2014

IES San Vicente

Enric Giménez Ribes
Anna Gemma Sanchis Perales

[GESTIÓ DE LA CARTERA D' INVERSIÓ]

La meua cartera d'inversió és una aplicació desenvolupada utilitzant el llenguatge de programació Java. Aquest programari permet gestionar la seua cartera d'inversió, això ho fa realitzant un llistat amb els valors de les accions de les diferents empreses que participen al mercat bursàtil espanyol. A més, mostra la informació sempre actualitzada, ja que mitjançant un fil cada 5 minuts accedeix a les dades de l'íbex35, així mateix davant un llistat permet realitzar les compres i les vendes, l'usuari selecciona l'empresa de la que vol comprar les accions i la quantitat i es procedeix a realitzar la compra, de la mateixa manera funcionen les vendes. A més, sempre tenim present el promig entre les compres i les vendes que l' inversionista fa a la seua cartera d'inversió.

ÍNDEX

1. INTRODUCCIÓ

El present projecte versa sobre el desenvolupament d'una aplicació anomenada "La meua cartera d'inversió", per al qual s'ha utilitzat el llenguatge de programació Java. El principal objectiu d'aquest programari és satisfer les necessitats dels inversors en borsa al mercat espanyol, permetent-los crear i gestionar la seua cartera d'inversió, entenent per cartera d'inversió al conjunt d'actius financers en els quals s'inverteix.

Als darrers anys aquesta tasca ha estat duta a terme per diversos professionals que s'han especialitzat en aquesta funció. Ara mateix, junt amb els professionals, cada vegada està sent més comú optar per una solució més intel·ligent: utilitzar les noves tecnologies a l'àmbit de la gestió d'inversions. Doncs, eixe ha estat el motiu pel qual s'ha decidit desenvolupar una ferramenta de gestió de la cartera d'inversió que permeta fer un seguiment de les carteres d'inversió, de manera senzilla, automàtica i al moment.

La meua cartera d'inversió és un programari amb les següents funcionalitats:

- Mostra en temps real quins són els preus de les accions i participacions en el mercat.
- Permet realitzar inversions mitjançant la compra/venda d'accions.
- Visualitza quin n'és el promig d'aquestes inversions.
- Realitza una valoració històrica i mostra informació actualitzada de les inversions.
- Organitza les inversions en diferents carteres.
- Compara l'evolució de les inversions i la dels principals índex del mercat d'una manera senzilla i molt visual.

Amb tot això, en el que resta de projecte es mostren quins són els objectius que es pretenen assolir amb aquest, a continuació es vorà el desenvolupament de l'aplicació mostrant totes les fases del cicle de vida de desenvolupament del software (anàlisi, disseny, implementació, documentació), tot seguit es mostren les conclusions a les que s'han arribat i per últim està la bibliografia utilitzada així com els pertinents annexos on es mostra més informació de l'aplicació com ara les llibreries utilitzades per implementar-la.

2. ANTECEDENTS

Abans d'iniciar-se amb el desenvolupament de la ferramenta es va realitzar un estudi de mercat per veure quines eren les aplicacions i pàgines web que realitzaven tasques semblants a la idea inicial de l'aplicació. Després de veure diferents solucions i llegir opinions al respecte, es pot dir que s'ha trobat el que a continuació s'exposa:

- Plantilles implementades a un full de càlcul
- Pàgines Web que ofereixen possibilitat de realitzar un seguiment de les carteres d'inversions: http://www.rankia.com/mi-cartera/carteras
- Aplicacions d'escriptori com Pcbolsa o ProRealTime

De tot l'anterior, no s'ha trobat cap producte que tinga tots els objectius marcats, doncs, amb un full de càlcul no hi ha suficient flexibilitat per realitzar el seguiment que s'ha proposat, per altra banda les pàgines Web suposen una bona alternativa però hi hauria que tindre en compte la gestió de la privacitat de les nostres inversions i pel que fa a les aplicacions d'escriptori no tenen una interfície massa intuïtiva, a més pareixen estar enfocades a inversors avançats.

3. DESCRIPCIÓ DEL PROBLEMA I OBJECTIUS

Descripció del problema

La meua cartera d'inversió és una aplicació desenvolupada utilitzant el llenguatge de programació Java. Aquest programari permet gestionar la seua cartera d'inversió, això ho fa realitzant un llistat amb els valors de les accions de les diferents empreses que participen al mercat bursàtil espanyol. A més, mostra la informació sempre actualitzada, ja que mitjançant un fil cada 5 minuts s'accedeix a les dades de l'íbex35, així mateix davant un llistat permet realitzar les compres i les vendes, l'usuari selecciona l'empresa de la que vol comprar les accions i la quantitat i es procedeix a realitzar la compra, de la mateixa manera funcionen les vendes. A més, sempre tenim present el promig entre les compres i les vendes que l'inversionista fa a la seua cartera d'inversió.

Tota aquesta informació queda resumida a un informe que genera la pròpia aplicació.

Objectius

Els objectius que es pretenen assolir amb aquest projecte són els que es mostren a continuació:

- Gestionar els usuaris de l'aplicació: permetent-los registrar-se per a després accedir a les funcionalitats de l'aplicació.
- Donar informació actualitzada i detallada de quin és l'estat actual del mercat bursàtil espanyol. Es mostrarà informació de l'empresa, l'últim valor, la diferència, el màxim, el mínim, el volum, l'efectiu, la data i l'hora.
- Mostrar informació diària de quin és l'estat de l'íbex, com ara el valor, el mínim, el màxim.
- Generar gràfics de quina ha estat l'evolució del valor de les accions de cadascuna de les empreses que participen a l'íbex, permetent-los elegir entre l'evolució mensual, al llarg de 6 mesos o bé un any.
- Realitzar compres d'accions
- Realitzar vendes d'accions.
- Visualitzar el promig entre les vendes i les compres realitzades en la seua cartera d'inversió.
- Generar informes de quin és l'estat actual de la seua cartera d'inversió.

4. DESENVOLUPAMENT DEL PROJECTE

Com ja s'ha descrit als anteriors apartats d'aquest projecte, el principal objectiu és el desenvolupament d'una aplicació per a la creació i gestió de la seua cartera d'inversió.

Quan es tracta de desenvolupar software el primer que hi ha que fer és tindre present tot el referent a l'enginyeria del software. Una de les primeres coses que ha aconseguit l'enginyeria del software ha estat identificar de manera precisa la forma que adopta el procés de desenvolupament del software indicant el passos que cal seguir per a la solució del problema.

