InoTek

Tecnologia-Integral-Industrial

SIGHERSA

PROYECTO

DOCUMENTACIÓN DE API

SIGHERSA

PLANTAS DE EMERGENCIA

PRESENTA

Zapopan, Jalisco, 16 de Febrero de 2021

-Método a utilizar en todas las API:

1- POST:

Es el correspondiente método necesario a utilizar, para mandar y recibir la información dentro de la API de forma adecuada y exitosa, se obtiene como resultado de su uso:


```
{
        "mensaje": "Conexion success"
}
```


2- GET, DELETE, PUT:

Al utilizar cualquiera de estos tres métodos: GET, DELETE O PUT, no se puede mandar y recibir la información dentro de la API de forma adecuada y exitosa, provocando que no se logre la conexión correspondiente, se obtiene como resultado de su uso:


```
{
        "mensaje": "Conexion fail : only use post conection"
}
```


-Api_usuario.php:

1- URL: http://sigersa.homelinux.com/screw/api/api_usuario.php

2- Variables a recibir:

Las variables con su correspondiente información, que son necesarias para poder realizar el login dentro del sistema las siguientes:

```
{

        token(dato obligatorio) - tipo: cadena
        id_token(dato obligatorio) - tipo: número entero
        user_name(dato obligatorio) - tipo: cadena
        user_password(dato obligatorio) - tipo: cadena

}
```

3- Variables que regresan:

3.1- Mensaje de éxito:

Nos regresa siempre un mensaje ya sea exitoso o de que existió un error durante la ejecución y conexión con la API, de ser exitoso muestra el siguiente mensaje:

```
{
    "user_name": "",
    "user_password": "",
    "id": "",
    "token": "",
    "mensaje": "Conexion success"
}
```

Además, se realiza el correspondiente inicio de sesión gracias al login realizado correctamente, con todas las variables o datos que fueron recibidos.

3.2- Mensajes de error:

De existir un error la hora de la conexión con la API, podrían existir los siguientes casos:

a) Si la evaluación de las variables recibidas de user namer y user password resulta incorrecta, no nos permitirá inicar sesión, dado que el login no ser realizado de forma correcta, por lo que el error a obtener será el siguiente:

```
{
    "mensaje": " Evaluation fail"
}
```

b) Si el token mandando por POST no es válido, dado que es erróneo o no es el mismo al que corresponde a ese correspondiente id de token , el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail : el token no es valido"
}
```

c) Si el token obtenido en la función obtener_token(gracias al id token enviado por POST) no existe, no esta activo o no es diferente de cero, el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail : el token no existe"
}
```

d) Si al ingresar las variables a recibir de él token o el id token se encuentra vacío, es cero o tiene algún error (desde la forma de envió o ser otro tipo de variable no adecuada). Nos indicaría que existió ese fallo en la variable a recibir, por lo que el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail: no se ingreso el token o id_token
    correctamente o exite un error dentro del mismo";
}
```

-Api_clientes_activos.php:

1- URL: http://sigersa.homelinux.com/screw/api/api_clientes_activos.php

2- Variables a recibir:

Las variables con su correspondiente información, que son necesarias para poder realizar la correspondiente consulta que nos dará los datos de todos los clientes correspondientes dentro de la API son las siguientes:

```
{

        token(dato obligatorio) - tipo: cadena
        id_token(dato obligatorio) - tipo: número entero

}
```

3- Variables que regresan:

3.1- Mensaje de éxito:

Nos regresa siempre un mensaje ya sea exitoso o de que existió un error durante la ejecución y conexión con la API, de ser exitoso muestra el siguiente mensaje:

```
{

        "mensaje": "Conexion success"
}
```

Además se realiza la correspondiente consulta para obtener los datos de todos los clientes existentes, donde se obtienen estas variables de regreso aparte del mensaje de éxito anterior:

```
{
    "idc ": {
        "id": "",
        "nombre": "",
        "nombre_comercial": "
        "direccion": "",
```

