkbox"/>				Borrar	118	2019-11-10	1	40
<input type="checkbox"/>				Borrar	119	2019-11-10	0	40
<input type="checkbox"/>				Borrar	120	2019-11-10	1	40
<input type="checkbox"/>				Borrar	121	2019-11-09	1	40
<input type="checkbox"/>				Borrar	122	2019-11-09	0	40

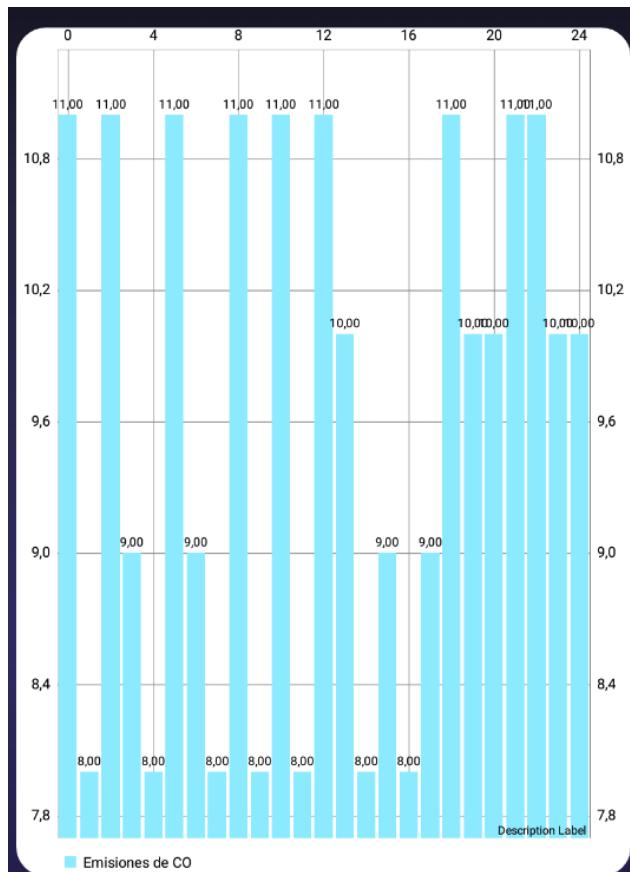
Resultado: EXITOSO

CP-008	Generación de grafica CO	Se debe generar una gráfica de los datos CO
--------	--------------------------	---

Evidencia de la información de los sensores

		id	nombre	magnitud
<input type="checkbox"/>	Editar  Copiar  Borrar	1	CO	PPM
<input type="checkbox"/>	Editar  Copiar  Borrar	2	HUMEDAD	%
<input type="checkbox"/>	Editar  Copiar  Borrar	3	TEMPERATURA	°C
<input type="checkbox"/>	Editar  Copiar  Borrar	4	MP	µm-PPM

En base a los datos recopilados generamos una grafica



En esta grafica generada a partir de los datos del servidor se evidencian las emisiones de CO en una fecha específica

