Imagen que contiene exterior, cielo, nieve, agua

Descripción generada con confianza muy alta

**ESCUELA SUPERIOR POLITÉNICA DEL LITORAL**

**PROYECTO DE CONMUTACIÓN Y ENRUTAMIENTO**

**MANUAL TÉCNICO**

SISTEMA DE CONFIGURACIÓN DEL PROTOCOLO DEL ENRUTAMIENTO BGP PARA LA INTERCONEXIÓN ENTRE EMPRESAS O ISP

**MG. ADRIANA COLLAGUAZO**

**GRUPO #2**

OCHOA OCHOA ARIANA

CRUZ OCHOA ALBERTO

SÁNCHEZ VALVERDE JOSUÉ

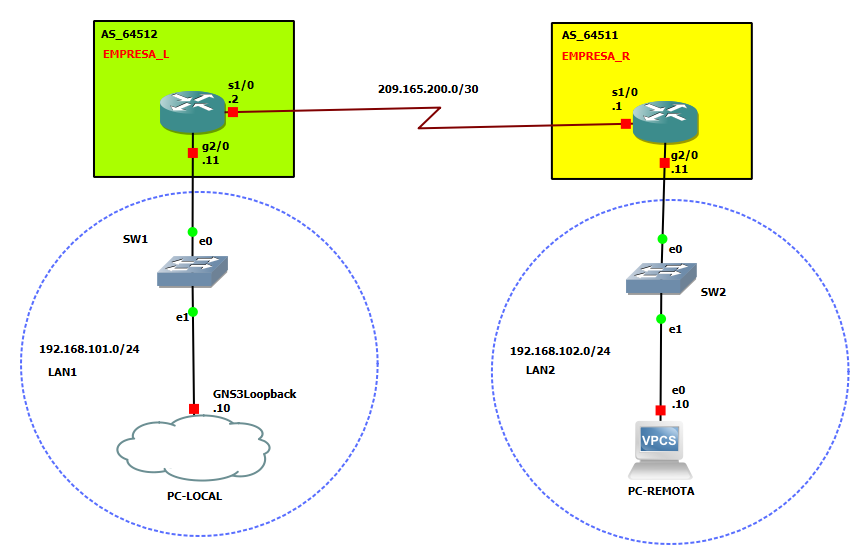
SOLEDISPA CARRASCO ALDAIR

**PARALELO 1**

2T – 2018