Les activitats bàsiques o fases que s'han de realitzar per desenvolupar software són:

- Anàlisi
- Disseny
- Codificació
- Prova
- Manteniment

Per a qualsevol sistema software, amb un cert nivell de complexitat, cadascuna de les fases que hem citat anteriorment, s'han de realitzar de manera formal. Al present projecte degut a les característiques que aquest presenta i a l'elecció del llenguatge de programació, s'ha decidit utilitzar un cicle de vida en espiral ja que es tracta d'un procés evolutiu i incremental.

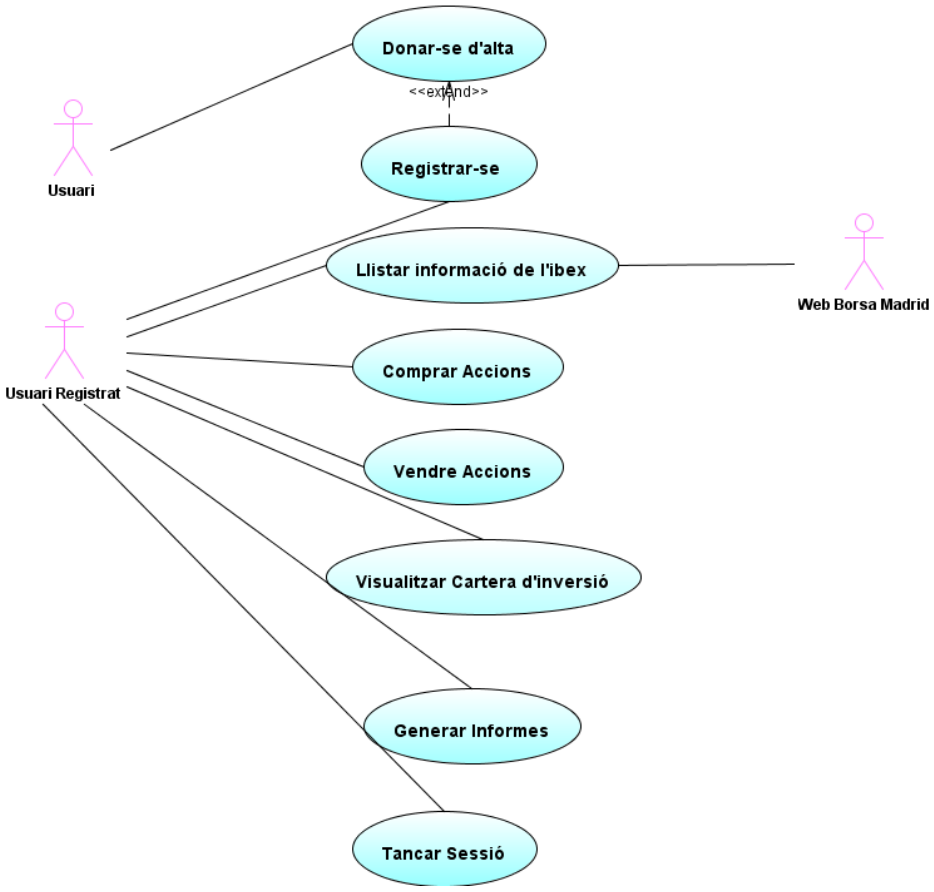
El desenvolupament de software precisa també d'una sèrie de metodologies per tal d'assegurar que tot procés estiga regit per un conjunt de normes i aspectes que puguen assegurar la qualitat i l'èxit del producte. Al igual que passava anteriorment i degut a les característiques del projecte s'utilitzarà una metodologia orientada a objectes, concretament utilitzarem l'UML (Unified Modeling Language) per modelar, construir i documentar el sistema software.

L'UML és un llenguatge unificat de modelització. UML és un conjunt de notacions gràfiques que serveixen per especificar, dissenyar, elaborar i documentar models de sistemes i, en particular, d'aplicacions informàtiques.

4.1 Anàlisi

Per documentar la fase d'anàlisi s'utilitzarà el diagrama de casos d'ús. Un diagrama de casos d'ús és una visualització gràfica dels requisits funcionals del sistema. La seua funció principal és dirigir el procés de creació de software, definint que s'espera d'ell, i el principal avantatge és la facilitat d'interpretar-lo.

Tot seguit es mostra quin és el diagrama de casos d'ús de l'aplicació "La meua cartera d'inversió", es pot veure quins són els actors que interactuaran en el sistema així com les funcionalitats que tindrà l'aplicació.

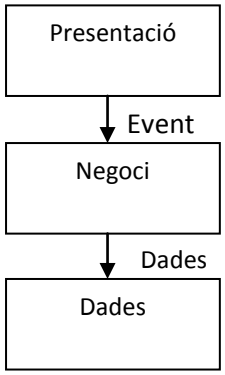


Imatge 1. Diagrama de casos d'ús de l'aplicació

4.2 Disseny

L'objectiu d'aquesta etapa es planificar una solució al problema especificat en el document de requeriments. Aquesta fase és el primer pas que ens porta des del domini del problema cap al domini de la solució, és a dir, hem començat amb què es necessita i ara en el disseny establim com hem de satisfer aquestes necessitats.

Aleshores, serà en aquesta fase on s'especifique com arribar a la solució plantejada a l'anàlisi, és ací on s'ha decidit que el codi s'estructurarà seguint el model per capes, el qual devideix l'aplicació en tres capes: capa de presentació, capa de negoci i capa de dades. Gràficament aquest model quedaria representat de la següent manera:



A més a més, és al disseny on es decideix quin model conceptual de dades utilitzarà el sistema, per definir-lo s'ha utilitzat el diagrama de classes.