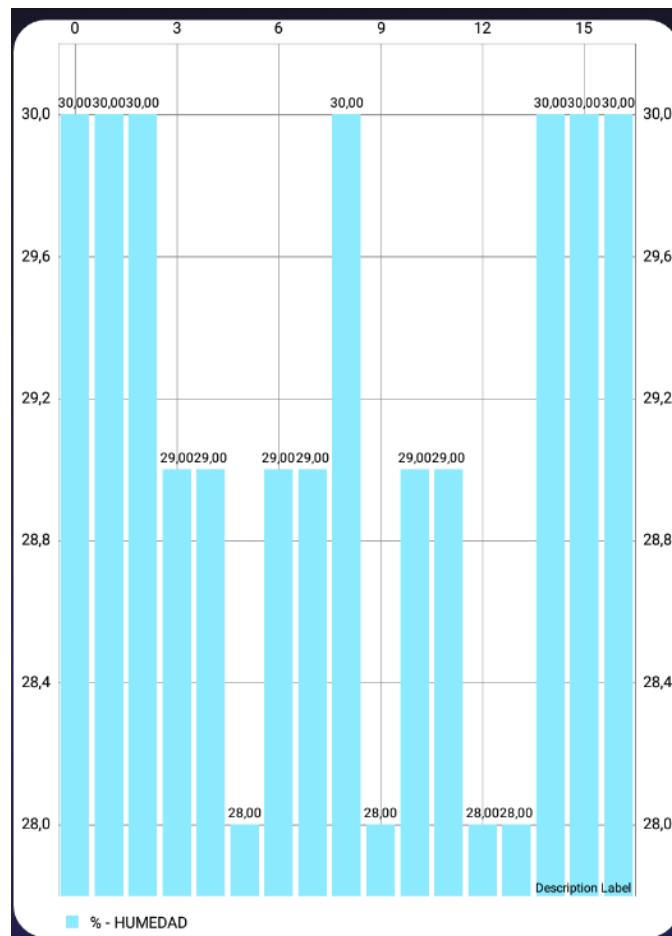
Resultado: **EXITOSO**

CP-009	Generación de grafica Humedad	Se debe generar una gráfica de los datos Humedad
--------	----------------------------------	---

Evidencia de la información de los sensores

		id	nombre	magnitud
<input type="checkbox"/>	Editar	1	CO	PPM
<input type="checkbox"/>	Editar	2	HUMEDAD	%
<input type="checkbox"/>	Editar	3	TEMPERATURA	°C
<input type="checkbox"/>	Editar	4	MP	µm-PPM

En base a los datos recopilados generamos una grafica



Evidenciamos el WebService que consume y como se ejecuta la consulta y la petición y la envía en JSON ENCODE

```

?php

require_once 'connect.php';

$fecha = $_GET["fecha"];
$sql = "SELECT valor FROM mensaje WHERE sensor = 30 and fecha='$fecha';"

$datos = Array();
$response = mysqli_query($conn, $sql);

while ($row = mysqli_fetch_object($response)){
    $datos[] = $row;
}
echo json_encode ($datos);
mysqli_close($conn);
?>

```

Y se compara que los datos de la gráfica sean los mismos de la base de datos donde se generó la consulta

	<input type="checkbox"/>	Editar	Copiar	Borrar	<b>id</b>	<b>fecha</b>	<b>valor</b>	<b>sensor</b>
		<input type="checkbox"/>			53	2019-11-09	11	20
		<input type="checkbox"/>			54	2019-11-10	13	20
		<input type="checkbox"/>			55	2019-11-10	13	20
		<input type="checkbox"/>			56	2019-11-09	17	20
		<input type="checkbox"/>			57	2019-11-10	16	20
		<input type="checkbox"/>			58	2019-11-10	14	20
		<input type="checkbox"/>			59	2019-11-09	17	20
		<input type="checkbox"/>			60	2019-11-09	14	20
		<input type="checkbox"/>			61	2019-11-09	12	20
		<input type="checkbox"/>			62	2019-11-09	17	20
		<input type="checkbox"/>			63	2019-11-09	12	20
		<input type="checkbox"/>			64	2019-11-09	12	20
		<input type="checkbox"/>			65	2019-11-09	12	20
		<input type="checkbox"/>			66	2019-11-09	14	20
		<input type="checkbox"/>			67	2019-11-10	11	20
		<input type="checkbox"/>			68	2019-11-10	11	20
		<input type="checkbox"/>			69	2019-11-09	13	20
		<input type="checkbox"/>			70	2019-11-09	11	20
		<input type="checkbox"/>			71	2019-11-10	12	20
		<input type="checkbox"/>			72	2019-11-10	12	20