```
            "ciudad": "",
            "estado": "",
            "codigo_postal": "",
            "telefono": "",
            "correo": "",
            "rfc": "",
            "curp": "",
            "estado_cliente": ""
        },
        "mensaje": "Conexion success"
    }
```


3.2- Mensajes de error:

De existir un error la hora de la conexión con la API, podrían existir los siguientes casos:

a) Si el token mandando por POST no es válido, dado que es erróneo o no es el mismo al que corresponde a ese correspondiente id de token , el error a obtener será el siguiente:


```
{
    "mensaje": "Conexion fail : el token no es valido"
}
```


b) Si el token obtenido en la función obtener_token(gracias al id token enviado por POST) no existe, no esta activo o no es diferente de cero, el error a obtener será el siguiente:


```
{
    "mensaje": "Conexion fail : el token no existe"
}
```


c) Si al realizar la consulta de los clientes, el token o el id token se encuentra vacío, es cero o tiene algún error (desde la forma de envió o ser otro tipo de variable no adecuada).

Nos indicaría que existió ese fallo en la variable a recibir, por lo que el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail: no se ingreso el token o id_token
    correctamente o exite un error dentro del mismo";
}
```

-Api_plantas_cliente.php:

1- URL: http://sigersa.homelinux.com/screw/api/api_plantas_cliente.php

2- Variables a recibir:

Las variables con su correspondiente información, que son necesarias para poder realizar la correspondiente consulta que nos dará los datos de la o las planta correspondiente al id_cliente ingresado dentro de la API son las siguientes:

```
{
        token(dato obligatorio) - tipo: cadena
        id_token(dato obligatorio) - tipo: número entero
        id_cliente(dato obligatorio) - tipo: número entero
}
```

3- Variables que regresan:

3.1- Mensaje de éxito:

Nos regresa siempre un mensaje ya sea exitoso o de que existió un error durante la ejecución y conexión con la API, de ser exitoso muestra el siguiente mensaje:

```
{
        "mensaje": "Conexion success"
}
```

Además, se realiza la correspondiente consulta para obtener los datos de la o las plantas solicitadas en id cliente correspondientes a ese cliente en particular, donde se obtienen estas variables de regreso aparte del mensaje de éxito anterior en una o más plantas:

```
{
    "id": {
        "id": "",
        "id_cliente": "",
        "nombre": "",
        "numero_serie": "",
        "kwh": "",
        "voltaje": "",
        "motor": "",
        "modelo": "",
        "cpl": "",
        "descripcion": "",
        "estado": ""
    },
    "mensaje": "Conexion success"
}
```

3.2- Mensajes de error:

De existir un error la hora de la conexión con la API, podrían existir los siguientes casos:

a) Si el token mandando por POST no es válido, dado que es erróneo o no es el mismo al que corresponde a ese correspondiente id de token , el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail : el token no es valido"
}
```

b) Si el token obtenido en la función obtener_token(gracias al id token enviado por POST) no existe, no está activo o no es diferente de cero, el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail : el token no existe"
}
```


c) Se dispone con un contador, el cual empieza en cero y va aumentando cuando se realizar la búsqueda de una planta, siempre y cuando el id cliente exista dentro de las plantas existentes. Donde al obtener en contador cualquier número mayor a cero, nos indicaría que existió ese fallo en la variable id cliente a recibir, por lo que el error a obtener será el siguiente:


```
{
    "mensaje": "Conexion fail : el id_cliente no existe"
}
```


d) Si al realizar la búsqueda de un cliente el token, el id token o id cliente se encuentra vacío, es cero o tiene algún error (desde la forma de envió o ser otro tipo de variable no adecuada). Nos indicaría que existió ese fallo en la variable a recibir, por lo que el error a obtener será el siguiente:


```
{
    "mensaje": "Conexion fail: no se ingreso el token, id_token o
    id_cliente correctamente o exite un error dentro del mismo";
}
```


-Api_reporte.php:

1- URL: http://sigersa.homelinux.com/screw/api/api_reporte.php

2- Variables a recibir:

