Projecte Síntesi

Gasolinera

Curs: DAM2 (Desenvolupament d’Aplicacions Multiplataforma)
Alumne: Aitor Marín González

Plantejament del Projecte

Realitzarem el diseny i la implementacio d’una base de dades i una serie d’aplicacions per a l’us i automatitzacio d’una gasolinera.

implementacions amb .net:

Una benzinera vol automatitzar la venda i distribucio del carburant, aixi con la reposicio del mateix, la estacio de servei en qüesto disposa de:

- **4 sortidors, el quals disposen de 2 tipus de combustibles.**
- **4 tipus de combustibles, distribuits entre 4 sortidors.**
- **4 deposits de combustible, on s’enmagatzemen el diferents combustibles**
- **una petita botiga on disposen de 4 articles relacionats amb el manteniment de l’automovil.**

Crearem una base de dades per emmagatzemar i poder consultar l’informacio necessaria, la base de dades contindra les següents taules:

Link diagrama sqlServer

Realitzarem 2 aplicacions de formulari, una client i una admin.

Client: Introduccio de les vendes d’articles i serveis de repostatge, generacio de tickets.

Admin: Mostrar de forma automatica l’estat dels sortidors, la capacitat actual dels deposits i el total de ventes diari i mensual per articles, combustibles i sortidors.

Implementacions amb java:

Crearem dos apps java, una que recollira dades i les pasara a un archiu “SERVEIS.txt” i l’altra agafara les dades del “SERVEIS.txt” generat i les processara i mostrara de determinades formes.

GetData

- Javadoc
- Diagrama Classes
- Diagrama Seqüencia
- Diagrama Casos d’Us

El format del fitxer “*SERVEIS.txt*” sera:

nº de sortidor; T ipus de carburant; HH:mm; litres

1 ; T2 ; 10:50 ; 50.20

ProcessData

- Javadoc
- Diagrama Classes
- Diagrama Seqüencia
- Diagrama Casos d’Us

Mostrarem, per tipus de carburant, quin server a gastat mes litres i quin menys.
Recorrerem els sortidors i mostrarem quin dels deposits esta mes buit i realitzarem una comanda per reomplir-lo.
Llistarem els carburants per ordre de mes a menys contaminant.
Demanarem un nom de carburant, o un nom de preveïdor (els quals mostrarem alhora de demanarlos), i cercarem els combustibles que ens ven cada proveïdor o quins proveïdors en venen cada carburant.
Demanarem un tipus de carburant i mostrarem les seves caracteristiques.

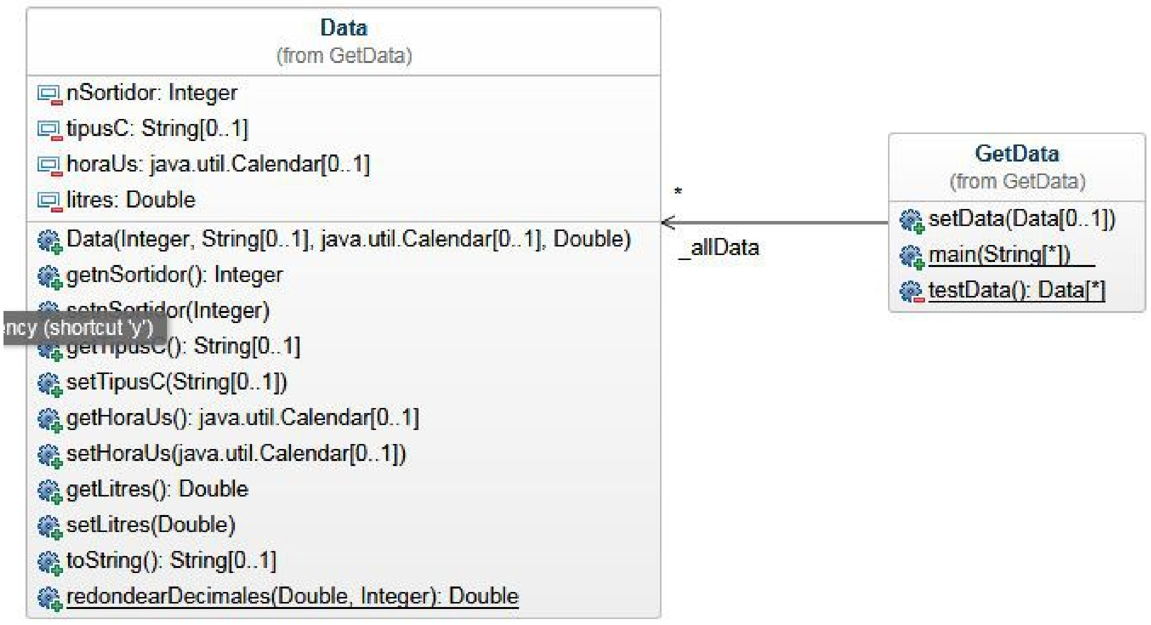
Implementacions amb android:

1ª: realitzar una app que llegeixi un fitxer (amb objectes tipus servei) on guardarem els serveis realitzats, l’app permetra llegir les dades existents e inserir noves dades.

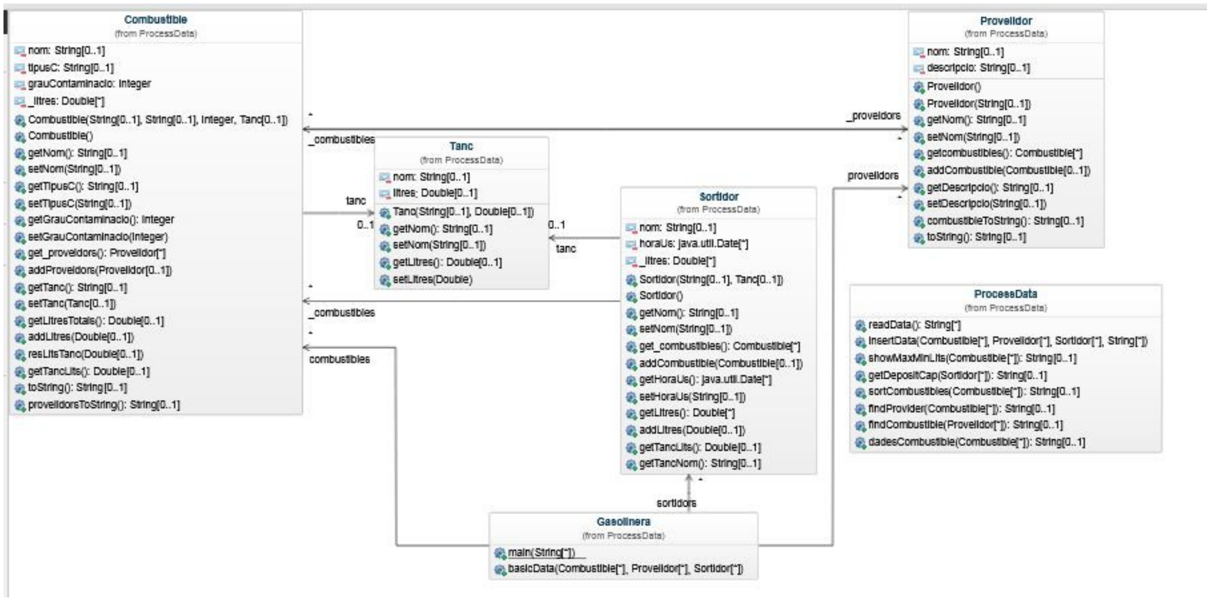
2ª: realitzar una app que mostri les gasolineres properes en relacio a la posicio actual del GPS. (Es tindra en compte com es mostren les dades)(esta tambien pa recu M8 GPS)

Model Conceptual

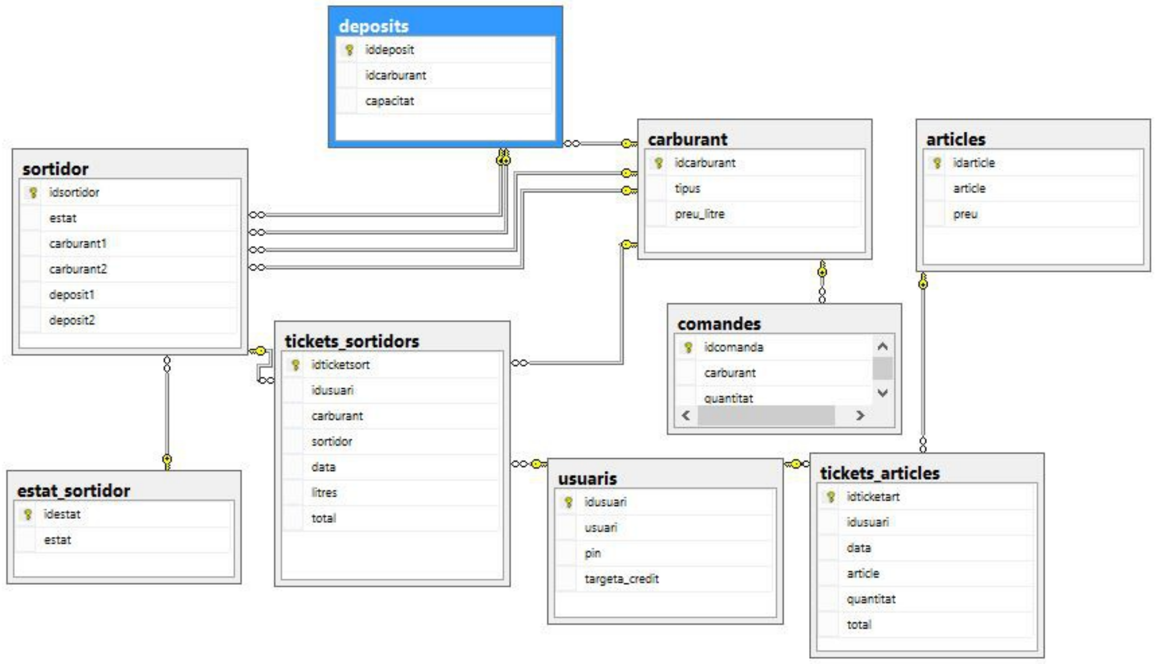
GetData Java:



ProcessData Java:


Combustible
(from ProcessData)
nom: String[0..1]
tipusC: String[0..1]
grauContaminacio: Integer
_litres: Double[*]
Combustible(String[0..1], String[0..1], Integer, Tanc[0..1])
Combustible()
getNom(): String[0..1]
setNom(String[0..1])
getTipusC(): String[0..1]
setTipusC(String[0..1])
getGrauContaminacio(): Integer
setGrauContaminacio(Integer)
get_proveidors(): Proveidor[*]
addProveidors(Proveidor[0..1])
getTanc(): String[0..1]
setTanc(Tanc[0..1])
getLitresTotals(): Double[0..1]
addLitres(Double[0..1])
resLitsTanc(Double[0..1])
getTancLits(): Double[0..1]
toString(): String[0..1]
proveidorsToString(): String[0..1]
_combustibles
tanc
0..1
Tanc
(from ProcessData)
nom: String[0..1]
litres: Double[0..1]
Tanc(String[0..1], Double[0..1])
getNom(): String[0..1]
setNom(String[0..1])
getLitres(): Double[0..1]
setLitres(Double)
0..1
tanc
_combustibles
combustibles
Sortidor
(from ProcessData)
nom: String[0..1]
horaUs: java.util.Date[*]
_litres: Double[*]
Sortidor(String[0..1], Tanc[0..1])
Sortidor()
getNom(): String[0..1]
setNom(String[0..1])
get_combustibles(): Combustible[*]
addCombustible(Combustible[0..1])
getHoraUs(): java.util.Date[*]
setHoraUs(String[0..1])
getLitres(): Double[*]
addLitres(Double[0..1])
getTancLits(): Double[0..1]
getTancNom(): String[0..1]
sortidors
Gasolinera
(from ProcessData)
main(String[*])
basicData(Combustible[*], Proveidor[*], Sortidor[*])
_proveidors
proveidors
Proveidor
(from ProcessData)
nom: String[0..1]
descripcio: String[0..1]
Proveidor()
Proveidor(String[0..1])
getNom(): String[0..1]
setNom(String[0..1])
getcombustibles(): Combustible[*]
addCombustible(Combustible[0..1])
getDescripcio(): String[0..1]
setDescripcio(String[0..1])
combustibleToString(): String[0..1]
toString(): String[0..1]
ProcessData
(from ProcessData)
readData(): String[*]
insertData(Combustible[*], Proveidor[*], Sortidor[*], String[*])
showMaxMinLits(Combustible[*]): String[0..1]
getDepositCap(Sortidor[*]): String[0..1]
sortCombustibles(Combustible[*]): String[0..1]
findProvider(Combustible[*]): String[0..1]
findCombustible(Proveidor[*]): String[0..1]
dadesCombustible(Combustible[*]): String[0..1]