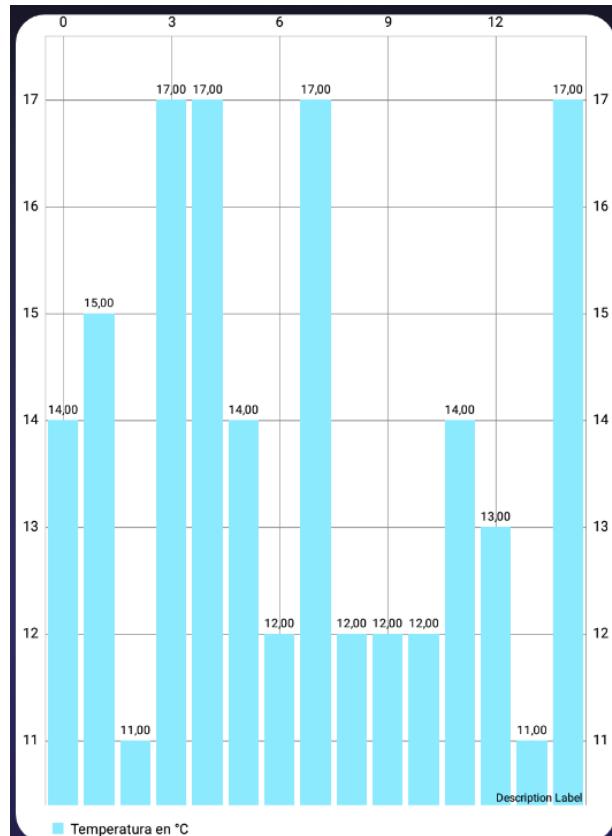
Resultado: **EXITOSO**

CP-010	Generación de grafica Temperatura	Se debe generar una gráfica de los datos Temperatura
--------	-----------------------------------	--

Evidencia de la información de los sensores

	id	nombre	magnitud
<input type="checkbox"/>	1	CO	PPM
<input type="checkbox"/>	2	HUMEDAD	%
<input type="checkbox"/>	3	TEMPERATURA	°C
<input type="checkbox"/>	4	MP	µm-PPM

En base a los datos recopilados generamos una grafica



En esta grafica generada a partir de los datos del servidor se evidencian las emisiones de CO en una fecha específica

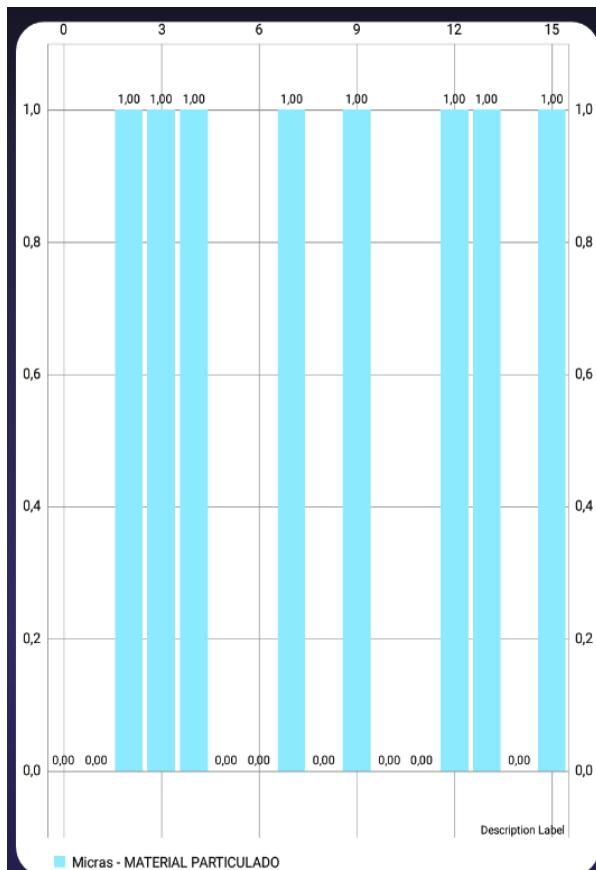
Resultado: **EXITOSO**

CP-011	Generación de grafica Material Particulado	Se debe generar una gráfica de los datos MP
--------	--	---

Evidencia de la información de los sensores

		id	nombre	magnitud
<input type="checkbox"/>	Editar  Copiar  Borrar	1	CO	PPM
<input type="checkbox"/>	Editar  Copiar  Borrar	2	HUMEDAD	%
<input type="checkbox"/>	Editar  Copiar  Borrar	3	TEMPERATURA	°C
<input type="checkbox"/>	Editar  Copiar  Borrar	4	MP	μm-PPM

