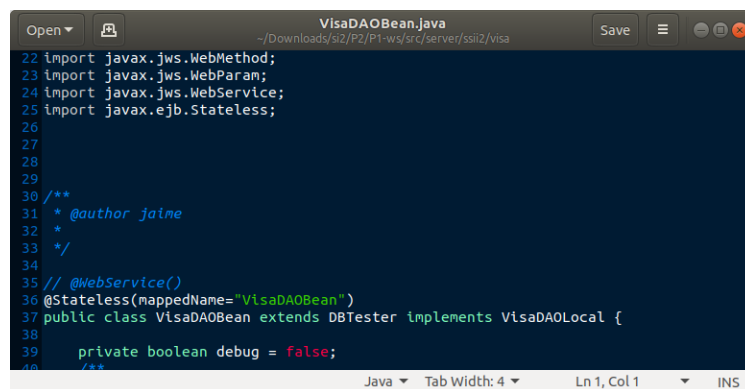
		Escuela Politécnica Superior Ingeniería Informática Prácticas de Sistemas Informáticos 2			
Grupo	T2363	Práctica	1B	Fecha	13/03/19
Alumno/a		Garcia, Fernandez, Roman			

Práctica 1B: Arquitectura de JAVA EE (Segunda parte)

Ejercicio 1

1. En el archivo VisaDAOBeans.java
 - a. Incorporar imports
 - i. Import javax.ejb.Stateless;
 - b. Vaciar constructor
 - c. Ajustar método getPagos() a la interfaz definida por VisaDAOLocal
 - i. Cambiar la cabecera de la función
 - ii. Cambiar el retorno para que sea acorde a la cabecera

```
return new PagoBean[pagos.size()].toArray();
```



```

22 import javax.jws.WebMethod;
23 import javax.jws.WebParam;
24 import javax.jws.WebService;
25 import javax.ejb.Stateless;
26
27
28
29
30 /**
31  * @author jaine
32  *
33  */
34
35 // @WebService()
36 @Stateless(mappedName="VisaDAOBean")
37 public class VisaDAOBean extends DBTester implements VisaDAOLocal {
38
39     private boolean debug = false;
40     /**

```

Ejercicio 2

1. En el archivo ProcesaPago.java
 - a. Incorporar
 - i. import javax.ejb.EJB;
 - ii. import ssii2.visa.VisaDAOLocal;
 - iii. Atributo privado dao


```
@EJB(name="VisaDAOBean", beanInterface=VisaDAOLocal.class)
private VisaDAOLocal dao;
```
 - b. Borrar
 - i. Imports de VisaDAOWS y VisaDAOWSService
 - ii. Creación de la instancia de VisaDAOWS
 - iii. Código de obtención de la referencia remota
 - iv. Referencias a BindingProvider
2. Hacer lo mismo para el resto de servlets: DelPagos y GetPagos (los errores de compilación ayudan a encontrarlos)

[illegible]

The image shows a code editor with three files open: **MainModule**, **PageController**, and **PageView**.

MainModule (Left):

```

1  // se encarga de cargar a pagecontroller.js. En caso de
2  // error corresponde al
3  //
4  //
5  //
6  //
7  //
8  //
9  //
10 package mainModule;
11
12 import java.io.IOException;
13 import java.io.PrintWriter;
14 import java.net.NetworkInterface;
15 import java.net.SocketException;
16 import java.net.UnknownHostException;
17 import java.util.ArrayList;
18 import java.util.HashMap;
19 import java.util.Iterator;
20 import java.util.List;
21 import java.util.Map;
22 import java.util.Set;
23 import java.util.concurrent.ConcurrentHashMap;
24
25 public class MainModule {
26     //
27     //
28     //
29     //
30     //
31     //
32     //
33     //
34     //
35     //
36     //
37     //
38     //
39     //
40     //
41     //
42     //
43     //
44     //
45     //
46     //
47     //
48     //
49     //
50     //
51     //
52     //
53     //
54     //
55     //
56     //
57     //
58     //
59     //
60     //
61     //
62     //
63     //
64     //
65     //
66     //
67     //
68     //
69     //
70     //
71     //
72     //
73     //
74     //
75     //
76     //
77     //
78     //
79     //
80     //
81     //
82     //
83     //
84     //
85     //
86     //
87     //
88     //
89     //
90     //
91     //
92     //
93     //
94     //
95     //
96     //
97     //
98     //
99     //
100    //
101    //
102    //
103    //
104    //
105    //
106    //
107    //
108    //
109    //
110    //
111    //
112    //
113    //
114    //
115    //
116    //
117    //
118    //
119    //
120    //
121    //
122    //
123    //
124    //
125    //
126    //
127    //
128    //
129    //
130    //
131    //
132    //
133    //
134    //
135    //
136    //
137    //
138    //
139    //
140    //
141    //
142    //
143    //
144    //
145    //
146    //
147    //
148    //
149    //
150    //
151    //
152    //
153    //
154    //
155    //
156    //
157    //
158    //
159    //
160    //
161    //
162    //
163    //
164    //
165    //
166    //
167    //
168    //
169    //
170    //
171    //
172    //
173    //
174    //
175    //
176    //
177    //
178    //
179    //
180    //
181    //
182    //
183    //
184    //
185    //
186    //
187    //
188    //
189    //
190    //
191    //
192    //
193    //
194    //
195    //
196    //
197    //
198    //
199    //
200    //
201    //
202    //
203    //
204    //
205    //
206    //
207    //
208    //
209    //
210    //
211    //
212    //
213    //
214    //
215    //
216    //
217    //
218    //
219    //
220    //
221    //
222    //
223    //
224    //
225    //
226    //
227    //
228    //
229    //
230    //
231    //
232    //
233    //
234    //
235    //
236    //
237    //
238    //
239    //
240    //
241    //
242    //
243    //
244    //
245    //
246    //
247    //
248    //
249    //
250    //
251    //
252    //
253    //
254    //
255    //
256    //
257    //
258    //
259    //
260    //
261    //
262    //
263    //
264    //
265    //
266    //
267    //
268    //
269    //
270    //
271    //
272    //
273    //
274    //
275    //
276    //
277    //
278    //
279    //
280    //
281    //
282    //
283    //
284    //
285    //
286    //
287    //
288    //
289    //
290    //
291    //
292    //
293    //
294    //
295    //
296    //
297    //
298    //
299    //
300    //
301    //
302    //
303    //
304    //
305    //
306    //
307    //
308    //
309    //
310    //
311    //
312    //
313    //
314    //
315    //
316    //
317    //
318    //
319    //
320    //
321    //
322    //
323    //
324    //
325    //
326    //
327    //
328    //
329    //
330    //
331    //
332    //
333    //
334    //
335    //
336    //
337    //
338    //
339    //
340    //
341    //
342    //
343    //
344    //
345    //
346    //
347    //
348    //
349    //
350    //
351    //
352    //
353    //
354    //
355    //
356    //
357    //
358    //
359    //
360    //
361    //
362    //
363    //
364    //
365    //
366    //
367    //
368    //
369    //
370    //
371    //
372    //
373    //
374    //
375    //
376    //
377    //
378    //
379    //
380    //
381    //
382    //
383    //
384    //
385    //
386    //
387    //
388    //
389    //
390    //
391    //
392    //
393    //
394    //
395    //
396    //
397    //
398    //
399    //
400    //
401    //
402    //
403    //
404    //
405    //
406    //
407    //
408    //
409    //
410    //
411    //
412    //
413    //
414    //
415    //
416    //
417    //
418    //
419    //
420    //
421    //
422    //
423    //
424    //
425    //
426    //
427    //
428    //
429    //
430    //
431    //
432    //
433    //
434    //
435    //
436    //
437    //
438    //
439    //
440    //
441    //
442    //
443    //
444    //
445    //
446    //
447    //
448    //
449    //
450    //
451    //
452    //
453    //
454    //
455    //
456    //
457    //
458    //
459    //
460    //
461    //
462    //
463    //
464    //
465    //
466    //
467    //
468    //
469    //
470    //
471    //
472    //
473    //
474    //
475    //
476    //
477    //
478    //
479    //
480    //
481    //
482    //
483    //
484    //
485    //
486    //
487    //
488    //
489    //
490    //
491    //
492    //
493    //
494    //
495    //
496    //
497    //
498    //
499    //
500    //
501    //
502    //
503    //
504    //
505    //
506    //
507    //
508    //
509    //
510    //
511    //
512    //
513    //
514    //
515    //
516    //
517    //
518    //
519    //
520    //
521    //
522    //
523    //
524    //
525    //
526    //
527    //
528    //
529    //
530    //
531    //
532    //
533    //
534    //
535    //
536    //
537    //
538    //
539    //
540    //
541    //
542    //
543    //
544    //
545    //
546    //
547    //
548    //
549    //
550    //
551    //
552    //
553    //
554    //
555    //
556    //
557    //
558    //
559    //
560    //
561    //
562    //
563    //
564    //
565    //
566    //
567    //
568    //
569    //
570    //
571    //
572    //
573    //
574    //
575    //
576    //
577    //
578    //
579    //
580    //
581    //
582    //
583    //
584    //
585    //
586    //
587    //
588    //
589    //
590    //
591    //
592    //
593    //
594    //
595    //
596    //
597    //
598    //
599    //
600    //
601    //
602    //
603    //
604    //
605    //
606    //
607    //
608    //
609    //
610    //
611    //
612    //
613    //
614    //
615    //
616    //
617    //
618    //
619    //
620    //
621    //
622    //
623    //
624    //
625    //
626    //
627    //
628    //
629    //
630    //
631    //
632    //
633    //
634    //
635    //
636    //
637    //
638    //
639    //
640    //
641    //
642    //
643    //
644    //
645    //
646    //
647    //
648    //
649    //
650    //
651    //
652    //
653    //
654    //
655    //
656    //
657    //
658    //
659    //
660    //
661    //
662    //
663    //
664    //
665   
```