**Manual Implemantación Técnica del Proyecto**

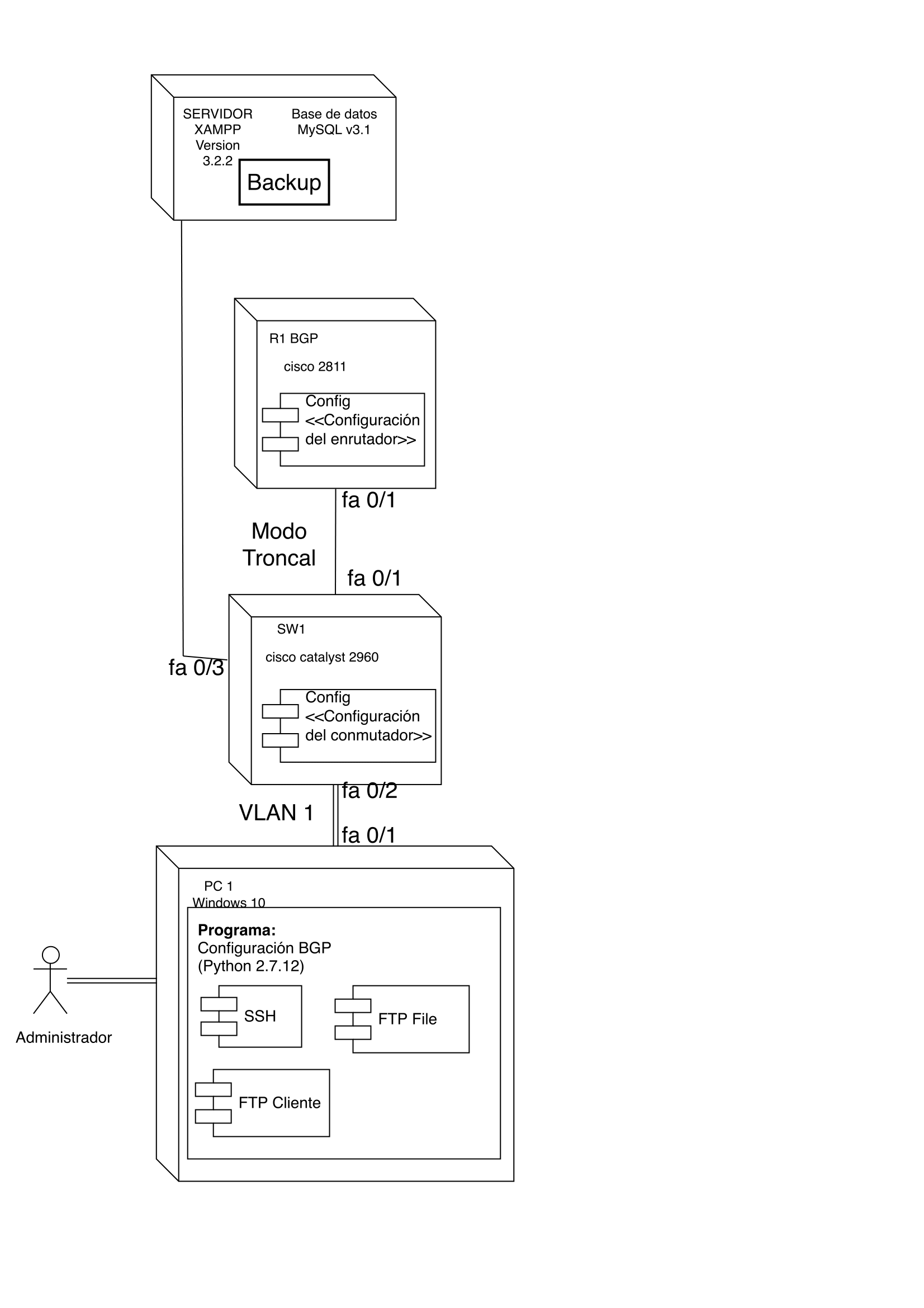
1. **Diagrama de red**



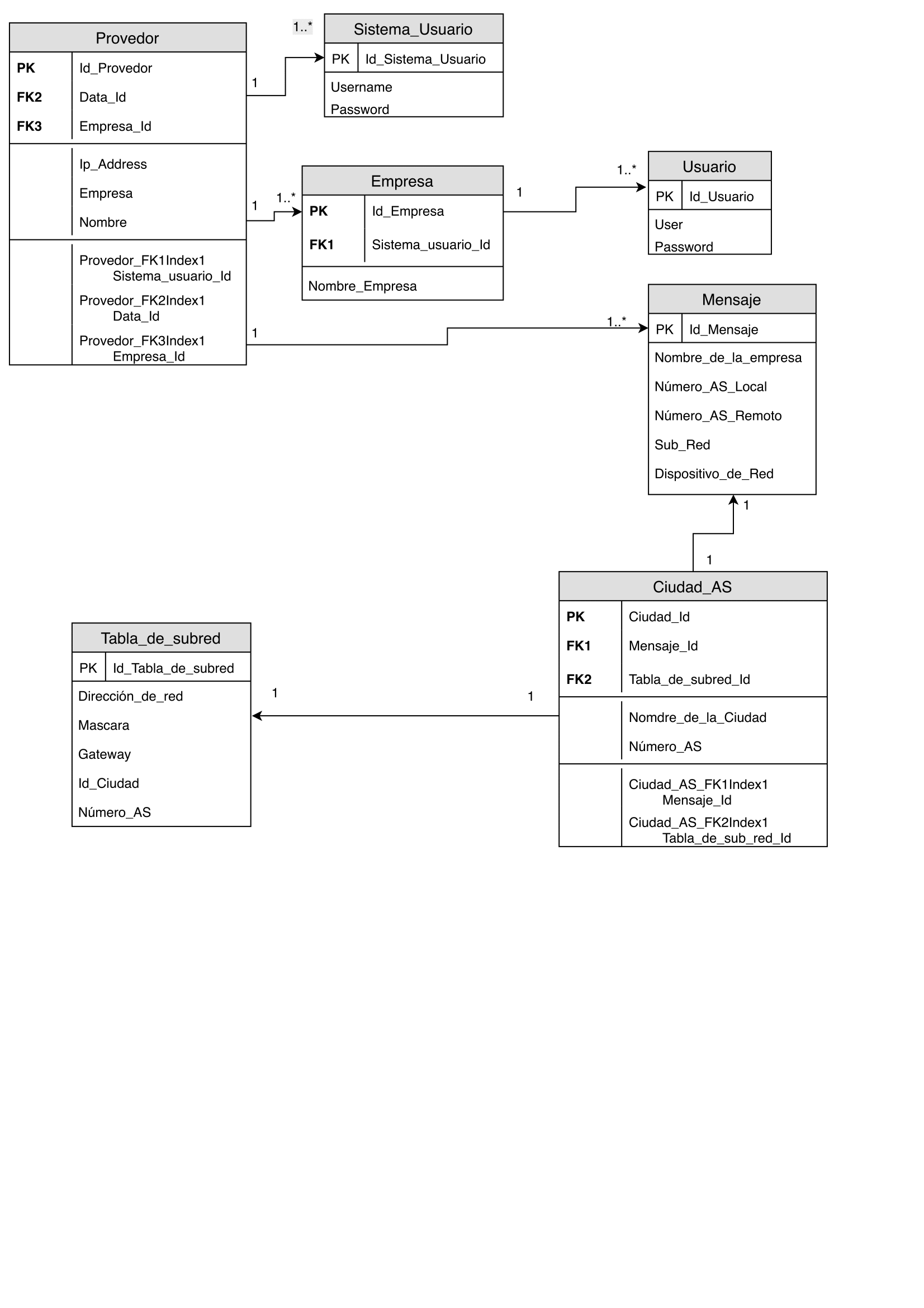
1. **Recursos**

|  |  |
| --- | --- |
| Hardware | Software |
| 2 enrutadores Cisco | GNS3 version 2.1.2 para Windows (64-bit) |
| Dos Computadores portátiles | Python 2.7.12 con las librerías: Telnetlib, easygui, wxPython, wx |
|  | WxFormBuilder versión 3.5 |
|  | Plataforma de desarrollo colaborativo GitHub |
|  | Aplicación web Asana |