En base a los datos recopilados generamos una grafica



En esta grafica generada a partir de los datos del servidor se evidencian las emisiones de CO en una fecha específica

Resultado: **EXITOSO**

CP-012	Los datos deben filtrarse por fechas	Los datos consultados se podrán consultar por fecha
--------	--------------------------------------	---

Se pueden filtrar los datos para cada sensor mediante una parametrización realizada en cada una de la consulta lo que permite seleccionar los datos con un datePicker que se despliega al tocar el listView lo que genera el dato con el parámetro escogido y genera la consulta.



Resultado: **EXITOSO**

### **6.1.3. Conclusiones De Las Pruebas Funcionales**

- App se comporta de manera fluida y responde bien a los tiempos estándares aun cuando depende del servidor.
- Los datos generados por filtro se generan solo día consultado y cada consulta se ejecuta por separado para no saturar la app ni el servidor.
- La aplicación grafica los datos de manera correcta para su visualización, para la generación de la gráfica también se debe filtrar por fecha los datos que se desea consultar ya que como son bastantes datos se necesita un parámetro para filtrar toda esta información.
- El login realiza su funcionalidad correctamente, aunque es un proceso sencillo la administración de datos no estaba contemplada en esta app por lo que solo se probó que se realizará la validación correctamente.
- El registro es un proceso normal dentro de los estándares de lo que debe ser, ya que encripta la contraseña y pide datos básicos, nada sensible.

### **6.2 Evaluación TAM**

Para esta fase del proyecto, se le aplicó la evaluación TAM a los siguientes Ingenieros y en base a sus repuestas y observaciones se analizó y replanteó lo que se debe mejorar y lo que se debe resaltar de la app y de GIEMA como tal, en este punto se comenzará con presentar a los ingenieros que participaron en la evaluación cada uno desempeñándose en un campo diferente y su opinión fue de gran importancia para este proyecto:

- Ingeniero de sistemas con experiencia de 5 años en gerencia de proyectos, aplicaciones móviles, Alejandro Poveda Ingeniero de sistemas – especialista en gerencia de proyectos – Evl 1.
- Ingeniera de Telecomunicaciones con dos años de experiencia en sistemas de conexiones inalámbricas, Natalia Torres Ingeniero Telecomunicaciones – Evl 2.
- Ingeniero de Telecomunicaciones con 3 años de experiencias en modelos IoT y conexiones vía satélite, Pablo Nonsoque Ingeniero de Telecomunicaciones – Evl 3.

- Ingeniero Mecánico con 2 años de experiencia en la automatización de máquinas y e ingeniería de materiales, Sebastián Florez Ingeniero Mecánico – Maestría en ingeniería materiales – Evl 4.

En base a estos resultados donde se evaluaron diferentes aspectos, se toma como referencia de modelo de mejora de la aplicación para una última versión antes de ser liberada

Para esta sección se estructuro una evaluación TAM en base a la aplicación en la cual se viera reflejada los diferentes componentes y funcionalidades de la aplicación, así como su estructura su diseño, la usabilidad y funcionalidades, esta sección los ingenieros que ayudaron con la retroalimentación evaluación y observaciones dadas al probar esta aplicación para crecimiento técnico del proyecto y retro alimentación y crecimiento personal como profesional a su autor.

Para la esta evaluación técnica realizada por profesionales el primer paso a evaluar fue el diseño, para esta sección se plantearon los siguientes ítems como criterio de evaluación de app para que puedan ser calificados.

Diseño	El texto usado en la aplicación es difícil de leer
	La paleta de colores usada en el prototipo tiene un contraste inadecuado
	Las imágenes de fondo no opacan la visibilidad de las pantallaz
	Los datos se vizualizan de manera armonioza
	El color de las graficas diferencia cada dato Señalado
	El diseño del prototipo no se ajusta a la funcionalidad del mismo
	La información transmitida con código de color son congruentes

En esta sección se observa todo lo referente al diseño de la aplicación como modelo de evaluación en este caso resaltando como se estructuro su diseño y la experiencia de usuario al ser manejada.

