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

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 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.

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.

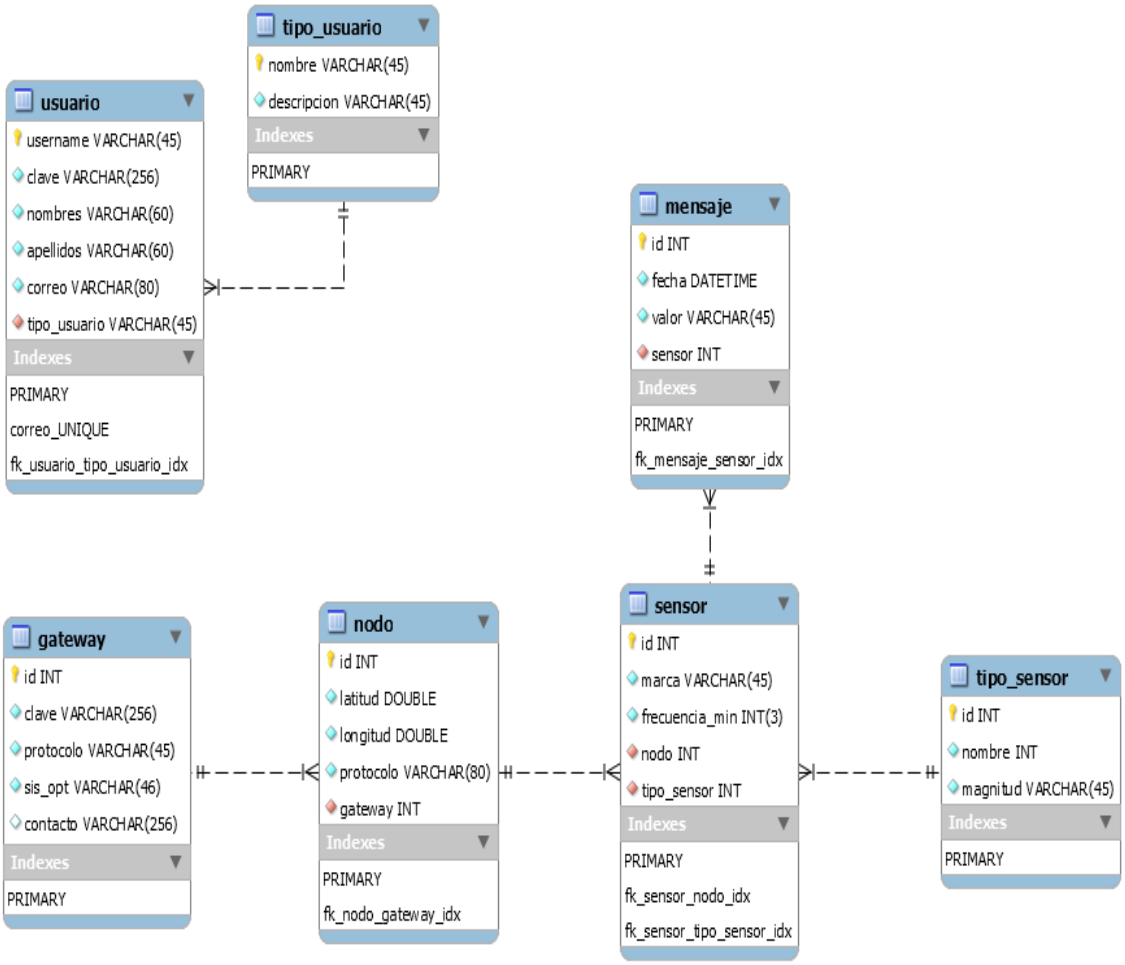


Figura 1 - Modelo de Datos

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 protoclo 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.