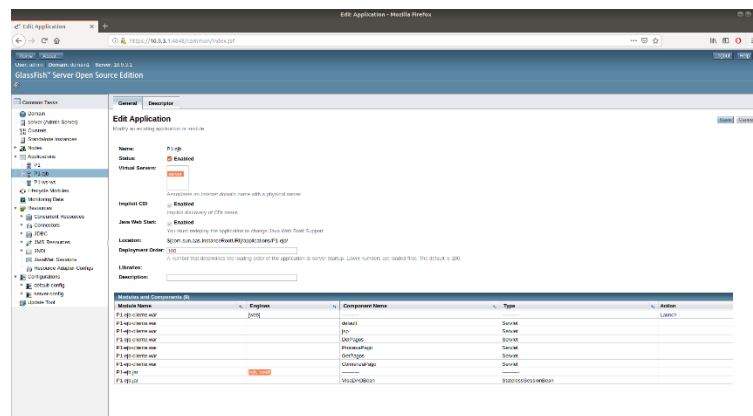
Ejercicio 3

Hay que cambiar los (*).properties para utilizar la nueva arquitectura. Para ello cambiar las ips del build.properties, que corresponde al servidor de la aplicación y al cliente de la aplicación, por la ip que corresponde a la vm1 (10.9.3.2)

En el postgres.properties hay que tener cuidado, ya que el cliente de la base de datos es quien va a consumir la base de datos y no el cliente que va a acceder a la aplicación (aunque en ese caso es el mismo). Por eso, en host ponemos la ip de la vm2 (10.9.3.1) y en cliente la de la vm1, que, como es también la del servidor de aplicaciones, es la que va a consumir la bbdd (10.9.3.2)

Ejercicio 4

Comprobación del despliegue del servicio



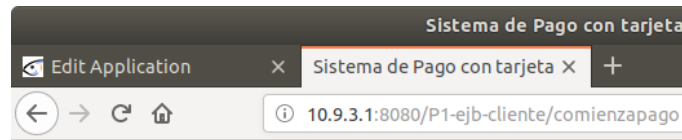
Creación de la transacción

Id Transacción:

Id Comercio:

Importe:

Transacción creada,
efectuando el pago
con tarjeta



Pago con tarjeta

Numero de visa:

Titular:

Fecha Emisión:

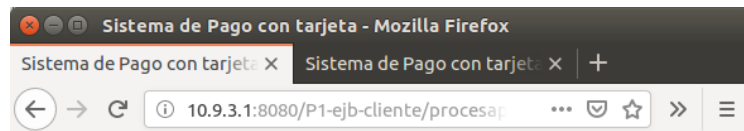
Fecha Caducidad:

CVV2:

Id Transacción: 1
Id Comercion: 1
Importe: 11.0

Prácticas de Sistemas Informáticos II

Comprobación del
pago correctamente
efectuado



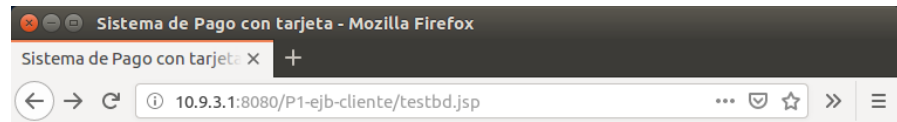
Pago con tarjeta

Pago realizado con éxito. A continuación se muestra el comprobante del mismo:

idTransaccion: 1
idComercio: 1
importe: 11.0
codRespuesta: 000
idAutorizacion: 1

[Volver al comercio](#)

Prácticas de Sistemas Informáticos II



Pago con tarjeta

Proceso de un pago

Id Transacción:

Id Comercio:

Importe:

Numero de visa:

Titular:

Fecha Emisión:

Fecha Caducidad:

CVV2:

Modo debug: ☐ True ☐ False

Direct Connection: ☐ True ☐ False

Use Prepared: ☐ True ☐ False

Consulta de pagos

Id Comercio:

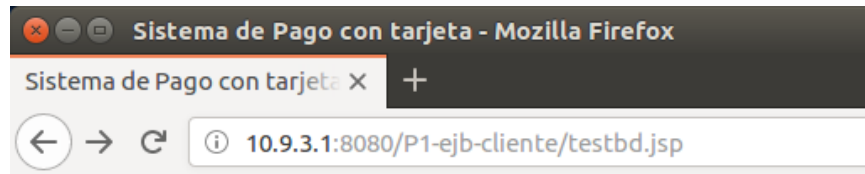
Borrado de pagos

Id Comercio:

Prácticas de Sistemas Informáticos II

Página de testbd.jsp.

Pago con tarjeta con testbd.jsp.

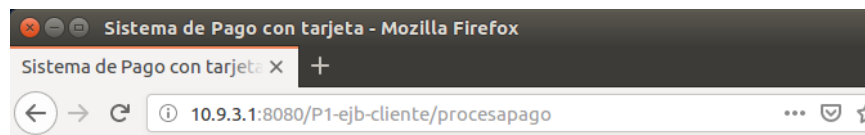


Pago con tarjeta

Proceso de un pago

Id Transacción:	<input type="text" value="2"/>
Id Comercio:	<input type="text" value="2"/>
Importe:	<input type="text" value="22"/>
Numero de visa:	<input type="text" value="1111 2222 3333 4444"/>
Titular:	<input type="text" value="Jose Garcia"/>
Fecha Emisión:	<input type="text" value="11/09"/>
Fecha Caducidad:	<input type="text" value="11/20"/>
CVV2:	<input type="text" value="123"/>
Modo debug:	<input type="radio"/> True <input type="radio"/> False
Direct Connection:	<input type="radio"/> True <input type="radio"/> False
Use Prepared:	<input type="radio"/> True <input type="radio"/> False
<input type="button" value="Pagar"/>	

Comprobación del pago correctamente efectuado.



Pago con tarjeta

Pago realizado con éxito. A continuación se muestra el comprobante del mismo:

idTransaccion: 2
idComercio: 2
importe: 22.0
codRespuesta: 000
idAutorizacion: 2

[Volver al comercio](#)

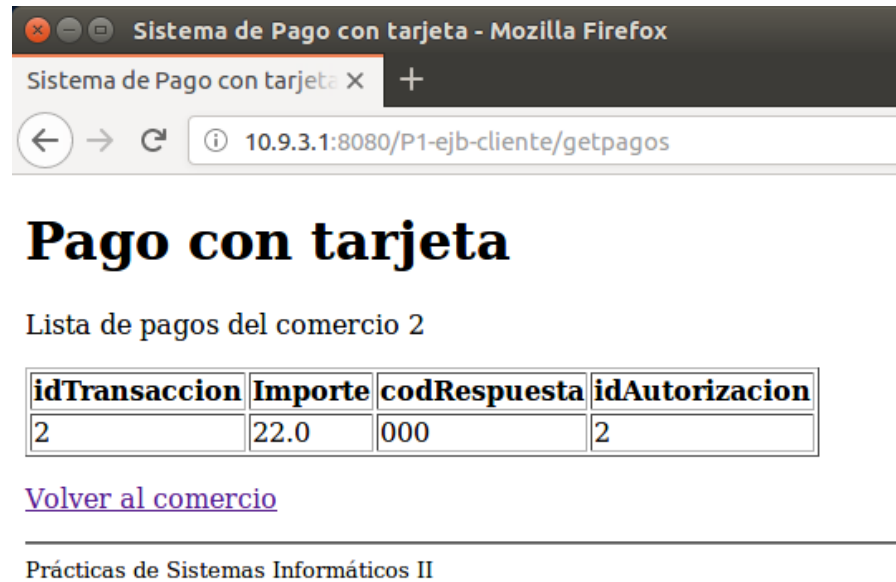
Prácticas de Sistemas Informáticos II

Consulta de pagos.

Consulta de pagos

Id Comercio:	<input type="text" value="2"/>
<input type="button" value="GetPagos"/>	

Resultado de la consulta.



Pago con tarjeta

Lista de pagos del comercio 2

idTransaccion	Importe	codRespuesta	idAutorizacion
2	22.0	000	2

[Volver al comercio](#)

Prácticas de Sistemas Informáticos II

Borrado de un pago (el creado con el testbd.jsp)

Borrado de pagos

Id Comercio:

Comprobación del borrado del pago.