SQL DataBase:



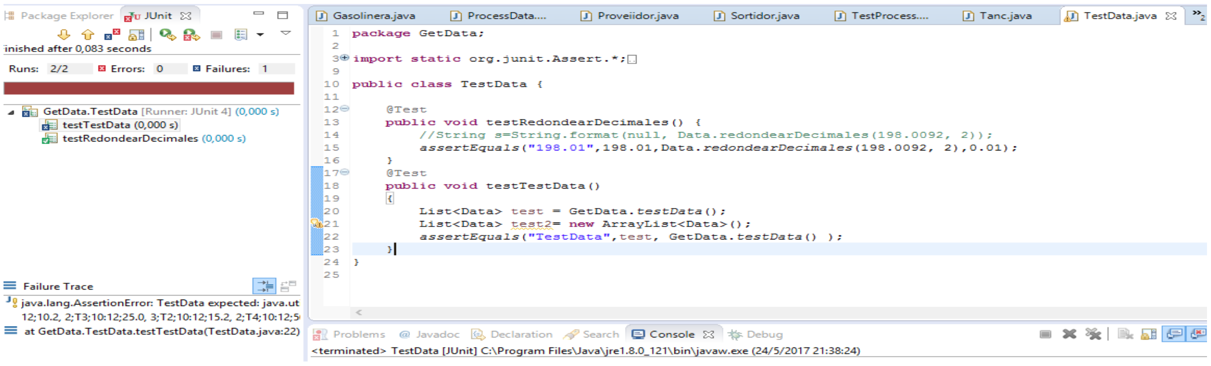
Codi Generacio BD:

```
CREATE DATABASE Gasolinera;
GO
USE Gasolinera;
GO
CREATE TABLE deposits
(
	iddeposit INT IDENTITY(1,1) PRIMARY KEY,
	idcarburant INT FOREIGN KEY REFERENCES carburant(idcarburant),
	capacitat DECIMAL(8,2) NOT NULL CHECK(Capacitat <= 10000),
)
CREATE TABLE carburant
(
	idcarburant INT IDENTITY(1,1) PRIMARY KEY,
	tipus VARCHAR(100) NOT NULL,
	preu_litre DECIMAL(4,2),
)
CREATE TABLE estat_sortidor
(
	idestat INT IDENTITY(1,1) PRIMARY KEY,
	estat VARCHAR(100) NOT NULL,
)
CREATE TABLE sortidor
(
	idsortidor INT IDENTITY(1,1) PRIMARY KEY,
	estat INT FOREIGN KEY REFERENCES estat_sortidor(idestat),
	carburant1 INT FOREIGN KEY REFERENCES carburant(idcarburant),
	carburant2 INT FOREIGN KEY REFERENCES carburant(idcarburant),
	deposit1 INT FOREIGN KEY REFERENCES deposits(iddeposit),
	deposit2 INT FOREIGN KEY REFERENCES deposits(iddeposit),
)
CREATE TABLE comandes
(
	idcomanda INT IDENTITY(1,1) PRIMARY KEY,
	carburant INT FOREIGN KEY REFERENCES carburant(idcarburant),
	quantitat INT NOT NULL,
)
CREATE TABLE articles
(
	idarticle INT IDENTITY(1,1) PRIMARY KEY,
	article VARCHAR(100) NOT NULL,
	preu DECIMAL(4,2) NOT NULL,
)
CREATE TABLE usuaris
(
	idusuari INT IDENTITY(1,1) PRIMARY KEY,
	usuari VARCHAR(100) NOT NULL,
	pin VARCHAR(100) NOT NULL,
	targeta_credit VARCHAR(100),
)
 CREATE TABLE tickets_sortidors
(
	idticketsort INT IDENTITY(1,1) PRiMARY KEY,
	idusuari INT FOREIGN KEY REFERENCES usuaris(idusuari),
	carburant INT FOREIGN KEY REFERENCES carburant(idcarburant),
	sortidor INT FOREIGN KEY REFERENCES sortidor(idsortidor),
	data DATE NOT NULL,
	litres DECIMAL(8,2) NOT NULL,
	total DECIMAL(5,2) ,
)
CREATE TABLE tickets_articles
(
	idticketart INT IDENTITY(1,1) PRIMARY KEY,
	idusuari INT FOREIGN KEY REFERENCES usuaris(idusuari),
	data DATE NOT NULL,
	article INT FOREIGN KEY REFERENCES articles(idarticle),
	quantitat SMALLINT NOT NULL,
	total DECIMAL(5,2) ,
)
--Triggers para pedido y para los totales de tiquets
CREATE TRIGGER tr_comanda
ON deposits
AFTER UPDATE
AS
DECLARE @carburant INT = (SELECT idcarburant FROM deposits WHERE capacitat <=
1000)
DECLARE @capacitat DECIMAL(8,2) = (SELECT capacitat FROM deposits WHERE
idcarburant=@carburant)
DECLARE @total DECIMAL(8,2) = (@capacitat+9000)
IF (@carburant IS NOT NULL)
BEGIN
	INSERT INTO comandes (carburant, quantitat) VALUES (@carburant,9000)
	UPDATE deposits SET capacitat = @total WHERE idcarburant = @carburant
END
GO
```

Joc de Proves d'Aplicacions

3 o 4 jocs de proves en JUnit per al codi java.

GetData:

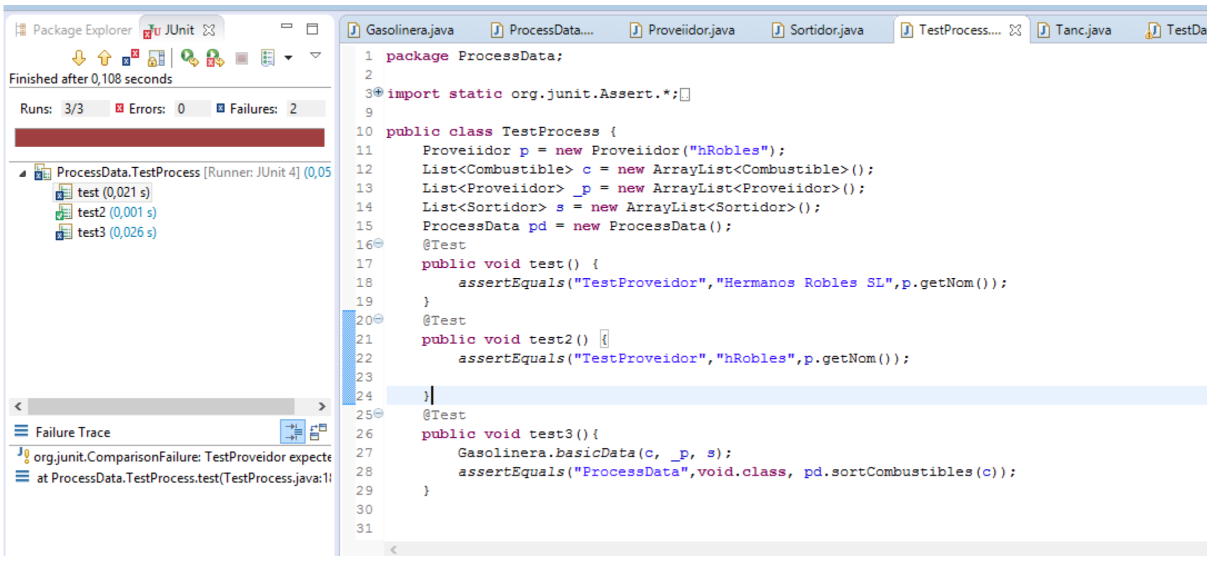


```
Package Explorer   JUnit
Finished after 0,083 seconds
Runs: 2/2    Errors: 0    Failures: 1

GetData.TestData [Runner: JUnit 4] (0,000 s)
    testTestData (0,000 s)
    testRedondearDecimales (0,000 s)

Failure Trace
java.lang.AssertionError: TestData expected: java.ut
12;10.2, 2;T3;10:12;25.0, 3;T2;10:12;15.2, 2;T4;10:12;5
at GetData.TestData.testTestData(TestData.java:22)
```

```java
package GetData;

import static org.junit.Assert.*;

public class TestData {

    @Test
    public void testRedondearDecimales() {
        //String s=String.format(null, Data.redondearDecimales(198.0092, 2));
        assertEquals("198.01",198.01,Data.redondearDecimales(198.0092, 2),0.01);
    }
    @Test
    public void testTestData()
    {
        List<Data> test = GetData.testData();
        List<Data> test2= new ArrayList<Data>();
        assertEquals("TestData",test, GetData.testData() );
    }
}
```

```
<terminated> TestData [JUnit] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (24/5/2017 21:38:24)
```

ProcessData:

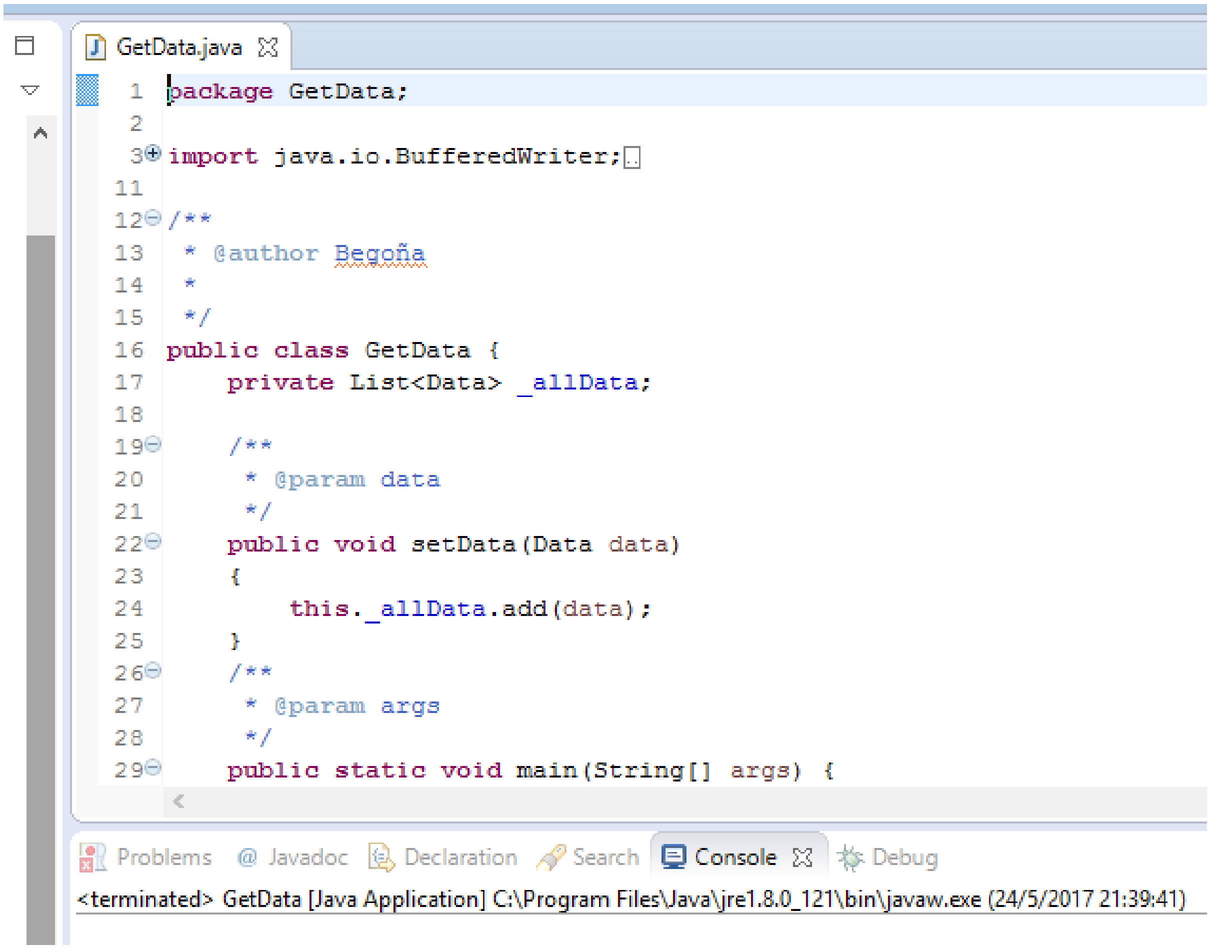


```
Package Explorer   JUnit
Finished after 0,108 seconds
Runs: 3/3    Errors: 0    Failures: 2

ProcessData.TestProcess [Runner: JUnit 4] (0,05
    test (0,021 s)
    test2 (0,001 s)
    test3 (0,026 s)

Failure Trace
org.junit.ComparisonFailure: TestProveidor expecte
at ProcessData.TestProcess.test(TestProcess.java:1
```

```java
package ProcessData;

import static org.junit.Assert.*;

public class TestProcess {
    Proveiidor p = new Proveiidor("hRobles");
    List<Combustible> c = new ArrayList<Combustible>();
    List<Proveiidor> _p = new ArrayList<Proveiidor>();
    List<Sortidor> s = new ArrayList<Sortidor>();
    ProcessData pd = new ProcessData();
    @Test
    public void test() {
        assertEquals("TestProveidor","Hermanos Robles SL",p.getNom());
    }
    @Test
    public void test2() {
        assertEquals("TestProveidor","hRobles",p.getNom());

    }
    @Test
    public void test3(){
        Gasolinera.basicData(c, _p, s);
        assertEquals("ProcessData",void.class, pd.sortCombustibles(c));
    }