El segundo aspecto para evaluar como criterio de evaluación fue la usabilidad, que tal iba al usuario usando la aplicación y su comodidad por medio de la experiencia de usuario manejo de excepciones y comportamiento de app a nivel técnico.

Usabilidad	Visibilidad del estado del sistema (el prototipo no da retroalimentación de las acciones realizadas por el usuario)
	Relación entre el sistema y el mundo real (el prototipo no presenta la información de manera sencilla y entendible)
	Control y libertad del usuario (el prototipo no presenta las opciones de salida de funciones o no se puede regresar)
	Consistencia y estándares (los iconos y frases del prototipo realizan diferentes acciones)
	Prevención de errores (el prototipo no es claro en los pasos al realizar una acción)
	Reconocimiento antes que recuerdo (el prototipo no presenta un acceso fácil a las acciones y funcionalidades)
	Flexibilidad y eficiencia de uso (el prototipo no presenta funciones específicas para cada tipo de usuario)
	Estética y diseño minimalista (el prototipo esta sobrecargado de información irrelevante y componentes innecesario o redundantes, no es minimalista)
	Ayudar a los usuarios a reconocer (el prototipo no informa los errores y colapsa)
	La generacion de graficas es rapida (el tiempo respuesta de la app para este proceso no tiene contratiempos)

Siguiente y último aspecto fue la funcionalidad, donde se evaluó que ofrecía cada componente y como podía ser integrado a nivel funcional creando una utilidad al usuario, para este paso los ingenieros probaron las opciones que tenían disponibles en la aplicación y realizaron sus observaciones.

Funcionalidad	La aplicación realiza un proceso de validacion (Login) para ingresar
	La aplicación filtra los datos por fecha
	La aplicación genera un reporte por fecha de los datos consultados
	La aplicación grafica los datos de los sensores meteorologicos
	La aplicacion genera una consulta individual por cada sensor disponible
	La aplicación muestra informacion sobre el nodo consultado

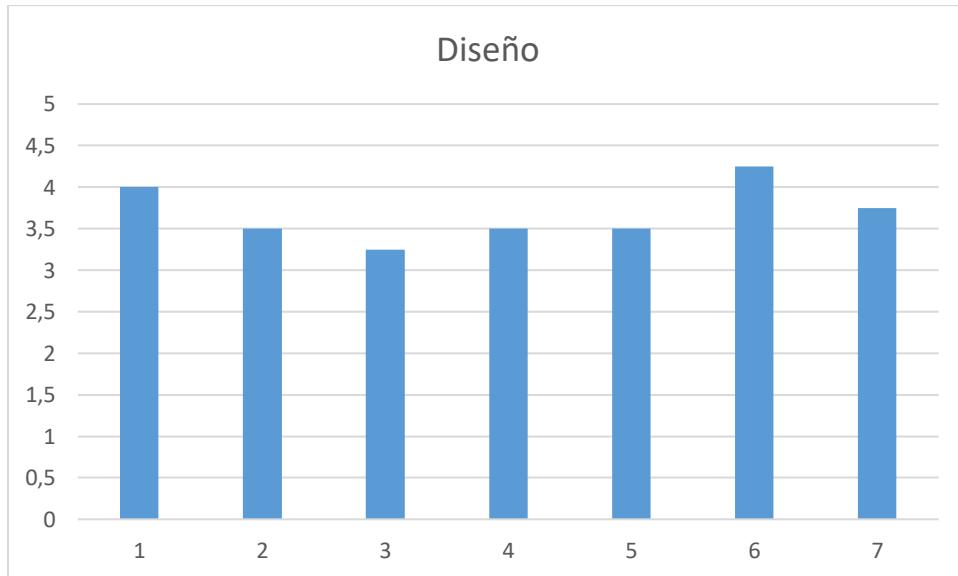
Donde los criterios de evaluación se basan en la siguiente el siguiente rango

5. Si considera que el aspecto es excelente
4. Si considera que el aspecto es bueno
3. Si considera que el aspecto es aceptable
2. Si considera que el aspecto regular
1. Si considera que el aspecto es malo

## Diseño

Según los criterios de evaluación se saca una media en base a las respuestas de los ingenieros que aplicaron la evaluación, para evaluar en este caso la parte de diseño, cada criterio está marcado con un número para identificar la media de cada dato en la evaluación TAM.