Pago con tarjeta

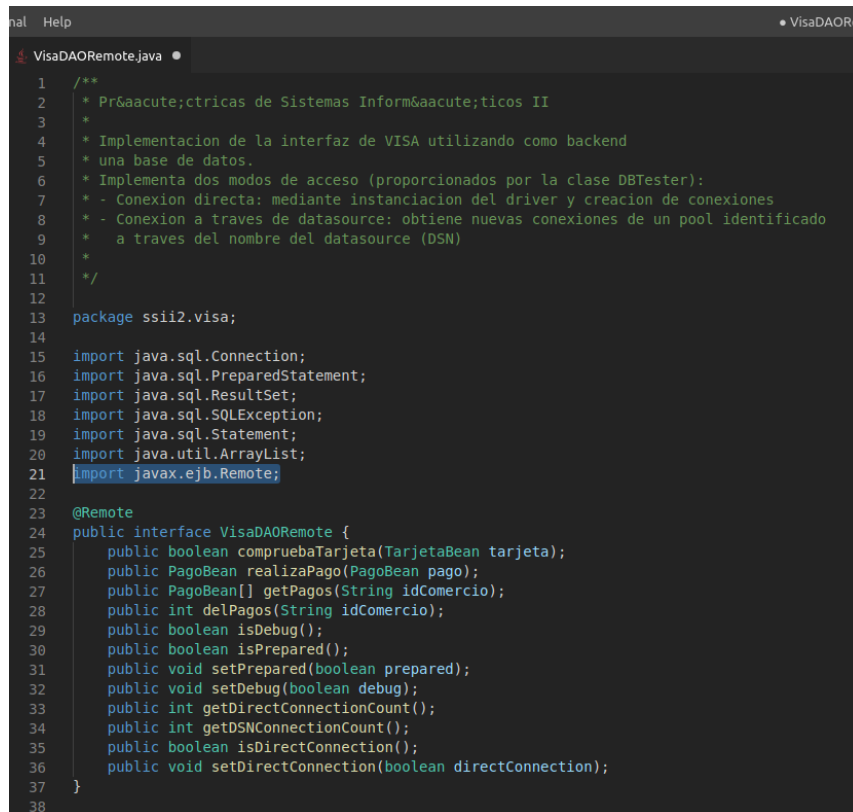
Se han borrado 1 pagos correctamente para el comercio 2

[Volver al comercio](#)

Prácticas de Sistemas Informáticos II

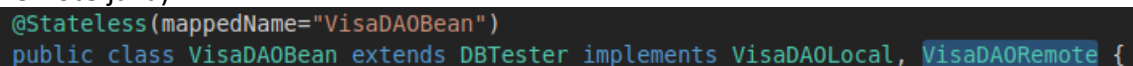
Ejercicio 5

1. Creamos el directorio copiando del P1-ejb: `cp -pr P1-ejb P1-ejb-servidor-remoto/`
2. Copiamos el archivo `VisaDAOLocal.java` a `VisaDAORemote.java`: `cp P1-ejb-servidor-remoto/src/server/ssii2/visa/VisaDAOLocal.java P1-ejb-servidor-remoto/src/server/ssii2/visa/VisaDAORemote.java`
 - a. Cambiar el nombre de la definición de la interfaz para que coincida con el nombre del archivo
 - b. Poner etiqueta `@Remote` donde estaba `@Local`
 - c. Cambiar el import de `javax.ejb.Local` por `javax.ejb.Remote`



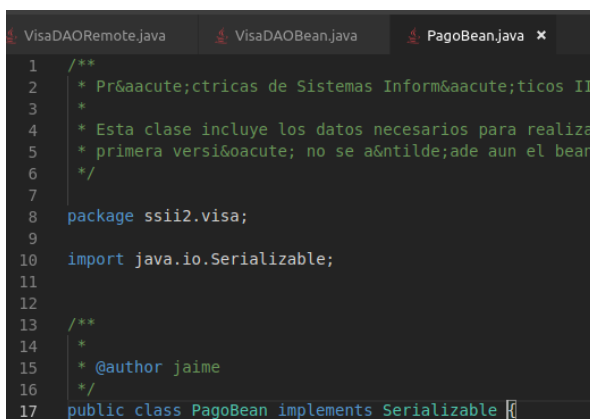
```
1  /**
2   * Prácticas de Sistemas Informáticos II
3   *
4   * Implementación de la interfaz de VISA utilizando como backend
5   * una base de datos.
6   * Implementa dos modos de acceso (proporcionados por la clase DBTester):
7   * - Conexión directa: mediante instanciación del driver y creación de conexiones
8   * - Conexión a través de datasource: obtiene nuevas conexiones de un pool identificado
9   *   a través del nombre del datasource (DSN)
10  */
11
12
13  package ssii2.visa;
14
15  import java.sql.Connection;
16  import java.sql.PreparedStatement;
17  import java.sql.ResultSet;
18  import java.sql.SQLException;
19  import java.sql.Statement;
20  import java.util.ArrayList;
21  import javax.ejb.Remote;
22
23  @Remote
24  public interface VisaDAORemote {
25      public boolean compruebaTarjeta(TarjetaBean tarjeta);
26      public PagoBean realizaPago(PagoBean pago);
27      public PagoBean[] getPagos(String idComercio);
28      public int delPagos(String idComercio);
29      public boolean isDebug();
30      public boolean isPrepared();
31      public void setPrepared(boolean prepared);
32      public void setDebug(boolean debug);
33      public int getDirectConnectionCount();
34      public int getDSNConnectionCount();
35      public boolean isDirectConnection();
36      public void setDirectConnection(boolean directConnection);
37  }
38
```

3. Hacer que `VisaDAOBean.java` implemente las dos interfaces (`VisaDAOLocal.java` y `VisaDAORemote.java`)

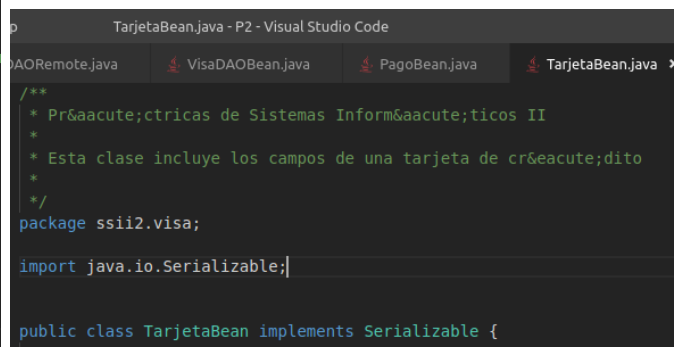


```
@Stateless(mappedName="VisaDAOBean")
public class VisaDAOBean extends DBTester implements VisaDAOLocal, VisaDAORemote {
```

4. Marcar como serializable `PagoBean` y `TarjetaBean`
 - a. Declarar el import: `java.io.Serializable`
 - b. Hacer que la clase implemente la interfaz `Serializable`
 - i. Agregar un atributo serial: `private static final long serialVersionUID = 1L;`



```
1  /**
2   * Prácticas de Sistemas Informáticos II
3   *
4   * Esta clase incluye los datos necesarios para realizar
5   * primera versión; no se añade aun el bean
6   */
7
8  package ssii2.visa;
9
10 import java.io.Serializable;
11
12
13 /**
14  *
15  * @author jaime
16  */
17 public class PagoBean implements Serializable {
```



```
1  /**
2   * Prácticas de Sistemas Informáticos II
3   *
4   * Esta clase incluye los campos de una tarjeta de crédito
5   */
6
7  package ssii2.visa;
8
9  import java.io.Serializable;
10
11
12 public class TarjetaBean implements Serializable {
```