Las variables con su correspondiente información, que son necesarias para poder guardar el nuevo reporte de servicio de la API son las siguientes:

```
{

        token(dato obligatorio) - tipo: cadena
        id_token(dato obligatorio) - tipo: número entero
        id_usuario(dato obligatorio) - tipo: número entero
        id_cliente(dato obligatorio) - tipo: número entero
        id_planta(dato obligatorio) - tipo: número entero
        tipo_servicio(dato obligatorio) - tipo: número bit(solo acepta dos valores: 1
        que corresponde al valor verdadero y 0 que corresponde al valor falso, donde
        cualquier otro numero ingresado diferente de cero será guardado como cero)
        recibe_nombre(dato obligatorio) - tipo: cadena
        puesto - tipo: cadena
        teléfono - tipo: cadena
        filt_prim - tipo: número bit(solo acepta dos valores: 1 que corresponde al
        valor verdadero y 0 que corresponde al valor falso, donde cualquier otro
        numero ingresado diferente de cero será guardado como cero)
        filt_secun - tipo: número bit(especificaciones anteriores de 1 y 0)
        filt_aceite - tipo: número bit(especificaciones anteriores de 1 y 0)
        filt_aire - tipo: número bit(especificaciones anteriores de 1 y 0)
        filt_agua - tipo: número bit(especificaciones anteriores de 1 y 0)
        filt_separador - tipo: número bit(especificaciones anteriores de 1 y 0)
        bateria_mca - tipo: número bit(especificaciones anteriores de 1 y 0)
        revision_reaprete - tipo: número bit(especificaciones anteriores de 1 y 0)
        nivel_combus - tipo: número bit(especificaciones anteriores de 1 y 0)
        revis_refri - tipo: número bit(especificaciones anteriores de 1 y 0)
        revis_panal - tipo: número bit(especificaciones anteriores de 1 y 0)
        revis_bandas - tipo: número bit(especificaciones anteriores de 1 y 0)
        revis_apriete - tipo: número bit(especificaciones anteriores de 1 y 0)
        revis_aceite - tipo: número bit(especificaciones anteriores de 1 y 0)
        revis_estado - tipo: número bit(especificaciones anteriores de 1 y 0)
        revis_precal - tipo: número bit(especificaciones anteriores de 1 y 0)
        revis_cableado - tipo: número bit(especificaciones anteriores de 1 y 0)
        estado_fisico - tipo: número bit(especificaciones anteriores de 1 y 0)
        pintura_equipo - tipo: número bit(especificaciones anteriores de 1 y 0)
        n_equipo - tipo: número bit(especificaciones anteriores de 1 y 0)
        voltaje - tipo: número bit(especificaciones anteriores de 1 y 0)
        modulo_control - tipo: número bit(especificaciones anteriores de 1 y 0)
        hrs_trabajo - tipo: número bit(especificaciones anteriores de 1 y 0)
        tipo_uni - tipo: número bit(especificaciones anteriores de 1 y 0)
        voltaje_red - tipo: número bit(especificaciones anteriores de 1 y 0)
        modelo_sensor - tipo: número bit(especificaciones anteriores de 1 y 0)
        reaprete_sistema - tipo: número bit(especificaciones anteriores de 1 y 0)
        inspeccion_fisica - tipo: número bit(especificaciones anteriores de 1 y 0)
        inspeccion_tierras - tipo: número bit(especificaciones anteriores de 1 y 0)
        inspeccion_neutros - tipo: número bit(especificaciones anteriores de 1 y 0)
        mod_carg - tipo: número bit(especificaciones anteriores de 1 y 0)
        cal_norm - tipo: número bit(especificaciones anteriores de 1 y 0)
        cal_carga - tipo: número bit(especificaciones anteriores de 1 y 0)
        cal_emerg - tipo: número bit(especificaciones anteriores de 1 y 0)
        pintura_general - tipo: número bit(especificaciones anteriores de 1 y 0)
        limpieza_general - tipo: número bit(especificaciones anteriores de 1 y 0)
```

```
prox_cam - tipo: número bit(especificaciones anteriores de 1 y 0)
presion_aceite - tipo: número bit(especificaciones  anteriores de 1 y 0)
temperatura_refrigerante - tipo: número bit(especificaciones anteriores de 1
y 0)
saque_presion
saque_generacion - tipo: número bit(especificaciones anteriores de 1 y 0)
voltaje_generacion - tipo: número bit(especificaciones anteriores de 1 y 0)
frecuencia - tipo: número bit(especificaciones anteriores de 1 y 0)
voltaje_alternador - tipo: número bit(especificaciones anteriores de 1 y 0)
baja_presion - tipo: número bit(especificaciones anteriores de 1 y 0)
alta_temperatura - tipo: número bit(especificaciones anteriores de 1 y 0)
falla_generacion - tipo: número bit(especificaciones anteriores de 1 y 0)
baja_velocidad - tipo: número bit(especificaciones anteriores de 1 y 0)
paro_emergencia - tipo: número bit(especificaciones anteriores de 1 y 0)
intentos_arranque - tipo: número bit(especificaciones anteriores de 1 y 0)
sobre_velocidad - tipo: número bit(especificaciones anteriores de 1 y 0)
sobre_carga - tipo: número bit(especificaciones anteriores de 1 y 0)
tiempo_arranque - tipo: número bit(especificaciones anteriores de 1 y 0)
tiempo_transferencia - tipo: número bit(especificaciones anteriores de 1 y 0)
tiempo_retransferencia - tipo: número bit(especificaciones anteriores de 1 y
0)
tiempo_desfoge - tipo: número bit(especificaciones anteriores de 1 y 0)
tiempo_prueba - tipo: número bit(especificaciones anteriores de 1 y 0)
voltaje_generacion_carga - tipo: número bit(especificaciones anteriores de 1
y 0)
frecuencia_carga - tipo: número bit(especificaciones anteriores de 1 y 0)
amparaje - tipo: número bit(especificaciones anteriores de 1 y 0)
presion_aceite_carga - tipo: número bit(especificaciones anteriores de 1 y 0)
temperatura_refrigerante_carga - tipo: número bit(especificaciones anteriores
de 1 y 0)
trabajos_realizados - tipo: cadena
recomendaciones_tecnico - tipo: cadena
material_utilizado - tipo: cadena
observaciones_cliente - tipo: cadena
firma_tecnico - tipo: cadena
firma_cliente - tipo: cadena

}
```