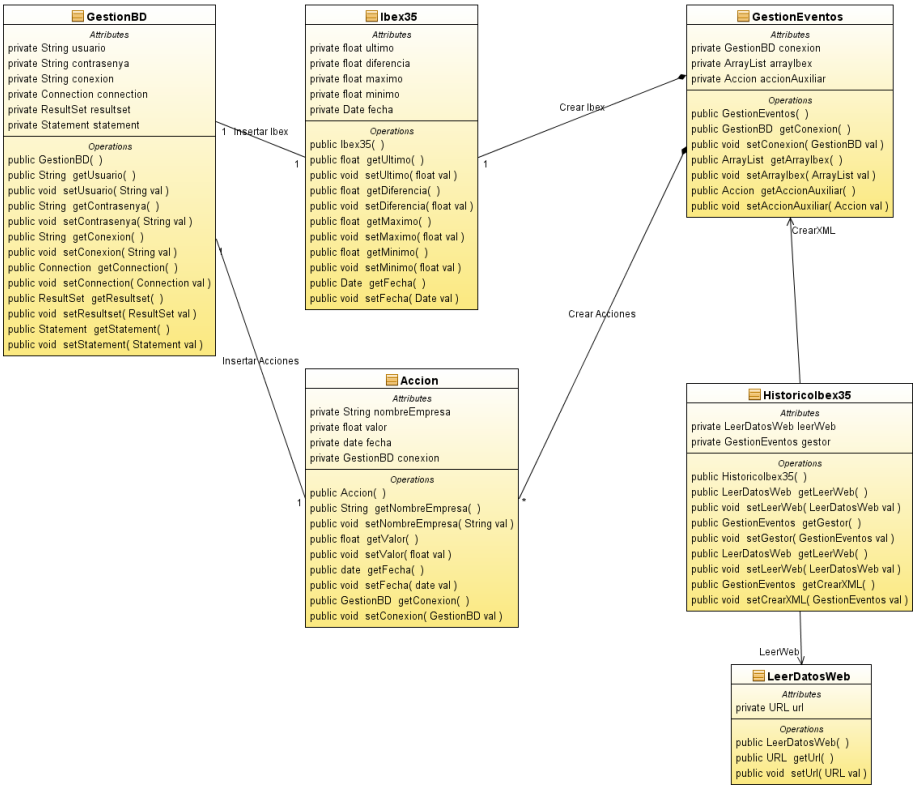
El diagrama de classes és un dels diagrames referents d'UML (Unified Modelling Language), està classificat dins els diagrames de tipus estàtic. És un dels diagrames més utilitzats a les metodologies d'anàlisi i de disseny que es basen en UML.

Un diagrama de classes representa les classes que seran utilitzades dins el sistema i les relacions que existeixen entre elles. Aquest tipus de diagrames són utilitzats durant la fase de disseny dels projectes de desenvolupament de programari. És en aquest moment on es comença a crear el model conceptual de les dades que farà servir el sistema. Per això s'identifiquen els components (amb els seus atributs i funcionalitats) que prendran part en els processos i es defineixen les relacions que hi haurà entre ells.

A aquest projecte en seran dos els diagrames de classes:

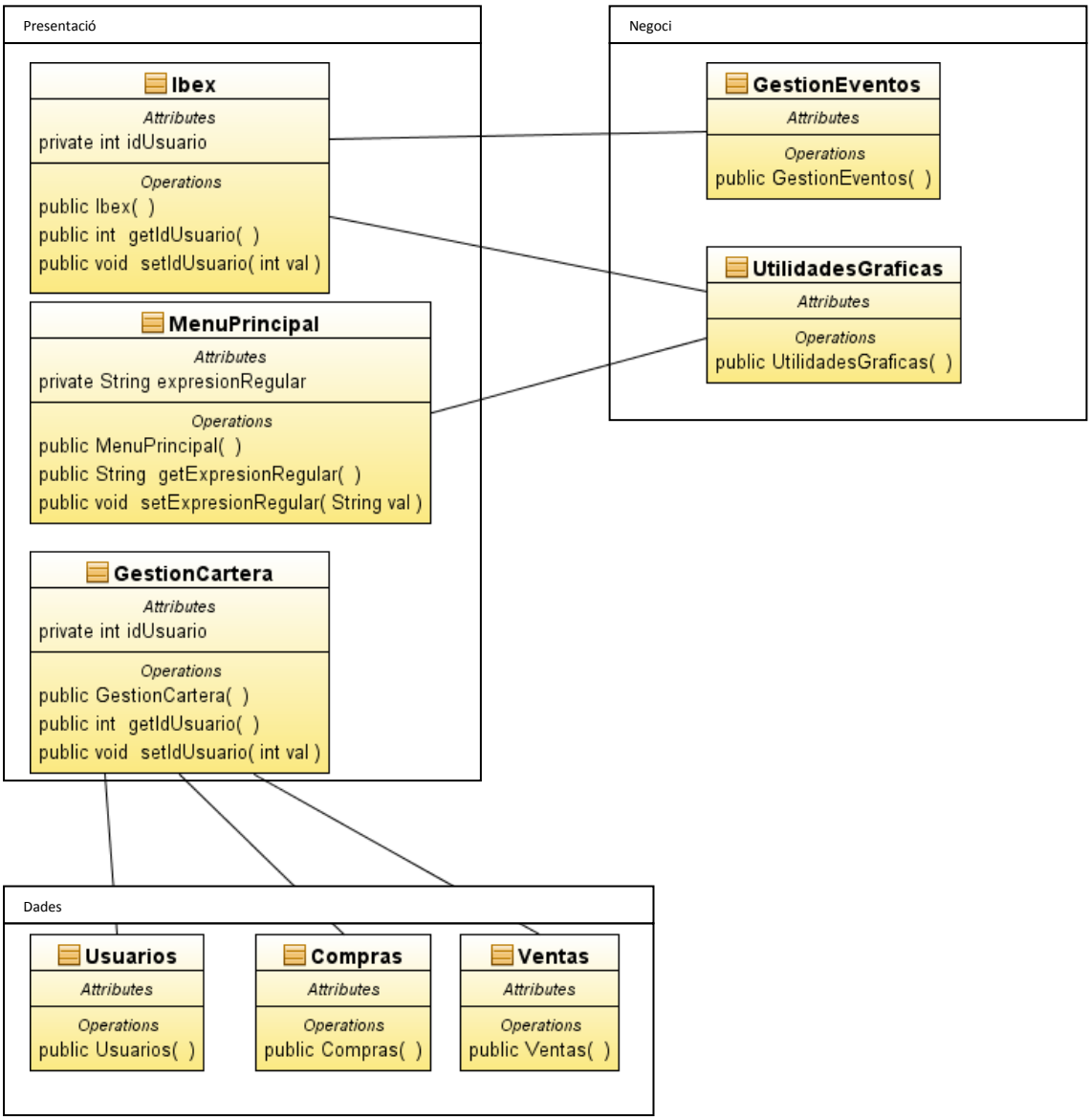
- Per una banda l'utilitzat per implementar el servici que actualitza diàriament la informació que conté la base de dades.
- I per altra banda l'utilitza't per desenvolupar l'aplicació que gestiona la cartera d'inversió.

A continuació es mostra el diagrama de classes de l'aplicació que actualitza diàriament la informació dels històrics a la base de dades. S'han identificat les classes, junt amb els atributs, els mètodes i les relacions entre elles.



Imatge 2. Diagrama de classes del servici que actualitza diàriament la informació de la BD

Tot seguit es mostra el diagrama de classes referent a l'aplicació per a la gestió de la cartera d'inversió.