1. **Diagrama de Despliegue**



1. **Modelo de entidad-relación de la base de datos.**

****

1. **Campos y tipos de datos creados en la base de datos**

A continuación se presenta en detalle los campos con sus respectivos tipos de datos creados en la base de datos del proyecto:

El campo ***PROVEEDOR*** corresponde al ISP, en donde presenta como datos el id\_proveedor que representa un ID único otorgado al ISP, además la Data\_Id que indica información acerca del proveedor de internet, Empresa\_Id indica el ID de la empresa para la posterior interconexión entre las mismas. Presenta como datos del campo la dirección IP, la empresa y el nombre de la misma.

El campo  ***SISTEMA\_USUARIO*** es aquel que se encarga de administrar la Id\_Sistema\_Usuario, es decir que tiene como datos la identificación del usuario para el ingreso al sistema, el username y el password.

El campo ***EMPRESA*** presenta como tipos de datos creados el ID\_Empresa el cual es la identificación única de la empresa a tratar para la interconexión, y el Sistema\_usuario\_Id que representa la identificación del usuario, datos como usuario y clave del mismo para el acceso.

El campo  ***USUARIO*** presenta la identificación única del usuario, user y password como reconocimiento del usuario.

El campo ***MENSAJE*** presenta un ID\_Mensaje, este campo se encarga de mostrar cuando exista la interconexión entre empresas tanto la local como la remota, con ayuda de la información de la misma como Nombre\_de\_la\_empresa, Número\_AS\_local correspondiente al AS de la empresa local, Número\_AS\_remoto que corresponde al AS de la empresa remota, sub\_Red que representa la dirección de la red en la cual se encuentran al momento ambas empresas y Dispositivo\_de\_Red que representa el equipo el cual se encuentra teniendo la conexión al momento.

El campo ***CIUDAD\_AS*** es aquel que se encarga de mostrar según sus tipos de datos las empresas que se desean interconectar mostrando sus respectivos AS, posee como dato el Nombre\_de\_la\_ciudad, y número\_de\_AS.

El campo ***TABLA\_DE\_SUBRED*** presenta el ID\_Tabla\_de\_subred como tipo de dato el cual se encarga de mostrar la identificación respectiva a la red de cada empresa ya sea la local o remota con su respectivo AS, en el presente campo se muestra la dirección\_de\_red, máscara, gateway, Id\_Ciudad y Número\_AS como parámetros principales para ordenar una tabla de subred para la respectiva interconexión y el correcto funcionamiento del proyecto.

1. **Configuraciones de los dispositivos de red utilizados.**
   1. Configuración de router *Empresa Local*

|  |  |
| --- | --- |
|  | conf t |
|  | hostname EMPRESA\_L |
|  | ip domain-name fiec.espol.edu.ec |
|  | no ip domain lookup |
|  | crypto key generate rsa |
|  | 1024 |
|  |  |
|  | banner motd #SOLO ACCESO A PERSONAL AUTORIZADO# |
|  | username admin privilege 15 secret admin |
|  | username alberto privilege 15 secret alberto |
|  | username aldair privilege 15 secret aldair |
|  | username ariana privilege 5 secret ariana |
|  | line console 0 |
|  |  |
|  |  |
|  | exec-timeout 3 3 |
|  | login local |
|  | exit |
|  | line vty 0 15 |
|  |  |
|  | exec-timeout 3 3 |
|  | login local |
|  | transport input all |
|  | exit |
|  | service password-encryption |
|  | int f0/0 |
|  | des Conectado a LAN1 |
|  | ip address 192.168.101.11 255.255.255.0 |
|  | no sh |
|  | int f0/1 |
|  | des Conectado a RED\_REM\_2 |
|  | ip address 192.168.100.1 255.255.255.0 |
|  | no sh |
|  | int s0/2/0 |
|  | des Enlace serial BGP hacia AS\_64511 |
|  | ip address 209.165.200.2 255.255.255.252 |
|  | no sh |
|  | end |

* 1. Configuración de router Empresa Remota

|  |  |
| --- | --- |
|  | enable |
|  | conf t |
|  | hostname EMPRESA\_R |
|  | ip domain-name fiec.espol.edu.ec |
|  | no ip domain lookup |
|  | crypto key generate rsa |
|  | 1024 |
|  | enable secret cisco |
|  | banner motd #SOLO ACCESO A PERSONAL AUTORIZADO# |
|  | username admin privilege 15 secret admin |
|  | username alberto privilege 15 secret alberto |
|  | username aldair privilege 15 secret aldair |
|  | username ariana privilege 5 secret ariana |
|  | line console 0 |
|  | logging synchronous |
|  | exec-timeout 3 3 |
|  | login local |
|  | exit |
|  | line vty 0 15 |
|  | logging synchronous |
|  | exec-timeout 3 3 |
|  | login local |
|  | transport input all |
|  | exit |
|  | service password-encryption |
|  | int f0/0 |
|  | des Conectado a LAN2 |
|  | ip address 192.168.102.11 255.255.255.0 |
|  | no sh |
|  | int s0/2/0 |
|  | des Enlace serial BGP hacia AS\_64512 |
|  | ip address 209.165.200.1 255.255.255.252 |
|  | no sh |
|  | end |