Ejercicio 6

1. Copiar carpeta P1-base (de la práctica P1A) y ponerle el nombre: P1-ejb-cliente-remoto
2. Cambiar el nombre de la aplicación en build.properties: `nombre=P1-ejb-cliente-remoto`
3. Eliminar el directorio ssii2/visa/dao: `rm -rf P1-ejb-cliente-remoto/src/ssii2/visa/dao`
4. Marcar como serializable PagoBean y TarjetaBean (los archivos de cliente, que son diferentes a los de servidor)
 - a. Declarar el import: `java.io.Serializable`
 - b. Hacer que la clase implemente Serializable
 - i. Agregar un atributo serial: `private static final long serialVersionUID = 1L;`
5. Cambiar la línea 179 de ProcesaPago.java (aproximadamente por ahí)
 - a. De `if(! dao.compruebaTarjeta(...))` a `if (dao.compruebaTarjeta(...) == false)`
6. Copiar la interfaz VisaDAORemote.java del servidor en el cliente: `cp P1-ejb-servidor-remoto/src/server/ssii2/visa/VisaDAORemote.java P1-ejb-cliente-remoto/src/ssii2/visa/VisaDAORemote.java`
7. Cambiar la obtención del objeto dao en los servlets (procesapagos.java, getpagos.java y del-pagos.java)
 - a. Borrar (o comentar) la llamada al constructor
 - b. Crear una variable estática que llama al objeto EJB:
`@EJB(name = "VisaDAOBean", beanInterface = VisaDAORemote.class)`
`private VisaDAORemote dao;`
 - c. Incorporar imports y borrar (o comentar) el correspondiente a la implementación anterior:
`// import ssii2.visa.dao.VisaDAO;`
`import javax.ejb.EJB;`
`import ssii2.visa.VisaDAORemote;`
8. Crear el archivo glassfish-web.xml en el directorio web/WEB-INF: `touch P1-ejb-cliente-remoto/web/WEB-INF/glassfish-web.xml`
 - a. Insertar en él, con la ip correspondiente a la máquina del servidor de aplicación, lo siguiente (10.9.3.2 en mi caso):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE glassfish-web-app PUBLIC "-//GlassFish.org//DTD
GlassFish Application Server 3.1 Servlet 3.0//EN"
"http://glassfish.org/dtds/glassfish-web-app_3_0-1.dtd">
<glassfish-web-app>
<ejb-ref>
<ejb-ref-name>VisaDAOBean</ejb-ref-name>
<jndi-name>corbaname:iiop:10.9.3.2:3700#java:global/P1-ejb/P1-ejb/VisaDAOBean!ssii2.visa.Visa-
DAORemote</jndi-name>
</ejb-ref>
</glassfish-web-app>
```
9. Asegurarse de que las ips de *.properties de ambos estan correctamente configuradas:
 - a. Servidor remoto
 - i. build.properties
 1. `as.host.client=10.9.3.2`
 2. `as.host.server=10.9.3.2`
 - ii. postgresql.properties
 1. `db.client.host=10.9.3.2`
 2. `db.host=10.9.3.1`
 - b. Cliente remoto
 - i. build.properties

1. `as.host=10.9.3.1`
- ii. `postgresql.properties`
 1. `db.client.host=10.9.3.1`
 2. `db.host=10.9.3.1`

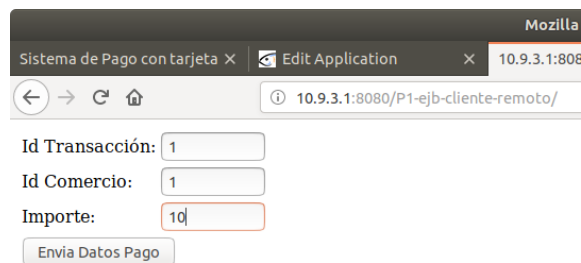
Por la configuración descrita vemos que si desplegamos ambos servicios se desplegarán dos clientes: uno en la ip (...)1 y otro en la ip (...)2

El primero por el cliente que hemos configurado al final y el segundo por la configuración de P1-ejb, la cual incluye un cliente además del servidor (como vemos en la configuración del `build.properties`, ambos se despliegan en la (...)2)

Para probar el cliente remoto, lo que nos interesa es conectarnos a la página desplegada en el (...)1

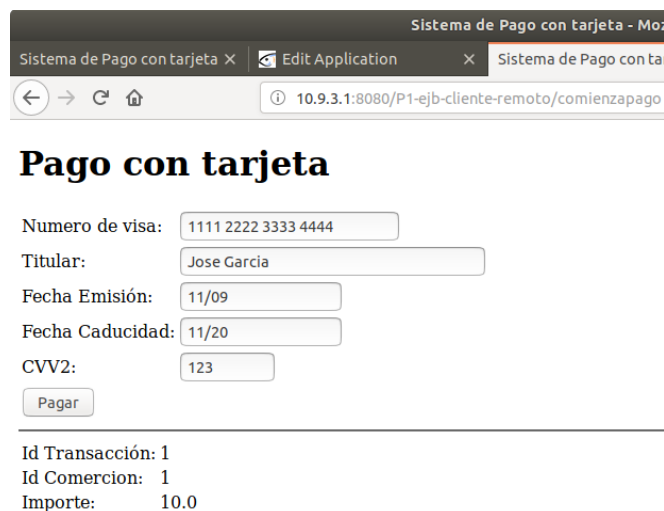
10. Compilar, empaquetar y desplegar

Crear la transacción



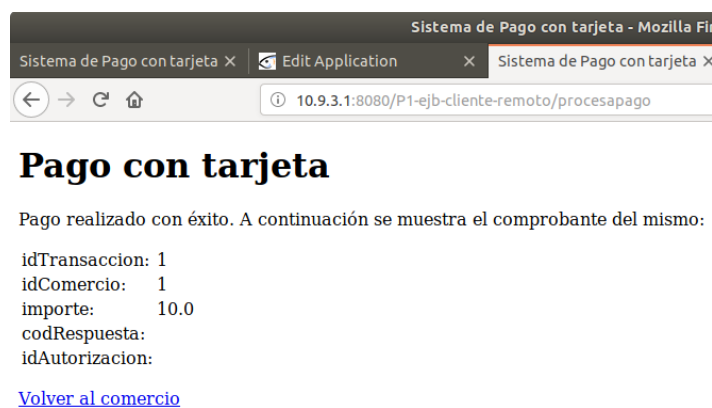
The screenshot shows a web browser window with the title 'Sistema de Pago con tarjeta'. The address bar shows '10.9.3.1:8080/P1-ejb-cliente-remoto/'. The form contains the following fields: 'Id Transacción:' with value '1', 'Id Comercio:' with value '1', and 'Importe:' with value '10'. There is a button labeled 'Envía Datos Pago'.

Crear el pago con tarjeta



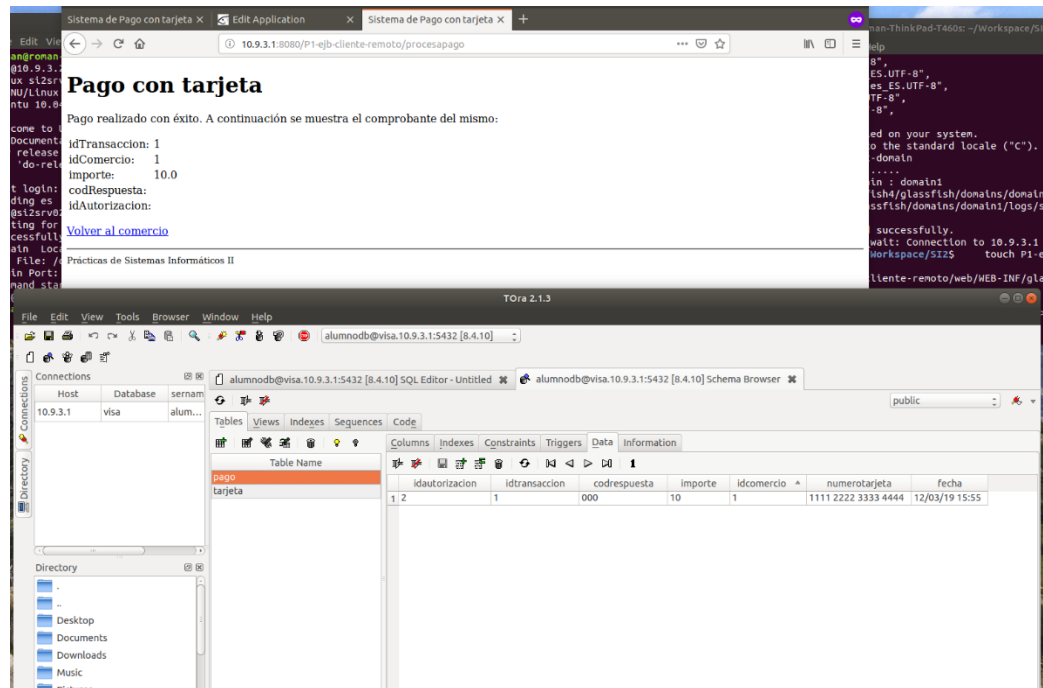
The screenshot shows a web browser window with the title 'Sistema de Pago con tarjeta - Mozilla Firefox'. The address bar shows '10.9.3.1:8080/P1-ejb-cliente-remoto/comienzapago'. The form is titled 'Pago con tarjeta' and contains the following fields: 'Numero de visa:' with value '1111 2222 3333 4444', 'Titular:' with value 'Jose Garcia', 'Fecha Emisión:' with value '11/09', 'Fecha Caducidad:' with value '11/20', and 'CVV2:' with value '123'. There is a button labeled 'Pagar'. Below the form, the following information is displayed: 'Id Transacción: 1', 'Id Comercion: 1', and 'Importe: 10.0'.

Comprobación del pago



The screenshot shows a web browser window with the title 'Sistema de Pago con tarjeta - Mozilla Firefox'. The address bar shows '10.9.3.1:8080/P1-ejb-cliente-remoto/procesapago'. The form is titled 'Pago con tarjeta' and contains the following text: 'Pago realizado con éxito. A continuación se muestra el comprobante del mismo:'. Below this, the following information is displayed: 'idTransaccion: 1', 'idComercio: 1', 'importe: 10.0', 'codRespuesta:', and 'idAutorizacion:'. There is a link labeled 'Volver al comercio'.

Comprobacion
por medio de
la base de
datos



Ejercicio 7

1. Creamos el directorio copiando desde P1-ejb y lo completamos con el contenido del archivo comprimido:
 - a. `cp -pr P1-ejb P1-ejb-transaccional/`
 - b. `cd P1-ejb-transaccional`
 - c. `tar -xzf ../P1-ejb-transaccional-base.tgz`
 - d. `ant limpiar-todo`
 - e.
2. En el archivo tarjetaBean.java
 - a. Agregar el atributo saldo y sus métodos de acceso

```
private double saldo;