Imatge 3. Diagrama de classes de l'aplicació

A l'annex 8.2 es poden veure totes les classes de l'aplicació separades per les capes anteriorment descrites.

4.3 Desenvolupament

És en aquest punt on es procedeix a implementar l'aplicació, per això s'ha decidit utilitzar el llenguatge de programació Java i l'entorn de desenvolupament Netbeans.

A partir dels diagrames generats a les anteriors fases s'ha procedit amb la implementació de l'aplicació.

4.3.1 Implementació del servici

El primer pas, ha consistit en implementar un servici que es connecta diàriament a la pàgina web de la borsa de Madrid:

http://www.bolsamadrid.es/esp/aspx/Mercados/Precios.aspx?indice=ESI100000000&punto=indice

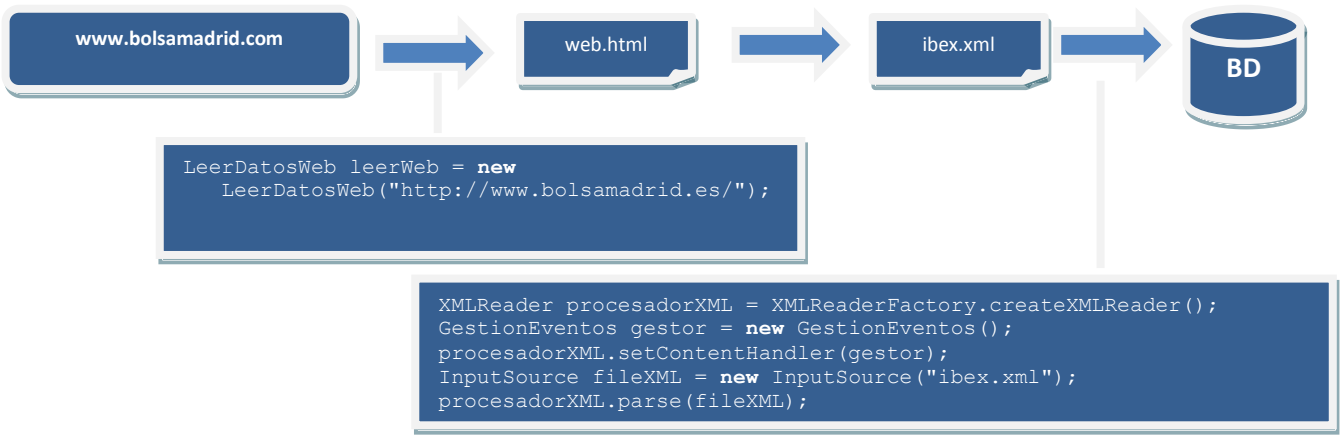
IBEX 35®

Nombre	Anterior	Último	% Dif.	Máximo	Mínimo	Fecha	Hora	% Dif. Año 2014
▲ IBEX 35®	10.567,00	10.587,20	0,19	10.600,90	10.533,40	13/05/2014	17:38	6,76

Nombre	Últ.	% Dif.	Máx.	Mín.	Volumen	Efectivo (miles €)	Fecha	Hora
▲ ABERTIS SE.A	16,8850	0,48	16,9300	16,7700	3.086.008	52.079,92	13/05/2014	Cierre
▲ ACCIONA	59,8700	0,79	59,9000	59,1700	179.915	10.732,03	13/05/2014	Cierre
▼ ACS	31,6500	-0,31	31,9000	31,3350	1.610.993	50.956,25	13/05/2014	Cierre

Imatge 4: informació que conté la pàgina Web de la borsa de Madrid

Per això s'ha implementat amb Java un mètode que ens escriu a un fitxer el codi html de la pàgina Web, després a partir d'eixe codi es genera un fitxer XML que conté tota la informació que es necessita a l'aplicació i que després mitjançant un parser s'aniran seleccionant els camps que es necessiten a la base de dades.



Imatge 5: Diagrama que mostra com s'obté la informació per a la base de dades

Per tal d'automatitzar aquesta funcionalitat s'ha creat un servici utilitzat el **crontab**, per a què diàriament s'actualitzen els històrics a la nostra base de dades.

```
30 18 * * * root /home/user/jar/script.sh
```

Com podem vorer al comandament de Crontab, el nostre script s'executarà cada dia de la setmana.

Per executar el programa JAVA s'ha creat un script en Linux que crida al crontab.

```
#! /bin/bash

java -jar /home/user/jar/Historicos.jar
```

4.3.2 Implementació de l'aplicació

Connexió a la Base de Dades

Per realitzar les connexions a la base de dades s'ha utilitzat la classe "Singletton", doncs, l'ús d'aquesta ens permet tenir únicament una connexió, ja que sols podem crear un objecte d'aquesta classe.

```
public final class GestionBD {

    public Connection conn;
    private Statement statement;
    public static GestionBD db;
    private ResultSet resultSet = null;

    private GestionBD() {
        String url = "jdbc:mysql://localhost:3306/";
        String dbName = "micartera";
        String driver = "com.mysql.jdbc.Driver";
        String userName = "root";
        String password = "";
        try {
            Class.forName(driver).newInstance();
            this.conn = (Connection) DriverManager.getConnection(url + dbName,
userName, password);
            System.out.println("Conectado a la base de datos [MiCartera]");
        } catch (Exception e) {
            System.out.println("ERROR [Conexión]: " + e.toString());
        }
    }

    public static synchronized GestionBD getDbCon() {
        if (db == null) {
            db = new GestionBD();
        }
        return db;
    }

    ...
}
```

Es pot veure que aquesta classe utilitza un instància d'ella mateixa (un objecte) que és "public" i "static". Al ser "public" fa que estiga accessible a tota l'aplicació, i "static" sols es pot instanciar un objecte d'aquest tipus. Per al desenrotllament de l'aplicació aquesta classe ens facilita molt la tasca, ja que sols es crea una connexió a l'inici de l'execució i es tanca al finalitzar l'aplicació.

Fil d'execució

Per tenir actualitzada a la base de dades les dades de l'íbex 35 hi ha un fil que s'executa cada 5 minuts.

```
public class ActualizarIbex extends Thread {

    private Utilidades util;

    public ActualizarIbex() {
        util = new Utilidades();
    }

    public void run() {
        while (true) {
            util.leerWeb();

            try {
                Thread.sleep(300000);

            } catch (InterruptedException ex) {
                System.err.println("ERROR: " + ex.getMessage());
            }
        }
    }
}
```

Una vegada l'aplicació és iniciada aquest fil es queda en segon pla i cada 5 minuts fa una lectura de la web www.bolsamadrid.es, açò permet treballar en dades reals de la borsa tot i que com a màxim pot haver un retràs de 25 minuts.