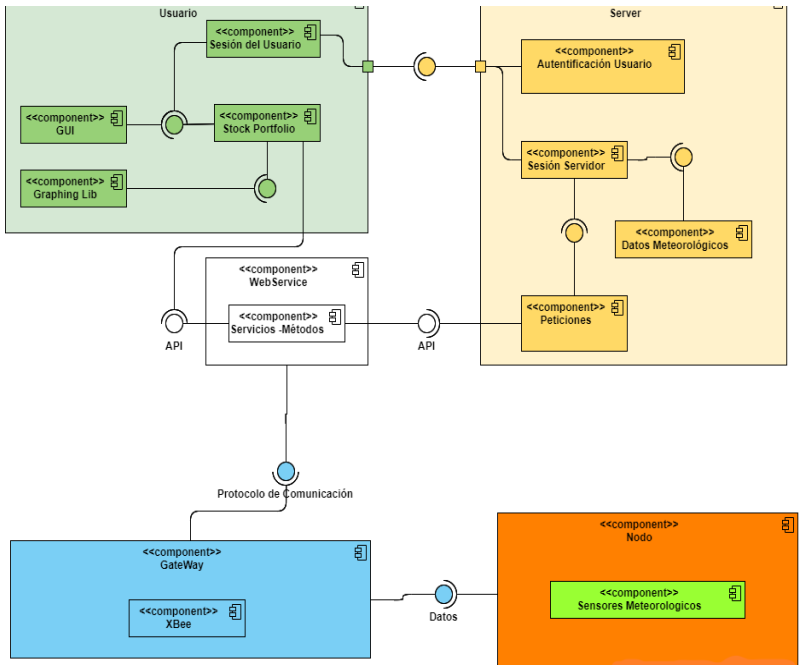


Figura 2 - Diagrama de componentes

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.

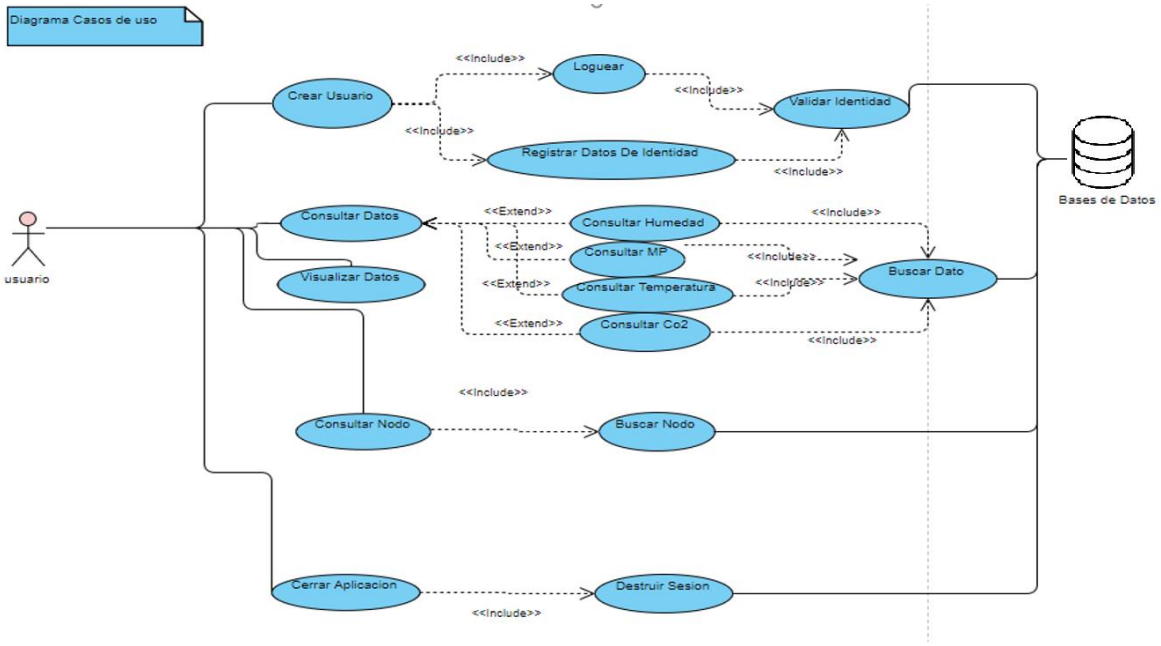


Figura 3- Diagrama de casos de uso

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.