```

Manual d'Usuari

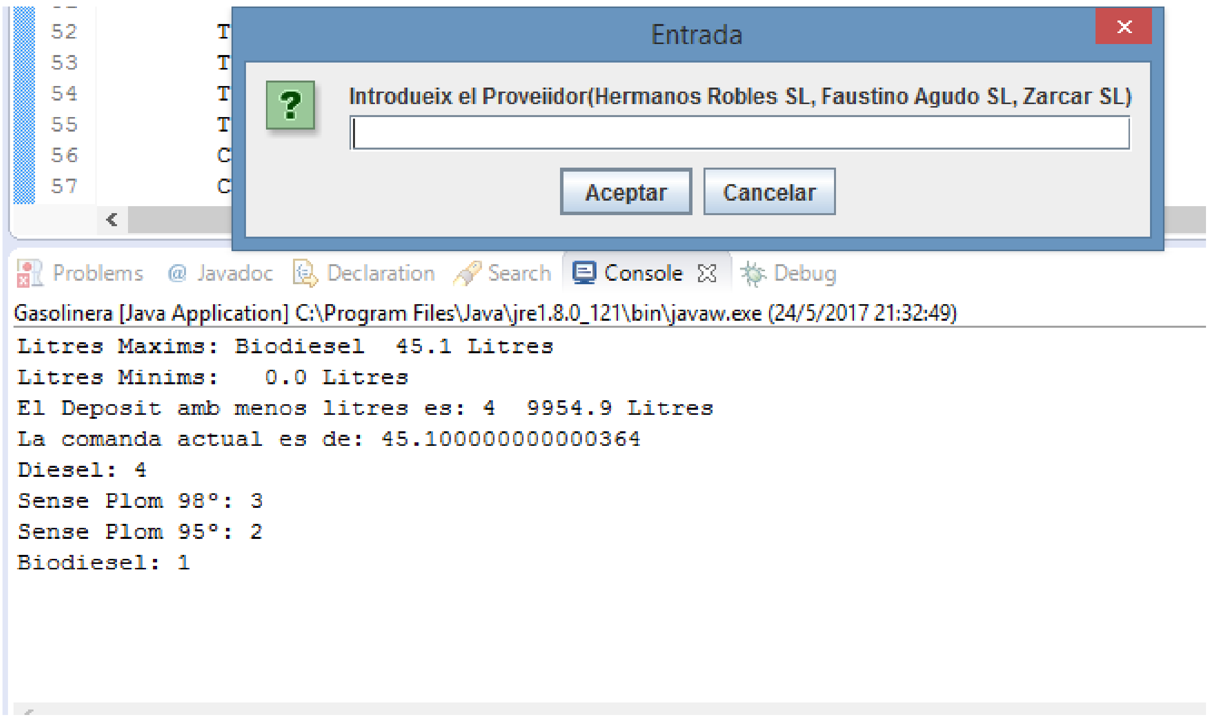
GetData: Nomes necessitem executar aquesta app i automaticament ens generara el fitxer "*SERVEIS.txt*".


GetData.java

```
package GetData;

import java.io.BufferedWriter;

/**
 * @author Begoña
 *
 */
public class GetData {
    private List<Data> _allData;

    /**
     * @param data
     */
    public void setData(Data data)
    {
        this._allData.add(data);
    }
    /**
     * @param args
     */
    public static void main(String[] args) {
```

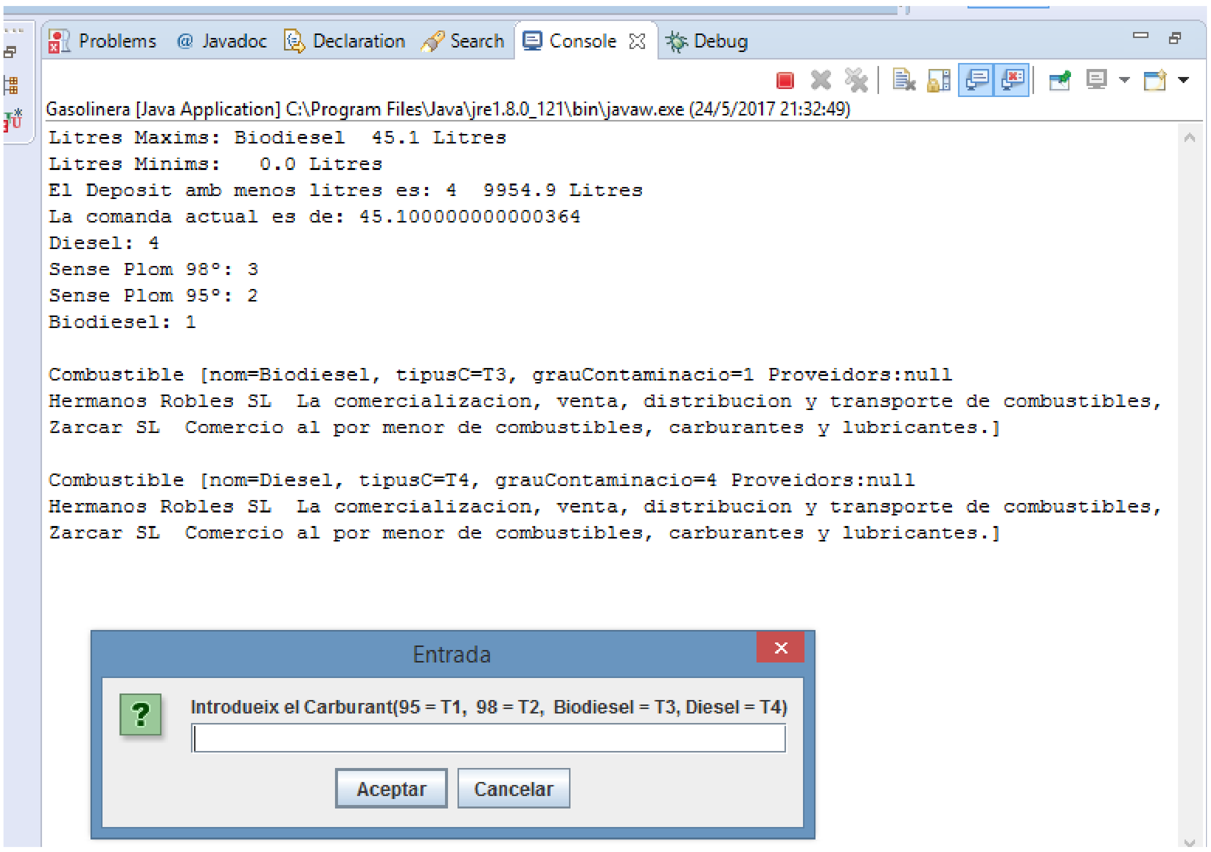
```
<terminated> GetData [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (24/5/2017 21:39:41)
```

ProcessData: Al iniciar la app s'ens mostraran les dades dels serveis amb litres minim i maxim i la llista de carburants ordenats de mes a menys contaminant.


Entrada
Introdueix el Proveiidor(Hermanos Robles SL, Faustino Agudo SL, Zarcar SL)
Aceptar
Cancelar
Problems
Javadoc
Declaration
Search
Console
Debug

```
Gasolinera [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (24/5/2017 21:32:49)
Litres Maxims: Biodiesel  45.1 Litres
Litres Minims:   0.0 Litres
El Deposit amb menos litres es: 4  9954.9 Litres
La comanda actual es de: 45.10000000000364
Diesel: 4
Sense Plom 98°: 3
Sense Plom 95°: 2
Biodiesel: 1
```

Com a opcions adicionals tindrem: Demanar un tipus de carburant a l'usuari, retornant les seves caracteristiques.



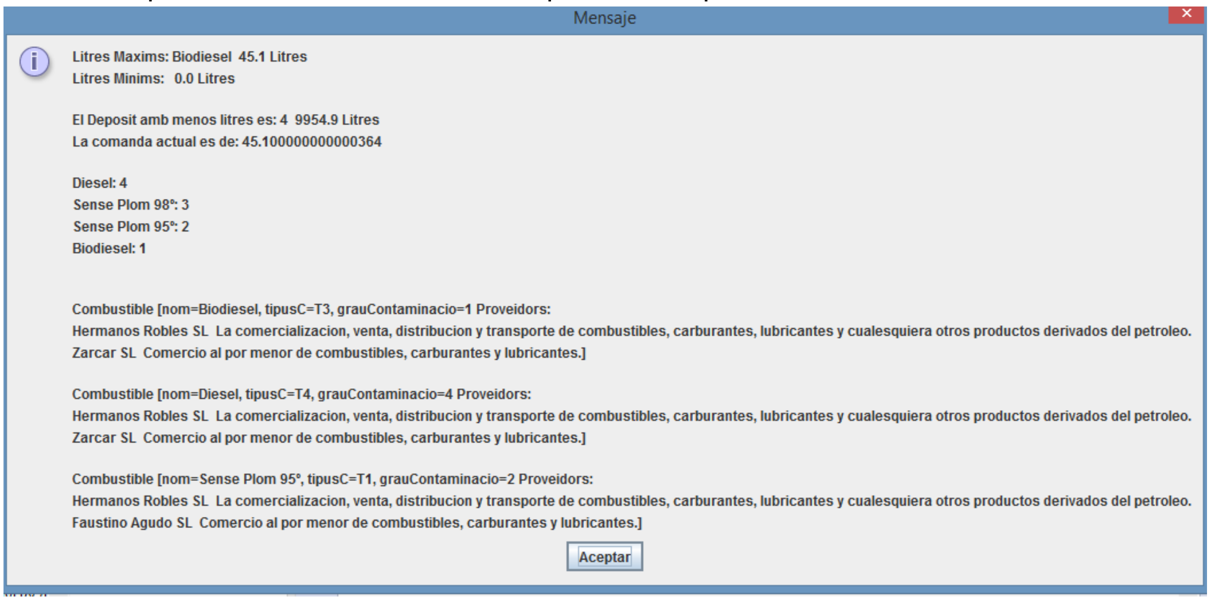
Gasolinera [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (24/5/2017 21:32:49)

```
Litres Maxims: Biodiesel  45.1 Litres
Litres Minims:   0.0 Litres
El Deposit amb menos litres es: 4  9954.9 Litres
La comanda actual es de: 45.100000000000364
Diesel: 4
Sense Plom 98°: 3
Sense Plom 95°: 2
Biodiesel: 1

Combustible [nom=Biodiesel, tipusC=T3, grauContaminacio=1 Proveidors:null
Hermanos Robles SL  La comercializacion, venta, distribucion y transporte de combustibles,
Zarcar SL  Comercio al por menor de combustibles, carburantes y lubricantes.]

Combustible [nom=Diesel, tipusC=T4, grauContaminacio=4 Proveidors:null
Hermanos Robles SL  La comercializacion, venta, distribucion y transporte de combustibles,
Zarcar SL  Comercio al por menor de combustibles, carburantes y lubricantes.]
```

Entrada

Introdueix el Carburant(95 = T1, 98 = T2, Biodiesel = T3, Diesel = T4)

Demanar tipus de carburant i retornar els proveïdors que ens el subministren.