1. **Explicación del código fuente desarrollado con los comentarios correspondientes**

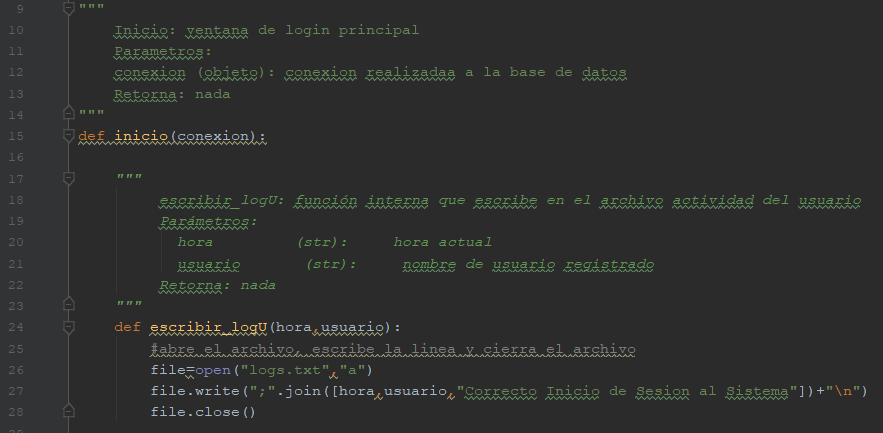
Durante la realización del proyecto se logró efectuar de manera correcta las respectivas historias de usuario planteadas en la rúbrica mediante un programa que permite al usuario ingresar datos y validar todos los posibles casos de error con herramientas como una base de datos obteniendo así resultados esperados.

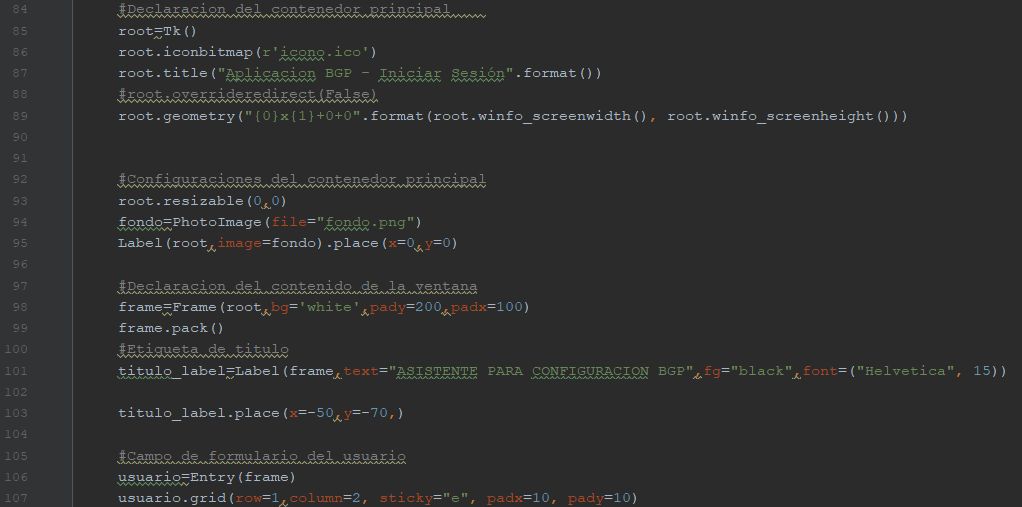
* 1. Explicación por archivo:
     1. **main.py**
* El archivo principal denominado main contiene la validación inicial de la conexión del programa con la base de datos, misma que contiene la información que permitirá realizar las validaciones respectivas así como la configuración mediante protocolo remoto. En caso de no existir conexión con la base de datos se le mostrara una ventana de notificación, misma que fue diseña como una función en el archivo graficos.py.

import graficos as gf  
import conexion\_mysql as sql  
"""  
Retorna grafico de error de conexion en caso de fallar la conexion con la base de datos  
o retorna el grafico de inicio para continuar a configurar los enrutadores.  
"""  
#Si existe conexion con la base de datos devuelve la direccion hexagecimal caso contrario devuevle -1  
conn=sql.realizar\_conexion()  
if conn==-1:  
 gf.error\_servidor()  
else:  
 gf.inicio(conn)  
 sql.cerrar\_conexion(conn)

