

# Descripción de Funcionalidades.

## Módulo: Bitácora.

Brinda al usuario la funcionalidad de registrar mensajes para manejar errores, advertencias, información y rastros.

Funcionalidades:

1. Almacena los registros en un archivo formato Json con la capacidad de acomodarlos por día.
2. Permite modificar el archivo de configuración durante tiempo de ejecución.
3. Envía correos electrónicos al e-mail definido en el archivo de configuración.

## Módulo: Registros.

Informar al usuario acerca de excepciones, información en partes del código en fase de prueba, registros tipo "trace", etc.

Funcionalidades:

1. Capacidad de definir el tipo de mensaje eligiendo entre cinco tipos: "fatal", "error", "warn", "info" y "trace".
2. Registra la fecha en que se efectuó el registro, información definida por el usuario, descripción del hardware y software de la computadora donde se generó el registro y la versión de este framework.

## Módulo: Archivo de configuración.

Permite al usuario definir el uso de operaciones que faciliten el trabajo con el framework.

Funcionalidades:

1. Define el tamaño máximo para los archivos de registro con el propósito de subdividirse en varios documentos.
2. Permite al usuario decidir la activación de los mensajes por correo electrónico igual le permite definir el correo y la contraseña del remitente.
3. Concede establecer un directorio donde se generarán los registros con sus carpetas contenedoras.
4. Tiene una funcionalidad extra que agrega al registro un dato adicional.

# Descripción de Componentes.

## Nombre: Bitácora

**Descripción:** Brinda al usuario la funcionalidad de registrar mensajes para manejar errores, advertencias, información y rastros.

**Dependencias con otros componentes:** El archivo de configuración.

### Interfaces de Salida:

La creación de los registros.

Modificación del archivo de configuración.

Envía los registros por correo electrónico.

### Interfaces de Entrada:

Ocupa obtener los registros anteriores de los archivos JSON.

Requiere el archivo de configuración para definir la ruta donde guardará los registros.

Si el modo “e-mail” está activado, requiere el archivo de configuración para activarlo y entonces generar una conexión con el servidor de correo electrónico de Gmail.

Para conocer el remitente y destinatario se necesita el archivo de configuración.

La información del software y hardware se obtiene mediante la librería Sigar.

### Artefactos:

Biblioteca Sigar, Gson y Mail.

## Nombre: Archivo de configuración

**Descripción:** Permite al usuario definir el uso de operaciones que faciliten el trabajo con el framework.

**Dependencias con otros componentes:** No tiene dependencias.

### Interfaces de Salida:

Define los siguientes valores:

- Tamaño del archivo de registros
- Version del framework
- Activa el modo “e-mail” y define el correo electrónico y su contraseña donde se enviarán los mensajes
- Puedo establecer un dato adicional
- La dirección donde se guardaron los archivos JSON de registros.
- Define la prioridad de los niveles

**Interfaces de Entrada:**

Requiere ser modificada por el usuario para definir los valores mencionados anteriormente.

**Artefactos:** No usa.

**Nombre:** Registros.

**Descripción:** Informar al usuario acerca de excepciones, información en partes del código en fase de prueba, registros tipo “trace”, etc.

**Dependencias con otros componentes:** Bitácora.

**Interfaces de Salida:**

Llevan consigo los mensajes definidos por el usuario junto con el nivel del mensaje.