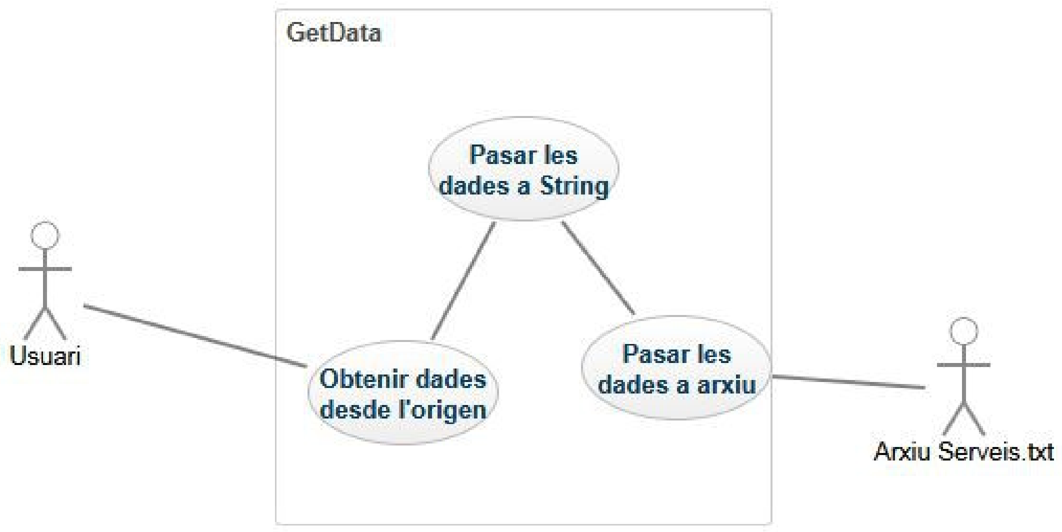


Mensaje

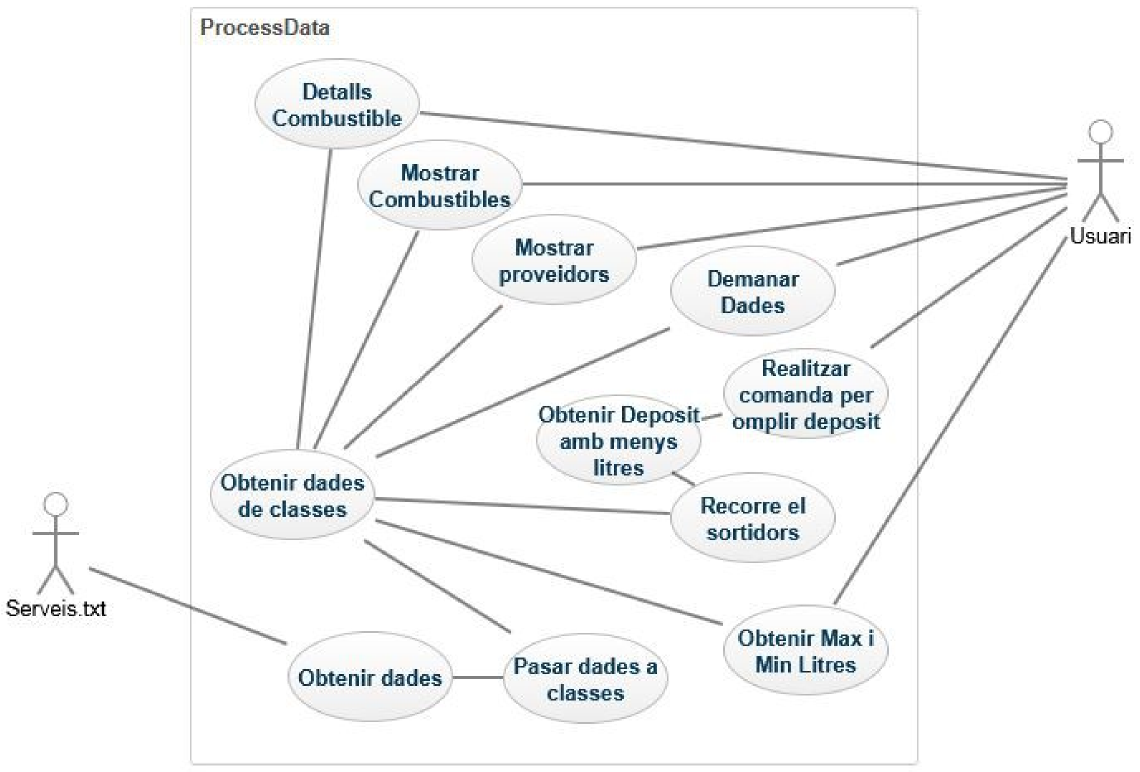
```
Litres Maxims: Biodiesel  45.1 Litres
Litres Minims:   0.0 Litres

El Deposit amb menos litres es: 4  9954.9 Litres
La comanda actual es de: 45.100000000000364

Diesel: 4
Sense Plom 98°: 3
Sense Plom 95°: 2
Biodiesel: 1


Combustible [nom=Biodiesel, tipusC=T3, grauContaminacio=1 Proveidors:
Hermanos Robles SL  La comercializacion, venta, distribucion y transporte de combustibles, carburantes, lubricantes y cualesquiera otros productos derivados del petroleo.
Zarcar SL  Comercio al por menor de combustibles, carburantes y lubricantes.]

Combustible [nom=Diesel, tipusC=T4, grauContaminacio=4 Proveidors:
Hermanos Robles SL  La comercializacion, venta, distribucion y transporte de combustibles, carburantes, lubricantes y cualesquiera otros productos derivados del petroleo.
Zarcar SL  Comercio al por menor de combustibles, carburantes y lubricantes.]

Combustible [nom=Sense Plom 95°, tipusC=T1, grauContaminacio=2 Proveidors:
Hermanos Robles SL  La comercializacion, venta, distribucion y transporte de combustibles, carburantes, lubricantes y cualesquiera otros productos derivados del petroleo.
Faustino Agudo SL  Comercio al por menor de combustibles, carburantes y lubricantes.]
```

Model Casos d’Us

GetData:

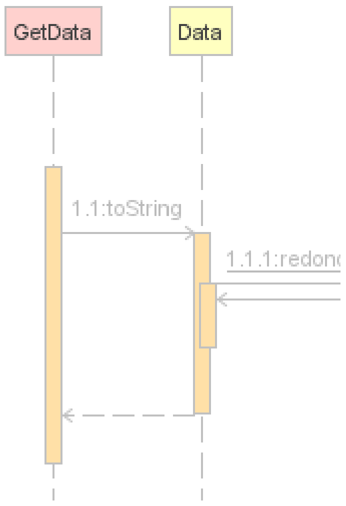


ProcessData:

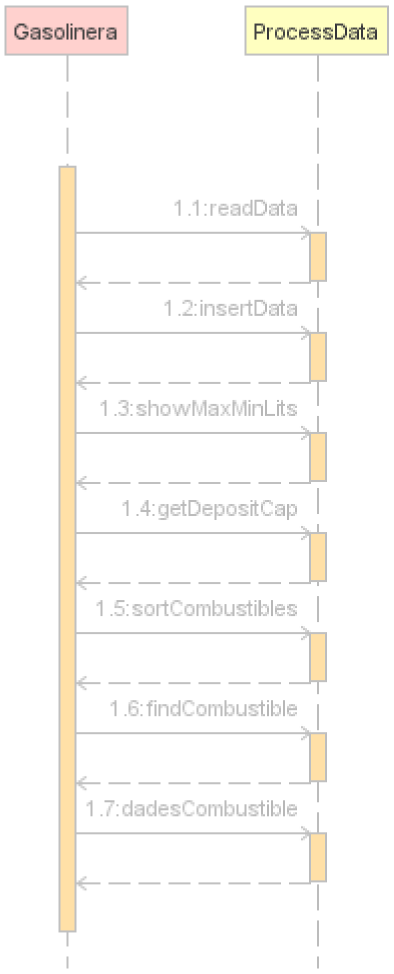


Diagrames de Seqüencia

GetData:



ProcessData:



Flux de Treball

Dillums	**Dimarts**	**Dimecres**	**Dijous**	**Divendres**
Plantejament projecte - **1:30 h** Implementacio amb java GetData - **2:30 h**	Javadoc + Uml classes + seqüencia + cas d'us GetData - **5 h**	Implementacio amb java ProcessData + Javadoc - **5 h**	Uml classes + seqüencia + cas d'us ProcessData - **3:30 h**	Manual d'usuari **0:30 h** Disseny BBDD - **2 h** Implementacio amb .net Client - **3 h**
Dillums	**Dimarts**	**Dimecres**	**Dijous**	**Divendres**
Implementacio amb .net Client - **2 h** Implementacio amb .net Admin - **4 h**	Implementacio amb andoid GPS	Implementacio amb android SERVEIS	*Uml classes + seqüencia + cas d'us android*	Junit tests – **2 h**

Gestio de Riscos

Estimacio Economica

Control del Projecte

Codi Java

GetData:

GetData.java

```
package GetData;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.List;

/**
 * @author Begoña
 *
 */
public class GetData {
	private List<Data> _allData;

	/**
	 * @param data
	 */
	public void setData(Data data)
	{
		this._allData.add(data);
	}
	/**
	 * @param args
	 */
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		GetData gd=new GetData();
		gd._allData=new ArrayList<Data>();
		gd._allData=testData();
		File serveis = new File("SERVEIS.txt");
		try {
			FileWriter escriure = new FileWriter(serveis);
			BufferedWriter bw = new BufferedWriter(escriure);
```

```
				for(int i = 0; i<gd._allData.size(); i++)
				{
					bw.write(gd._allData.get(i).toString());
					bw.newLine();
				}
				bw.close();
				escriure.close();
			} catch (IOException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}

	}

	/* Data Generated for Test Purposes */
	/**
	 * @return
	 */
	private static List<Data> testData()
	{
		List<Data> _test = new ArrayList<Data>();
		double a= 9.5;
		Calendar c1 = new GregorianCalendar();
		c1.set(Calendar.HOUR_OF_DAY, 12);
		c1.set(Calendar.MINUTE, 40);
		Data d1 = new Data(1,"T4",c1,a);
		_test.add(d1);
		a= 10.2;
		c1.set(Calendar.HOUR_OF_DAY, 13);
		c1.set(Calendar.MINUTE, 04);
		Data d2 = new Data(1,"T3",c1,a);
		_test.add(d2);
		a= 25.0;
		c1.set(Calendar.HOUR_OF_DAY, 10);
		c1.set(Calendar.MINUTE, 11);
		Data d3 = new Data(2,"T3",c1,a);
		_test.add(d3);
		a= 15.18;
		c1.set(Calendar.HOUR_OF_DAY, 9);
		c1.set(Calendar.MINUTE, 45);
		Data d4 = new Data(3,"T2",c1,a);
		_test.add(d4);
		a= 50.9;
		c1.set(Calendar.HOUR_OF_DAY, 17);
		c1.set(Calendar.MINUTE, 30);
		Data d5 = new Data(2,"T4",c1,a);
```

```
                _test.add(d5);
                a= 25.0;
                c1.set(Calendar.HOUR_OF_DAY, 11);
                c1.set(Calendar.MINUTE, 46);
                Data d6 = new Data(3,"T3",c1,a);
                _test.add(d6);
                a= 11.5;
                c1.set(Calendar.HOUR_OF_DAY, 10);
                c1.set(Calendar.MINUTE, 15);
                Data d7 = new Data(2,"T2",c1,a);
                _test.add(d7);
                a= 20.11;
                c1.set(Calendar.HOUR_OF_DAY, 20);
                c1.set(Calendar.MINUTE, 26);
                Data d8 = new Data(3,"T3",c1,a);
                _test.add(d8);

                return _test;
        }
}
```