1	El texto usado en la aplicación es difícil de leer
2	La paleta de colores usada en el prototipo tiene un contraste inadecuado
3	Las imágenes de fondo no opacan la visibilidad de las pantallaz
4	Los datos se vizualizan de manera armoniosa
5	El color de las graficas diferencia cada dato Señalado
6	El diseño del prototipo no se ajusta a la funcionalidad del mismo
7	La información transmitida con código de color son congruentes



Grafica 1 - Criterio de Evaluación Diseño

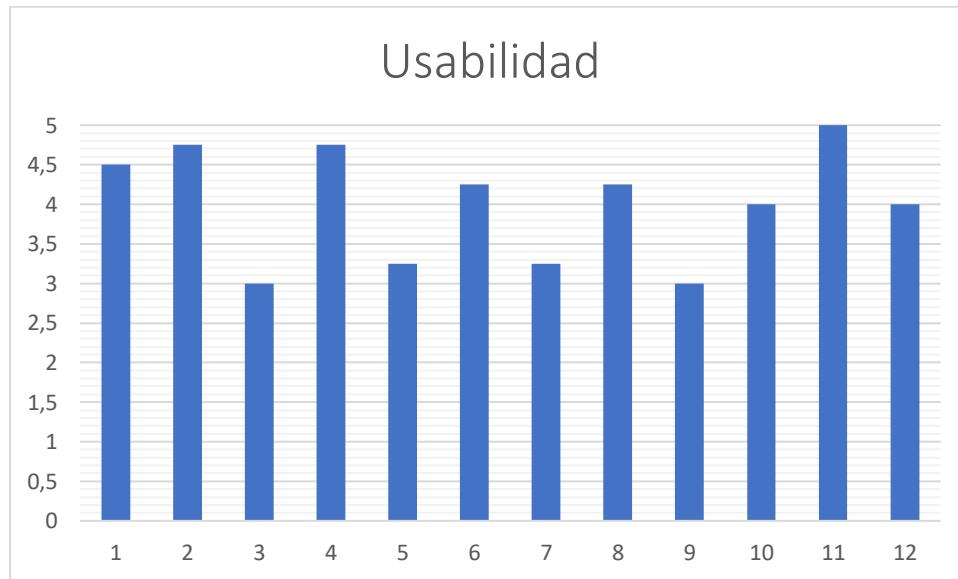
Como se muestra en la gráfica 1, los resultados para diseño fueron satisfactorios en la gran mayoría donde la distribución de la aplicación y la forma en la que se transmite la aplicación fueron gratamente evaluadas y aceptadas por los ingenieros evaluadores, de parte de los aspectos recomendados y observaciones para futuros cambios con opción de mejora encontramos los siguientes aspectos.

- Las imágenes en su dimensión deberían ser un poco menos pesadas y esto ayudaría a aliviar la carga en la app cuando se carga cada pantalla
- El apartado grafico puede mejorar haciendo énfasis en una buena paleta de colores lo que mejorara notablemente el aspecto de la aplicación y la experiencia de usuario
- Las gráficas pueden ser más coloridas y con mayor amplitud, un cambio en el aspecto que podría darle un aire grafico a la app

## Usabilidad

Los criterios de evaluación planteados aquí son a todas esas características a la que el usuario estará conectado de manera continua durante toda su experiencia en app, ya que la APP de GIEMA es una app netamente concentrada en la información de datos meteorológicos, fue un gran enfoque concentrarnos mucho en la usabilidad y los eventos que se pueden presentar al usar la app de manera continua , manejo de errores que en si es la experiencia de usuario dentro de app , para que sea lo más cómoda posible, se evaluaron los siguientes criterios de evaluación.

