```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DatabaseMetaData;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.Timestamp;
import java.util.Calendar;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import org.apache.tomcat.util.buf.TimeStamp;

//////////////////////////////////////////////////
//                                              //
//                    ALQUILER                  //
//                                              //
//////////////////////////////////////////////////



public class conexionBD {
    private String url = Cadenas.BD_URL;
    private String usr = Cadenas.BD_USR;
    private String pwd = Cadenas.BD_PWD;
    private boolean hayConexion = false;
    private Connection db=null;
    private Statement st;
    private ResultSet rs;
    private String sentenciaSQL;


    //INICIALIZAMOS LOS OBJETOS para la conexiÃ³n con la Base de Datos
    private boolean iniBD(){
////++//  Cadenas.alert("Â¿conexion?="+hayConexion);
        if (!hayConexion) {
            try
            {
              Class.forName("org.postgresql.Driver");
              db = DriverManager.getConnection(url,usr,pwd);
              hayConexion = true; // QuitÃ¡ndolo hay mÃºltiple conexiÃ³n o quita
r el if/else
              /*
                      DatabaseMetaData SS= db.getMetaData();
                      ResultSet res = SS.getTables(null, null, null,new String[]
 {"TABLE"});
                      System.out.println("List of tables: ");
                      String mm="";
                      while (res.next()) {
                         mm+=(
                            "   "+res.getString("TABLE_CAT")
                          + ", "+res.getString("TABLE_SCHEM")
                          + ", "+res.getString("TABLE_NAME")
                          + ", "+res.getString("TABLE_TYPE")
                          + ", "+res.getString("REMARKS")+" \n");
                      }
```

```
                    res.close();
                    System.out.println(mm);
                    ////++//  Cadenas.alert("*** EN TEORIA HAY CONEXION \n\n"+
mm);
                 */

            }
            catch(Exception e)
            {
                GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
                // //++//  Cadenas.alert("**IniBD**-> "+e.toString());
                return false;
            }
        }
        return true;
    }

    // Funciones para la manipulaciÃ³n de las BD, el string de la consulta
    private ResultSet laConsulta(){
        if (iniBD()){
            try {
                st = db.createStatement();
                rs = st.executeQuery(sentenciaSQL);
                return rs;
            }
            catch(Exception e){
                // GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());

                //**//++//  Cadenas.alert("++  "+e.toString());
                return null;
            }
        } else {
            return null;
        }
    }

    private boolean laAsignacion(){
        if (iniBD()) {
            try {
                st = db.createStatement();
                int filasAfectadas = st.executeUpdate(sentenciaSQL);
                return (filasAfectadas>0);
            }
            catch(Exception e){
                // GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());

                return false;
            }
        } else {
            return false;
        }
    }

    public boolean UsuarioOK(String login,String password) {
        sentenciaSQL = "SELECT * FROM usuarios WHERE nombre='"+login+"' AND clav
e='"+password+"'";

        try {
            rs= laConsulta();
            return ( rs.next() );
```

```
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
            return false;
        }
    }

    public ResultSet dimePuestos() {
        sentenciaSQL = "SELECT * FROM puestos WHERE activo ORDER BY to_number(nu
m_puesto,'999999') ASC;";
        try {
            rs= laConsulta();
            //GestionLogs.escribirLog(null, "PRUEBA en dimePuestos " );
            return ( rs );
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
            return null;
        }

    }