Registrar-se

Aquesta pantalla es l'encarregada de registrar els nous usuaris a l'aplicació.

Imatge 6: Interfície per registrar-se a l'aplicació

A més el camps queden validats per expressions regulars:

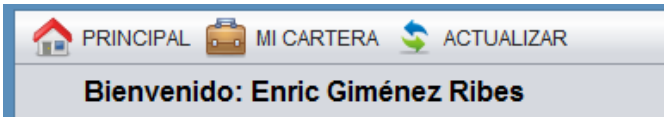
```
private static final String VALIDADNI = "([0-9]{8}[A-Z]{1})";

public boolean validar(String txt, String expresionRegular) {

    Pattern pattern = Pattern.compile(expresionRegular);

    Matcher matcher = pattern.matcher(txt);

    return matcher.matches();

}
```

Aquest mètode torna un booleà que indica si el "String" passat compleix amb l'expressió esperada.

Aquest formulari de registre valida tots els camps, ja siguem enters, cadenes i també les contrasenyes, que s'enregistren encriptant-les amb l'algorisme SHA1.

Menú

Una vegada els usuaris es donen d'alta al sistema, apareix el següent menú amb les opcions de:

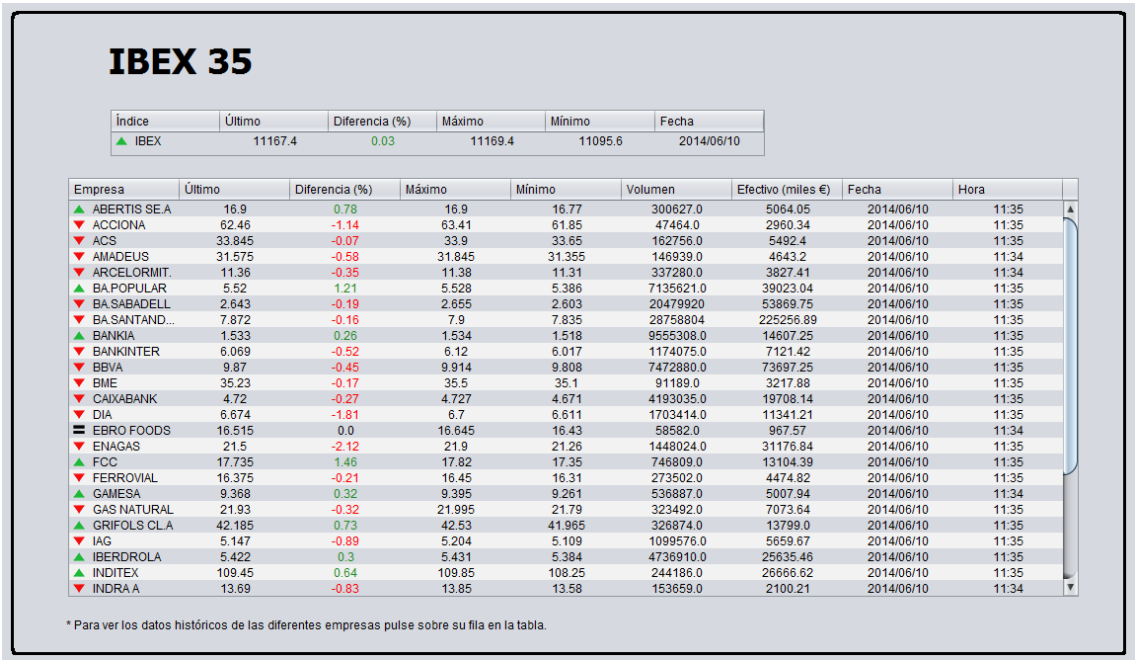


Imatge 7: Menú principal

- Principal: des d'on podem tancar l'aplicació, tancar la sessió de l'usuari i mostrar la informació de l'íbex.
- La meua cartera: que conté les opcions per gestionar la cartera, realitzar compres i vendes i generar informes.
- Actualitzar: com s'ha comentat prèviament la informació per registrar els històrics a la base de dades està implementada com un servici que s'executa a Linux, però com què no tots els usuaris són capaços de poder crear el servici, s'ha creat aquesta opció del menú que actualitza automàticament els històrics simplement seleccionant l'opció actualitzar del menú. A més s'ha controlat que la informació únicament puga ser actualitzada una vegada al dia al tancament de la borsa.

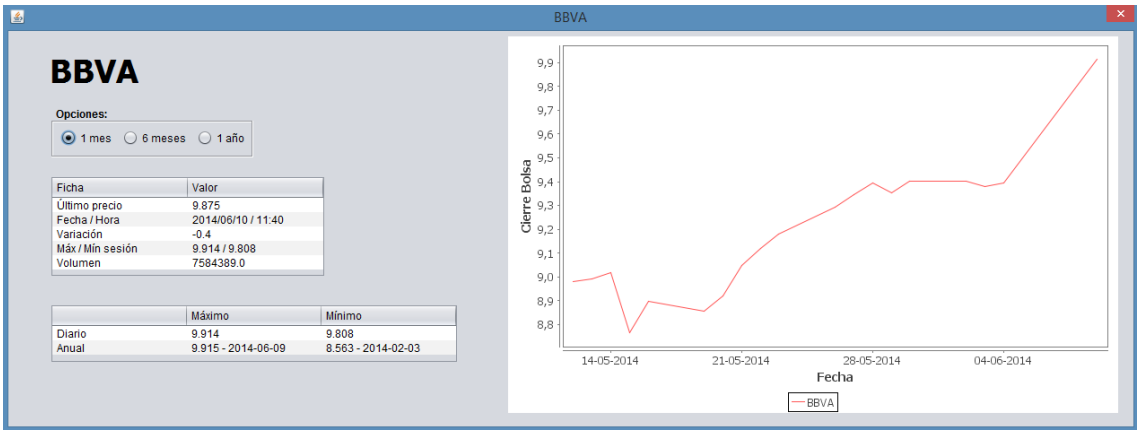
Íbex 35

És en aquesta interfície on es pot trobar tota la informació relativa a l'estat actual de les empreses de l'ÍBEX 35.



Imatge 8: Resum de l'íbex 35

A més, al fer clic en les files de la taula el diàleg ens mostra quina ha estat l'evolució de l'empresa, permetent-nos elegir entre el període d'un mes, sis mesos o bé un any.