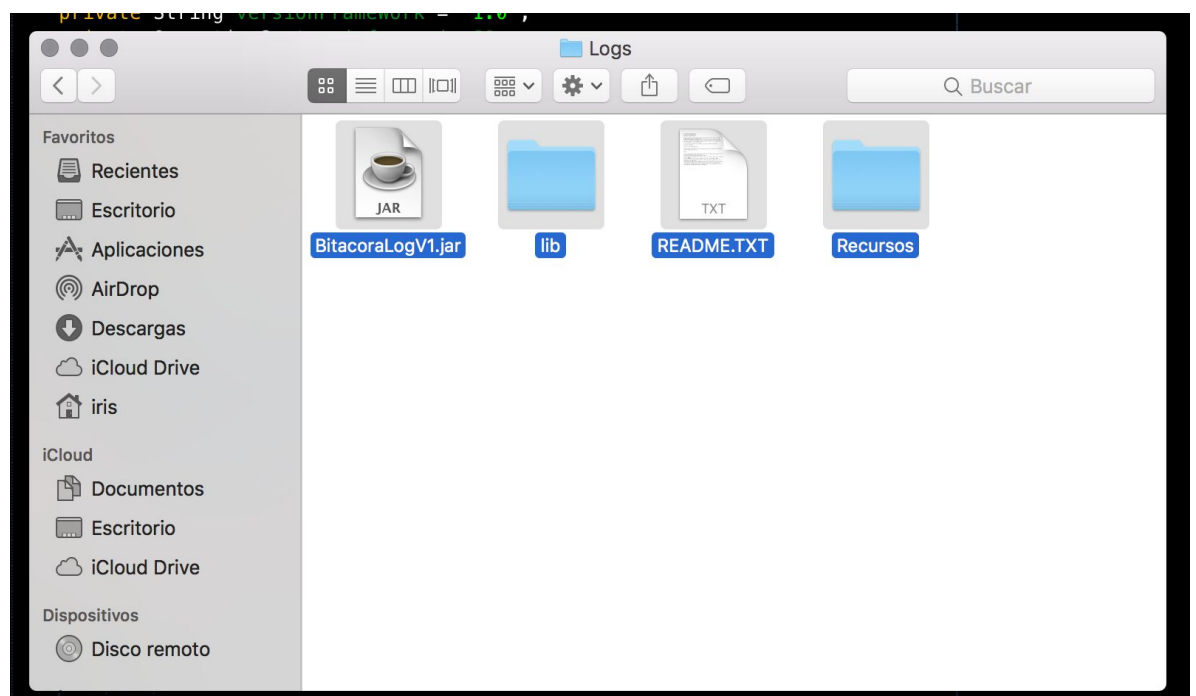
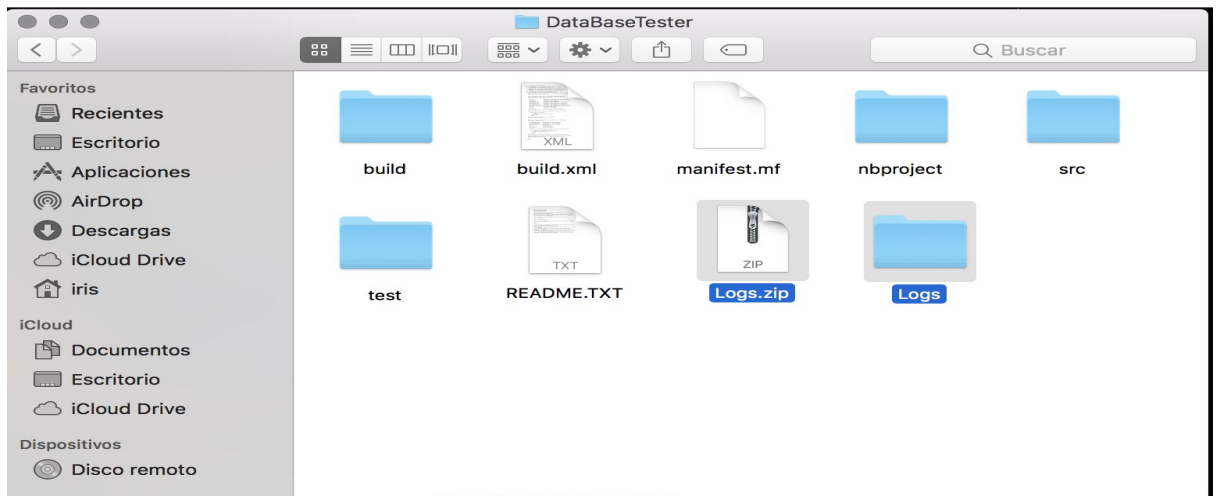
**Interfaces de Entrada:**