Data.java

```
package GetData;

import java.util.Calendar;

/**
 * @author Begoña
 *
 */
public class Data {
      /**
       * @param nSortidor
       * @param tipusC
       * @param horaUs
       * @param litres
       */
      public Data(int nSortidor, String tipusC, Calendar horaUs, double
litres) {
            this.nSortidor = nSortidor;
            this.tipusC = tipusC;
            this.horaUs = horaUs;
            this.litres = litres;
      }
      private int nSortidor;
      private String tipusC;
      private Calendar horaUs;
      private double litres;
```

```
/**
 * @return
 */
public int getnSortidor() {
      return nSortidor;
}
/**
 * @param nSortidor
 */
public void setnSortidor(int nSortidor) {
      this.nSortidor = nSortidor;
}
/**
 * @return
 */
public String getTipusC() {
      return tipusC;
}
/**
 * @param tipusC
 */
public void setTipusC(String tipusC) {
      this.tipusC = tipusC;
}
/**
 * @return
 */
public Calendar getHoraUs() {

      return horaUs;
}
/**
 * @param horaUs
 */
public void setHoraUs(Calendar horaUs) {
      this.horaUs = horaUs;
}
/**
 * @return
 */
public double getLitres() {
      return litres;
}
/**
 * @param litres
 */
public void setLitres(double litres) {
      this.litres = litres;
}
/* (non-Javadoc)
```

```
         * @see java.lang.Object#toString()
         */
        @SuppressWarnings("static-access")
        public String toString()
        {

              return (Integer.toString(nSortidor)
+";"+tipusC+";"+horaUs.HOUR+":"+horaUs.MINUTE+";"+redondearDecimales(litres
,1));
        }
        /**
         * @param valorInicial
         * @param numeroDecimales
         * @return
         */
        public static double redondearDecimales(double valorInicial, int
numeroDecimales) {
          double parteEntera, resultado;
          resultado = valorInicial;
          parteEntera = Math.floor(resultado);
          resultado=(resultado-parteEntera)*Math.pow(10, numeroDecimales);
          resultado=Math.round(resultado);
          resultado=(resultado/Math.pow(10, numeroDecimales))+parteEntera;
          return resultado;
      }
}
```

ProcessData:

Gasolinera.java

```
package ProcessData;

import java.util.ArrayList;
import java.util.List;

import javax.swing.JOptionPane;

/**
 * @author Begoña
 *
 */
public class Gasolinera {

      private List<Combustible> combustibles = new ArrayList<Combustible>();
      private List<Proveiidor> proveiidors = new ArrayList<Proveiidor>();
      private List<Sortidor> sortidors = new ArrayList<Sortidor>();
```

```
        /**
         * @param args
         */
        public static void main(String[] args) {
                String[] data;
                Gasolinera g1 = new Gasolinera();
                basicData(g1.combustibles, g1.proveiidors, g1.sortidors);
                ProcessData process = new ProcessData();
                List<String> readedData = process.readData();
                StringBuilder sb =new StringBuilder();
                for(int i=0;i<readedData.size();i++)
                {
                        sb.append(readedData.get(i)+"\n");
                        data = readedData.get(i).split(";");
                        process.insertData(g1.combustibles, g1.proveiidors, g1.sortidors,
data);
                }
                //JOptionPane.showMessageDialog(null, sb.toString());
                String litsMN = process.showMaxMinLits(g1.combustibles); //devuelve litros
max y min por carburante (es return)
                String depCap = process.getDepositCap(g1.sortidors);
                String Contaminants =
process.sortCombustibles(g1.combustibles).replace("null", " ");
                String CombFromProv =
process.findCombustible(g1.proveiidors).replace("null", " ");
                //String ProvFromComb = process.findProvider(g1.combustibles);
                String dadcomb =
process.dadesCombustible(g1.combustibles).replace("null", " ");

                JOptionPane.showMessageDialog(null,
litsMN+"\n\n"+depCap+"\n\n"+Contaminants+"\n\n"+CombFromProv+"\n\n"+dadcomb);
        }

        /**
         * @param c
         * @param p
         * @param s
         */
        public static void basicData(List<Combustible> c, List<Proveiidor> p, List<Sortidor>
s) {

                Tanc t95 = new Tanc("Deposit 95", 10000.00);
                Tanc t98 = new Tanc("Deposit 98", 10000.00);
                Tanc tbiodiesel = new Tanc("Deposit Biodiesel", 10000.00);
                Tanc tdiesel = new Tanc("Deposit Diesel", 10000.00);
```

```
Combustible sp95 = new Combustible("Sense Plom 95º","T1",2,t95);
Combustible sp98 = new Combustible("Sense Plom 98º","T2",3,t98);
Combustible biodiesel = new Combustible("Biodiesel","T3",1,tbiodiesel);
Combustible diesel = new Combustible("Diesel","T4",4,tdiesel);
Sortidor s1 = new Sortidor("1",t95);
Sortidor s2 = new Sortidor("2",t98);
Sortidor s3 = new Sortidor("3",tbiodiesel);
Sortidor s4 = new Sortidor("4",tdiesel);
Proveiidor hRobles = new Proveiidor("Hermanos Robles SL");
Proveiidor fAgudo = new Proveiidor("Faustino Agudo SL");
Proveiidor zarcar = new Proveiidor("Zarcar SL");

s1.addCombustible(sp95);
s1.addCombustible(sp98);
s2.addCombustible(sp95);
s2.addCombustible(sp98);
s3.addCombustible(biodiesel);
s3.addCombustible(diesel);
s4.addCombustible(biodiesel);
s4.addCombustible(diesel);
hRobles.addCombustible(sp95);
hRobles.addCombustible(sp98);
hRobles.addCombustible(biodiesel);
hRobles.addCombustible(diesel);
fAgudo.addCombustible(sp95);
fAgudo.addCombustible(sp98);
zarcar.addCombustible(biodiesel);
zarcar.addCombustible(diesel);
sp95.addProveidors(hRobles);
sp95.addProveidors(fAgudo);
sp98.addProveidors(hRobles);
sp98.addProveidors(fAgudo);
biodiesel.addProveidors(hRobles);
biodiesel.addProveidors(zarcar);
diesel.addProveidors(hRobles);
diesel.addProveidors(zarcar);
c.add(sp95);
c.add(sp98);
c.add(biodiesel);
c.add(diesel);
p.add(hRobles);
p.add(fAgudo);
p.add(zarcar);
s.add(s1);
s.add(s2);
s.add(s3);
```

```
		s.add(s4);
	}

}
```

Combustible.java

```
package ProcessData;
import java.util.ArrayList;
import java.util.List;

/**
 * @author Begoña
 *
 */
public class Combustible {

	/**
	 * @param nom
	 * @param tipusC
	 * @param grauContaminacio
	 * @param tanc
	 */
	public Combustible(String nom, String tipusC, int grauContaminacio, Tanc tanc) {
		this.nom = nom;
		this.tipusC = tipusC;
		this.grauContaminacio = grauContaminacio;
		this.setTanc(tanc);
	}

	public Combustible() {
	}

	private String nom;
	private String tipusC;
	private int grauContaminacio;
	private Tanc tanc;
	private List<Proveiidor> _proveidors = new ArrayList<Proveiidor>();
	private List<Double> _litres = new ArrayList<Double>();

	/**
	 * @return
	 */
	public String getNom() {
		return nom;
}

/**
 * @param nom
 */
public void setNom(String nom) {
	this.nom = nom;
}

/**
 * @return
 */
public String getTipusC() {
	return tipusC;
}

/**
 * @param tipusC
 */
public void setTipusC(String tipusC) {
	this.tipusC = tipusC;
}

/**
 * @return
 */
public int getGrauContaminacio() {
	return grauContaminacio;
}

/**
 * @param grauContaminacio
 */
public void setGrauContaminacio(int grauContaminacio) {
	this.grauContaminacio = grauContaminacio;
}

/**
 * @return
 */
public List<Proveiidor> get_proveidors() {
	return _proveidors;
}

/**
 * @param proveidor
 */
public void addProveidors(Proveiidor proveidor) {
          this._proveidors.add(proveidor);
}

/**
 * @return
 */
public String getTanc() {
          return tanc.getNom();
}

/**
 * @param tanc
 */
public void setTanc(Tanc tanc) {
          this.tanc = tanc;
}

/**
 * @return
 */
public Double getLitresTotals() {
          Double sum = 0.0;
          for(Double d : this._litres)
          {
                    sum+=d;
          }
          return sum;
}

/**
 * @param _litres
 */
public void addLitres(Double _litres) {
          this._litres.add(_litres);
}
/**
 * @param lits
 */
public void resLitsTanc(Double lits)
{
          this.tanc.setLitres(lits);
}
/**
 * @return
```

```
	 */
	public Double getTancLits()
	{
		return this.tanc.getLitres();
	}

	/* (non-Javadoc)
	 * @see java.lang.Object#toString()
	 */
	@Override
	public String toString() {
		return "Combustible [nom=" + nom + ", tipusC=" + tipusC
				+ ", grauContaminacio=" + grauContaminacio +"
Proveidors:"+proveiidorsToString()+ "]";
	}
	/**
	 * @return
	 */
	public String proveiidorsToString()
	{
		String s=null;
		for(int i=0;i<this._proveidors.size();i++)
		{
			s+= "\n"+this._proveidors.get(i).getNom()+"
"+this._proveidors.get(i).getDescripcio();
		}
		return s;
	}




}
```

Tanc.java

```
package ProcessData;

/**
 * @author Begoña
 *
 */
public class Tanc {
	/**
	 * @param nom
	 * @param litres
	 */
	public Tanc(String nom, Double litres) {
		this.nom = nom;
		this.litres = litres;
	}
	private String nom;
	private Double litres;
	/**
	 * @return
	 */
	public String getNom() {
		return nom;
	}
	/**
	 * @param nom
	 */
	public void setNom(String nom) {
		this.nom = nom;
	}
	/**
	 * @return
	 */
	public Double getLitres() {
		return litres;
	}
	/**
	 * @param litres
	 */
	public void setLitres(double litres) {
		this.litres -= litres;
	}

}
```

