

20085694-R

24/12/2021

SD	Sistemas Distribuidos
21/22	Práctica no guiada: Seguridad y API Rest
	Gestor de colas en parques temáticos “Fun with Queues”



**Universitat
d'Alacant**

JUAN GARCÍA MARTÍNEZ

SISTEMAS DISTRIBUIDOS
UNIVERSIDAD ALICANTE

COMPONENTES SOFTWARE

API REGISTRY

En la práctica anterior empleamos sockets para la comunicación del visitante con el componente **Registro**. En esta segunda practica hemos realizado una API que replica el comportamiento del componente **Registro** pero con dos mejoras, seguridad y registro de auditoría.

Para realizar la seguridad en la API hemos empleado cifrado asimétrico con certificado HTTPS y cifrado irreversible para las contraseñas de los usuarios.

Para ello, instalamos una aplicación llamada openSSL e introducimos una serie de comandos, generando así un certificado, que es la clave publica, y una clave privada.

Con estas claves, crearemos el servidor HTTPS que empleará la API Registry:

```
const OPTIONS_HTTPS = {
  key: fs.readFileSync('./cert/key.pem'),
  cert: fs.readFileSync('./cert/cert.pem')
};

https.createServer(OPTIONS_HTTPS, app).listen(port, () => {
  console.log('Ejecutando la API Registro en el puerto ${port}');
});
```

Respecto al registro de auditoría, hemos creado una base de datos en la que se guardan todos los eventos producidos en la API Registro, tanto eventos correctos como erróneos.

A continuación, un ejemplo del registro de auditoria:

ip	fecha	accion	parametros
::ffff:127.0.0.1	Wed Dec 22 2021 15:20:44 ...	Listar un usuario.	admin
::ffff:127.0.0.1	Wed Dec 22 2021 15:24:15 ...	Listar un usuario.	jose
::ffff:192.168.56.1	Wed Dec 22 2021 15:24:51 ...	Listar un usuario.	jose
::ffff:192.168.56.1	Wed Dec 22 2021 15:26:16 ...	Listar un usuario.	jose
::ffff:192.168.56.1	Thu Dec 23 2021 20:39:34 G...	Error en la creacion del usuario. SQLITE_CONSTRAINT: UNIQUE ...	perico
::ffff:192.168.56.1	Thu Dec 23 2021 20:41:55 G...	Se ha registrado un nuevo usuario	pablo
::ffff:192.168.56.1	Thu Dec 23 2021 20:42:24 G...	Error en la creacion del usuario. SQLITE_CONSTRAINT: UNIQUE ...	perico
::ffff:192.168.56.1	Thu Dec 23 2021 20:45:06 G...	Error en la creacion del usuario. SQLITE_CONSTRAINT: UNIQUE ...	admin