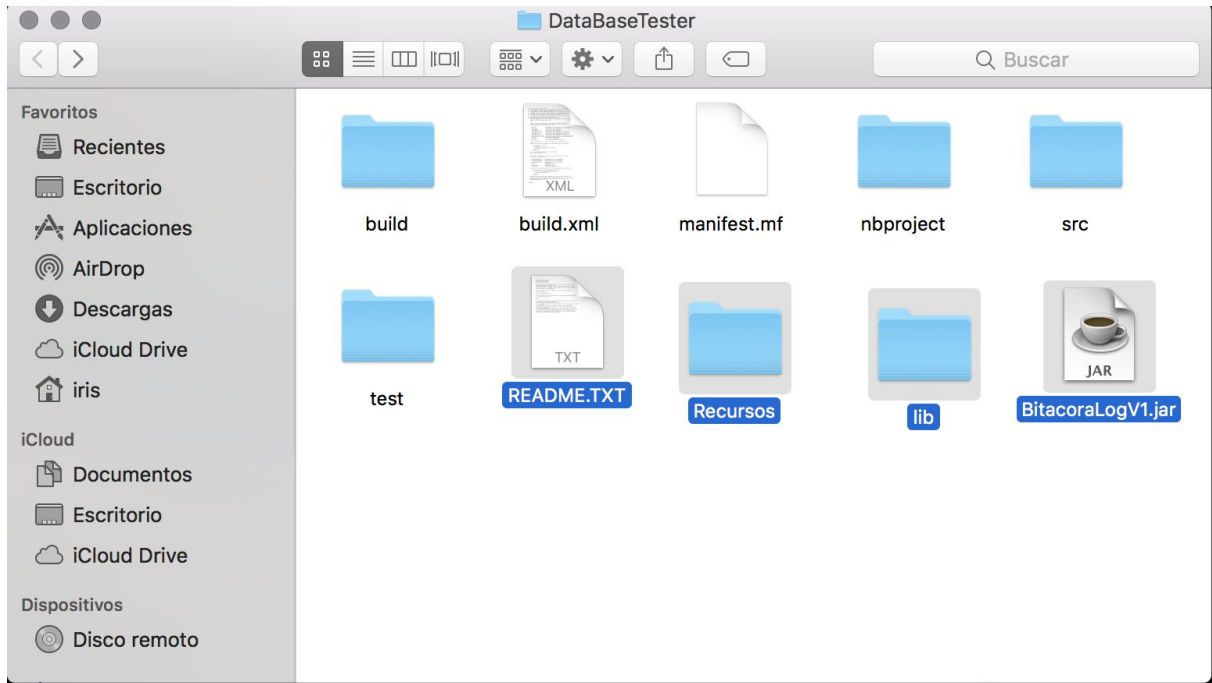
Requiere ser creados por la bitácora definiendo su nivel y el mensaje que va a llevar con el.

**Artefactos:** Biblioteca Gson.

## Ejemplo de uso:

- 1) Descomprimos el archivo .zip en la carpeta contenedora de nuestro proyecto.