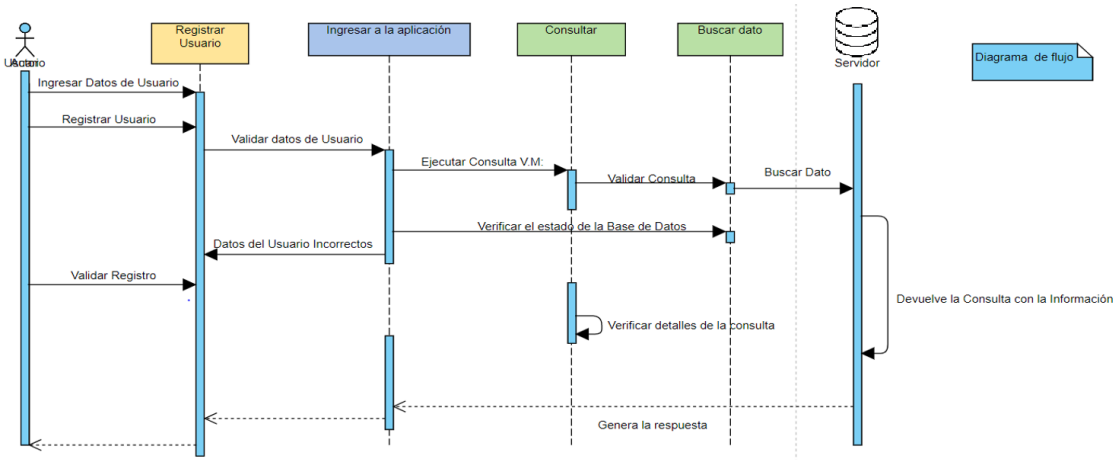


Figura 4 - Diagrama de flujo

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.