Imatge 9: Gràfic que representa l'evolució de l'empresa que participa a l'íbex 35

Per realitzar aquests gràfics s'ha utilitzat la llibreria "JFreeChart", la qual ofereix mètodes per dibuixar gràfics de línies i de barres.

```
public ChartFrame dibujarGraficoLineas(JFrame frame, String nombreEmpresa, int
num) {
    ChartFrame charFrame = null;
    TimeSeries pop = new TimeSeries(nombreEmpresa, Day.class);
    Calendar calendario = Calendar.getInstance();
    long milis1 = calendario.getTimeInMillis();
    Calendar cal = Calendar.getInstance();
    try {
        while (rs.next()) {
            Date fecha = rs.getDate("fecha");
            Double ultimoValor = rs.getDouble(ultimo);
            cal.setTime(fecha);
            long milis2 = cal.getTimeInMillis();
            long diff = milis1 - milis2;
            long difDias = diff / (24 * 60 * 60 * 1000);
            if (difDias <= num) {
                int dia = Integer.parseInt(dateFormatDia.format(fecha));
                int mes = Integer.parseInt(dateFormatMes.format(fecha));
                int anyo = Integer.parseInt(dateFormatAnyo.format(fecha));
                pop.add(new Day(dia, mes, anyo), ultimoValor);
           }
        }
        TimeSeriesCollection dataset = new TimeSeriesCollection();
        dataset.addSeries(pop);
        JFreeChart chart = ChartFactory.createTimeSeriesChart("", "Fecha",
"Cierre Bolsa", dataset, true, true, false);
        chart.getPlot().setBackgroundPaint(Color.WHITE);
        XYPlot plot = chart.getXYPlot();
        DateAxis axis = (DateAxis) plot.getDomainAxis();
        axis.setDateFormatOverride(new SimpleDateFormat("dd-MM-yyyy"));
        charFrame = new ChartFrame("Gráfico histórico " +
nombreEmpresa.toUpperCase(), chart);
        charFrame.pack();
    } catch (SQLException ex) {
        System.err.println("ERROR: " + ex.getMessage());
    }
    return charFrame;
}
```

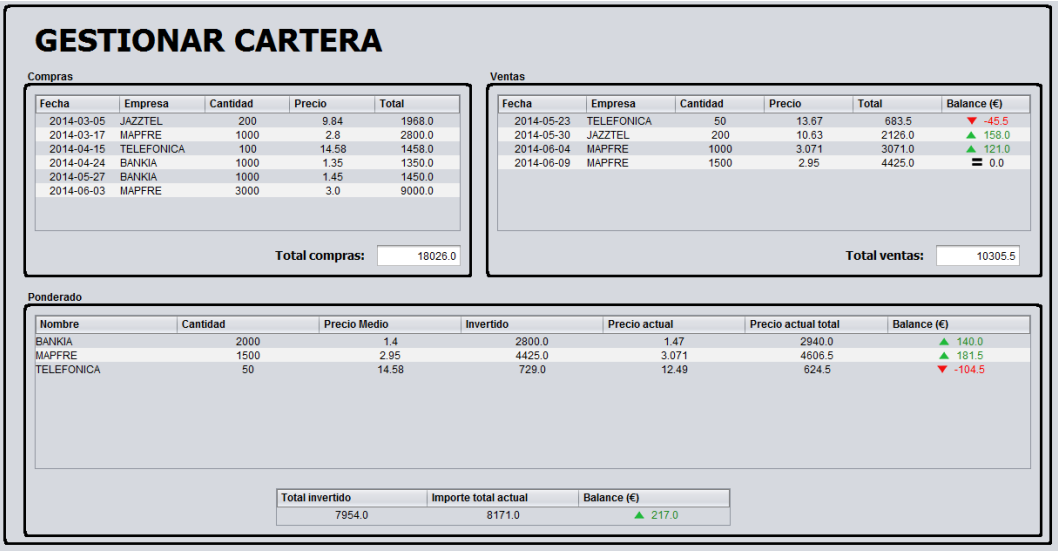
Compres i vendes

Aquesta pantalla s'utilitza per realitzar les compres i les vendes de les accions, els camps a emplenar són els següents: nom de l'empresa, que s'emplena automàticament si ho seleccionem del llistat, el preu, que s'emplena d'igual manera i el nombre d'accions, que en el cas de les compres ha de ser qualsevol número major a 0 i en el cas de les vendes sempre haurà de ser un nombre menor o igual al número de les accions que d'eixa empresa tenim. Tots aquests aspectes es controlen mitjançant validacions amb expressions regulars.

Imatge 10: Interfície per realitzar les compres i les vendes

Gestionar la meua cartera

Aquesta interfície permet fer la gestió de la cartera d'inversió, és ací on es poden veure les compres, les vendes i els preus ponderats de les accions.



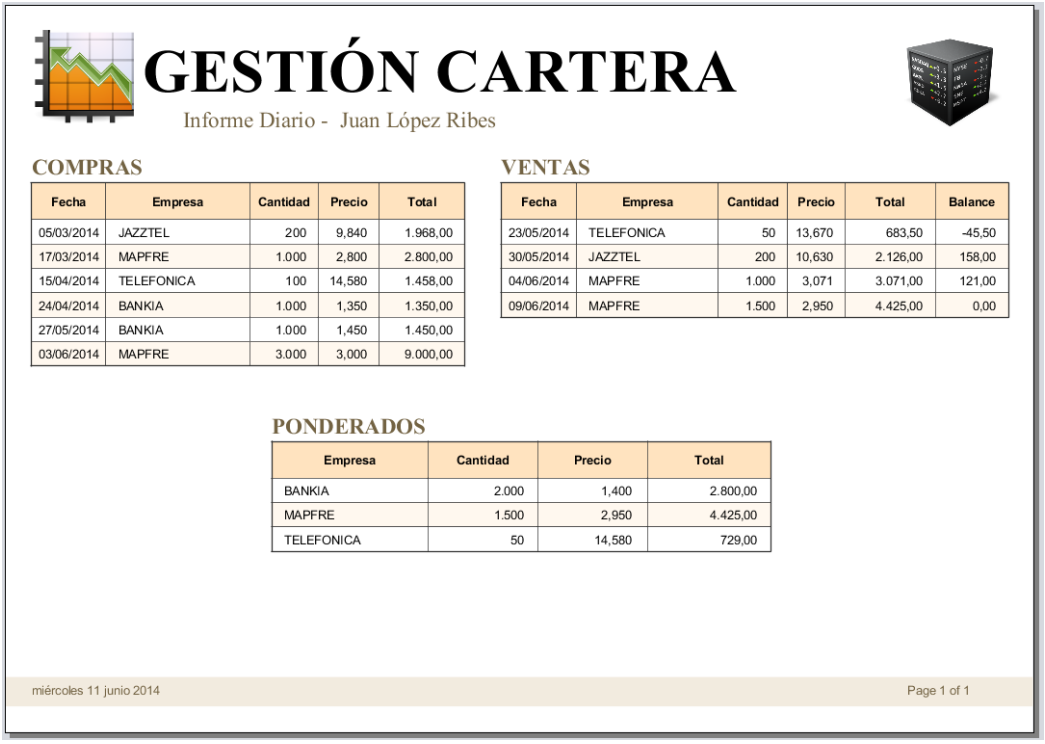
Imatge 11: Intefície per a la gestió de la cartera d'inversió