API ENGINE

Es un API Rest que contiene los métodos necesarios para que el **FRONT** consuma de él. Este API no contiene seguridad ni registro de auditoría. A continuación, unos métodos de la API:

```
//Listamos los usuarios
app.get('/usuarios',function (req, res) {
  console.log('Listado de todos los usuarios');

  sql = 'select * from user'
  dbUser.all(sql, [], (err, rows) => {
    if (err) {
      return console.error(err.message)
    }else{
      if (rows.length > 0){
        res.json(rows)
      }else{
        res.send('No hay resultados');
      }
    }
  })
})

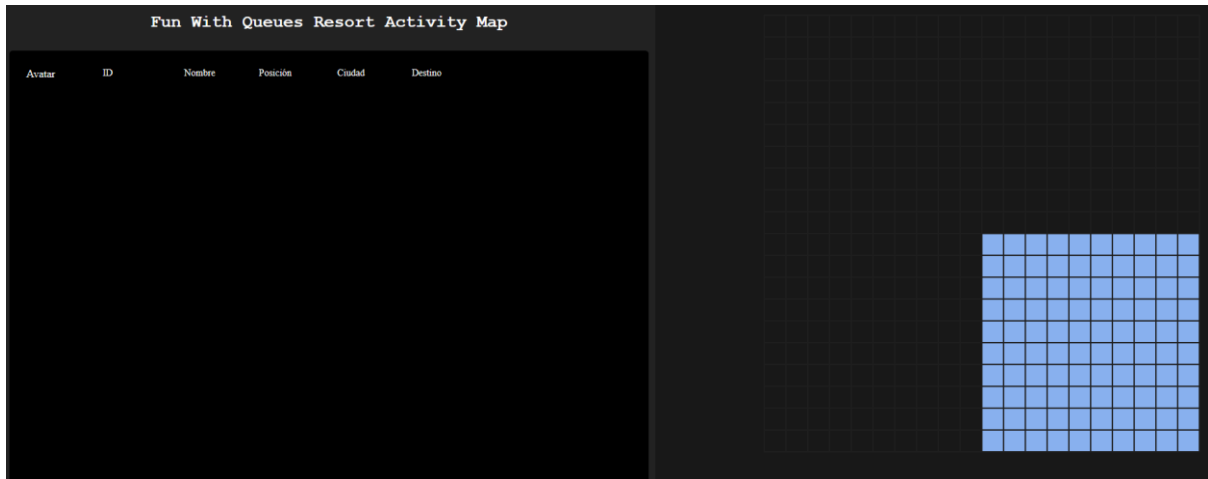
//Listamos los usuarios que estan en el mapa
app.get('/usuarios/inPark',function (req, res) {
  console.log('Listado de todos los usuarios que estan en el parque');

  sql = 'select * from user where inPark = 1'
  dbUser.all(sql, [], (err, rows) => {
    if (err) {
      return console.error(err.message)
    }else{
      if (rows.length > 0){
        res.json(rows)
      }else{
        res.send([]);
      }
    }
  })
})
```

FRONT

Este componente será nuestra pagina web que dará visual a nuestra **API Engine**. Para realizar este front-end he empleado Django (Python, CSS, HTML)

El front-end recoge los errores que puedan surgir con las peticiones a el **API Engine**, así como si se desconecta alguna atracción, sensor, servidor de tiempos de espera o incluso el Engine.



Así se vería inicialmente con todo desconectado y cuando pasan 6 segundos y el mapa no cambia, el sistema detecta que algo falla en el Engine y tira un mensaje de error:

Parece que el componente Engine esta desconectado. Intentalo de nuevo mas tarde

Si por ejemplo nos equivocásemos en la llamada a la API, tiraría otro mensaje de error distinto:

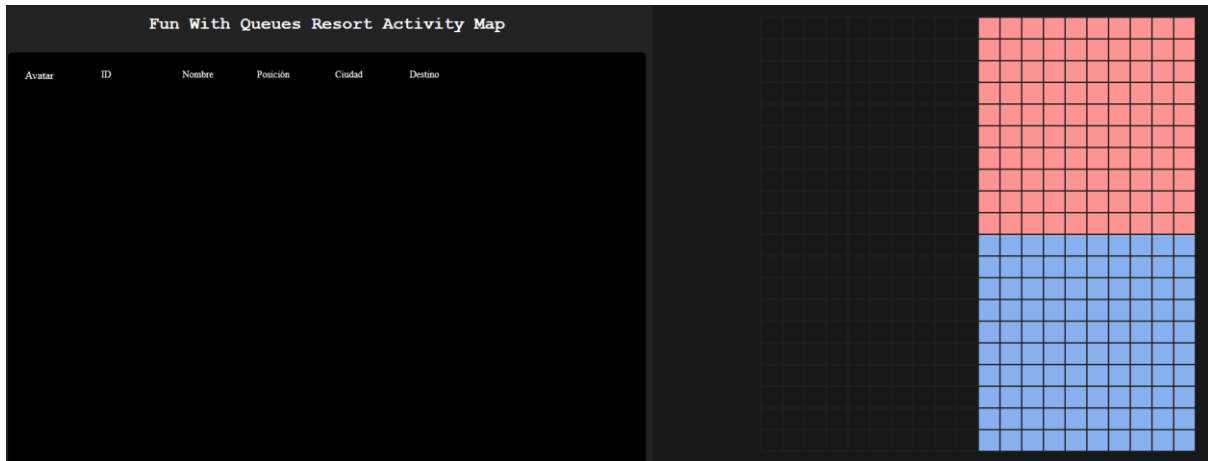
Se ha producido un error al conectarse a la API Engine. Intentalo de nuevo más tarde.