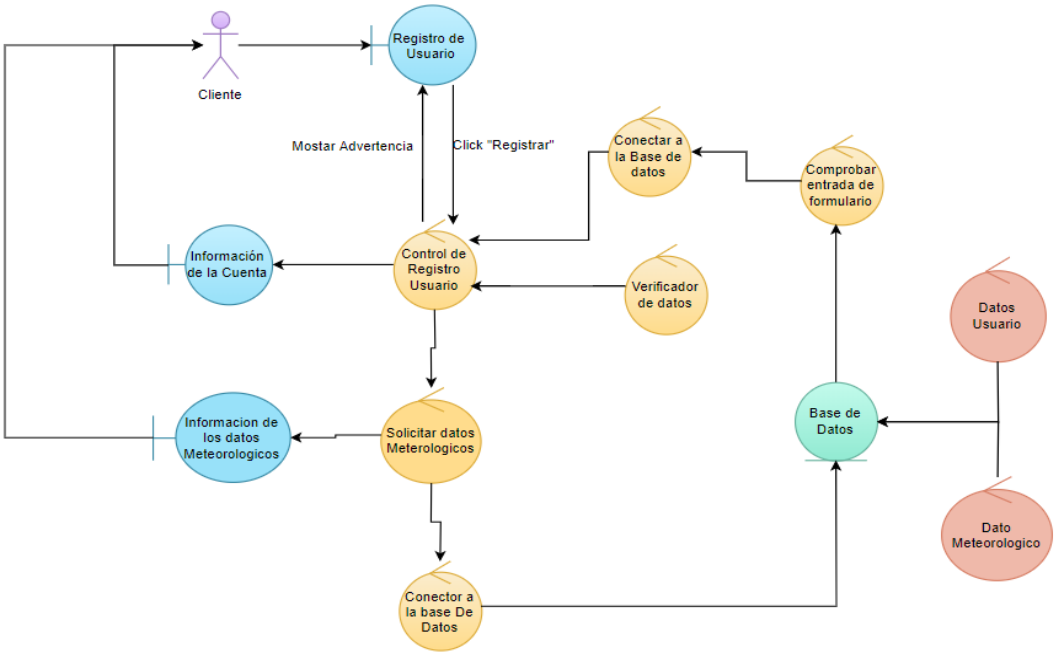


Figura 5 - Diagrama de comunicación

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 diferente 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.

```php
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.

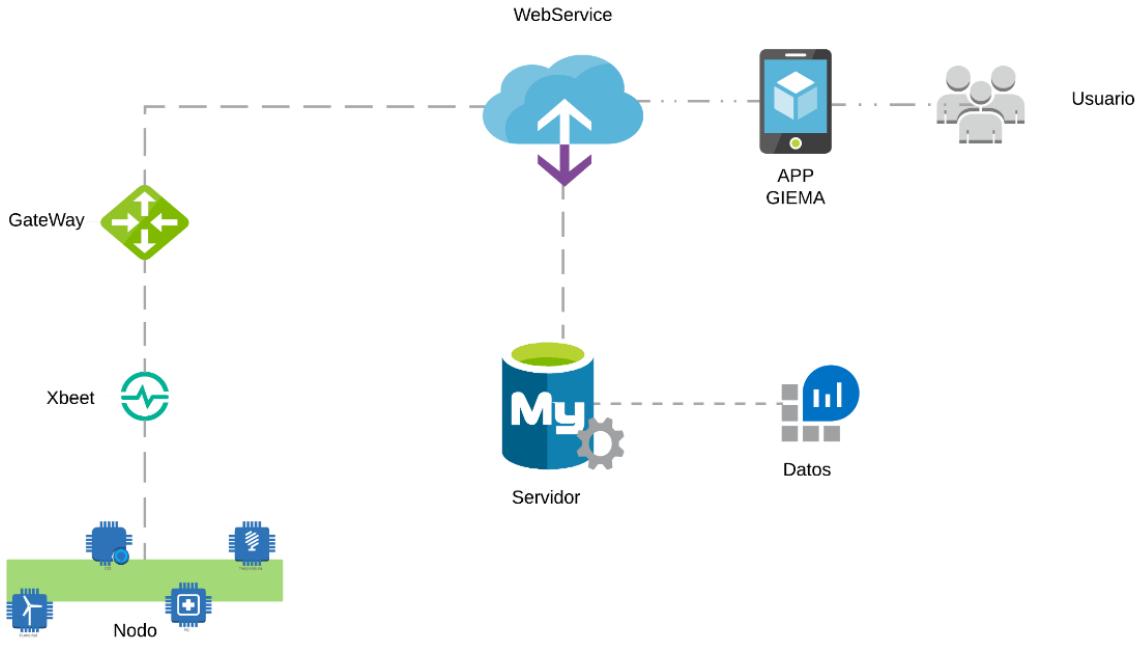


Figura 7 - Diagrama De Arquitectura GIEMA

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.

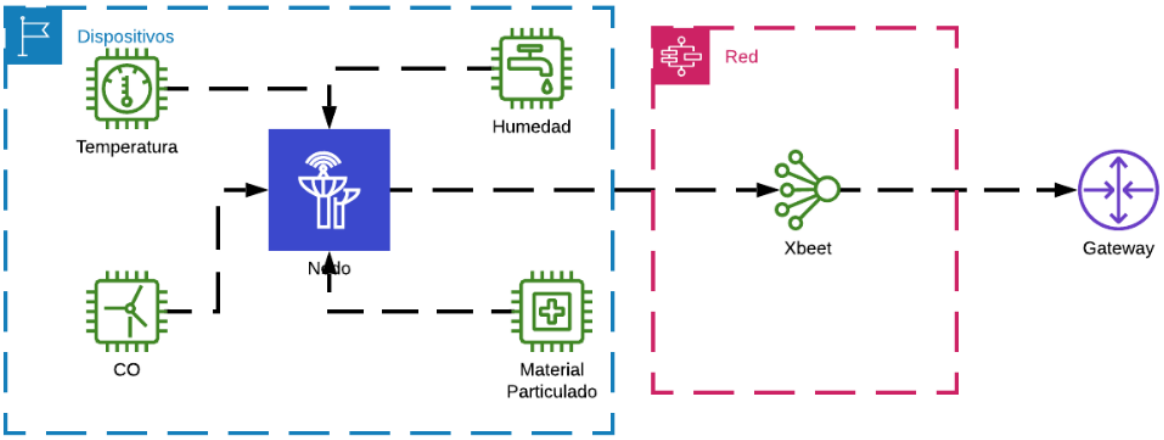