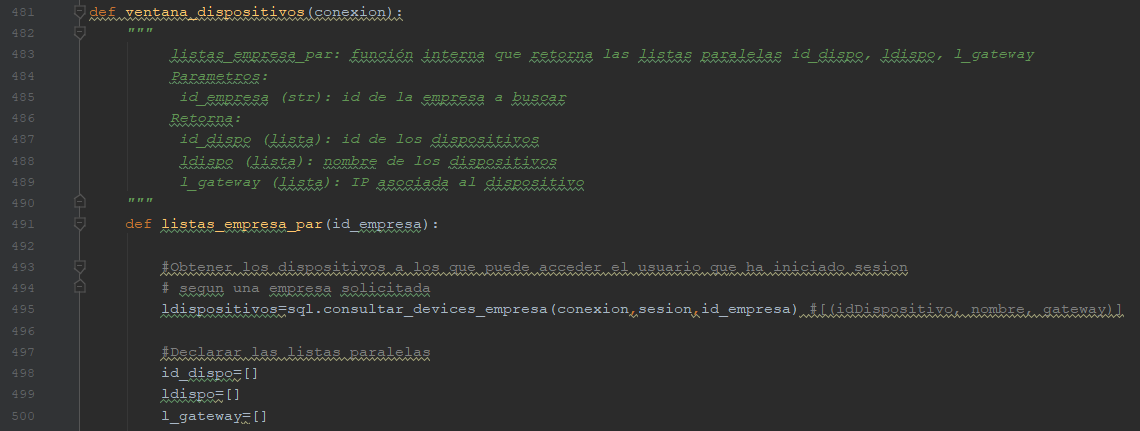
* + 1. **graficos.py**
* Debido a que basándonos en los requerimientos del programa, la interacción del usuario con el mismo era indispensable para cumplir con el objetivo del proyecto, por lo que se implementó interfaz gráfica para el ingreso de datos y permitir configurar parámetros necesarios en el establecimiento de sesión BGP.
* Entre las funciones que se implementaron para captar y validar información por parte del usuario se encuentran:

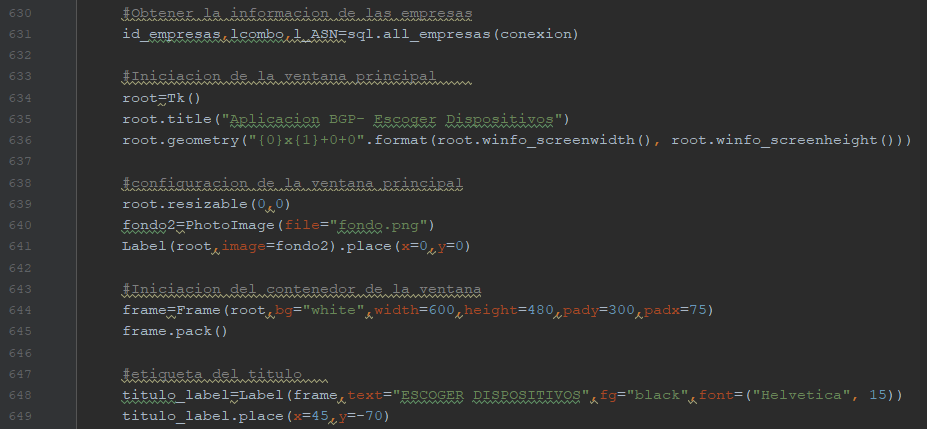
**def inicio**: ventana principal de inicio de sesión al sistema para validad las credenciales de acceso guardadas en la base de datos.





**def ventana\_dispositivos**: Ventana de ingreso de información para la configuración de dispositivos local y remoto. Aquí también se hace uso de la información de la base de datos para mostrar los dispositivos que el administrador puede configurar.





* Otras funciones fueron implementadas para la configuración de las redes que se desean agregar al peer BGP, así también como ventanas que muestras las menajes de error por las validaciones pero también las operaciones realizadas con éxito.
  + 1. **funciones\_Sesion\_BGP2.py**
* En este archivo podemos encontrar las funciones responsables del envió de las líneas de comando hacia los dispositivos para así configurar de forma remota los parámetros necesarios en BGP. A continuación se pueden apreciar algunas de las funciones implementadas con una breve descripción.

Imagen que contiene texto, captura de pantalla

Descripción generada con confianza alta

* Esta función habilita el protocolo BGP mediante una conexión ssh, donde determina en los parámetros el dispositivo a configurar y el AS del mismo que desea ingresar.
* En cuanto a la tercera historia de usuario, se requiere crear redes para los dispositivos seleccionados de la empresa local y remota, indicando el nombre del dispositivo y determinando los prefijos de red en la sesión BGP.

Imagen que contiene texto, captura de pantalla

Descripción generada con confianza alta

* La función configurar\_Network anuncia las redes que son ingresadas como parámetro de entrada, en la variable lista\_Network se presenta lista de strings de IP y máscara de red que corresponden al grupo de networks a configurar en el router.
* En cuanto a la segunda historia de usuario, se menciona que se requiere realizar una configuración el sistema autónomo, seleccionando a la vez el nombre de la empresa, nombre de dispositivo que se requiere interconectar usando el protocolo BGP, seleccionando el número de AS respectivo de cada dispositivo.
* A continuación, se presenta breve parte del código donde se realiza a cabalidad cada uno de los criterios de aceptación para el resultado final de la segunda historia de usuario.