3- Variables que regresan:

3.1- Mensaje de éxito:

Nos regresa siempre un mensaje ya sea exitoso o de que existió un error durante la ejecución y conexión con la API, de ser exitoso muestra el siguiente mensaje:

```
{
        "id_planta": "",
        "mensaje": "Conexion success"
```

```
}
```

Además, se realiza la correspondiente guardado de información del nuevo reporte de servicio, con todas las variables o datos que fueron recibidos y nos brindara el número de id_planta correspondiente.

3.2- Mensajes de error:

De existir un error la hora de la conexión con la API, podrían existir los siguientes casos:

a) Si el token mandando por POST no es válido, dado que es erróneo o no es el mismo al que corresponde a ese correspondiente id de token , el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail : el token no es valido"
}
```

b) Si el token obtenido en la función obtener_token(gracias al id token enviado por POST) no existe, no esta activo o no es diferente de cero, el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail : el token no existe"
}
```

c) Se dispone con un contador, el cual empieza en cero y va aumentando cada vez que una variable a recibir se encuentra vacía, no es enviada o se envió de manera errónea, esto solo sucede con las variables que son un dato obligatorio (son token, id_token, id_usuario, id_cliente, id_planta, tipo_servicio y recibe_nombre). Donde al obtener en contador cualquier número mayor a 0, nos indicaría que existió ese fallo en las variables a recibir, por lo que el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail : faltan datos obligatorios"
```

```
}
```

d) Si al ingresar las variables a recibir de él token o el id token se encuentra vacío, es cero o tiene algún error (desde la forma de envió o ser otro tipo de variable no adecuada). Nos indicaría que existió ese fallo en la variable a recibir, por lo que el error a obtener será el siguiente:

```
{
    "mensaje": "Conexion fail: no se ingreso el token o id_token
    correctamente o exite un error dentro del mismo";
}
```

e) Si al ingresar las variables a recibir de carácter obligatorio (son token, id_token, id_usuario, id_cliente, id_planta, tipo_servicio y recibe_nombre), cualquiera se encuentra vacía, es cero o tiene algún error (desde la forma de envió o ser otro tipo de variable no adecuada). Nos indicaría que existió ese fallo en la variable a recibir obligatoria, por lo que el error a obtener será el siguiente:

```
{
    "mensaje": " El id_usuario(variable en particular) no existe o se
    encuentra vacio";
}
```