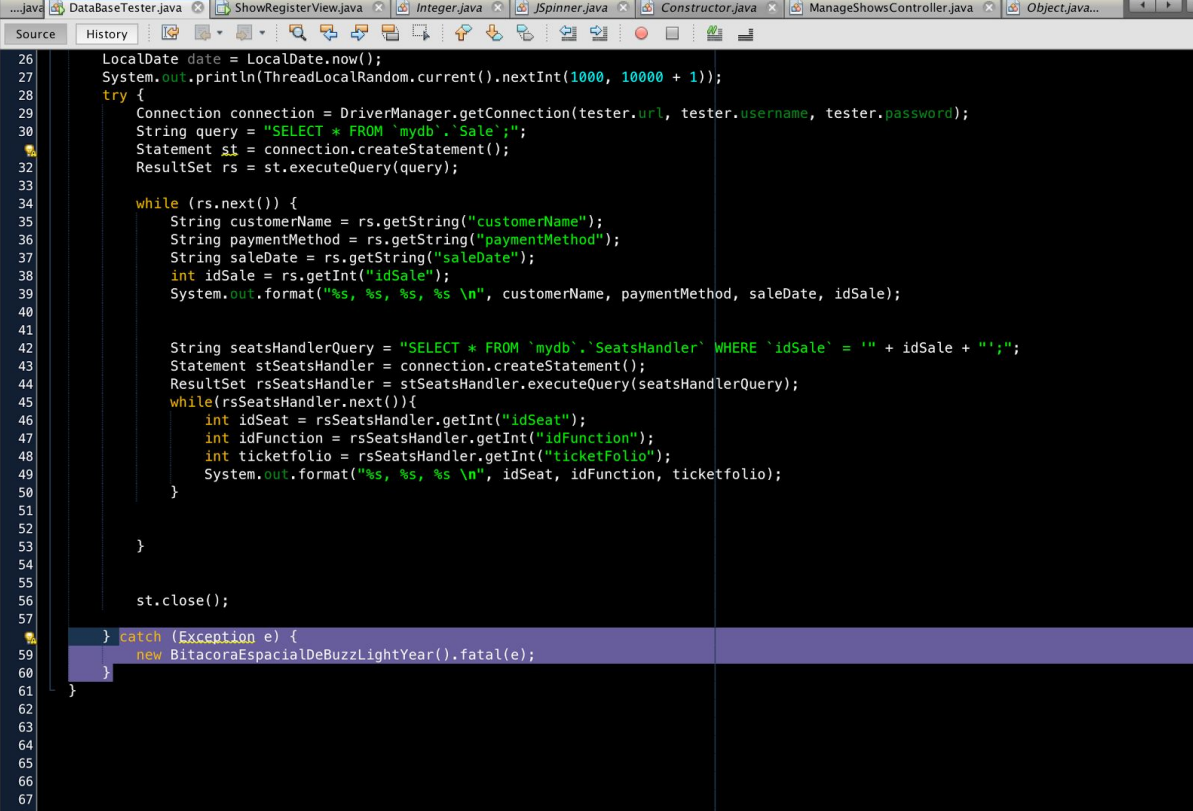
2) Modificamos el archivo de configuración con los datos requeridos.

```

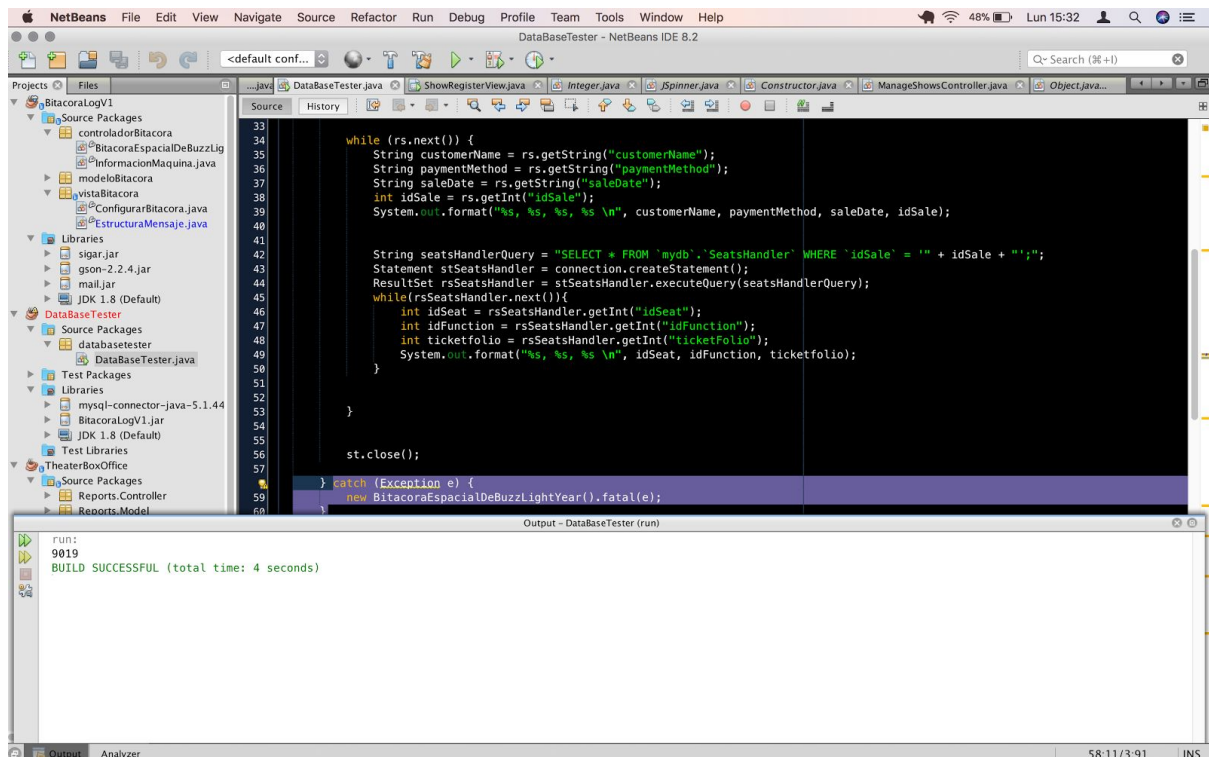
ArchivoConfiguracion.txt
<?xml version = "1.0"?>
<class>
  <Bitacora version = "1.0">
    <Nivel>Info</Nivel>
    <version> 1.0 </version>
    <Tamaño>1000</Tamaño>
    <ModoCorreo>True</ModoCorreo>
    <CorreoElectronico>bitacoraespacialbuzzlightyear</CorreoElectronico>
    <ContraseñaCorreo>bitacbraespacialdebuzzlightyear</ContraseñaCorreo>
    <DireccionGuardado>Users/iris/NetBeansProjects/DataBaseTester/Recursos/
  </Bitacora>
  <ContenidoExtra>Usuario: Desconocido</ContenidoExtra>
</class>