Figura 8 - Diagrama dispositivos GIEMA

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.

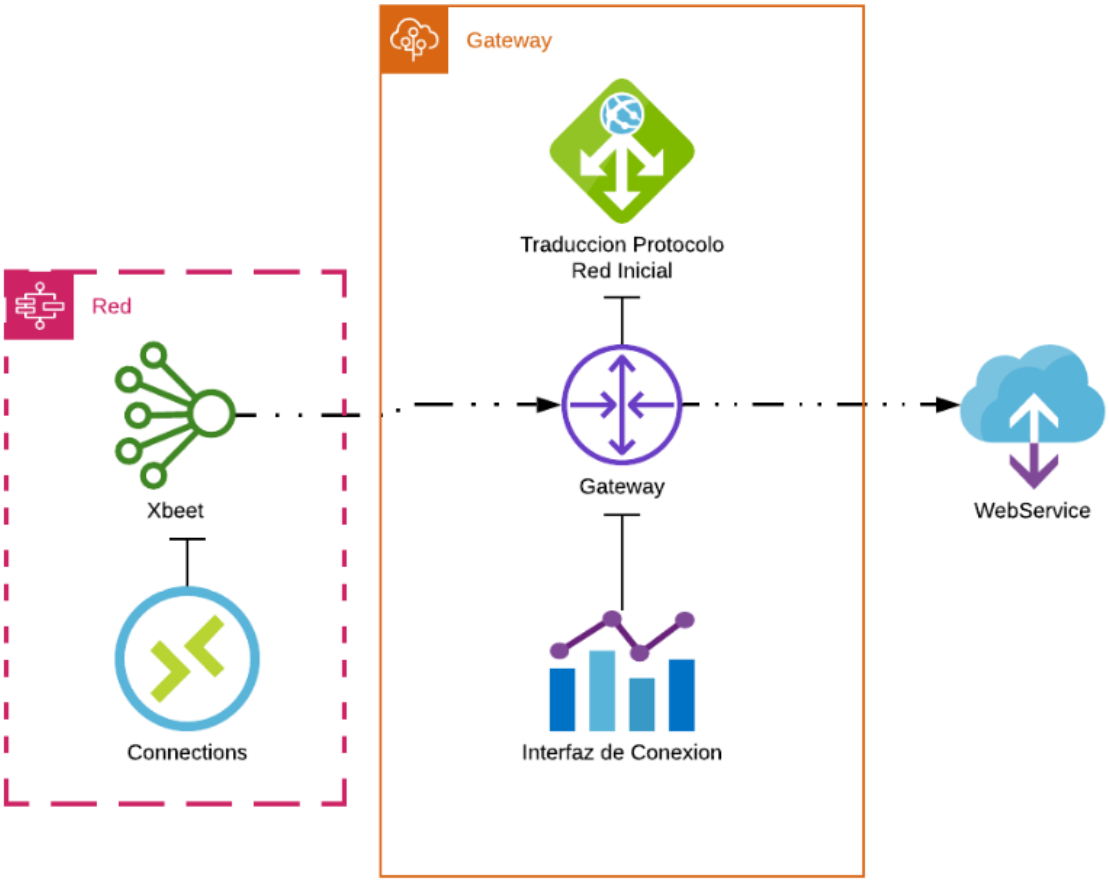


Figura 9 - Diagrama WebLogic GIEMA

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.

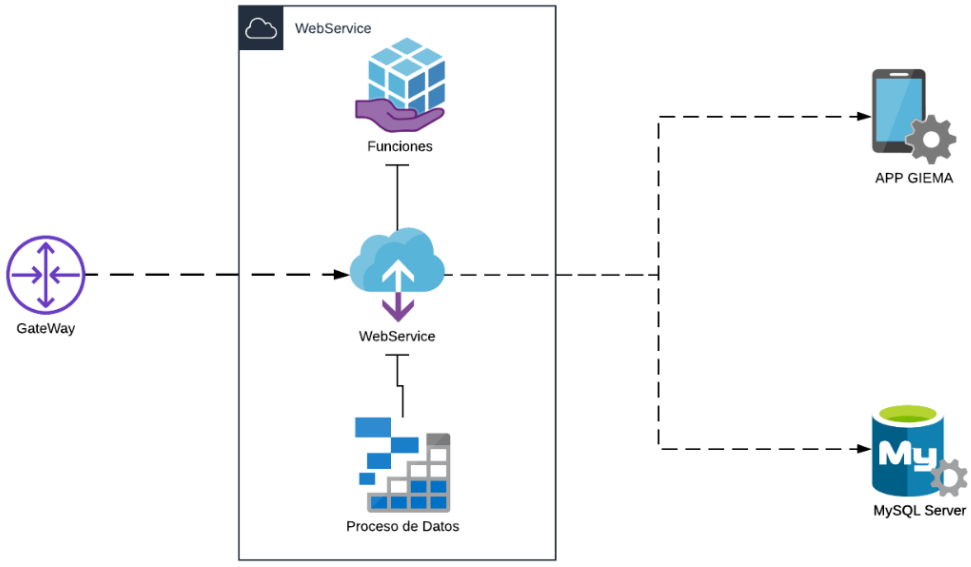


Figura 10 - Diagrama GateWay GIEMA

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.

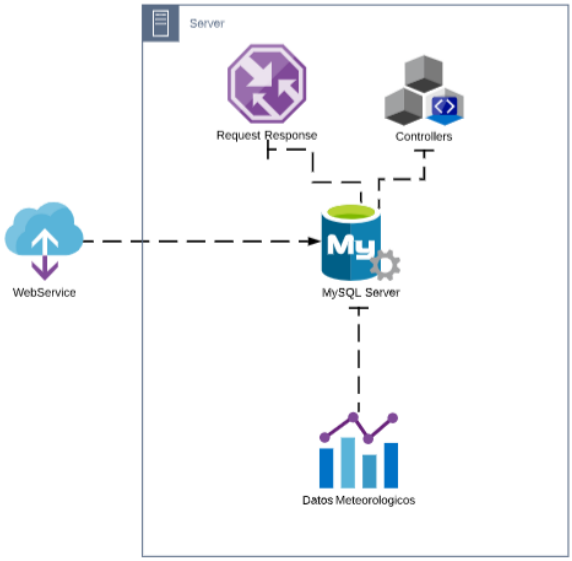


Figura 11 Server GIEMA