/**
 * @return the saldo
 */
public double getSaldo() {
    return saldo;
}

/**
 * @param saldo the saldo to set
 */
public void setSaldo(double saldo) {
    this.saldo = saldo;
}
```

3. En VisaDAOBean.java
 - a. Agregar Imports
 - i. `import javax.ejb.EJBException;`
 - b. Agregar consultas predefinidas, de tipo String, que recuperen e inserten el saldo en la base de datos

```
private static final String SELECT_SALDO_TARJETA_QRY =
```

```
"select saldo from tarjeta " +
"where numeroTarjeta=? " +
" and titular=? " +
" and validaDesde=? " +
" and validaHasta=? " +
" and codigoVerificacion=? ";
```

```
private static final String UPDATE_SALDO_TARJETA_QRY =
"update tarjeta " +
"set saldo=? " +
"where numeroTarjeta=? " +
" and titular=? " +
" and validaDesde=? " +
" and validaHasta=? " +
" and codigoVerificacion=? ";
```

- c. Modificar la función realizaPago para que inserte y actualice los datos en la base de datos usando las sentencias anteriores

```
double saldo = 0;
```

```
String insert = SELECT_SALDO_TARJETA_QRY;
errorLog(insert);
pstmt = con.prepareStatement(insert);
pstmt.setString(1, pago.getTarjeta().getNumero());
pstmt.setString(2, pago.getTarjeta().getTitular());
pstmt.setString(3, pago.getTarjeta().getFechaEmision());
pstmt.setString(4, pago.getTarjeta().getFechaCaducidad());
pstmt.setString(5, pago.getTarjeta().getCodigoVerificacion());
```

```
rs = pstmt.executeQuery();
if(rs.next()) {
    saldo = rs.getDouble("saldo");
    if(pago.getImporte() > saldo){
        pago.setIdAutorizacion(null);
        pago = null;
        throw new EJBException("Saldo insuficiente. Pago denegado.");
    }
}
```

```
saldo -= pago.getImporte();
```

```
insert = UPDATE_SALDO_TARJETA_QRY;
errorLog(insert);
pstmt = con.prepareStatement(insert);
pstmt.setDouble(1, saldo);
pstmt.setString(2, pago.getTarjeta().getNumero());
pstmt.setString(3, pago.getTarjeta().getTitular());
pstmt.setString(4, pago.getTarjeta().getFechaEmision());
pstmt.setString(5, pago.getTarjeta().getFechaCaducidad());
pstmt.setString(6, pago.getTarjeta().getCodigoVerificacion());
pstmt.executeUpdate();
```

4. En ProcesaPago.java

- a. Agregar imports

- i. `import javax.ejb.EJBException;`

- b. Modificamos el método procesaRequest para que capture la excepción lanzada por realizaPago de VisaDAOBean.java y, en caso de que exista la excepción, cierre la sesión y envíe a la página de error

```
// Almacenamos la tarjeta en el pago
pago.setTarjeta(tarjeta);

if (! dao.compruebaTarjeta(tarjeta)) {
    enviaError(new Exception("Tarjeta no autorizada:"), request, response);
    return;
}

try{
    if (dao.realizaPago(pago) == null) {
        enviaError(new Exception("Pago incorrecto"), request, response);
        return;
    }

    request.setAttribute(ComienzaPago.ATTR_PAGO, pago);
    if (sesion != null) sesion.invalidate();
    reenvia("/pagoexito.jsp", request, response);
    return;
} catch (EJBException e){
    enviaError(e, request, response);
    sesion.invalidate();
    return;
}
```

Ejercicio 8

Página de creación de pagos

Mozilla Firefox

FileEditViewHistoryBookmarksToolsHelp

10.9.3.2:8080/P1-ejb-cliente X +

←→↻🏠

10.9.3.2:8080/P1-ejb-cliente/

Id Transacción:

Id Comercio:

Importe:

Envía Datos Pago

Realizamos un pago con tarjeta

Sistema de Pago con tarjeta - Mozilla Firefox

Sistema de Pago con tarjeta X +

←→↻🏠

10.9.3.2:8080/P1-ejb-cliente/comenzapago

Pago con tarjeta

Numero de visa:

1111 2222 3333 4444

Titular:

Jose Garcia

Fecha Emisión:

11/09

Fecha Caducidad:

11/20

CVV2:

123

Pagar

Id Transacción: 1

Id Comercion: 1

Importe: 10.0

Prácticas de Sistemas Informáticos II

Sistema de Pago con tarjeta - Mozilla Firefox (Private Brow

FileEditViewHistoryBookmarksToolsHelp

Sistema de Pago con tarjeta X +

←→↻🏠

10.9.3.2:8080/P1-ejb-cliente/procesapago

Pago con tarjeta

Comprobación del pago

Pago realizado con éxito. A continuación se muestra el comprobante del mismo:

idTransaccion: 1

idComercio: 1

importe: 10.0

codRespuesta: 000

idAutorizacion: 1

[Volver al comercio](#)

Comprobación del saldo del cliente

Prácticas de Sistemas Informáticos II

SQL Editor - Untitled

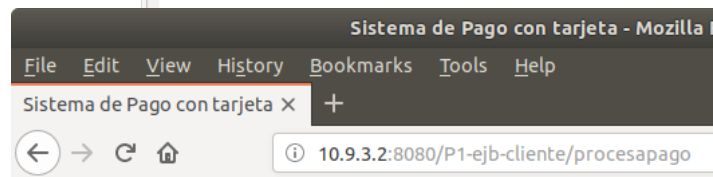
	numeroTarjeta	titular	validadesde	validahasta	codigoverificacion	saldo
1	1111 2222 3333 4444	José García	11/00	11/20	123	500
2	3288 3529 2313 3372	Luisa Pomares Sparrow	08/10	04/20	498	1000
3	2444 4974 2609 7643	Jack Garau Pomares	08/10	08/20	669	1000
4	5271 1595 0261 5762	Alberto Lopez Mas	04/08	05/20	528	1000

Comprobación del pago en la base de datos

SQL Editor - Untitled

	idautorizacion	idtransaccion	codrespuesta	importe	idcomercio	numeroTarjeta	fecha
1	1	1	000	10	1	1111 2222 3333 4444	11/03/19 11:38

Volvemos a pedir el mismo pago con el mismo id de transacción y comercio



Pago con tarjeta

Pago incorrecto

Prácticas de Sistemas Informáticos II

Comprobamos que el saldo no ha variado

SQL Editor - Untitled

	numeroTarjeta	titular	validadesde	validahasta	codigoverificacion	saldo
1	1111 2222 3333 4444	José García	11/00	11/20	123	500
2	9502 2454 9308 7508	Clodoveo Lobo Ribas	03/10	07/20	854	1000
3	9188 1495 2663 9025	Jack Dans Mas	08/08	04/20	129	1000
4	8788 0889 6993 2509	Gabriel Moreno Mojamuto	03/10	08/20	988	1000
5	8365 5667 6698 6481	Enlato Gonzalez Torres	03/09	11/20	421	1000

Comprobamos que no se ha realizado el pago

SQL Editor - Untitled

	idautorizacion	idtransaccion	codrespuesta	importe	idcomercio	numeroTarjeta	fecha
1	1	1	000	10	1	1111 2222 3333 4444	11/03/19 11:38

Ejercicio 9

New JMS Connection Factory - Mozilla Firefox

New JMS Connection Fac X

← → ↺ 🏠

🔒 https://10.9.3.2:4848/common/index.jsf

⋮ 🛡️ ☆

🔍 📄 🔴 ☰

Home About...

Logout Help

User: admin | Role: domain1 | Server: 10.9.3.2

GlassFish™ Server Open Source Edition

🔔 Total # of available updates : 1

Common Tasks

Domain

server (Admin Server)

Clusters

Standalone Instances

Nodes

Applications

P1-ejb

Lifecycle Modules

Monitoring Data

Resources

Concurrent Resources

Connectors

JDBC

JMS Resources

Connection Factories

Destination Resources

JNDI

JavaMail Sessions

Resource Adapter Configs

Configurations

default-config

server-config

Update Tool

New JMS Connection Factory

OK Cancel

The creation of a new Java Message Service (JMS) connection factory also creates a connector connection pool for the factory and a connector resource.

General Settings

JNDI Name: * jms/VisaConnectionFactory

Resource Type: javax.jms.QueueConnectionFactory

Description: Factoria de conexiones a la cola de pagos

Status: ☒ Enabled

Pool Settings

Initial and Minimum Pool Size: 8 Connections
Minimum and initial number of connections maintained in the pool

Maximum Pool Size: 32 Connections
Maximum number of connections that can be created to satisfy client requests

Pool Resize Quantity: 2 Connections
Number of connections to be removed when pool idle timeout expires

Idle Timeout: 300 Seconds
Maximum time that connection can remain idle in the pool

Max Wait Time: 60000 Milliseconds
Amount of time caller waits before connection timeout is sent

On Any Failure: ☐ Close All Connections
Close all connections and reconnect on failure, otherwise reconnect only when used

Transaction Support:

⌵

Level of transaction support. Overwrite the transaction support attribute in the Resource Adapter in a downward compatible way.

Connection Validation: ☐ Required
Validate connections, allow server to reconnect in case of failure

Additional Properties (0)

Add Property Delete Properties

Select	Name	Value	Description
No items found.			

Ejercicio 10

New JMS Destination Resource - Mozilla Firefox

New JMS Destination Resource X

← → ↻ 🏠

🔒 https://10.9.3.2:4848/common/index.jsf

⋮ 🛡️ ☆

☰ 📄 🔴

Home About...

User: admin | Role: domain1 | Server: 10.9.3.2

Logout Help

GlassFish™ Server Open Source Edition

Total # of available updates : 1

Tree

Common Tasks

Domain

- server (Admin Server)

Clusters

Standalone Instances

Nodes

- Applications
 - P1-ejb
- Lifecycle Modules
- Monitoring Data

Resources

- Concurrent Resources
- Connectors
- JDBC
- JMS Resources
 - Connection Factories
 - Destination Resources
- JNDI
 - Custom Resources
 - External Resources
- JavaMail Sessions
- Resource Adapter Configs

Configurations

- default-config
- server-config

Update Tool

New JMS Destination Resource

OK Cancel

The creation of a new Java Message Service (JMS) destination resource also creates an admin object resource.

JNDI Name: *

jms/VisaPagosQueue

Physical Destination Name *

VisaPagosQueue

Destination name in the Message Queue broker. If the destination does not exist, it will be created automatically when needed.

Resource Type: *

javax.jms.Queue

Description:

Cola de pagos VISA

Status:

☒ Enabled

Additional Properties (0)

Add Property Delete Properties

Select	Name	Value	Description
No items found.			

OK Cancel

Ejercicio 11

1. Descomprimir el P1-jms
2. Cambiar el archivo sun-ejb-jar.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sun-ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD Application Server 9.0
EJB 3.0//EN" "http://www.sun.com/software/appserver/dtds/sun-ejb-jar_3_0-0.dtd">
<sun-ejb-jar>
<enterprise-beans>
<ejb>
<ejb-name>VisaCancelacionJMSBean</ejb-name>
<mdb-connection-factory>
<jndi-name>jms/VisaConnectionFactory</jndi-name>
</mdb-connection-factory>
</ejb>
</enterprise-beans>
</sun-ejb-jar>
```