    //dice el numero maximo de bicis en un puesto
    public int dimeNumMaxBicisEnPuesto(String nPuesto) {
        sentenciaSQL = "SELECT max_bicis FROM puestos WHERE activo AND num_puest
o='"+nPuesto+"';";
        try {
            rs= laConsulta();
            if (rs.next()){
                return (Integer.parseInt(rs.getString(1)) );
            }
            else
                return 0;
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
            return 0;
        }

    }

    //dice los puestos cercanos al dado
    public String dimePuestosCercanos(String nPuesto) {
        sentenciaSQL = "SELECT puestos_cercanos FROM puestos WHERE num_puesto='"
+nPuesto+"' AND activo";
        //GestionLogs.escribirLog(null, sentenciaSQL);
        try {
            rs= laConsulta();
            if (rs.next())
                return rs.getString(1).trim() ;
            else
                return null;
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
            return null;
        }
    }

    public boolean existePuesto(String puesto) {
```

```
        sentenciaSQL = "SELECT * FROM puestos WHERE num_puesto='"+puesto+"'";
        try {
            rs= laConsulta();
            return ( rs.next() );
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
            return false;
        }

    }

    public String queBiciEn(String puesto,String posteSelBC){
        sentenciaSQL = "SELECT * FROM bicis_puestos WHERE num_puesto='"+puesto+"
' AND lugar='"+posteSelBC+"'";
////++//  Cadenas.alert("queBiciEn:"+sentenciaSQL);
        try {
            rs= laConsulta();
            if ( rs.next() ){
                return rs.getString(2);
            }else
                return null;
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
            return null;
        }
    }


    private boolean retirarBiciPorAlquiler(String puesto,String numBiciBC){
        sentenciaSQL = "UPDATE bicis_puestos SET num_bici='-',fecha=null  WHERE
num_puesto='"+puesto+"' AND num_bici='"+numBiciBC+"'";
//++//  Cadenas.alert("alquilar:"+sentenciaSQL);
        return  laAsignacion();

    }

    public boolean  AlquilarBici(clsTicket ticket){
        // se alquila una bici
        boolean accion=retirarBiciPorAlquiler(ticket.getPuesto(),ticket.getnumBi
ci());
        //if (!accion) return false;
//++//  Cadenas.alert("alquilar-accion:"+accion);
        sentenciaSQL="INSERT INTO historico (usuario,bici,idt,puesto_coge,fecha_
coge)";
        sentenciaSQL+= " VALUES (";
        sentenciaSQL+="'"+ticket.getUsuario()+"',";
        sentenciaSQL+="'"+ticket.getnumBici()+"',";
        sentenciaSQL+="'"+ticket.getidTicket()+"',";
        sentenciaSQL+="'"+ticket.getPuesto()+"',";
        sentenciaSQL+="'"+ticket.getCuando()+"'";
        sentenciaSQL+= ")";
        //+*+//GestionLogs.escribirLog(null,"alquilar-historico:"+sentenciaSQL )
;
//++//  Cadenas.alert("alquilar-historico:"+sentenciaSQL);


        return laAsignacion();
    }
```

```
    private boolean devolverBiciPorAlquiler(String puesto,String numBici){
        //lo guarda en el primero que encuentre libre ( que lo hay porque si no
se llega aqui )

        sentenciaSQL = "SELECT lugar FROM bicis_puestos WHERE num_puesto='"+pues
to+"' AND num_bici='-' Order By to_number(lugar,'9999')";
        //+*+//Cadenas.alert("devolverBici obtenerLugar:"+sentenciaSQL);

//+*+//GestionLogs.escribirLog(null, "devolverBici obtenerLugar:"+sentenciaSQL);


        rs=laConsulta();

        try {
            if (rs.next()){
                String lugar=rs.getString(1);
                //+*+//Cadenas.alert("devolverBici Lugar= :"+lugar);

                sentenciaSQL = "UPDATE bicis_puestos SET " +
                            "num_bici='"+numBici+"'," +
                            "fecha='"+obtenerFechaSQL()+"'" +
                            "  WHERE num_puesto='"+puesto+"' AND num_bici='-' AN
D lugar='"+lugar+"'";
                //+*+//Cadenas.alert("devolverBici :"+sentenciaSQL);

                //+*+//GestionLogs.escribirLog(null, "devolverBici :"+sentenciaS
QL);
                return laAsignacion();
            }
        } catch (SQLException e) {