ENGINE

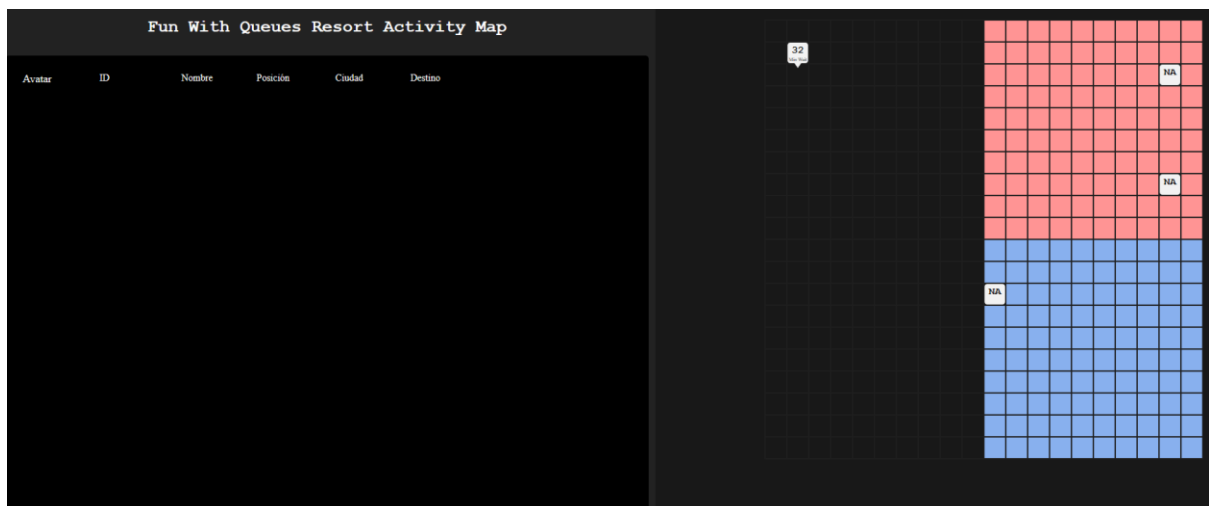
Este componente va a ser exactamente el mismo, solo que añadiendo un consumo del API de OpenWeatherMap.

Una vez llamamos a la API con 4 ciudades distintas, dividiremos el mapa en 4 cuadrantes, donde cada uno de ellos pertenecerá a una ciudad. Si la temperatura es menor que 20 o mayor que 30 esa ciudad y todas sus atracciones quedarán inaccesibles para los visitantes.

Ejemplo de 2 ciudades accesibles y 2 inaccesibles (una por frio y otra por calor):



Ahora conectamos el servidor de tiempos de espera con sus sensores, como podemos observar, las atracciones que están en ciudades inaccesibles están desconectadas.



VISITANTE

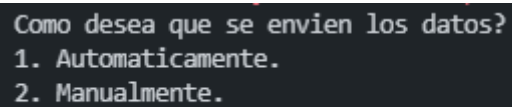
Este componente solo cambiará la comunicación con el registro que en este caso se realizará mediante una API Rest.

Para ello, el visitante tendrá que consumir de esta API para poder registrarse y poder modificar su perfil.