Per poder centrar els texts en les cel·les de la taula o poder canviar els colors del contingut depenent d'algunes circumstancies s'utilitza un mètode per renderitzar la taula.

```
public void renderTablaCartera(JTable table, final int numColNombre,
        final int numColGanancias) {
    TableCellRenderer render = new TableCellRenderer() {
    int filas = 0;


        @Override
        public Component getTableCellRendererComponent(JTable table,
                Object value, boolean isSelected, boolean hasFocus,
                int row, int column) {

            JLabel lbl = new JLabel(value == null ? "" : value.toString());

            ...

            return lbl;
        }
    };

    for (int i = 0; i < table.getColumnCount(); i++) {
        table.getColumnModel().getColumn(i).setCellRenderer(render);
    }
}
```

Generació d'informes

Una altra opció que té l'aplicació és la generació d'informes. Aquesta ens crea un arxiu pdf que conté la situació actual de la seua cartera d'inversió, amb les compres, vendes i el promig d'aquestes.



GESTIÓN CARTERA

Informe Diario - Juan López Ribes

COMPRAS

Fecha	Empresa	Cantidad	Precio	Total
05/03/2014	JAZZTEL	200	9,840	1.968,00
17/03/2014	MAPFRE	1.000	2,800	2.800,00
15/04/2014	TELEFONICA	100	14,580	1.458,00
24/04/2014	BANKIA	1.000	1,350	1.350,00
27/05/2014	BANKIA	1.000	1,450	1.450,00
03/06/2014	MAPFRE	3.000	3,000	9.000,00

VENTAS

Fecha	Empresa	Cantidad	Precio	Total	Balance
23/05/2014	TELEFONICA	50	13,670	683,50	-45,50
30/05/2014	JAZZTEL	200	10,630	2.126,00	158,00
04/06/2014	MAPFRE	1.000	3,071	3.071,00	121,00
09/06/2014	MAPFRE	1.500	2,950	4.425,00	0,00

PONDERADOS

Empresa	Cantidad	Precio	Total
BANKIA	2.000	1,400	2.800,00
MAPFRE	1.500	2,950	4.425,00
TELEFONICA	50	14,580	729,00

miércoles 11 junio 2014 Page 1 of 1

Imatge 12: Vista de l'informe

Per a la creació de l'informe s'ha utilitzat la llibreria "Jasperreports" de la qual s'ha triat una de les plantilles , s'ha personalitzat amb les pertinents imatges, s'ha seleccionat la informació a mostrar i amb una consulta sql s' extrau la informació necessària de la base de dades.

A continuació es mostra el codi que s'ha utilitzat per crear l'informe:

```
public void crearInforme(int idUsuario) {
    HashMap param = new HashMap();
    param.put("idUsuarioPar", idUsuario);
    param.put("nombreUsuarioPar", asignarNombreSesion(idUsuario));

    JasperReport report = null;
    String ruta = "/reports/informeFinal.jrxml";
    try {
        report =
JasperCompileManager.compileReport(System.getProperty("user.dir").conc
at(ruta));
        JasperPrint print;
        print = JasperFillManager.fillReport(report, param,
GestionBD.getDbCon().getCon());
        JasperViewer view = new JasperViewer(print, false);
        view.setTitle("GESTIÓN DE MI CARTERA DE INVERSIÓN");
        view.setExtendedState(Frame.MAXIMIZED_BOTH);
        view.setVisible(true);
    } catch (JRException ex) {
        System.err.println("ERROR: " + ex.getMessage());
    }
}
```

5. AVALUACIÓ DE RESULTATS

Si es fa referència als objectius que es van marcar a l'inici d'aquest projecte es pot afirmar que s'han complit inclús s'han implementat algunes ampliacions. Al llarg del desenrotllament del projecte s'ha dut a terme el següent:

- Gestió dels usuaris de l'aplicació. Qualsevol persona pot registrar-se a l'aplicació i utilitzar-la. S'ha implementat de manera que un usuari no pot tenir la sessió iniciada en dos ordinadors al mateix temps.
- Visualitzar les dades de l'ÍBEX35 en una taula i visualitzar amb un gràfic les dades històriques que s'han anat emmagatzemant a la base de dades.
- Interfície gràfica intuïtiva on mitjançant les taules es mostra informació de les compres, vendes i els preus actuals de les accions.

D'altra banda s'han pensat algunes ampliacions per a l'aplicació:

- Poder configurar el servidor des de les opcions d'un menú, d'aquesta manera aconseguiríem que qualsevol usuari es podés instal·lar l'aplicació i en tan sols canviar els paràmetres li seria possible executar-la al seu ordinador.
- Creació de la base de dades automàticament al iniciar l'aplicació per primera vegada, així l'usuari sols hauria de posar en funcionament el SGBD i no hauria d'executar cap script, tot estaria automatitzat.
- Fer una versió Web de l'aplicació o bé una aplicació per a dispositius mòbils.

6. CONCLUSIONS

Arribats a aquest punt, podem dir que estem a la fi del desenrotllament del projecte. Després de realitzar l'avaluació de resultats es pot afirmar que s'han assolit tots i cadascun del objectius que es van plantejar inicialment. Amb tot això, també es pot dir que la duració ha estat l'esperada i la que es va planificar inicialment.

Al llarg del projecte ens hem trobat amb alguns problemes, com ara la gestió de les connexions a la base de dades per evitar concurrències i que es quedaren les connexions obertes, per solucionar-ho hem utilitzat la classe singletton. És a dir, aquests problemes s'han anat superant i ens ha servit per aprendre, a més de profunditzar en la matèria tant del llenguatge JAVA, com de la llibreria SWING per implementar la interfície gràfica, a més d'estructurar el projecte amb el model per capes: presentació, negoci i dades. La qual cosa ens ha servit per tindre un codi millor estructurat, a més de ser més fàcil de mantindre i reutilitzar.