1	Visibilidad del estado del sistema (el prototipo no da retroalimentación de las acciones realizadas por el usuario)
2	Relación entre el sistema y el mundo real (el prototipo no presenta la información de manera sencilla y entendible)
3	Control y libertad del usuario (el prototipo no presenta las opciones de salida de funciones o no se puede regresar)
4	Consistencia y estándares (los iconos y frases del prototipo realizan diferentes acciones)
5	Prevención de errores (el prototipo no es claro en los pasos al realizar una acción)
6	Reconocimiento antes que recuerdo (el prototipo no presenta un acceso fácil a las acciones y funcionalidades)
7	Flexibilidad y eficiencia de uso (el prototipo no presenta funciones específicas para cada tipo de usuario)
8	Estética y diseño minimalista (el prototipo esta sobrecargado de información irrelevante y componentes innecesario o redundantes, no es minimalista)
9	Ayudar a los usuarios a reconocer (el prototipo no informa los errores y colapsa)
10	La generacion de graficas es rapida (el tiempo respuesta de la app para este proceso no tiene contratiempos)
11	El tiempo de respuesta de la aplicación es rápida
12	La aplicación realiza peticiones indicadas en las opciones establecidas



Grafica 2 - Usabilidad Criterio TAM

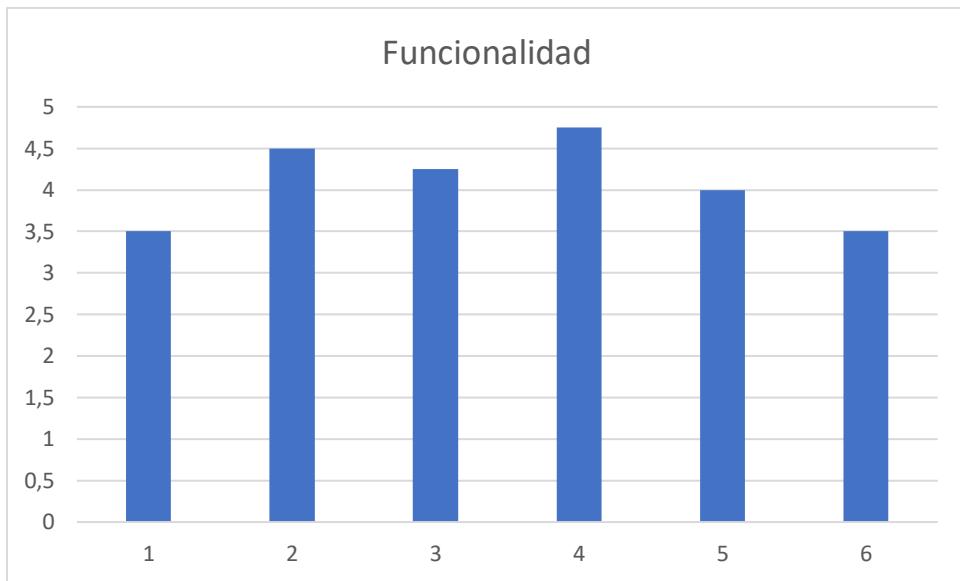
Como se muestra en la gráfica 2, aunque el desempeño de la app fue evaluado de manera sobresaliente y se resaltó de gran manera su rapidez y facilidad al usar, algunos aspectos necesitan un especial trabajo dentro de los criterios de evaluación se tienen como observación los siguientes ítems, para futuras correcciones que aunque no impactan de manera negativa en la app son grandes observaciones,

- Implementación de botones para el retorno de la app, esto con el fin que se use botones Return de la misma app sin necesidad de usar el botón atrás integrado del celular
- La parametrización de errores debe ser más clara, con un mensaje entendible para un usuario normal, sin conocimiento de que está sucediendo detrás de la app.
- Aclarar un poco más a la hora de implementar y parametrizar los errores de la app.

## Funcionalidad

En esta sección se evaluó la parte funcional de app como respondía a cada función y como se desempeñó la app bajo estas pruebas aplicadas por nuestros evaluadores.