            //+*+//Cadenas.alert("Excepcion :"+e.toString());
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: devolverBiciPorAlquiler"
+ e.toString());
        }
        return false;
    }


    public boolean  devolverBici(clsTicket ticket,String puesto){
        String mPuesto=String.valueOf(Integer.parseInt(puesto));
        // se devuelve una bici
        boolean accion=devolverBiciPorAlquiler(mPuesto,ticket.getnumBici());
        //+*+//Cadenas.alert("devolver bici a bicis puesto:"+accion);
        //if (!accion) return false;
        sentenciaSQL="UPDATE historico SET ";
        sentenciaSQL+=" fecha_deja='"+obtenerFechaSQL()+"',";
        sentenciaSQL+=" puesto_deja='"+mPuesto+"'";
        sentenciaSQL+= " WHERE idT='"+ticket.getidTicket()+"'";
        //+*+//GestionLogs.escribirLog(null, sentenciaSQL)  ;
        return laAsignacion();

    }


    public boolean existePuestoEnBiciPuestos(String puesto) {
        sentenciaSQL = "SELECT * FROM bicis_puestos WHERE num_puesto='"+puesto+"
'";
        try {
            rs= laConsulta();
```

```
            return ( rs.next() );
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
            return false;
        }

    }

    public boolean existeBiciEnBiciPuestos(String puesto,String bici) {
        sentenciaSQL = "SELECT * FROM bicis_puestos WHERE num_puesto='"+puesto+"
' AND num_bici='"+bici+"'";
        try {
            rs= laConsulta();
            return ( rs.next() );
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
            return false;
        }

    }

    public int numeroBicisLibresEnPuesto(String puesto) {
        int nLibres=0;
        sentenciaSQL = "SELECT COUNT(num_puesto) FROM bicis_puestos WHERE num_pu
esto='"+puesto+"' AND num_bici='-'";
        try {
            rs= laConsulta();
            if ( rs.next() )
                nLibres=rs.getInt(1);
            GestionLogs.escribirLog(null, "bicis libres (numBicisPOrPuesto): " +
 nLibres);
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N (numBicisPOrPuesto): " + e
.toString());
        }
        return nLibres;
    }

    //devuelve el numero de bicis ocupadas en el puesto
    public int numeroBicisOcupadasEnPuesto(String puesto) {
        int nOcupadas=0;
        //sentenciaSQL = "SELECT * FROM bicis_puestos WHERE num_puesto='"+puesto
+"' AND num_bici<>'-' AND num_bici!=NULL";
        sentenciaSQL ="SELECT COUNT(num_puesto) FROM bicis_puestos WHERE num_pue
sto='"+puesto+"' AND num_bici<>'-'";
        try {
            rs= laConsulta();
            if ( rs.next() )
                nOcupadas=rs.getInt(1);
            GestionLogs.escribirLog(null, "bicis libres (numBicisPOrPuesto): " +
 nOcupadas);
        }
        catch(Exception e){
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N (numBicisPOrPuesto): " + e
.toString());
        }
        return nOcupadas ;
```

```
    }


    public boolean escribirLogBD(String idTransaccion,String quien,String accion
,Timestamp cuando,String donde,String numBici){
        sentenciaSQL = "INSERT INTO Historico " +
                "(usuario,accion,cuando,donde,nBici,idt) VALUES" +
                "('"+quien+"','"+accion+"','"+cuando+"','"+donde+"','"+numBici+"
'"+idTransaccion+"')";
        return laAsignacion();
    }

    public boolean escribirIncidenciaBD(String incidencia,String datos){
        sentenciaSQL = "INSERT INTO Incidencias " +
                "(incidencia,datos) VALUES" +
                "('"+incidencia+"','"+datos+"')";
//+**++//Cadenas.alert("escribir Incidencias:"+sentenciaSQL);
//+**++//GestionLogs.escribirLog(null, sentenciaSQL)    ;

        return laAsignacion();
    }
    public boolean escribirSolucionIncidenciaBD(String solucion){
        sentenciaSQL = "INSERT INTO Incidencias " +
                "(solucion,fecha_solucion) VALUES" +
                "('"+solucion+"','"+obtenerFechaSQL()+"',')";
        //se puede hacer directamente asi FECHA==> TIMESTAMP WITHOUT TIME ZONE '
now()'
        return laAsignacion();
    }


    //devuelve las bicis disponibles en un puesto
    public ResultSet dimeBicisDisponiblesPuesto(String puesto) {
        sentenciaSQL = "SELECT * FROM Bicis_Puestos WHERE num_puesto='"+puesto+"