Amb tot, podem concloure que s'ha aconseguit desenvolupar l'aplicació "La meua cartera d'inversió", totalment implementada amb el llenguatge Java. Que ens permet gestionar la cartera d'inversió, això ho hem fet realitzant un llistat amb els valors de les accions de les diferents empreses que participen al mercat bursàtil espanyol. A més, mostra la informació sempre actualitzada, ja que mitjançant un fil cada 5 minuts accedeix a les dades de l'íbex35, així mateix davant un llistat permet realitzar les compres i les vendes, l'usuari selecciona l'empresa de la que vol comprar les accions i la quantitat i es procedeix a realitzar la compra, de la mateixa manera funcionen les vendes. A més, sempre tenim present el promig entre les compres i les vendes que l' inversionista fa a la seua cartera d'inversió.

7. BIBLIOGRAFIA

Per al desenrotllament del projecte s'han utilitzat els següents recursos:

Recursos per a Java i el model vista controlador:

- https://www.fdi.ucm.es/profesor/jpavon/poo/2.14.MVC.pdf
- http://codejavu.blogspot.com.es/2013/06/ejemplo-modelo-vista-controlador.html
- http://codejavu.blogspot.com.es/2013/07/ejemplo-patron-singleton.html
- http://www.tutorialspoint.com/java/java_using_singleton.htm
- http://www.chuidiang.com/java/tablas/tablarender/tablarender.php

Pàgina de la borsa de Madrid, des d'on s'obté la informació per a la base de dades:

- http://www.bolsamadrid.es/esp/aspx/Mercados/Precios.aspx?indice=ESI100000000

Recursos per a la creació de gràfics:

- https://www.youtube.com/watch?v=zS8pW1lPk5k
- http://www.jc-mouse.net/java/grafico-de-lineas-con-jfreechart
- http://javalangnullpointer.wordpress.com/2007/02/22/graficas-con-java/

Recursos per a la creació dels informes:

- http://www.javatutoriales.com/2009/02/creacion-de-reportes-con-jasperrepots-y.html
- https://www.youtube.com/watch?v=A-wBTTLtg_g
- https://www.youtube.com/watch?v=Zxqh41bB3tY
- http://siempredesdeelcurro.blogspot.com.es/2010/05/crear-un-jasper-report-con-varias.html

8. ANNEXOS

8.1. Biblioteques utilitzades per a la implementació del projecte

Mysql

Llibreria utilitzada per a connectar la nostra aplicació java en una base de dates relacional que en aquest cas es Mysql. Aquesta llibreria disposa de mètodes per a fer les tasques habituals en una base de dades com son “INSERT”, “DELETE”, “UPDATE”, etc.

JCalendar

Llibreria que ens proporciona un selector de dates d’una manera gràfica i senzilla. Aquesta llibreria la utilitzem per a fer d’una manera senzilla la inserció de dates per part dels usuaris.

JFreeChart

Llibreria utilitzada per a dibuixar les nostres gràfiques. Aquesta llibreria ens permet crear gràfics d’una manera senzilla. JFreeChart necessita també de la llibreria JCommon per a mostrar les gràfiques al usuari.

JCommon

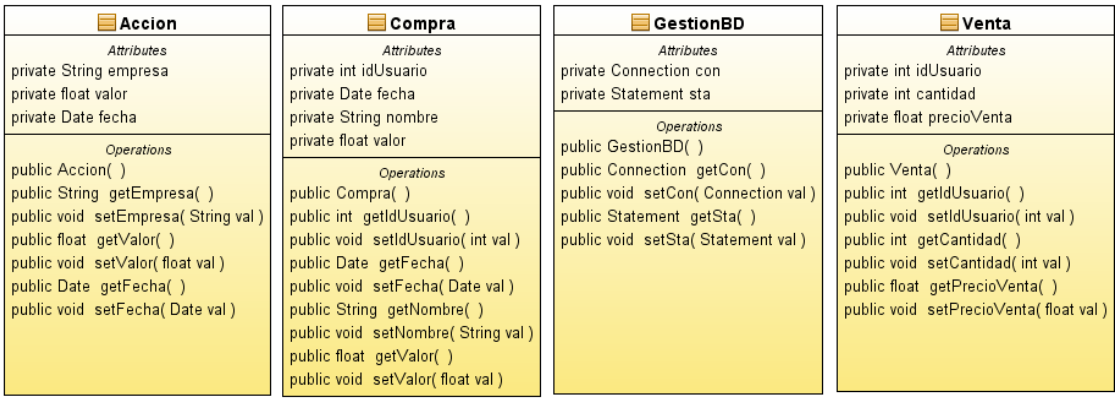
Llibreria utilitzada per incloure classes d’interfície gràfica al dibuixar les nostres gràfiques. Aquesta llibreria la utilitzem per a donar un aspecte visual mes atractiu de cara al usuari.

Jasperreports

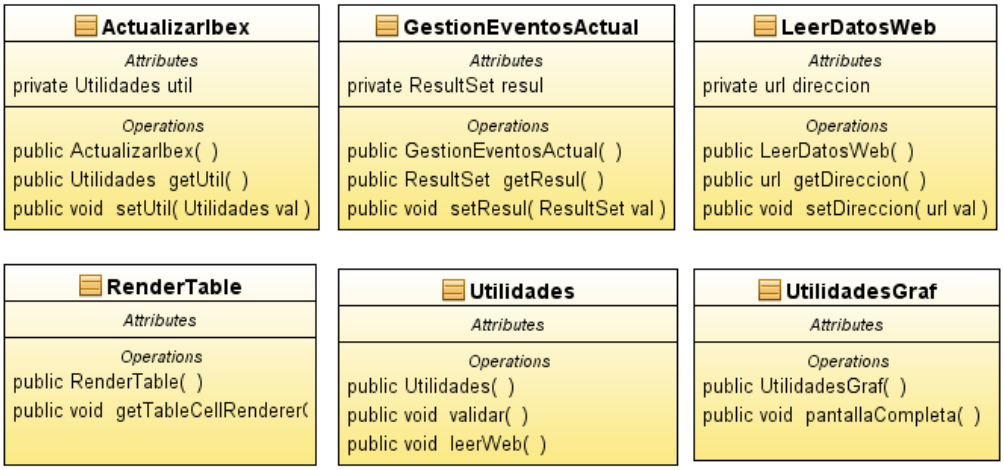
Llibreria utilitzada per a la creació de informes. Aquesta llibreria te la habilitat de entregar contingut al monitor o a la impressora o a fitxers PDF, HTML, XLS, CSV y XML.

8.2. Vista de les classes de l'aplicació

DADES



NEGOCI



PRESENTACIÓ