```

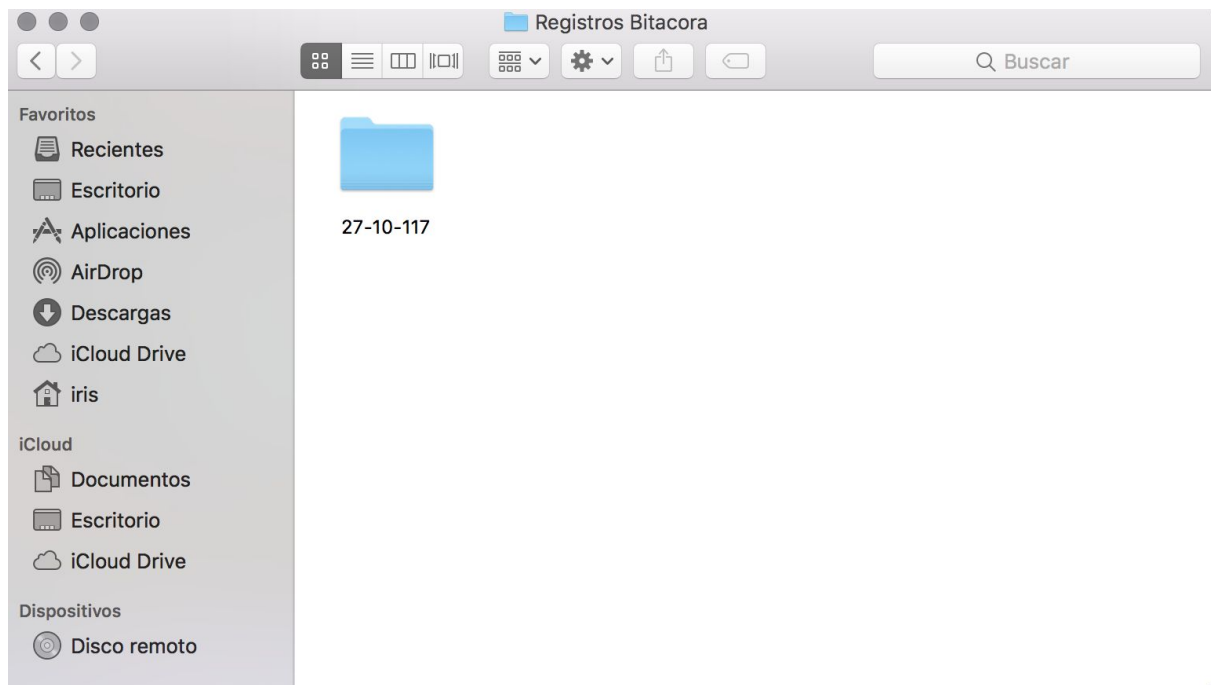
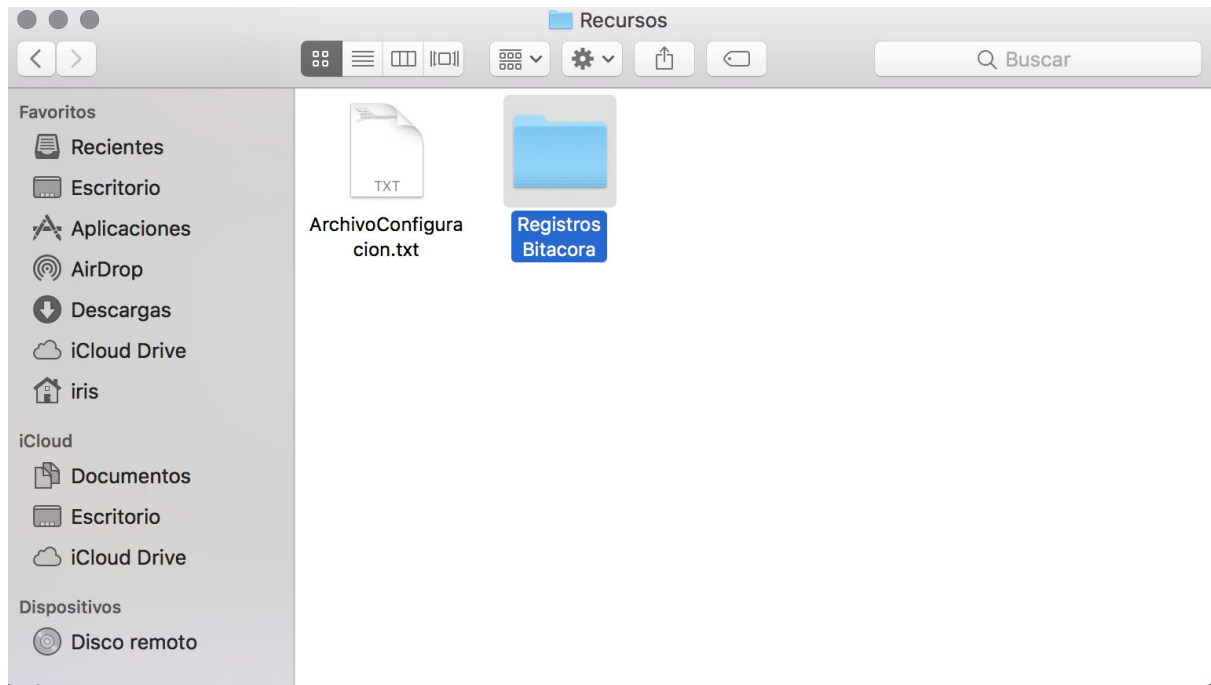
### 3) Preparamos un mensaje de tipo fatal cuando exista una excepción de SQL.