' AND num_bici<>'-' ORDER BY to_number(lugar,'999')";
//GestionLogs.escribirLog(null, sentenciaSQL);
        return laConsulta();
    }

    public clsTicket crearTicketDesdeBD(String usuario,String numBici){
        //OJO SE MIRA SI LA BICI que esta en el ticket es del usuario y no esta
devuelta ( lo normal )
        //  no se comprueba que:
        //      - la bici este alquilada en otro usuario (alquilada o no )
        //      - si el usuario tiene varias bicis sin devolver (no se debe de h
acer aqui)
        sentenciaSQL = "SELECT * FROM Historico WHERE LOWER(usuario)=LOWER('"+us
uario+"') AND fecha_deja is null AND bici='"+numBici+"'";
        rs=laConsulta();

        try {
            if (rs.next()){
//++//  Cadenas.alert("crearTicketDesdeBD 0 :encontrado ticketBD");

                //se ha encontrado una entrada de alquiler sin dejar
                                    //new clsTicket(idSesion,            cuando
,             puesto,        numBici,        usuario)
                clsTicket nuevoTicket=new clsTicket(rs.getString(3),rs.getTimest
amp(4), rs.getString(6), rs.getString(2),rs.getString(1));
                return nuevoTicket;
```

```
            }// si no se encuentra se devuelve un null que se trata para crear u
na incidencia

        } catch (SQLException e) {
            // TODO Auto-generated catch block
//++//  Cadenas.alert("crearTicketDesdeBD 1 :excepcion");
            GestionLogs.escribirLog(null, "EXCEPCIÃ“N (crearTicketCon...): " + e
.toString());
        }
        return null;
    }


/*****************************
    @Deprecated
    public clsTicket sigTicket(String idEmpresa,String idSeccion) {
        int numSiguiente = 9998;
        Timestamp cuando ;
        Timestamp ahora = obtenerFechaSQL();
        boolean esPrimerNumero=false;
        clsTicket lTicket;

        empresaSeccionSQL=" WHERE Empresa='"+idEmpresa+"' AND Seccion='"+idSecci
on+"'";
//++//  Cadenas.alert(empresaSeccionSQL);
        //Buscamos el nÂº de la Empresa+Seccion
        sentenciaSQL = "SELECT siguiente,cuando FROM numero " +empresaSeccionSQL
;

        ResultSet rs = laConsulta();   // siguiente=getInt(1)
        try {
            esPrimerNumero=true;
            if ( rs.next() ) {
                // si ya existe hay que mirar cual es el nÂº ( el que hay es el
Siguiente NO EL ULTIMO!! )
//++//  Cadenas.alert("entro");
                numSiguiente = rs.getInt(1);
////++//  Cadenas.alert("leido numero = "+ String.valueOf(numSiguiente));
                cuando=rs.getTimestamp(2);
////++//  Cadenas.alert("leido fecha = " + cuando.toString());

                //comprobamos si el periodo de tiempo es el mismo ( en este caso
 yo escogi un Dia, pero puede ser MaÃ±ana/Tarde, etc ... )
                if (! esNuevoDia(cuando,ahora)){
                    //si NO es nuevo dia el numero sera=siguiente
                    sentenciaSQL = "UPDATE numero SET siguiente=(siguiente+1),cu
ando="+cuando+ empresaSeccionSQL;
                    esPrimerNumero=false;
                    ahora=cuando;
                }
                if (esPrimerNumero) {
////++//  Cadenas.alert("primerNUevo");
                    sentenciaSQL="UPDATE numero set siguiente=2,cuando=TIMESTAMP
 WITHOUT TIME ZONE 'now()' " + empresaSeccionSQL;
                    numSiguiente=1;
                }
            }else{

//++//  Cadenas.alert("NO entro");
                // no existe la empresa-seccion se crea, se pone el siguiente nu
```

```
mero a 2 ya que el 1 se acaba de asignar
                sentenciaSQL="INSERT INTO numero (siguiente,cuando,empresa,secci
on) VALUES " +
                                                " (2,TIMESTAMP WITHOUT TIME ZONE 'now()'
,'"+idEmpresa+"','"+idSeccion+"')";
                numSiguiente=1;
            }
//++//  Cadenas.alert(sentenciaSQL);
            laAsignacion();  //Solo hacemos una accion SQL para modificar la BD
con lo escrito en "sentenciaSQL"
            lTicket=new clsTicket (numSiguiente,ahora,idEmpresa,idSeccion);

        } catch (Exception e) {
            //++//  Cadenas.alert(e.toString());
            lTicket=new clsTicket ("9999",ahora);
        }