Todas las funcionalidades son exactamente las mismas que la Práctica 1.

SENSOR

El sensor hace lo mismo que la práctica anterior, pero falto implementar que se pudiesen meter los tiempos por consola. Ahora te da la opción para que puedas realizarlo a la forma que más guste.



```
Como desea que se envíen los datos?  
1. Automáticamente.  
2. Manualmente.  
█
```

DESPLIEGUE DEL SISTEMA

API ENGINE

```
PS C:\Users\johns\Desktop\SDPractica\FWQPython\API_Engine> node .\api_engine.js 192.168.56.1 3001
Ejecutando la aplicación API REST de Engine en 192.168.56.1 con el puerto 3001
Preparado para guardar logs.
Conectado a la base de datos del parque correctamente.
Conectado a la base de datos de los usuarios correctamente.
```

□

API REGISTRO

```
PS C:\Users\johns\Desktop\SDPractica\FWQPython\API_Registro> node .\api_registroNoCert.js 192.168.56.1 3001
Ejecutando la API Registro en 192.168.56.1 con puerto 3001
Preparado para guardar logs.
Conectado a la base de datos de logs correctamente.
Conectado a la base de datos de usuarios correctamente.
```

■

KAFKA

```
C:\kafka> .\bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties
[2021-12-25 14:26:49,179] INFO Reading configuration from: .\config\zookeeper.properties
[2021-12-25 14:26:49,179] INFO ClientConfig: Loading many client properties to the configuration
[2021-12-25 14:26:49,185] INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.server.NIOServerCnxn)
C:\kafka> .\bin\windows\kafka-server-start.bat .\config\server.properties
[2021-12-25 17:19:37,658] INFO Registered kafka:type=kafka.Log4jController MBean
[2021-12-25 17:19:37,658] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotiation=true
[2021-12-25 17:19:37,658] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotiation=true
[2021-12-25 17:19:38,050] INFO starting (kafka.server.KafkaServer)
[2021-12-25 17:19:38,051] INFO Connecting to zookeeper on localhost:2181 (kafka.server.KafkaServer)
```

ENGINE

```
PS C:\Users\johns\Desktop\SDPractica\FWQPython> python .\FWQ_Engine.py 192.168.56.1 9092 1000 192.168.56.1 1500
[LISTENING] Escuchando mensajes de Kafka en el servidor 192.168.56.1:9092
La temperatura en Caracas es de 23.31
La temperatura en Ati es de 27.87
La temperatura en Dubai es de 24.92
La temperatura en Benidorm es de 16.43
```

VISITANTE

```
PS C:\Users\johns\Desktop\SDPractica\FWQPython> python FwQ_Visitor.py 192.168.0.173 9092 192.168.56.1 3000
***** WELCOME TO OUR BRILLIANT SD UA CURSO 2020/2021 SOCKET CLIENT *****
0.Iniciar Sesión.
1.Crear perfil.
```

WAITING TIME SERVER

```
PS C:\Users\johns\Desktop\SDPractica\FWQPython> python .\FWQ_WaitingTimeServer.py 1500 192.168.56.1 9092
[STARTING] FWQ_WaitingTimeServer inicializándose...
[LISTENING] FWQ_WaitingTimeServer a la escucha en ('192.168.56.1', 1500)
[LISTENING] Escuchando mensajes de Kafka en el servidor 192.168.56.1:9092
```

SENSOR

```
PS C:\Users\johns\Desktop\SDPractica\FWQPython> python .\FWQ_Sensor.py 0 192.168.56.1 9092
Como desea que se envíen los datos?
1. Automáticamente.
2. Manualmente.
```

FRONT

```
(django) C:\Users\johns\Desktop\SDPractica\FWQPython\FWQ_Visitor>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
December 25, 2021 - 17:39:37
Django version 3.2.9, using settings 'FWQ_Visitor.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```