ProcessData.java

```
package ProcessData;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import javax.swing.JOptionPane;


/**
 * @author Begoña
 *
 */
public class ProcessData {

	/**
	 * @return
	 */
	public List<String> readData()
	{
		try{
			FileReader lector;
				try {
					lector = new FileReader("SERVEIS.txt");

				BufferedReader bf = new BufferedReader(lector);
				String linea;
				List<String> arlist = new ArrayList<String>();
					while((linea=bf.readLine())!=null){
						arlist.add(linea);
					}

					bf.close();

					lector.close();

				return arlist;
				} catch (FileNotFoundException e) {
					// TODO Auto-generated catch block
					e.printStackTrace();
				}
			} catch(IOException e){
				e.printStackTrace();
			}
		return null;
		}

	/**
	 * @param combustibles
	 * @param proveiidors
	 * @param sortidors
	 * @param data
	 */
	public void insertData(List<Combustible> combustibles, List<Proveiidor> proveiidors,
List<Sortidor> sortidors, String[] data)
	{
		if (data[0].equals("1")){
			sortidors.get(0).setHoraUs(data[2]);
```

```
                    sortidors.get(0).addLitres(Double.parseDouble(data[3]));
                    sortidors.get(0).addLitres(Double.parseDouble(data[3]));
                    if(data[1].equals("T1")){
                            combustibles.get(0).addLitres(Double.parseDouble(data[3]));
                    } else if(data[1].equals("T2")){
                            combustibles.get(1).addLitres(Double.parseDouble(data[3]));
                    }

            } else if (data[0].equals("2")){
                    sortidors.get(1).setHoraUs(data[2]);
                    sortidors.get(1).addLitres(Double.parseDouble(data[3]));
                    sortidors.get(1).addLitres(Double.parseDouble(data[3]));
                    if(data[1].equals("T1")){
                            combustibles.get(0).addLitres(Double.parseDouble(data[3]));
                            combustibles.get(0).resLitsTanc(Double.parseDouble(data[3]));
                    } else if(data[1].equals("T2")){
                            combustibles.get(1).addLitres(Double.parseDouble(data[3]));
                            combustibles.get(1).resLitsTanc(Double.parseDouble(data[3]));
                    }
                            }else if (data[0].equals("3")){
                                    sortidors.get(2).setHoraUs(data[2]);

    sortidors.get(2).addLitres(Double.parseDouble(data[3]));

    sortidors.get(2).addLitres(Double.parseDouble(data[3]));
                                    if(data[1].equals("T3")){

    combustibles.get(2).addLitres(Double.parseDouble(data[3]));

    combustibles.get(2).resLitsTanc(Double.parseDouble(data[3]));
                                    } else if(data[1].equals("T4")){

    combustibles.get(3).addLitres(Double.parseDouble(data[3]));

    combustibles.get(3).resLitsTanc(Double.parseDouble(data[3]));
                                    }
                            } else if (data[0].equals("4")){
                                    sortidors.get(3).setHoraUs(data[2]);

    sortidors.get(3).addLitres(Double.parseDouble(data[3]));

    sortidors.get(3).addLitres(Double.parseDouble(data[3]));
                                    if(data[1].equals("T3")){

    combustibles.get(2).addLitres(Double.parseDouble(data[3]));
                                    } else if(data[1].equals("T4")){
```

```
combustibles.get(3).addLitres(Double.parseDouble(data[3]));
                                        }
                              }
}
/**
 * @param combustibles
 * @return
 */
public String showMaxMinLits(List<Combustible> combustibles)
{
          Double d1, d2, d3, d4;
          String n1, n2, n3, n4, nmax = "", nmin = "";

          d1=combustibles.get(0).getLitresTotals();
          n1=combustibles.get(0).getNom();
          d2=combustibles.get(1).getLitresTotals();
          n2=combustibles.get(1).getNom();
          d3=combustibles.get(2).getLitresTotals();
          n3=combustibles.get(2).getNom();
          d4=combustibles.get(3).getLitresTotals();
          n4=combustibles.get(3).getNom();

          Double max = 0.0;
          if(d1>d2 && d1>d3 && d1>d4){
                    max=d1;
                    nmax=n1;
          } else if(d2>d1 && d2>d3 && d2>d4){
                    max=d2;
                    nmax=n2;
          } else if(d3>d1 && d3>d2 && d3>d4){
                    max=d3;
                    nmax=n3;
          } else if(d4>d1 && d4>d2 && d4>d3){
                    max=d4;
                    nmax=n4;
          }

          Double min = 0.0;
          if(d1<d2 && d1<d3 && d1<d4){
                    min=d1;
                    nmin=n1;
          } else if(d2<d1 && d2<d3 && d2<d4){
                    min=d2;
                    nmin=n2;
          } else if(d3<d1 && d3<d2 && d3<d4){
                min=d3;
                nmin=n3;
            } else if(d4<d1 && d4<d2 && d4<d3){
                min=d4;
                nmin=n4;
            }
            String s = ("Litres Maxims: "+nmax.toString()+"  "+max.toString()+" Litres\n"+
                        "Litres Minims: "+nmin.toString()+"  "+min.toString()+" Litres");
            System.out.println(s);
            //JOptionPane.showMessageDialog(null,s);
            return s;
    }
    /**
     * @param sortidors
     * @return
     */
    public String getDepositCap(List<Sortidor> sortidors)
    {
            Double t1, t2, t3, t4, tt = null;
            String n1, n2, n3, n4 = null, nn=null;
            t1=sortidors.get(0).getTancLits();
            t2=sortidors.get(1).getTancLits();
            t3=sortidors.get(2).getTancLits();
            t4=sortidors.get(3).getTancLits();
            n1=sortidors.get(0).getNom();
            n2=sortidors.get(1).getNom();
            n3=sortidors.get(2).getNom();
            n3=sortidors.get(3).getNom();
            if(t1<t2 && t1<t3 && t1<t4){
                tt= t1;
                nn=n1;
            }else if(t2<t1 && t2<t3 && t2<t3){
                tt=t2;
                nn=n2;
            }else if(t3<t1 && t3<t2 && t3<t4){
                tt=t3;
                nn=n3;
            }else if(t4<t1 && t4<t2 && t4<t3){
                tt=t4;
                nn=n4;
            }
            double totcom = 10000.00-tt;
            System.out.println("El Deposit amb menos litres es: "+nn+"  "+tt.toString()+"
Litres"+"\n"
            +"La comanda actual es de: "+totcom);
            return "El Deposit amb menos litres es: "+nn+"  "+tt.toString()+" Litres"+"\n"
```

```
                        +"La comanda actual es de: "+totcom;
}

/**
 * @param combustibles
 * @return
 */
public String sortCombustibles(List<Combustible> combustibles)
{
        int c1 = 0, c2 = 0, c3 = 0, c4 = 0;
        String n1 = null, n2 = null, n3 = null, n4 = null;
        for(Combustible c : combustibles)
        {
                if(c.getGrauContaminacio()==4){
                        c1=c.getGrauContaminacio();
                        n1=c.getNom();
                }
                if(c.getGrauContaminacio()==3){
                        c2=c.getGrauContaminacio();
                        n2=c.getNom();
                }else if(c.getGrauContaminacio()==2 ){
                        c3=c.getGrauContaminacio();
                        n3=c.getNom();
                }else if(c.getGrauContaminacio()==1 ){
                        c4=c.getGrauContaminacio();
                        n4=c.getNom();
                }

        }
        System.out.println(n1+": "+c1+"\n"
                        +n2+": "+c2+"\n"
                        +n3+": "+c3+"\n"
                        +n4+": "+c4+"\n");
        return n1+": "+c1+"\n"
                        +n2+": "+c2+"\n"
                        +n3+": "+c3+"\n"
                        +n4+": "+c4+"\n";
}
/**
 * @param combustibles
 * @return
 */
public String findProvider(List<Combustible> combustibles)
{
        List<Proveiidor> lc = new ArrayList<Proveiidor>();
```

```
		String s = (JOptionPane.showInputDialog("Introdueix el Carburant(95 = T1,
98 = T2,  Biodiesel = T3, Diesel = T4)"));
		if(s==combustibles.get(0).getTipusC()){
			lc=combustibles.get(0).get_proveidors();
		}
		if(s==combustibles.get(1).getTipusC()){
			lc=combustibles.get(1).get_proveidors();
		}
		if(s==combustibles.get(2).getTipusC()){
			lc=combustibles.get(2).get_proveidors();
		}
		if(s==combustibles.get(3).getTipusC()){
			lc=combustibles.get(3).get_proveidors();
		}
		if (lc.size()==4)
		{
			System.out.println(lc.get(0).toString()+"\n\n"+lc.get(1).toString()
+"\n\n"+lc.get(2).toString()+"\n\n"+lc.get(3).toString());
			return "\n"+lc.get(0).toString()+"\n\n"+lc.get(1).toString()
+"\n\n"+lc.get(2).toString()+"\n\n"+lc.get(3).toString();
		}else{
		//System.out.println("\n"+lc.get(0).toString()+"\n\n"+lc.get(1).toString());
		return "\n"+lc.get(0).toString()+"\n\n"+lc.get(1).toString();
		}
	}

	/**
	 * @param proveiidors
	 * @return
	 */
	public String findCombustible(List<Proveiidor> proveiidors)
	{
		List<Combustible> lc = new ArrayList<Combustible>();
		String s = (JOptionPane.showInputDialog("Introdueix el Proveiidor(Hermanos
Robles SL, Faustino Agudo SL, Zarcar SL)"));
		String busc = proveiidors.get(0).getNom().trim().toLowerCase();
		String ss = s.trim().toLowerCase();
		if (busc.equals(ss)){
			lc=proveiidors.get(0).getcombustibles();
		}
		busc = proveiidors.get(1).getNom().trim().toLowerCase();
		if (busc.equals(ss)){
			lc=proveiidors.get(1).getcombustibles();
		}
		busc = proveiidors.get(2).getNom().trim().toLowerCase();
		if (busc.equals(ss)){
```

```
                    lc=proveiidors.get(2).getcombustibles();
                }
                if (lc.size()==4)
                {
                    System.out.println(lc.get(0).toString()+"\n\n"+lc.get(1).toString()
+"\n\n"+lc.get(2).toString()+"\n\n"+lc.get(3).toString());
                    return lc.get(0).toString()+"\n\n"+lc.get(1).toString()
+"\n\n"+lc.get(2).toString()+"\n\n"+lc.get(3).toString();
                }else{
                System.out.println(lc.get(0).toString()+"\n\n"+lc.get(1).toString());
                return lc.get(0).toString()+"\n\n"+lc.get(1).toString();
                }
        }
        /**
         * @param combustibles
         * @return
         */
        public String dadesCombustible(List<Combustible> combustibles)
        {
                String s = (JOptionPane.showInputDialog("Introdueix el Carburant(95 = T1,
98 = T2,  Biodiesel = T3, Diesel = T4)"));


if(combustibles.get(0).getTipusC().toLowerCase().trim().equals(s.toLowerCase().trim()))
                        {
                                System.out.println(combustibles.get(0).toString());
                                return combustibles.get(0).toString();
                        }else
if(combustibles.get(1).getTipusC().toLowerCase().trim().equals(s.toLowerCase().trim()))
                        {
                                System.out.println(combustibles.get(1).toString());
                                return combustibles.get(1).toString();
                        }else
if(combustibles.get(2).getTipusC().toLowerCase().trim().equals(s.toLowerCase().trim()))
                        {
                                System.out.println(combustibles.get(2).toString());
                                return combustibles.get(2).toString();
                        }else
if(combustibles.get(3).getTipusC().toLowerCase().trim().equals(s.toLowerCase().trim()))
                        {
                                System.out.println(combustibles.get(3).toString());
                                return combustibles.get(3).toString();
                        }
                return "No s'ha trobat la busqueda: "+s;
                }
        }
```

Proveiidor.java

```
package ProcessData;
import java.util.ArrayList;
import java.util.List;

/**
 * @author Begoña
 *
 */
public class Proveiidor {
	public Proveiidor() {
	}

	/**
	 * @param nom
	 */
	public Proveiidor(String nom) {
		this.nom = nom;
		this.setDescripcio(nom);
	}

	private String nom;
	private String descripcio;
	private List<Combustible> _combustibles = new ArrayList<Combustible>();

	/**
	 * @return
	 */
	public String getNom() {
		return nom;
	}

	/**
	 * @param nom
	 */
	public void setNom(String nom) {
		this.nom = nom;
	}

	/**
	 * @return
	 */
	public List<Combustible> getcombustibles() {
		return _combustibles;
	}

	/**
	 * @param combustible
	 */
	public void addCombustible(Combustible combustible) {
		this._combustibles.add(combustible);
	}

	/**
	 * @return
	 */
	public String getDescripcio() {
		return descripcio;
	}
	/**
	 * @param nom
	 */
	public void setDescripcio(String nom) {
		if (nom.equals("Hermanos Robles SL")){
			this.descripcio = "La comercializacion, venta, distribucion y transporte
de combustibles, carburantes, lubricantes y cualesquiera otros productos derivados del
petroleo.";
		}
		else if (nom.equals("Faustino Agudo SL")){
			this.descripcio = "Comercio al por menor de combustibles,
carburantes y lubricantes.";
		}
		else if(nom.equals("Zarcar SL")){
			this.descripcio = "Comercio al por menor de combustibles,
carburantes y lubricantes.";
		}
	}
	/**
	 * @return
	 */
	public String combustibleToString()
	{
		String s=null;
		for(int i=0;i<this._combustibles.size();i++)
		{
			s+= "\n"+this._combustibles.get(i).getNom()+"
"+this._combustibles.get(i).getTipusC()+"
"+this._combustibles.get(i).getGrauContaminacio()+"  "+this._combustibles.get(i).getTanc();
		}
		return s;
	}

	/* (non-Javadoc)
	 * @see java.lang.Object#toString()
	 */
	@Override
	public String toString() {
		return "Proveiidor [nom=" + nom + ", descripcio=" + descripcio
+combustibleToString()+ "]";
	}
}
```