Imagen que contiene captura de pantalla

Descripción generada con confianza muy alta

* La función config\_Vecino configura el enrutador vecino con su AS, teniendo como parámetros el objeto que se conectará por ssh, la dirección IP del dispositivo vecino, SA del router de la empresa local, SA del router de la empresa remota, y finalmente una lista de strings que corresponden al grupo de redes a configurar en el router. En la presente función se realiza la configuración BGP de ambos dispositivos local y remoto.
  + 1. **conexion\_mysql,py**
* En este archivo se encuentran funciones para el establecimiento de la sesión con la base de datos así como funciones que permiten obtener información especifica para realizar las validaciones necesarias en varios etapas del programa.
* En el caso de la primera historia de usuario donde se menciona que se requiere ingresar a un dispositivo que se encuentre activo en la red, accediendo a la base de datos donde el dispositivo se encuentra asociado a una dirección IP, se presenta parte del código para el correcto funcionamiento y acceso al dispositivo para establecer una sesión remota:

Imagen que contiene captura de pantalla

Descripción generada con confianza alta

* El presente código se establece para realizar la conexión a la base de datos de MySQL, definiendo a servidor como nombre del servidor, nombre de usuario, contraseña del usuario y bd como nombre de la base de datos.

Imagen que contiene captura de pantalla

Descripción generada con confianza muy alta

* La función consultar\_empresas se encarga de realizar la consulta de todas las empresas siempre y cuando la conexión en la base de datos sea exitosa, donde retorna la lista de empresas existentes determinan el Id de la misma, el nombre y ASN de la empresa.

Imagen que contiene captura de pantalla

Descripción generada con confianza muy alta

* La función all\_empresas hace una consulta a la BD, y retorna una tupla de listas paralelas con las id, nombre y ASNs de las empresas .

Imagen que contiene captura de pantalla

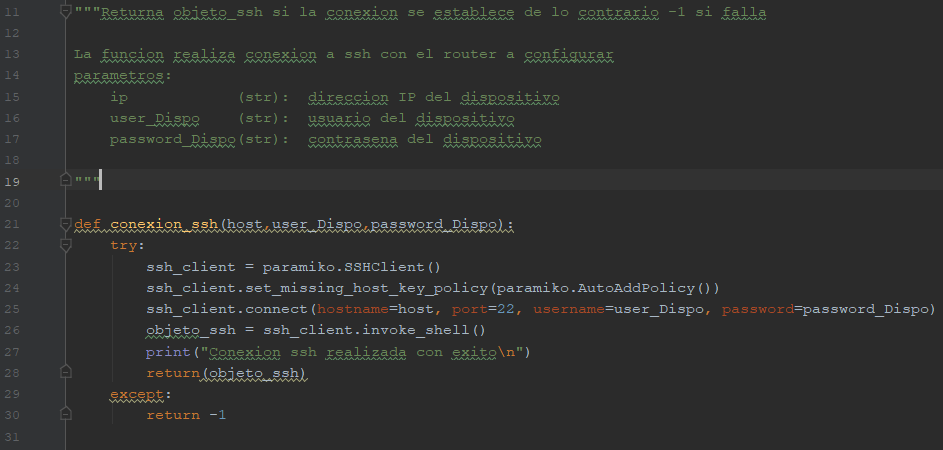
Descripción generada con confianza muy alta

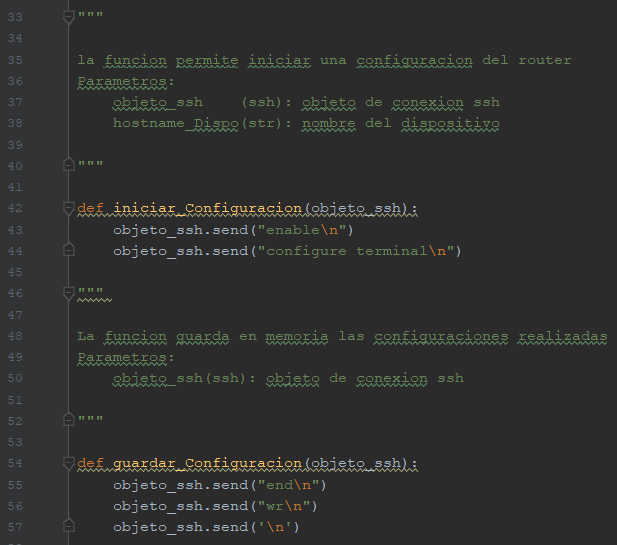
* La función consultar\_interfaces se encarga realizar una consulta de todas las interfaces de un dispositivo siempre y cuando la conexión sea exitosa.