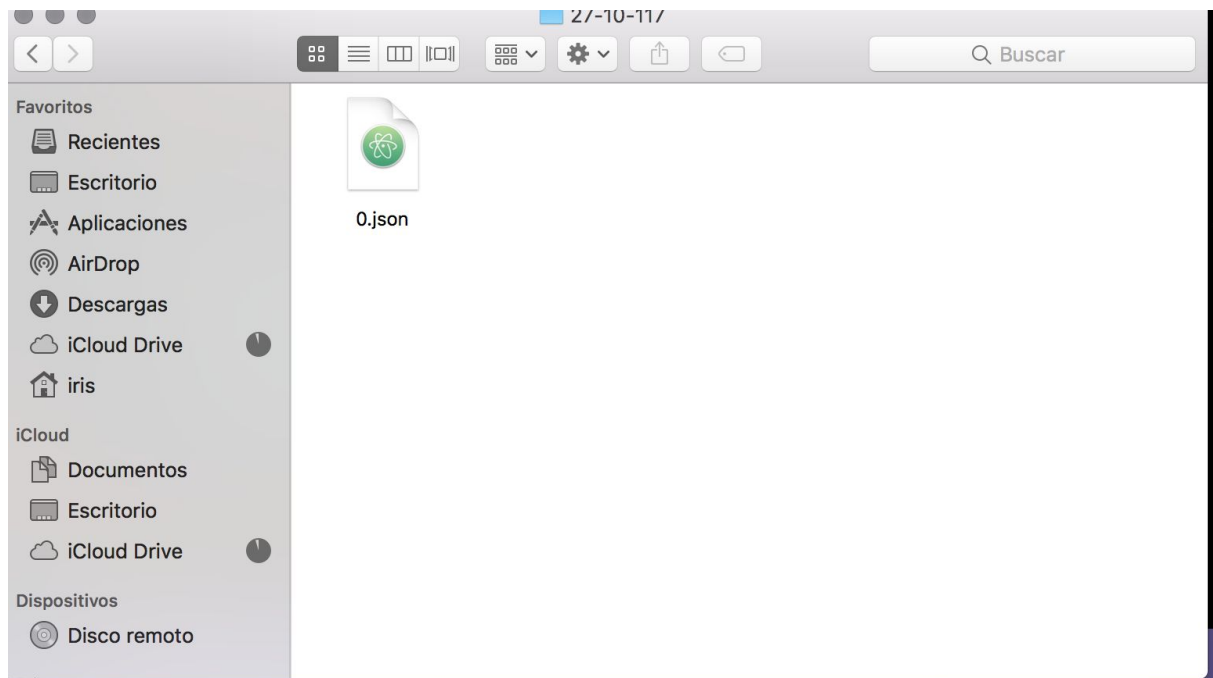
```
26 LocalDate date = LocalDate.now();
27 System.out.println(ThreadLocalRandom.current().nextInt(1000, 10000 + 1));
28 try {
29     Connection connection = DriverManager.getConnection(tester.url, tester.username, tester.password);
30     String query = "SELECT * FROM 'mydb'. 'Sale' ";
31     Statement st = connection.createStatement();
32     ResultSet rs = st.executeQuery(query);
33
34     while (rs.next()) {
35         String customerName = rs.getString("customerName");
36         String paymentMethod = rs.getString("paymentMethod");
37         String saleDate = rs.getString("saleDate");
38         int idSale = rs.getInt("idSale");
39         System.out.format("%s, %s, %s, %s \n", customerName, paymentMethod, saleDate, idSale);
40
41         String seatsHandlerQuery = "SELECT * FROM 'mydb'. 'SeatsHandler' WHERE 'idSale' = " + idSale + " ";
42         Statement stSeatsHandler = connection.createStatement();
43         ResultSet rsSeatsHandler = stSeatsHandler.executeQuery(seatsHandlerQuery);
44         while(rsSeatsHandler.next()){
45             int idSeat = rsSeatsHandler.getInt("idSeat");
46             int idFunction = rsSeatsHandler.getInt("idFunction");
47             int ticketfolio = rsSeatsHandler.getInt("ticketFolio");
48             System.out.format("%s, %s, %s \n", idSeat, idFunction, ticketfolio);
49         }
50     }
51
52     st.close();
53
54 } catch (Exception e) {
55     new BitacoraEspacialDeBuzzLightYear().fatal(e);
56 }
57 }
```