//++//  Cadenas.alert(" [ TICKET ConexionBD ] \n "+  String.valueOf(lTicket.getN
umero())+" \n " + Cadenas.escribeFecha( lTicket.getCuando() ) );
        return lTicket;

    }

    private boolean esNuevoDia(Timestamp vCuando,Timestamp vAhora){
      //    vCuando.getYear()...etc de Timestamp estan deprecated

      Calendar ahora=Calendar.getInstance();
      Calendar cuando=Calendar.getInstance();
      ahora.setTime(vAhora);
      cuando.setTime(vCuando);
//++//  Cadenas.alert("ahora ="+String.valueOf(ahora.get(Calendar.DAY_OF_YEAR))+
"\n cuando="+String.valueOf(cuando.get(Calendar.DAY_OF_YEAR))+"\n"+
        String.valueOf( ahora.get(Calendar.DAY_OF_YEAR)==cuando.get(Calendar.DAY
_OF_YEAR) ));
      return ( ahora.get(Calendar.DAY_OF_YEAR)!=cuando.get(Calendar.DAY_OF_YEAR)
 );

   // return ( ahora.get(Calendar.DAY_OF_MONTH)==cuando.get(Calendar.DAY_OF_MONT
H)
   //       && ahora.get(Calendar.MONTH)==cuando.get(Calendar.MONTH)
   //       && ahora.get(Calendar.YEAR)==cuando.get(Calendar.YEAR) );

    }

*****************************/


    private Timestamp obtenerFechaSQL() {
        java.util.Date today = new java.util.Date();
        return new java.sql.Timestamp(today.getTime());
        //
        // se puede hacer directamente con una sentencia SQL todo
        //UPDATE numero set sig_numero=sig_numero+1,cuando=TIMESTAMP WITHOUT TIM
E ZONE 'now()';
        //
    }

    public void cerrarConexion() {
        try {
            if (hayConexion) {
```

```
                db.close();
                hayConexion = false;
            }
        } catch(Exception e){
            // GestionLogs.escribirLog(null, "EXCEPCIÃ“N: " + e.toString());
        }
    }


    public void llenaBD() {
        // INUTILIZADO... SOLO PARA RELLENAR LA BASE DE DATOS PARA PROBAR){
        int maxBicis=40 ;
        int nBici=0 ;

        return;


        /*********************************************

        String puestosCercanos="";

        for (int i=1 ; i<=1000;i++){

            if (! existePuesto(String.valueOf(i))) {
                maxBicis = (int) (Math.random()*30+10);
                if (i>2 && i<998)
                    puestosCercanos=String.format("%d", i-2)+","+
                                    String.format("%d", i-1)+","+
                                    String.format("%d", i+1)+","+
                                    String.format("%d", i+2);
                if (i==1)
                    puestosCercanos="999,1000,2,3";
                if (i==2)
                    puestosCercanos="1000,1,3,4";
                if (i==1000)
                    puestosCercanos="999,998,1,2";
                if (i==999)
                    puestosCercanos="997,998,1000,1";

                sentenciaSQL="INSERT INTO puestos (num_puesto,max_bicis,nombre,c
alle,numero_calle,cod_postal,puestos_cercanos)" +
                        "VALUES ('"+String.format("%d", i)+"'," +
//num_puesto
                                "'"+String.valueOf(maxBicis) +"'," +
//max_bicis
                                "'Puesto "+String.valueOf(i) +"'," +
//nombre
                                "'"+"calle puesto "+String.valueOf(i)+"'," +
//calle
                                "'"+String.valueOf(maxBicis)+"'," +
//numero_calle
                                "'"+"46"+String.valueOf(i)+"'," +
//codigo postal
                                "'"+puestosCercanos+"')";
//puestos cercanos


                boolean s=laAsignacion();
            }
            maxBicis=dimeNumMaxBicisEnPuesto(String.format("%d", i));
```

```
            for (int j=1 ;j<=maxBicis;j++){
                    sentenciaSQL="INSERT INTO bicis_puestos (num_puesto,num_bici
,lugar)" +
                                "VALUES ('"+String.format("%d", i)+"'," +
//num_puesto
                                        "'"+String.format("%07d", ++nBici) +"'," +
//num_bici
                                        "'"+String.valueOf(maxBicis-j+1) +"')" ;
//lugar dentro del puesto es el

//  poste en el que esta anclada la bici

                    boolean s=laAsignacion();
            }
        }
        *******************************************************/
    }
}
```