Imagen que contiene captura de pantalla

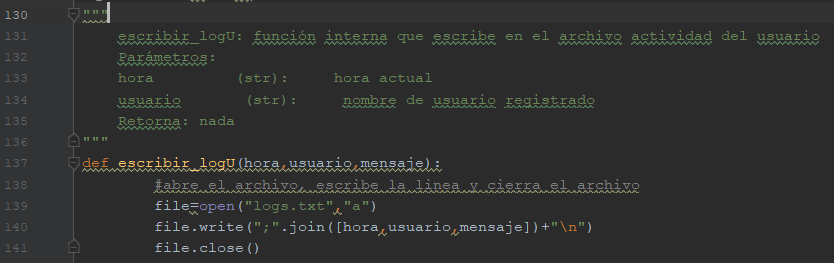
Descripción generada con confianza alta

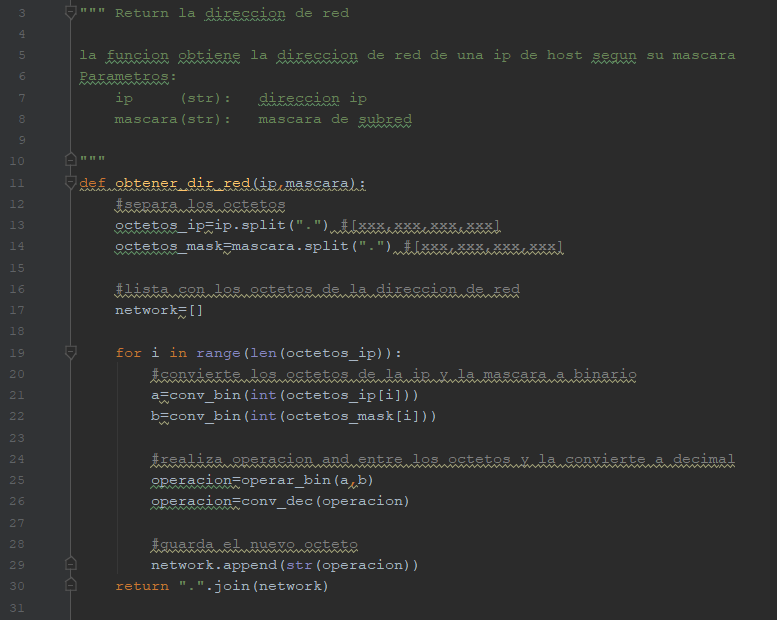
* La función buscar\_id\_dispositivo busca el dispositivo según el nombre y el id de la empresa siempre y cuando la conexión que se tenga en la base de datos sea exitosa. A su vez también se presenta una función denominada buscar\_ip\_dispositivo, la misma realiza la misma función anterior pero la diferencia es que determina la IP del dispositivo a seleccionar.
  + 1. **conexion\_ssh.py**
* En este archivo podemos encontrar funciones que nos permiten establecer la sesión remota SSH con los dispositivos remotos, enviar información así como guardar las configuraciones realizadas en los dispositivos.
* La función conexion\_ssh es encargada de gestionar la conexión remota haciendo uso del protocolo SecureShell. En caso de realizarse correctamente se notifica por consola que se ha establecido conexión.





* La función iniciar\_Configuracion y guardar son las encargadas de mandar mediante conexión las líneas de comando para habilitar la configuración de los dispositivos y una vez realizada guardar permanentemente dicha configuración respectivamente.
  + 1. **auxiliares.py**
* En este archivo se pueden encontrar funciones para validar los datos que el usuario ingresas y poder en base a los resultados esperados mostrar las ventanas de notificación creadas en el archivo graficos.py si es necesario. Entre las funciones se pueden encontrar obtener\_dir\_red, función que permite encontrar la red a la que pertenece una IP conociendo su máscara. Así también como escribir\_logU que abre el archivo de logs para escribir una línea sobre las acciones que se están realizando con el programa.





1. **Archivo de configuración SQL para la creación de la base de datos.**

**SET** SQL\_MODE="NO\_AUTO\_VALUE\_ON\_ZERO";  
**SET time\_zone** = "+00:00";  
  
  
/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;  
/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;  
/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;  
/\*!40101 SET NAMES utf8 \*/;  
  
--  
-- Base de datos: `bgp`  
--  
  
-- --------------------------------------------------------  
  
--  
-- Estructura de tabla para la tabla `dispositivos`  
--  
  
**CREATE TABLE** IF **NOT EXISTS** `dispositivos` (  
 `idDispositivo` *int*(10) **NOT NULL** AUTO\_INCREMENT **COMMENT** 'id dispositivo',  
 `nombre` *varchar*(25) **NOT NULL COMMENT** 'nombre dispositivo',  
 `empresa` *int*(10) **NOT NULL COMMENT** 'id empresa',  
 `gateway` *varchar*(20) **DEFAULT NULL**,  
 `estado` *varchar*(10) **NOT NULL**,  
 **PRIMARY KEY** (`idDispositivo`),  
 **KEY** `empresa` (`empresa`)  
) ENGINE=InnoDB **DEFAULT** CHARSET=latin1 AUTO\_INCREMENT=3 ;  
  
--  
-- Volcado de datos para la tabla `dispositivos`  
--  
  
**INSERT INTO** `dispositivos` (`idDispositivo`, `nombre`, `empresa`, `gateway`, `estado`) **VALUES**(1, 'RouterLocal', 1, **NULL**, 'activo'),  
(2, 'RouterRemoto', 2, **NULL**, 'activo');  
  
-- --------------------------------------------------------  
  
--  
-- Estructura de tabla para la tabla `empresa`  
--  
  
**CREATE TABLE** IF **NOT EXISTS** `empresa` (  
 `idEmpresa` *int*(10) **NOT NULL** AUTO\_INCREMENT **COMMENT** 'id de la empresa',  
 `nombre` *varchar*(50) **NOT NULL COMMENT** 'nombre de la empresa',  
 `ASN` *int*(10) **NOT NULL**,  
 **PRIMARY KEY** (`idEmpresa`)  
) ENGINE=InnoDB **DEFAULT** CHARSET=latin1 AUTO\_INCREMENT=3 ;  
  