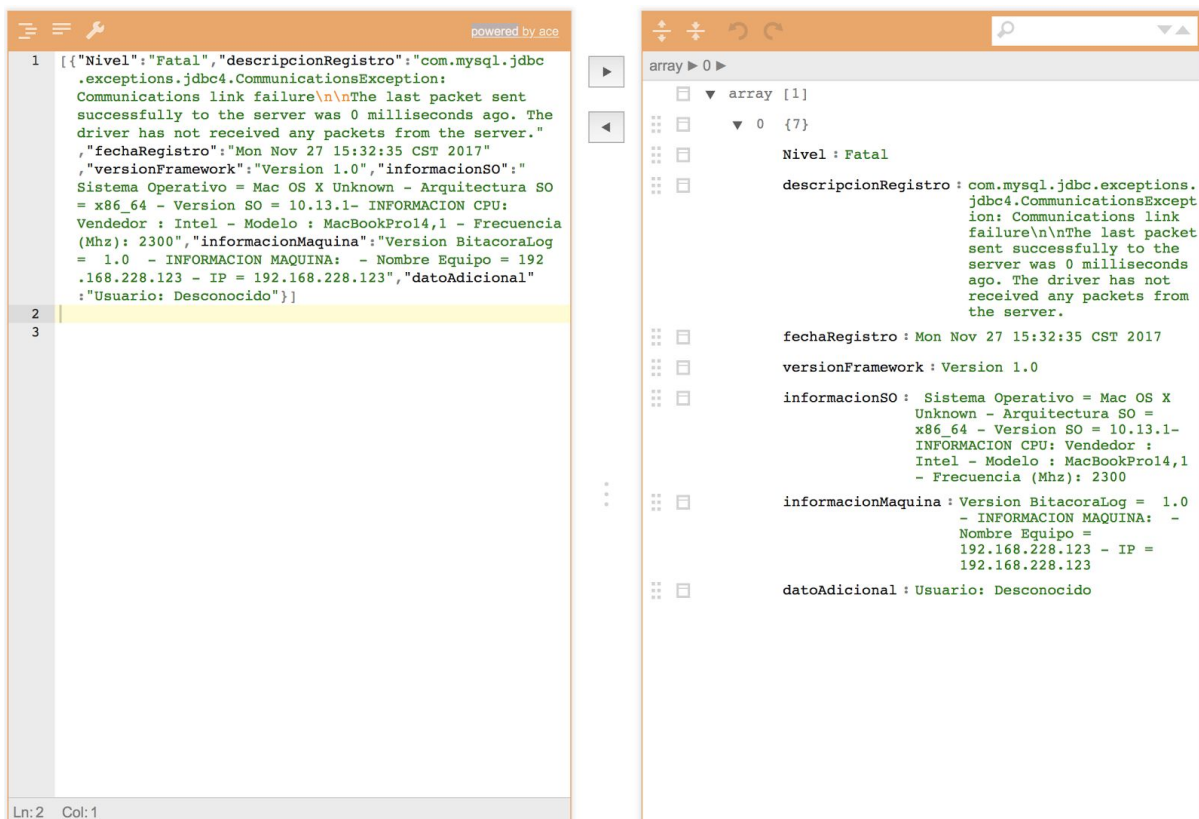
5) Carpeta de los registros en la dirección que establecimos en el archivo de configuración.



## 6) Archivo json creado



## 7) Archivo JSON compilado.





## 8) Mensaje Gmail.

Google

Gmail

Más

1-4 de 4

<

>

Es

REDACTAR

Recibidos (3)

Destacados

Enviados

Borradores

Más

Buzz

Principal

Social

Promociones

<input type="checkbox"/>	<input type="checkbox"/>	yo	Mon Nov 27 15:32:35 CST 2017 - Fatal - [{"Nivel":"Fatal","descripcionRegistro":"com.mysql.jdbc.exceptions.jdbc"}]	15:32
<input type="checkbox"/>	<input type="checkbox"/>	yo	Mon Nov 27 14:40:21 CST 2017 - Fatal - [{"Nivel":"Fatal","descripcionRegistro":"com.mysql.jdbc.exceptions.jdbc"}]	14:40
<input type="checkbox"/>	<input type="checkbox"/>	yo	Mon Nov 27 14:32:07 CST 2017 - Fatal - [{"Nivel":"Fatal","descripcionRegistro":"com.mysql.jdbc.exceptions.jdbc4"}]	14:32
<input type="checkbox"/>	<input type="checkbox"/>	Lore de Google	Buzz, sácale más partido a tu nueva cuenta de Google - Hola, Buzz: Me alegro de que hayas decidido probar	14:30

0 GB (0%) ocupados de 15 GB

Administrar

Condiciones - Privacidad

Última actividad de la cuenta: hace 1 minuto

Información detallada