3. Implementar sentencias que permitan el uso de las colas en VisaCancelacionJMSBean:

```
private static final String UPDATE_CANCELA_QRY =
"update pago " +
"set codRespuesta=999 " +
"where idAutorizacion=? ";
```

```
private static final String UPDATE_RECTIFICAR_SALDO_QRY =
"update tarjeta " +
"set saldo=? " +
"where numeroTarjeta=? ";
```

```
private static final String SELECT_TARJETA_IDAUTORIZACION_QRY =
"select numeroTarjeta " +
" from pago " +
" where idAutorizacion = ?";
```

```
private static final String SELECT_SALDO_QRY =
"select saldo " +
" from tarjeta " +
" where numeroTarjeta = ?";
```

```
private static final String SELECT_IMPORTE_QRY =
"select importe " +
" from pago " +
" where idAutorizacion = ?";
```

4. Implementar la función onMessage para que
 - a. Cree una conexión o la obtenga del pool
 - b. Conseguimos el número de tarjeta y comprobamos que haya realizado un pago
 - c. Si no hay ningún problema, obtenemos el saldo de la cuenta y su el importe de la transacción
 - d. Actualizamos el saldo

```
public void onMessage(Message inMessage) {
    TextMessage msg = null;
```

```

PreparedStatement pstmt = null;
Connection con = null;
double saldo, importe;
ResultSet result = null;
int res;
String tarjeta;

try {
    con = getConnection();

    if (inMessage instanceof TextMessage) {

        msg = (TextMessage) inMessage;
        logger.info("MESSAGE BEAN: Message received: " + msg.getText());

        String insert = UPDATE_CANCELA_QRY;
        logger.info(insert);

        pstmt = con.prepareStatement(insert);
        pstmt.setInt(1, Integer.parseInt(msg.getText()));

        pstmt.execute();

        insert = SELECT_TARJETA_IDAUTORIZACION_QRY;
        logger.info(insert);
        pstmt = con.prepareStatement(insert);

        pstmt.setInt(1, Integer.parseInt(msg.getText()));
        result = pstmt.executeQuery();

        if(result.next()){
            tarjeta = result.getString("numeroTarjeta");
        } else{
            throw new JMSEException("Error, esta tarjeta no ha reaizado ningun pago");
        }

        insert = SELECT_SALDO_QRY;
        logger.info(insert);
        pstmt = con.prepareStatement(insert);

        pstmt.setString(1, tarjeta);
        result = pstmt.executeQuery();

        if(result.next()){
            saldo = result.getDouble("saldo");

```

```

    } else {
        throw new JMSEException("Error al capturar el saldo de la tarjeta");
    }

    insert = SELECT_IMPORTE_QRY;
    logger.info(insert);
    pstmt = con.prepareStatement(insert);

    pstmt.setInt(1, Integer.parseInt(msg.getText()));
    result = pstmt.executeQuery();

    if(result.next()){
        importe = result.getDouble("importe");
    } else {
        throw new JMSEException("Error al capturar el importe de la operacion cancelada");
    }

    saldo += importe;

    insert = UPDATE_RECTIFICAR_SALDO_QRY;
    logger.info(insert);

    pstmt = con.prepareStatement(insert);
    pstmt.setDouble(1, saldo);
    pstmt.setString(2, tarjeta);

    res = pstmt.executeUpdate();
    } else {
        logger.warning(
            "Message of wrong type: "
            + inMessage.getClass().getName());
    }
    } catch (JMSEException e) {
        e.printStackTrace();
        mdc.setRollbackOnly();
    } catch (Throwable te) {
        te.printStackTrace();
    }
}

```

Ejercicio 12

1. Agregar las etiquetas de recurso para que localice la cola de mensajes, en la clase VisaQueueMessageProducer

```

@Resource(mappedName = "jms/VisaConnectionFactory")
private static ConnectionFactory connectionFactory;

@Resource(mappedName = "jms/VisaPagosQueue")
private static Queue queue;

```

2. Implementar ambos métodos (estático y dinámico) y dejar comentado el dinámico

```
try {
// TODO: Inicializar connectionFactory
// y queue mediante JNDI
/*Metodo de conexion dinamico*/
/*InitialContext jndi = new InitialContext();
connectionFactory = (ConnectionFactory)jndi.lookup("jms/VisaConnectionFactory");
queue = (Queue)jndi.lookup("jms/VisaPagosQueue");*/

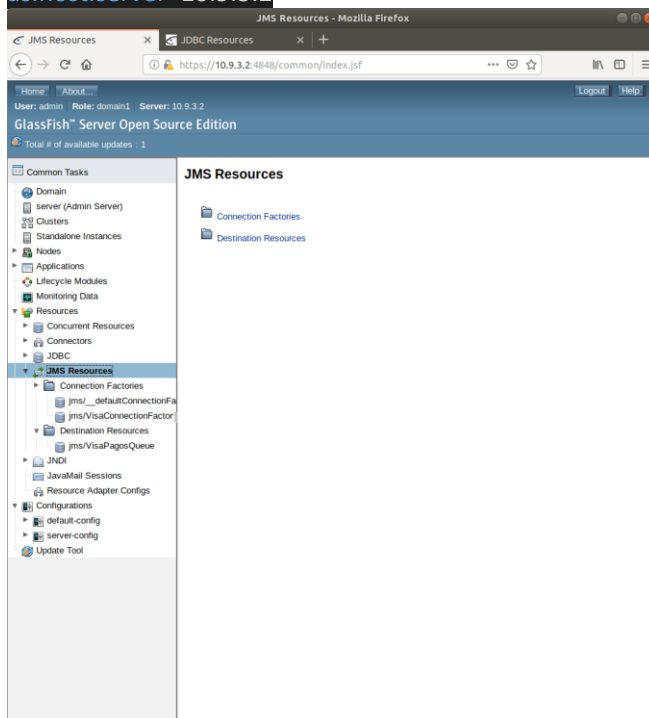
connection = connectionFactory.createConnection();
session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
```

3. El método estático tiene la ventaja de ser más rápido, pero tiene la desventaja de necesitar conocer el nombre del recurso en tiempo de compilación. El dinámico, sin embargo, lo resuelve en tiempo de ejecución, pero es lleva más procesamiento.