Sortidor.java

```
package ProcessData;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.List;

//import javax.swing.JOptionPane;

/**
 * @author Begoña
 *
 */
public class Sortidor {
	/**
	 * @param nom
	 * @param tanc
	 */
	public Sortidor(String nom, Tanc tanc) {
		this.nom = nom;
		this.tanc=tanc;
	}

	public Sortidor() {
	}

	private String nom;
	private List<Date> horaUs = new ArrayList<Date>();
	private List<Double> _litres = new ArrayList<Double>();
private List<Combustible> _combustibles = new ArrayList<Combustible>();
private Tanc tanc;

/**
 * @return
 */
public String getNom() {
        return nom;
}

/**
 * @param nom
 */
public void setNom(String nom) {
        this.nom = nom;
}

/**
 * @return
 */
public List<Combustible> get_combustibles() {
        return _combustibles;
}

/**
 * @param combustible
 */
public void addCombustible(Combustible combustible) {
        this._combustibles.add(combustible);
}


/**
 * @return
 */
public List<Date> getHoraUs() {
        return horaUs;
}

/**
 * @param horaUs
 */
public void setHoraUs(String horaUs)
{
        try {
        SimpleDateFormat sdf = new SimpleDateFormat("HH:mm");
```

```
			Calendar cal = Calendar.getInstance();

				cal.setTime(sdf.parse(horaUs));
			this.horaUs.add(cal.getTime());
			} catch (ParseException e) {
				e.printStackTrace();
			}
		}

		/**
		 * @return
		 */
		public List<Double> getLitres() {
			return _litres;
		}

		/**
		 * @param _litres
		 */
		public void addLitres(Double _litres) {
			this._litres.add(_litres);
		}
		/**
		 * @return
		 */
		public Double getTancLits()
		{
			return this.tanc.getLitres();
		}
		/**
		 * @return
		 */
		public String getTancNom()
		{
			return this.tanc.getNom();
		}

}
```

Codi Android + APK’s

SintesiServei

package org.example.sintesiservei;

```
import android.content.ContentValues;
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.view.View;
import android.widget.Toast;

/**
 * Created by Begoña on 25/05/2017.
 */

public class DBHelper extends SQLiteOpenHelper {
   public DBHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int
version) {
       super(context, name, factory, version);
   }
   @Override
   public void onCreate(SQLiteDatabase db) {
       db.execSQL("create table servs(sortidor int ,tipusc text,hora text, litres text)");
   }


   @Override
   public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

   }

}
```

Servei.java

```
package org.example.sintesiservei;

import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.provider.BaseColumns;

/**
 * Created by Begoña on 24/05/2017.
 */

public class Servei {
    private String NumSort;
    private String TipusC;
    private String HoraUs;
    private String Litres;

    public Servei(String numSort, String tipusC, String horaUs, String litres) {
        NumSort = numSort;
        TipusC = tipusC;
        HoraUs = horaUs;
        Litres = litres;
    }

    public String getNumSort() {
        return NumSort;
    }

    public void setNumSort(String numSort) {
        NumSort = numSort;
    }

    public String getTipusC() {
        return TipusC;
    }

    public void setTipusC(String tipusC) {
        TipusC = tipusC;
    }

    public String getHoraUs() {
        return HoraUs;
    }

    public void setHoraUs(String horaUs) {
        HoraUs = horaUs;
    }

    public String getLitres() {
        return Litres;
    }

    public void setLitres(String litres) {
        Litres = litres;
    }
    public String stringToExport(){
        return NumSort+";"+TipusC+";"+HoraUs+";"+Litres;

    }
    @Override
    public String toString() {
        return "Servei{" +
                "NumSort='" + NumSort + '\'' +
                ", TipusC='" + TipusC + '\'' +
                ", HoraUs=" + HoraUs +
                ", Litres=" + Litres +
                '}';
    }
}
```

SintesiServei.java

```
package org.example.sintesiservei;


import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;
import android.widget.TextView;
import android.widget.Toast;

import java.util.List;

public class SintesiServei extends AppCompatActivity {
    TextView tv1;
    List<Servei> ls;
    EditText et1, et2, et3, et4;
```

```
SQLiteDatabase db;
Button bmos, bsav;
ListView lv;

@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_sintesi_servei);
   bmos = (Button)findViewById(R.id.button);
   bsav= (Button)findViewById(R.id.button2);
   et1=(EditText)findViewById(R.id.editText);
   et2=(EditText)findViewById(R.id.editText2);
   et3=(EditText)findViewById(R.id.editText3);
   et4=(EditText)findViewById(R.id.editText4);
   //tv1=(TextView) findViewById(R.id.textView6);
   lv=(ListView)findViewById(R.id.listView1);

}
public void alta(View v) {
   DBHelper admin = new DBHelper(this,
          "serveis", null, 1);
   SQLiteDatabase bd = admin.getWritableDatabase();
   String sort = et1.getText().toString();
   String tipusc = et2.getText().toString();
   String hora = et3.getText().toString();
   String litres = et4.getText().toString();
   ContentValues registro = new ContentValues();
   registro.put("sortidor", sort);
   registro.put("tipusc", tipusc);
   registro.put("hora", hora);
   registro.put("litres",litres);
   bd.insert("servs", null, registro);
   bd.close();
   et1.setText("");
   et2.setText("");
   et3.setText("");
   et3.setText("");
   et4.setText("");
   Toast.makeText(this, "Se cargaron los datos del servicio",
          Toast.LENGTH_SHORT).show();
}

public void consultaporcodigo(View v) {
   DBHelper admin = new DBHelper(this,
          "serveis", null, 1);
   SQLiteDatabase bd = admin.getWritableDatabase();
```

```
        String[] columns = new String[] {"sortidor", "tipusc", "hora", "litres" };
        int[] to = new int[] {1,2,3,4};




        Cursor fila = bd.rawQuery(
                "select * from servs", null);

        SimpleCursorAdapter mAdap = new SimpleCursorAdapter(this,
R.layout.activity_sintesi_servei, fila, columns, to);
        lv.setAdapter(mAdap);

        //this.setListAdapter(mAdap);
        /*StringBuilder sb = new StringBuilder();
        int i=0;
        while (fila.getString(i) != null) {
            sb.append(fila.getString(i));
                //et2.setText(fila.getString(0));
                //et3.setText(fila.getString(1));

        }*/
        bd.close();

        //Toast.makeText(this, sb.toString(), Toast.LENGTH_LONG).show();
    }


}
```

SearchNearByPlaces

Constant.java

```
package org.example.searchnearbyplaces;

/**
 * Created by Begoña on 25/05/2017.
 */

public class Constant {
    public static String PLACE_API_BASE_URL = "https://maps.googleapis.com/maps/";
}
```

Geometry.java

```
package org.example.searchnearbyplaces;

/**
 * Created by Begoña on 25/05/2017.
 */


import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

/**
 * Created by Parth Dave on 31/3/17.
 * Spaceo Technologies Pvt Ltd.
 * parthd.spaceo@gmail.com
 */
public class Geometry {

    @SerializedName("location")
    @Expose
    private Location location;

    /**
     *
     * @return
     * The location
     */
    public Location getLocation() {
        return location;
    }

    /**
     *
     * @param location
     * The location
     */
    public void setLocation(Location location) {
        this.location = location;
    }

}
```

Location.javapackage org.example.searchnearbyplaces;

```
/**
 * Created by Begoña on 25/05/2017.
 */

import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

/**
 * Created by Parth Dave on 31/3/17.
 * Spaceo Technologies Pvt Ltd.
 * parthd.spaceo@gmail.com
 */
public class Location {

    @SerializedName("lat")
    @Expose
    private Double lat;
    @SerializedName("lng")
    @Expose
    private Double lng;

    /**
     *
     * @return
     * The lat
     */
    public Double getLat() {
        return lat;
    }

    /**
     *
     * @param lat
     * The lat
     */
    public void setLat(Double lat) {
        this.lat = lat;
    }

    /**
     *
     * @return
     * The lng
     */
    public Double getLng() {
        return lng;
    }

    /**
     *
     * @param lng
     * The lng
     */
    public void setLng(Double lng) {
        this.lng = lng;
    }

}
```

MyApplication.java

```
package org.example.searchnearbyplaces;

import android.app.Application;

import com.google.gson.ExclusionStrategy;
import com.google.gson.FieldAttributes;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.google.gson.reflect.TypeToken;

import java.util.Collection;
import java.util.concurrent.TimeUnit;

import okhttp3.OkHttpClient;
import okhttp3.logging.HttpLoggingInterceptor;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

/**
 * Created by Parth Dave on 31/3/17.
 * Spaceo Technologies Pvt Ltd.
 * parthd.spaceo@gmail.com
 */

public class MyApplication extends Application {

    NearByApi nearByApi = null;
    static MyApplication app;

    @Override
    public void onCreate() {
        super.onCreate();
        app = this;
    }


    public NearByApi getApiService() {
        if (nearByApi == null) {
            HttpLoggingInterceptor interceptor = new HttpLoggingInterceptor();
            interceptor.setLevel(HttpLoggingInterceptor.Level.BODY);
            OkHttpClient client = new
OkHttpClient.Builder().retryOnConnectionFailure(true).readTimeout(80,
TimeUnit.SECONDS).connectTimeout(80,
TimeUnit.SECONDS).addInterceptor(interceptor).build();

            Retrofit retrofit = new
Retrofit.Builder().baseUrl(Constant.PLACE_API_BASE_URL).addConverterFactory(getApiC
onvertorFactory()).client(client).build();

            nearByApi = retrofit.create(NearByApi.class);
            return nearByApi;
        } else {
            return nearByApi;
        }
    }

    private static GsonConverterFactory getApiConvertorFactory() {
        return GsonConverterFactory.create();
    }


    public static MyApplication getApp() {
        return app;
    }

}
```

NearByApi.java

```
package org.example.searchnearbyplaces;

import retrofit2.Call;
import retrofit2.http.GET;
import retrofit2.http.Query;
```

```
/**
 * Created by Begoña on 25/05/2017.
 */
public interface NearByApi {

    @GET("api/place/nearbysearch/json?
sensor=true&key=AIzaSyCe0L2pON1GBKGzTCYu6-T2d2cbt-OlHNo")
    Call<NearByApiResponse> getNearbyPlaces(@Query("type") String type,
@Query("location") String location, @Query("radius") int radius);
}
```

NearByApiResponse.java

```
package org.example.searchnearbyplaces;


import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by Parth Dave on 31/3/17.
 * Spaceo Technologies Pvt Ltd.
 * parthd.spaceo@gmail.com
 */
public class NearByApiResponse {

    @SerializedName("html_attributions")
    @Expose
    private List<Object> htmlAttributions = new ArrayList<Object>();
    @SerializedName("next_page_token")
    @Expose
    private String nextPageToken;
    @SerializedName("results")
    @Expose
    private List<Result> results = new ArrayList<Result>();
    @SerializedName("status")
    @Expose
    private String status;

    /**
     *
     * @return
     * The htmlAttributions
     */
    public List<Object> getHtmlAttributions() {
        return htmlAttributions;
    }

    /**
     *
     * @param htmlAttributions
     * The html_attributions
     */
    public void setHtmlAttributions(List<Object> htmlAttributions) {
        this.htmlAttributions = htmlAttributions;
    }

    /**
     *
     * @return
     * The nextPageToken
     */
    public String getNextPageToken() {
        return nextPageToken;
    }

    /**
     *
     * @param nextPageToken
     * The next_page_token
     */
    public void setNextPageToken(String nextPageToken) {
        this.nextPageToken = nextPageToken;
    }

    /**
     *
     * @return
     * The results
     */
    public List<Result> getResults() {
        return results;
    }

    /**
     *
     * @param results
     * The results
```

```
     */
    public void setResults(List<Result> results) {
        this.results = results;
    }

    /**
     *
     * @return
     * The status
     */
    public String getStatus() {
        return status;
    }

    /**
     *
     * @param status
     * The status
     */
    public void setStatus(String status) {
        this.status = status;
    }

}
```