3.5. Mockups y Navegación

En base a la arquitectura planteada, se diseñó el flujo de la aplicación, ya que este es el método por el cual el usuario se comunicará, el diseño en basa 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.

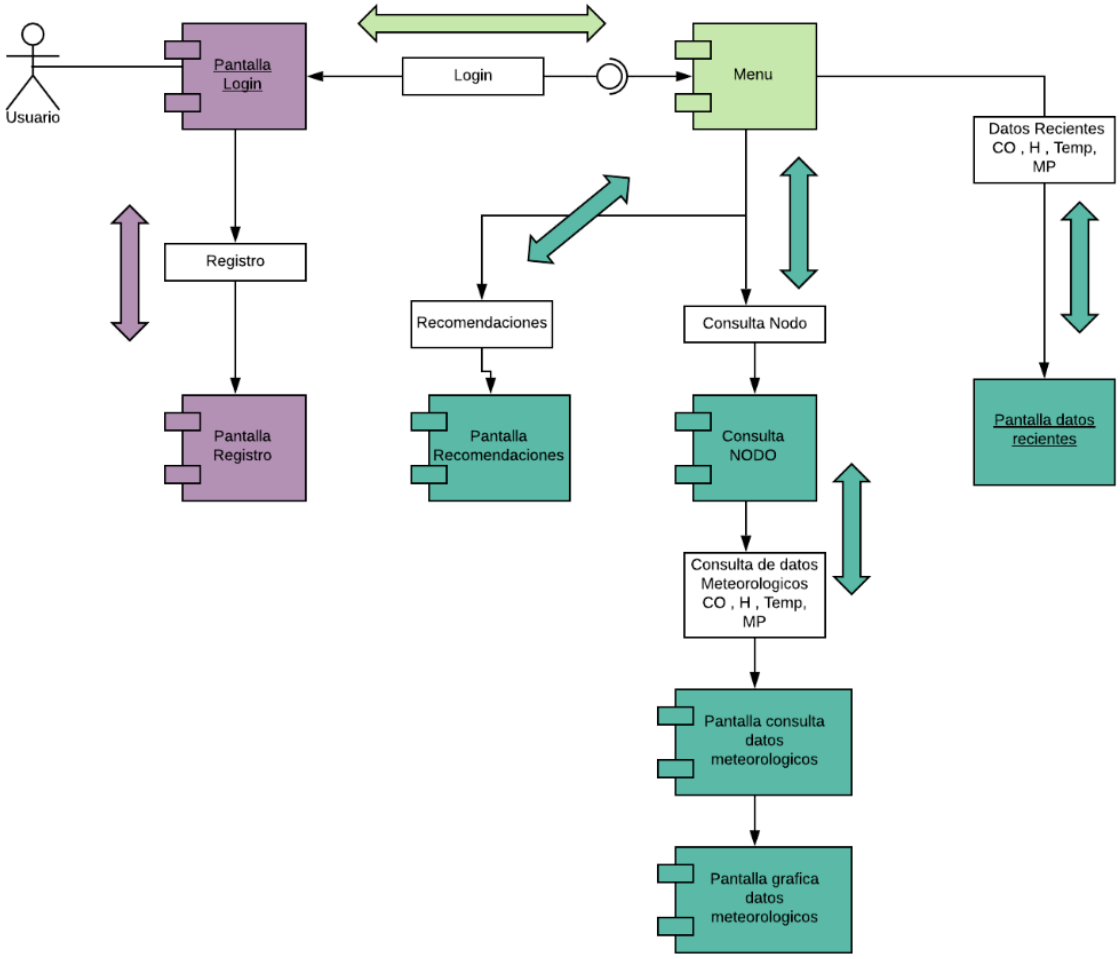
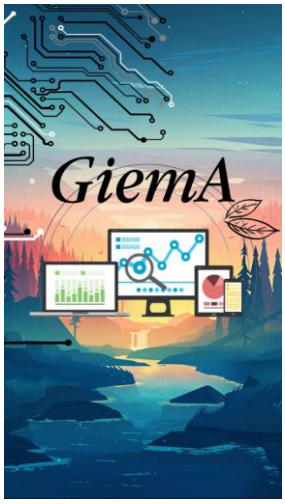
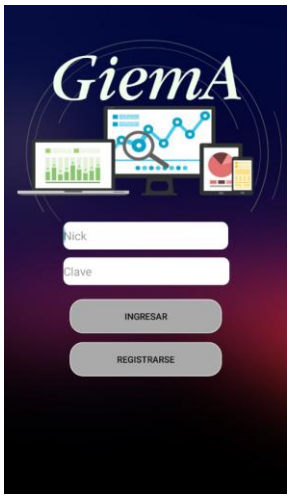


Figura 12 Diagrama de navegación

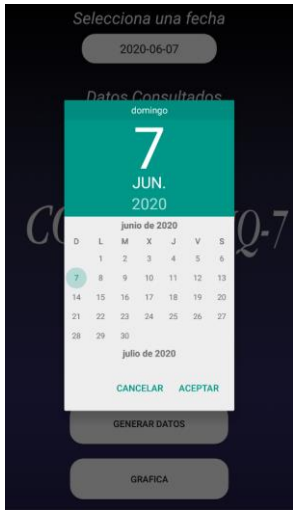