Ejercicio 13

1. Cambiamos los parámetros referentes a colas en `jms.properties` y estableciendo las ips correspondientes

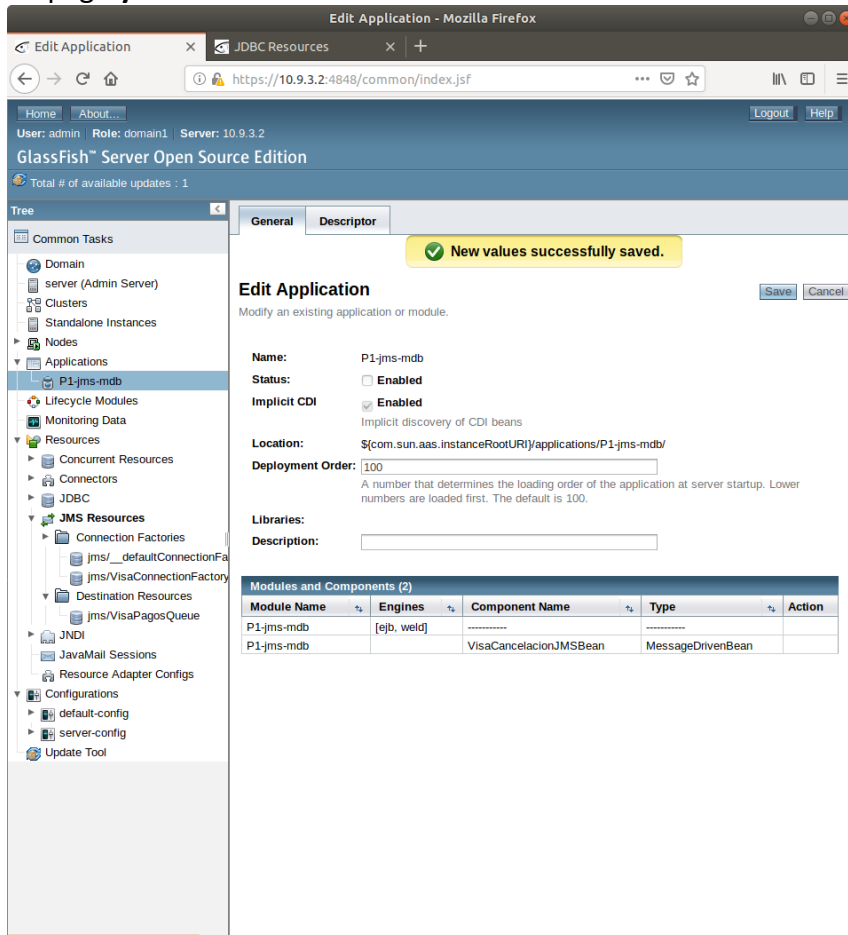
```
jms.factoryname=jms/VisaConnectionFactory
jms.name=jms/VisaPagosQueue
jms.physname=VisaPagosQueue
as.host.mdb=10.9.3.2
as.host.server=10.9.3.2
```



2. Se usan estas ips porque son las correspondientes a la maquina en la que se desplegarán los recursos de las colas.
3. En el fichero `jms.xml`, el comando equivalente para crear una cola JMS es: `target: create-jms-resource`.

Ejercicio 14

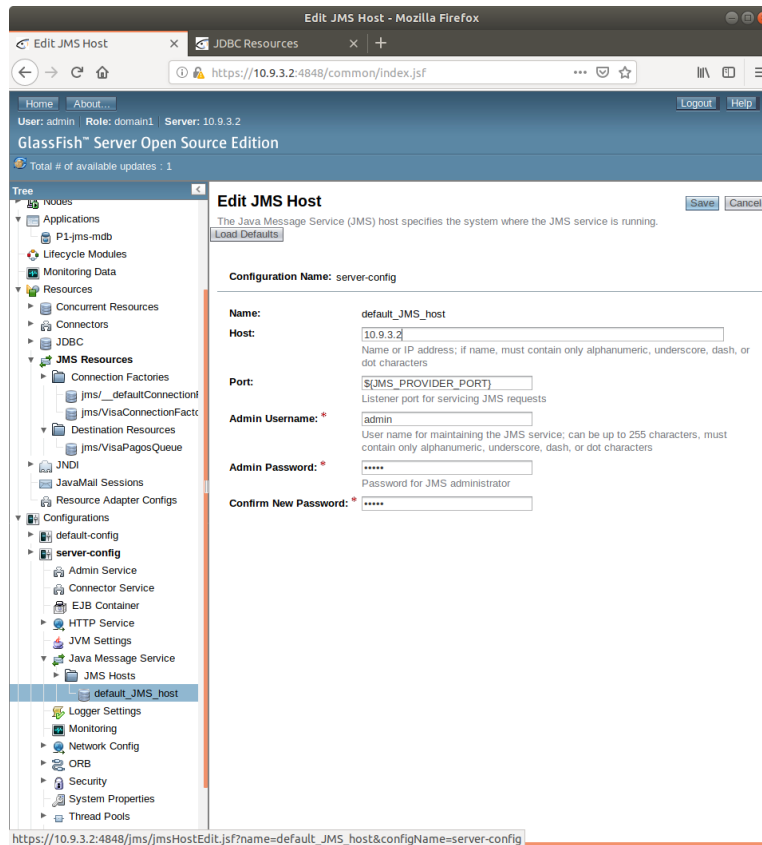
1. Detener la ejecución del MDB con la consola de glassfish como muestra la imagen: desmarcando el tick en enable **sin replegar el servicio**
 - a. Esto sirve para desactivar la cola de mensajes y, de esta forma, hacer que los mensajes se queden almacenados en la cola en vez de procesarlos
Después lo usaremos para comprobar su funcionamiento a través de la cancelación de un pago **ya realizado**



2. Modificar el archivo VisaQueueMessageProducer.java para que envíe un argumento pasado por parámetro a la llamada de la función como mensaje de texto

```
messageProducer = session.createProducer(queue);  
message = session.createTextMessage();  
message.setText(args[0]);  
messageProducer.send(message);  
messageProducer.close();  
session.close();
```

3. Cambiar el campo host (que contiene hostname) por la ip en la que corre el MDB (10.9.3.2)
 - a. Console web -> Configurations -> server-conf -> Java Message Service -> JMS Hosts -> default_JMS_host



4. Vamos a comprobar que la cola de mensajes funciona correctamente
 - a. Ejecutamos el cliente proporcionado en la ruta P1-jms/dist/clientjms/P1-jms-clientjms.jar, desde la ruta de aplicación de cliente de glassfish, indicando la ip de la maquina en que se ejecuta jms (10.9.3.2):
 - i. `/opt/glassfish4/glassfish/bin/appclient -targetserver 10.9.3.2 -client Workspace/SI2/P2/P1-jms/dist/clientjms/P1-jms-clientjms.jar idAutorizacion`
 - ii. El idAutorizacion corresponde al atributo idAutorizacion de la tabla pago de la base de datos
 - b. Ejecutamos el mismo comando, pero cambiando el atributo para ver los mensajes de la cola de mensajes
 - i. `/opt/glassfish4/glassfish/bin/appclient -targetserver 10.9.3.2 -client Workspace/SI2/P2/P1-jms/dist/clientjms/P1-jms-clientjms.jar -browse`
 - ii. Este comando hara que se ejecute el codigo que modificamos anteriormente, de esta forma veremos listado el contenido de la cola por pantalla. En este caso, deberia aparecer idAutorizacion como mensaje dentro de la cola de mensajes

En la siguiente foto podemos ver la secuencia de comandos y su resultado:

1. Comprobar la cola
2. Añadir un elemento
3. Comprobar la cola de nuevo

```
roman@roman-ThinkPad-T460s: ~  
File Edit View Search Terminal Help  
roman@roman-ThinkPad-T460s:~$ /opt/glassfish4/glassfish/bin/appclient -targetserver 10.9.3.2 -client Workspace/SI2/P2/P1-jms/dist/clientjms/P1-jms-clientjms.jar -browse  
Mar 12, 2019 9:09:17 PM org.hibernate.validator.internal.util.Version <clinit>  
INFO: HV000001: Hibernate Validator 5.1.2.Final  
Mar 12, 2019 9:09:18 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version: 5.1.1 (Build 2-c) Compile: March 17 2015 1045  
Mar 12, 2019 9:09:18 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is REMOTE, connection mode is TCP  
Mar 12, 2019 9:09:18 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:REMOTE  
Cola de mensajes vacía!  
roman@roman-ThinkPad-T460s:~$ /opt/glassfish4/glassfish/bin/appclient -targetserver 10.9.3.2 -client Workspace/SI2/P2/P1-jms/dist/clientjms/P1-jms-clientjms.jar idAutorizacion  
Mar 12, 2019 9:10:30 PM org.hibernate.validator.internal.util.Version <clinit>  
INFO: HV000001: Hibernate Validator 5.1.2.Final  
Mar 12, 2019 9:10:31 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version: 5.1.1 (Build 2-c) Compile: March 17 2015 1045  
Mar 12, 2019 9:10:31 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is REMOTE, connection mode is TCP  
Mar 12, 2019 9:10:31 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:REMOTE  
roman@roman-ThinkPad-T460s:~$ /opt/glassfish4/glassfish/bin/appclient -targetserver 10.9.3.2 -client Workspace/SI2/P2/P1-jms/dist/clientjms/P1-jms-clientjms.jar -browse  
Mar 12, 2019 9:10:55 PM org.hibernate.validator.internal.util.Version <clinit>  
INFO: HV000001: Hibernate Validator 5.1.2.Final  
Mar 12, 2019 9:10:55 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version: 5.1.1 (Build 2-c) Compile: March 17 2015 1045  
Mar 12, 2019 9:10:55 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is REMOTE, connection mode is TCP  
Mar 12, 2019 9:10:55 PM com.sun.messaging.jms.ra.ResourceAdapter start  
INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:REMOTE  
Mensajes en cola:  
idAutorizacion  
roman@roman-ThinkPad-T460s:~$
```

5. Una vez comprobada procedemos a comprobar la cancelación de pagos mediante la cola de mensajes

Para ello:

- a. Desplegamos P1-ejb y realizamos un pago correcto
- b. Ejecutamos el comando anterior sustituyendo idAutorizacion por el idAutorizacion correspondiente al pago efectuado
- c. Comprobamos que se encuentra en la cola de mensajes con el comando anterior y el parametro –browse
- d. Vamos a la consola de glassfish y activamos el MDB (habilitando el check de status que desactivamos anteriormente)
- e. Comprobamos de nuevo la cola de mensajes
 - i. Esta vez debe estar vacía
- f. Comprobamos en Tora que el pago se ha cancelado (el atributo codrespuesta tiene valor 999)
- g. Comprobamos que el saldo no se ha modificado

Tora 2.1.3

File Edit View Tools Browser Window Help

alumnodb@visa.10.9.3.1:5432 [8.4.10]

Connections

Host	Database	sernam
10.9.3.1	visa	alum...