1	La aplicación realiza un proceso de validacion (Login) para ingresar
2	La aplicación filtra los datos por fecha
3	La aplicación genera un reporte por fecha de los datos consultados
4	La aplicación grafica los datos de los sensores meteorológicos
5	La aplicación genera una consulta individual por cada sensor disponible
6	La aplicación muestra información sobre el nodo consultado



*Grafica 3 - Funcionalidad Criterio TAM*

Según la gráfica 3, los criterios funcionales fueron muy bien acogidos ya que, aunque tuvimos dos aspectos regulares fueron los aspectos que implementaron más por regulación de app que por pedido de requerimiento con el propósito esta app siga creciendo en el futuro, entre las observaciones más relevantes tenemos,

- El login es un proceso muy básico, aunque por debajo está bien estructurado ya que encripta la contraseña y valida los datos con métodos POST, es necesario agregar un poco más de seguridad y gestión al usuario, pero como esto no está contemplado es solo una observación para futuros cambios que se quieran hacer cuando esta app crezca.
- También se aconseja que para un futuro cambio se pueda consultar más datos acerca del nodo no solo el nombre, con estos dos ítems la app puede seguir creciendo y ser mejor en futuros cambios.

Para concluir con el análisis se realiza un ponderado general para obtener una conclusión clara sobre los datos obtenidos mediante esta evaluación TAM como se muestra en la Tabla 2.

*Tabla 2 - Resultados Evaluación TAM*

Criterios	Evl 1	Evl 2	Evl 3	Evl 4	Promedio del Criterio
Diseño	3,86	3,71	3,86	3,71	3,79
Usabilidad	4,00	3,92	4,08	4,00	4,00
Funcionalidad	4,17	3,83	4,00	4,33	4,08
Promedio evaluador	4,01	3,82	3,98	4,02	4,01

En base al análisis anterior como se muestra en la tabla 1 se concluye que la aplicación cumple con los requerimientos planteados y aunque hay elementos agregados que se integraron para fines de que la experiencia de usuario sea más grata se espera que en el futuro esta aplicación siga creciendo, el objetivo de aplicar esta evaluación a Ingenieros reales fue el dar un espacio de opinión técnica y dar un panorama más amplio de lo que es GIEMA y cómo puede mejorar en un futuro, dicho esto también se desea concluir a nivel general sobre su funcionamiento diseño y usabilidad en base a las observaciones que dieron los evaluadores los siguientes aspectos.

- Mejora en el apartado gráfico, ya que el objetivo de GIEMA es la practicidad para la lectura y manejo de datos de manera informativa un punto que se puede mejorar a futuro es la reinención de un nuevo apartado grafico esto con el fin de hacer la aplicación más agradable y con esto aumentar la experiencia de usuario notablemente, entre estas están las cajas de texto forma de la fuente, imágenes de fondo botones y graficas.
- Para el criterio de usabilidad solo se tocaron puntos como más orientación en la información en la aplicación y mejora e integración en botones de return dentro de la aplicación, además de manejo de errores y excepciones más claras, pueden ayudar mucho a pulir la aplicación en el futuro.
- En cuanto a la funcionalidad , aunque fue uno de los elementos mejor valorados , las observaciones estuvieron más orientadas a temas como el login o visualización del nodo , temas que se desarrollaron como un adicional y fueron bien recibidas esto con el propósito de que en el futuro se siga integrando más funcionalidades respecto a estos componentes adicionales como manejo de información personal y del nodo, dicho esto podemos concluir

que la funcionalidad en lo que reafirma sus objetivos están completos y las observaciones solo van dirigidas a futuros ítems de mejora en el futuro.

Se concluye que GIEMA cumple con las funcionalidades y requerimientos planteados durante su diseño y gracias a las observaciones de los evaluadores que dan un panorama más amplio respecto a lo técnico de cómo puede crecer esta app en el futuro y ser integrada en temas más grandes, ya que fue bien recibida su diseño y su propósito para que en el futuro sea una app en constante mejora y pueda ser integrada a trabajos de datos ambientales más pesados.