--  
-- Volcado de datos para la tabla `empresa`  
--  
  
**INSERT INTO** `empresa` (`idEmpresa`, `nombre`, `ASN`) **VALUES**(1, 'Empresa 1', 64512),  
(2, 'Empresa 2', 64511);  
  
-- --------------------------------------------------------  
  
--  
-- Estructura de tabla para la tabla `interfaces`  
--  
  
**CREATE TABLE** IF **NOT EXISTS** `interfaces` (  
 `idDispositivo` *int*(10) **NOT NULL**,  
 `nombre` *varchar*(10) **NOT NULL**,  
 `ipAddress` *varchar*(20) **NOT NULL**,  
 `mascara` *varchar*(20) **NOT NULL**,  
 **KEY** `idDispositivo` (`idDispositivo`)  
) ENGINE=InnoDB **DEFAULT** CHARSET=latin1;  
  
--  
-- Volcado de datos para la tabla `interfaces`  
--  
  
**INSERT INTO** `interfaces` (`idDispositivo`, `nombre`, `ipAddress`, `mascara`) **VALUES**(1, 'f0/1', '192.168.100.1', '255.255.255.0'),  
(1, 's0/2/0', '209.165.200.2', '255.255.255.252'),  
(2, 'f0/1', '192.168.103.1', '255.255.255.0'),  
(2, 's0/2/0', '209.165.200.1', '255.255.255.252'),  
(1, 'f0/0', '192.168.101.11', '255.255.255.0'),  
(2, 'f0/0', '192.168.102.11', '255.255.255.0');  
  
-- --------------------------------------------------------  
  
--  
-- Estructura de tabla para la tabla `usuarios`  
--  
  
**CREATE TABLE** IF **NOT EXISTS** `usuarios` (  
 `idUsuario` *int*(10) **NOT NULL** AUTO\_INCREMENT **COMMENT** 'Id de usuario',  
 `tipo` *varchar*(20) **NOT NULL**,  
 `**user**` *varchar*(30) **NOT NULL**,  
 `password` *varchar*(50) **NOT NULL**,  
 **PRIMARY KEY** (`idUsuario`)  
) ENGINE=InnoDB **DEFAULT** CHARSET=latin1 AUTO\_INCREMENT=5 ;  
  
--  
-- Volcado de datos para la tabla `usuarios`  
--  
  
**INSERT INTO** `usuarios` (`idUsuario`, `tipo`, `**user**`, `password`) **VALUES**(1, 'admin', 'jocelyn', 'jocelyn'),  
(2, 'admin', 'viviana', 'viviana'),  
(3, 'admin', 'edisson', 'edisson'),  
(4, 'monitoreo', 'monitoreo', 'monitoreo');  
  
-- --------------------------------------------------------  
  
--  
-- Estructura de tabla para la tabla `usuario\_dispositivo`  
--  
  
**CREATE TABLE** IF **NOT EXISTS** `usuario\_dispositivo` (  
 `idUsuario` *int*(10) **NOT NULL**,  
 `idDispositivo` *int*(10) **NOT NULL**,  
 **KEY** `idUsuario` (`idUsuario`,`idDispositivo`),  
 **KEY** `idDispositivo` (`idDispositivo`)  
) ENGINE=InnoDB **DEFAULT** CHARSET=latin1;  
  
--  
-- Volcado de datos para la tabla `usuario\_dispositivo`  
--  
  
**INSERT INTO** `usuario\_dispositivo` (`idUsuario`, `idDispositivo`) **VALUES**(1, 1),  
(1, 2),  
(2, 1),  
(2, 2),  
(3, 1),  
(3, 2);  
  
--  
-- Restricciones para tablas volcadas  
--  
  
--  
-- Filtros para la tabla `dispositivos`  
--  
**ALTER TABLE** `dispositivos`  
 **ADD CONSTRAINT** `dispositivos\_ibfk\_1` **FOREIGN KEY** (`empresa`) **REFERENCES** `empresa` (`idEmpresa`) **ON DELETE CASCADE ON UPDATE CASCADE**;  
  
--  
-- Filtros para la tabla `interfaces`  
--  
**ALTER TABLE** `interfaces`  
 **ADD CONSTRAINT** `interfaces\_ibfk\_1` **FOREIGN KEY** (`idDispositivo`) **REFERENCES** `dispositivos` (`idDispositivo`) **ON DELETE CASCADE ON UPDATE CASCADE**;  
  
--  
-- Filtros para la tabla `usuario\_dispositivo`  
--  
**ALTER TABLE** `usuario\_dispositivo`  
 **ADD CONSTRAINT** `usuario\_dispositivo\_ibfk\_1` **FOREIGN KEY** (`idUsuario`) **REFERENCES** `usuarios` (`idUsuario`) **ON DELETE CASCADE ON UPDATE CASCADE**,  
 **ADD CONSTRAINT** `usuario\_dispositivo\_ibfk\_2` **FOREIGN KEY** (`idDispositivo`) **REFERENCES** `dispositivos` (`idDispositivo`) **ON DELETE CASCADE ON UPDATE CASCADE**;  
  
/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;  
/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;  
/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;