Directory

- ..
- Desktop
- Documents
- Downloads
- Music
- Pictures

alumnodb@visa.10.9.3.1:5432 [8.4.10] SQL Editor - Untitled

alumnodb@visa.10.9.3.1:5432 [8.4.10] Schema Browser

public

Tables Views Indexes Sequences Code

Columns Indexes Constraints Triggers Data Information

Table Name

pagotarjeta

	numerotarjeta	titular	validadesde	validahasta	codigoverificacion	saldo
991	2403 0077 2538 1568	Gonzalo Vallejo Coll	01/10	07/20	496	1000
992	5343 1218 6446 2069	Kate Ribera Cozar	06/08	02/20	619	1000
993	7667 2372 6507 7928	Armando Coll Ribas	10/08	08/20	020	1000
994	7442 5242 7104 2062	Camilo Narvaez Dans	09/08	05/20	578	1000
995	5988 9974 0089 0633	Hugo Coll Gracia	03/10	06/20	763	1000
996	1655 3223 3160 7910	Eva Vallejo Locke	06/09	02/20	910	1000
997	6593 8663 5794 4246	Luisa Cozar Locke	01/10	08/20	014	1000
998	5474 3403 2899 0556	John Locke Vallejo	03/08	04/20	895	1000
999	8792 0855 2501 4428	Jose Locke Avila	06/09	11/20	168	1000
1000	7365 7998 7497 2704	Bias Punset Ribera	09/10	09/20	726	1000
1001	1111 2222 3333 4444	Jose Garcia	11/09	11/20	123	889

Row: 1 Col: 1

Tora 2.1.3

File Edit View Tools Browser Window Help

alumnodb@visa.10.9.3.1:5432 [8.4.10]

Connections

Host	Database	sernam
10.9.3.1	visa	alum...

Directory

- ..
- Desktop
- Documents
- Downloads

alumnodb@visa.10.9.3.1:5432 [8.4.10] SQL Editor - Untitled

alumnodb@visa.10.9.3.1:5432 [8.4.10] Schema Browser

public

Tables Views Indexes Sequences Code

Columns Indexes Constraints Triggers Data Information

Table Name

pagotarjeta

	idautorizacion	idtransaccion	codrespuesta	importe	idcomercio	numerotarjeta	fecha
1 2	2	000	111	1	1111 2222 3333 4444	12/03/19 13:10	
2 1	1	999	11	1	1111 2222 3333 4444	12/03/19 13:05	

Row: 1 Col: 1

Ln 15, Col 30 Spaces: 4 UTF-8 LF Java A modo de resumen, para esta práctica se espera encontrar dentro del archivo SI2P1B <grupo> pareja>.zip

se

Mar 12, 2019 9:10:55 PM org.hibernate.validator.internal.util.Version <clinit>
 INFO: HV000001: Hibernate Validator 5.1.2.Final

Mar 12, 2019 9:10:55 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version: 5.1.1 (Build 2-c) Compile: March 17 2015 1045

Mar 12, 2019 9:10:55 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is REMOTE, connection mode is TCP

Mar 12, 2019 9:10:55 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:REMOTE

Mensajes en cola:

idAutorizacion

roman@roman-ThinkPad-T460s:~\$ /opt/glassfish4/glassfish/bin/appclient -targetserver 10.9.3.2 -client Workspace/SI2/P2/P1-jms/dist/clientjms/P1-jms-clientjms.jar 1

Mar 12, 2019 9:13:50 PM org.hibernate.validator.internal.util.Version <clinit>

INFO: HV000001: Hibernate Validator 5.1.2.Final

Mar 12, 2019 9:13:50 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version: 5.1.1 (Build 2-c) Compile: March 17 2015 1045

Mar 12, 2019 9:13:50 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is REMOTE, connection mode is TCP

Mar 12, 2019 9:13:51 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:REMOTE

roman@roman-ThinkPad-T460s:~\$ /opt/glassfish4/glassfish/bin/appclient -targetserver 10.9.3.2 -client Workspace/SI2/P2/P1-jms/dist/clientjms/P1-jms-clientjms.jar -brow

se

Mar 12, 2019 9:14:13 PM org.hibernate.validator.internal.util.Version <clinit>

INFO: HV000001: Hibernate Validator 5.1.2.Final

Mar 12, 2019 9:14:14 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version: 5.1.1 (Build 2-c) Compile: March 17 2015 1045

Mar 12, 2019 9:14:14 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is REMOTE, connection mode is TCP

Mar 12, 2019 9:14:14 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:REMOTE

Mensajes en cola:

idAutorizacion

1

roman@roman-ThinkPad-T460s:~\$ /opt/glassfish4/glassfish/bin/appclient -targetserver 10.9.3.2 -client Workspace/SI2/P2/P1-jms/dist/clientjms/P1-jms-clientjms.jar -brow

se

Mar 12, 2019 9:15:16 PM org.hibernate.validator.internal.util.Version <clinit>

INFO: HV000001: Hibernate Validator 5.1.2.Final

Mar 12, 2019 9:15:17 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter: Version: 5.1.1 (Build 2-c) Compile: March 17 2015 1045

Mar 12, 2019 9:15:17 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter starting: broker is REMOTE, connection mode is TCP

Mar 12, 2019 9:15:17 PM com.sun.messaging.jms.ra.ResourceAdapter start

INFO: MQJMSRA_RA1101: GlassFish MQ JMS Resource Adapter Started:REMOTE

Cola de mensajes vacía!

roman@roman-ThinkPad-T460s:~\$

Activities

Tora

mar 21:19

Edit Application - Mozilla Firefox (Private Browsing)

Sistema de Pago con tarjeta x

Edit Application x

https://10.9.3.2:4848/common/index.jsf

Home About...

User: admin Role: domain1 Server: 10.9.3.2

GlassFish™ Server Open Source Edition

Total # of available updates : 1

Tree

Common Tasks

Domain

server (Admin Server)

Clusters

Standalone Instances

Nodes

Applications

P1-epb

P1-jms-mdb

Lifecycle Modules

Monitoring Data

Resources

Concurrent Resources

Connectors

JDBC

JMS Resources

JNDI

JavaMail Sessions

Resource Adapter Configs

Configurations

default-config

server-config

Admin Service

Connector Service

EJB Container

HTTP Service

JVM Settings

Java Message Service

JMS Hosts

default_JMS_host

Logger Settings

Monitoring

Network Config

ORB

Security

System Properties

General

Descriptor

New values successfully saved.

Edit Application

Modify an existing application or module.

Save

Cancel

Name: P1-jms-mdb

Status: ☒ Enabled

Implicit CDI ☒ Enabled

Implicit discovery of CDI beans

Location: \${com.sun.aas.instanceRootURI}/applications/P1-jms-mdb/

Deployment Order: 100

A number that determines the loading order of the application at server startup. Lower numbers are loaded first. The default is 100.

Libraries:

Description:

Modules and Components (2)

Module Name	Engines	Component Name	Type	Action
P1-jms-mdb	[ejb, weld]	-----	-----	
P1-jms-mdb		VisaCancelacionJMSBean	MessageDrivenBean	

27

Cuestión 1:

Abrir el archivo VisaDAOLocal.java y comprobar la definición de dicha interfaz. Anote en la memoria comentarios sobre las librerías Java EE importadas y las anotaciones utilizadas. ¿Para qué se utilizan?

En las librerías encontramos clases que tienen que ver con interacción con bases de datos por medio de sql. Además encontramos la librería ejb que contiene las clases e interfaces que definen la interacción entre enterprise bean y sus clientes y entre enterprise bean y el contenedor EJB

Cuestión 2:

Abrir el archivo application.xml y explicar su contenido. Verifique el contenido de todos los archivos .jar / .war / .ear que se han construido hasta el momento (empleando el comando jar -tvf). Anote sus comentarios y evidencias en la memoria.

En application.xml se puede ver que se han generado 2 módulos:

P1-ejb.jar, que hace referencia a la aplicación desde el punto de vista del servidor (el que va a tener las funcionalidades de gestión de pagos).

P1-ejb-cliente.war está contenido dentro de un tag denominado <web-uri>, lo cual nos da una idea de que estará relacionado con el identificador de recurso universal (es decir, la dirección a la que tendrá que conectarse el cliente para poder llevar a cabo los pagos). Esto es coherente con los URL de las figuras del ejercicio 4.