El flujo de la aplicación es el siguiente según lo planteado en el diagrama de navegación en la figura 12.

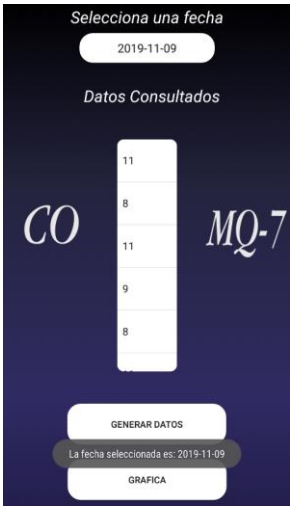

GiemA

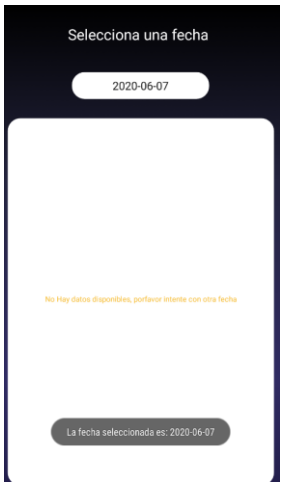

GiemA
Nick
Clave
INGRESAR
REGISTRARSE

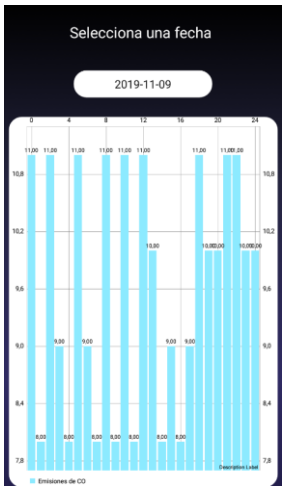

GiemA
DATOS METEROLOGICOS DIARIOS
CONSULTA DE DATOS - METEOROLOGICOS
RECOMENDACIONES
CERRAR


GiemA
Nodo Actual:
U. San Buenaventura
Sensores disponibles:
CO
HUMEDAD
TEMPERATURA
MATERIAL PARTICULADO


Selecciona una fecha
2020-06-07
domingo
7
JUN.
2020
junio de 2020
julio de 2020
CANCELAR
ACEPTAR
GENERAR DATOS
GRAFICA


Selecciona una fecha
2019-11-09
Datos Consultados
CO
MQ-7
11
8
11
9
8
GENERAR DATOS
La fecha seleccionada es: 2019-11-09
GRAFICA


Selecciona una fecha
2020-06-07
No Hay datos disponibles, porfavor intente con otra fecha
La fecha seleccionada es: 2020-06-07


Selecciona una fecha
2019-11-09
Emisiones de CO