OpeningHours.java

```
package org.example.searchnearbyplaces;


import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by Parth Dave on 31/3/17.
 * Spaceo Technologies Pvt Ltd.
 * parthd.spaceo@gmail.com
 */
public class OpeningHours {

    @SerializedName("open_now")
    @Expose
    private Boolean openNow;
    @SerializedName("weekday_text")
    @Expose
    private List<Object> weekdayText = new ArrayList<Object>();

    /**
     *
     * @return
     * The openNow
     */
    public Boolean getOpenNow() {
        return openNow;
    }

    /**
     *
     * @param openNow
     * The open_now
     */
    public void setOpenNow(Boolean openNow) {
        this.openNow = openNow;
    }

    /**
     *
     * @return
     * The weekdayText
     */
    public List<Object> getWeekdayText() {
        return weekdayText;
    }

    /**
     *
     * @param weekdayText
     * The weekday_text
     */
    public void setWeekdayText(List<Object> weekdayText) {
        this.weekdayText = weekdayText;
    }

}
```

Photo.java

```
package org.example.searchnearbyplaces;

import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by Parth Dave on 31/3/17.
 * Spaceo Technologies Pvt Ltd.
 * parthd.spaceo@gmail.com
 */
public class Photo {

    @SerializedName("height")
    @Expose
    private Integer height;
    @SerializedName("html_attributions")
    @Expose
    private List<String> htmlAttributions = new ArrayList<String>();
    @SerializedName("photo_reference")
    @Expose
    private String photoReference;
    @SerializedName("width")
    @Expose
    private Integer width;

    /**
     *
     * @return
     * The height
     */
    public Integer getHeight() {
        return height;
    }

    /**
     *
     * @param height
     * The height
     */
```

```
public void setHeight(Integer height) {
   this.height = height;
}

/**
 *
 * @return
 * The htmlAttributions
 */
public List<String> getHtmlAttributions() {
   return htmlAttributions;
}

/**
 *
 * @param htmlAttributions
 * The html_attributions
 */
public void setHtmlAttributions(List<String> htmlAttributions) {
   this.htmlAttributions = htmlAttributions;
}

/**
 *
 * @return
 * The photoReference
 */
public String getPhotoReference() {
   return photoReference;
}

/**
 *
 * @param photoReference
 * The photo_reference
 */
public void setPhotoReference(String photoReference) {
   this.photoReference = photoReference;
}

/**
 *
 * @return
 * The width
 */
public Integer getWidth() {
        return width;
    }

    /**
     *
     * @param width
     * The width
     */
    public void setWidth(Integer width) {
        this.width = width;
    }

}
```

Result.java

```
package org.example.searchnearbyplaces;


import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by Parth Dave on 31/3/17.
 * Spaceo Technologies Pvt Ltd.
 * parthd.spaceo@gmail.com
 */
public class Result {

    @SerializedName("geometry")
    @Expose
    private Geometry geometry;
    @SerializedName("icon")
    @Expose
    private String icon;
    @SerializedName("id")
    @Expose
    private String id;
    @SerializedName("name")
    @Expose
    private String name;
    @SerializedName("opening_hours")
```

```
@Expose
private OpeningHours openingHours;
@SerializedName("photos")
@Expose
private List<Photo> photos = new ArrayList<Photo>();
@SerializedName("place_id")
@Expose
private String placeId;
@SerializedName("rating")
@Expose
private Double rating;
@SerializedName("reference")
@Expose
private String reference;
@SerializedName("scope")
@Expose
private String scope;
@SerializedName("types")
@Expose
private List<String> types = new ArrayList<String>();
@SerializedName("vicinity")
@Expose
private String vicinity;
@SerializedName("price_level")
@Expose
private Integer priceLevel;

/**
 *
 * @return
 * The geometry
 */
public Geometry getGeometry() {
   return geometry;
}

/**
 *
 * @param geometry
 * The geometry
 */
public void setGeometry(Geometry geometry) {
   this.geometry = geometry;
}

/**
```

```
 *
 * @return
 * The icon
 */
public String getIcon() {
    return icon;
}

/**
 *
 * @param icon
 * The icon
 */
public void setIcon(String icon) {
    this.icon = icon;
}

/**
 *
 * @return
 * The id
 */
public String getId() {
    return id;
}

/**
 *
 * @param id
 * The id
 */
public void setId(String id) {
    this.id = id;
}

/**
 *
 * @return
 * The name
 */
public String getName() {
    return name;
}

/**
 *
 * @param name
 * The name
 */
public void setName(String name) {
    this.name = name;
}

/**
 *
 * @return
 * The openingHours
 */
public OpeningHours getOpeningHours() {
    return openingHours;
}

/**
 *
 * @param openingHours
 * The opening_hours
 */
public void setOpeningHours(OpeningHours openingHours) {
    this.openingHours = openingHours;
}

/**
 *
 * @return
 * The photos
 */
public List<Photo> getPhotos() {
    return photos;
}

/**
 *
 * @param photos
 * The photos
 */
public void setPhotos(List<Photo> photos) {
    this.photos = photos;
}

/**
 *
 * @return
 * The placeId
 */
public String getPlaceId() {
    return placeId;
}

/**
 *
 * @param placeId
 * The place_id
 */
public void setPlaceId(String placeId) {
    this.placeId = placeId;
}

/**
 *
 * @return
 * The rating
 */
public Double getRating() {
    return rating;
}

/**
 *
 * @param rating
 * The rating
 */
public void setRating(Double rating) {
    this.rating = rating;
}

/**
 *
 * @return
 * The reference
 */
public String getReference() {
    return reference;
}

/**
 *
 * @param reference
 * The reference
 */
public void setReference(String reference) {
    this.reference = reference;
}

/**
 *
 * @return
 * The scope
 */
public String getScope() {
    return scope;
}

/**
 *
 * @param scope
 * The scope
 */
public void setScope(String scope) {
    this.scope = scope;
}

/**
 *
 * @return
 * The types
 */
public List<String> getTypes() {
    return types;
}

/**
 *
 * @param types
 * The types
 */
public void setTypes(List<String> types) {
    this.types = types;
}

/**
 *
 * @return
 * The vicinity
 */
```

```
    public String getVicinity() {
        return vicinity;
    }

    /**
     *
     * @param vicinity
     * The vicinity
     */
    public void setVicinity(String vicinity) {
        this.vicinity = vicinity;
    }

    /**
     *
     * @return
     * The priceLevel
     */
    public Integer getPriceLevel() {
        return priceLevel;
    }

    /**
     *
     * @param priceLevel
     * The price_level
     */
    public void setPriceLevel(Integer priceLevel) {
        this.priceLevel = priceLevel;
    }

}
```

SerchNearByPlaces.java

```
package org.example.searchnearbyplaces;

import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Build;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.GoogleApiAvailability;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.location.LocationListener;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;

import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;

public class SearchNearByPlaces extends AppCompatActivity implements
OnMapReadyCallback, GoogleApiClient.ConnectionCallbacks,
GoogleApiClient.OnConnectionFailedListener,LocationListener {

    private GoogleMap googleMap;
    private GoogleApiClient mGoogleApiClient;
    private Button btnRestorentFind,btnHospitalFind;
    private LocationRequest mLocationRequest;
    private Location location;
    private int PROXIMITY_RADIUS = 8000;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_search_near_by_places);

        //To check permissions above M as below it making issue and gives permission denied
on samsung and other phones.
        if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            checkLocationPermission();
        }

        btnRestorentFind = (Button) findViewById(R.id.btnRestorentFind);
        btnHospitalFind = (Button) findViewById(R.id.btnHospitalFind);
        btnRestorentFind.setEnabled(true);
        btnHospitalFind.setEnabled(true);

        //To check google play service available
        if(!isGooglePlayServicesAvailable()){
            Toast.makeText(this,"Google Play Services not
available.",Toast.LENGTH_LONG).show();
            finish();
        }else{
            // when the map is ready to be used.
            SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager().findFragmentById(R.id.map);
            mapFragment.getMapAsync(this);

        }
    }

    private boolean isGooglePlayServicesAvailable() {
        GoogleApiAvailability googleAPI = GoogleApiAvailability.getInstance();
        int result = googleAPI.isGooglePlayServicesAvailable(this);
        if (result != ConnectionResult.SUCCESS) {
            if (googleAPI.isUserResolvableError(result)) {
                googleAPI.getErrorDialog(this, result, 0).show();
            }
            return false;
        }
        return true;
    }


    @Override
    public void onMapReady(GoogleMap googleMap) {
        /*LatLng cero = new LatLng(41.602427, 0.586540);
        googleMap.addMarker(new MarkerOptions().position(cero).title("Posicio
actual").icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_BLUE)))
;
        googleMap.moveCamera(CameraUpdateFactory.newLatLng(cero));*/

        this.googleMap = googleMap;
        googleMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
        //Initialize Google Play Services
        if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
                buildGoogleApiClient();
                googleMap.setMyLocationEnabled(true);
            }
        } else {
            buildGoogleApiClient();
            googleMap.setMyLocationEnabled(true);
        }
    }


    protected synchronized void buildGoogleApiClient() {
        mGoogleApiClient = new
GoogleApiClient.Builder(this).addConnectionCallbacks(this).addOnConnectionFailedListener
(this).addApi(LocationServices.API).build();
        mGoogleApiClient.connect();
    }

    public void onRestorentFindClick(View view){
        findPlaces("Gas station");
    }

    public void onHospitalsFindClick(View view){
        findPlaces("Hospital");
    }
//AKIII
    public void findPlaces(String placeType){
        Call<NearByApiResponse> call =
MyApplication.getApp().getApiService().getNearbyPlaces(placeType,
41.602427+","+0.586540/*location.getLatitude() + "," + location.getLongitude()*/,
PROXIMITY_RADIUS);

        call.enqueue(new Callback<NearByApiResponse>() {
            @Override
            public void onResponse(Call<NearByApiResponse> call,
Response<NearByApiResponse> response) {
                try {
                    googleMap.clear();
                    // This loop will go through all the results and add marker on each location.
                    for (int i = 0; i < response.body().getResults().size(); i++) {
                        Double lat =
response.body().getResults().get(i).getGeometry().getLocation().getLat();
```

```
                    Double lng =
response.body().getResults().get(i).getGeometry().getLocation().getLng();
                    String placeName = response.body().getResults().get(i).getName();
                    String vicinity = response.body().getResults().get(i).getVicinity();
                    MarkerOptions markerOptions = new MarkerOptions();
                    LatLng latLng = new LatLng(lat, lng);
                    // Location of Marker on Map
                    markerOptions.position(latLng);
                    // Title for Marker
                    markerOptions.title(placeName + " : " + vicinity);
                    // Color or drawable for marker

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_
AZURE));
                    // add marker
                    Marker m = googleMap.addMarker(markerOptions);
                    // move map camera
                    googleMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
                    googleMap.animateCamera(CameraUpdateFactory.zoomTo(13));
                }
            } catch (Exception e) {
                Log.d("onResponse", "There is an error");
                e.printStackTrace();
            }
        }

        @Override
        public void onFailure(Call<NearByApiResponse> call, Throwable t) {
            Log.d("onFailure", t.toString());
            t.printStackTrace();
            PROXIMITY_RADIUS += 10000;
        }
    });
}



public static final int MY_PERMISSIONS_REQUEST_LOCATION = 99;

public boolean checkLocationPermission() {
    if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {

        // Asking user if explanation is needed
```

```
          if (ActivityCompat.shouldShowRequestPermissionRationale(this,
Manifest.permission.ACCESS_FINE_LOCATION)) {

              // Show an explanation to the user *asynchronously* -- don't block
              // this thread waiting for the user's response! After the user
              // sees the explanation, try again to request the permission.

              //Prompt the user once explanation has been shown
              ActivityCompat.requestPermissions(this, new String[]
{Manifest.permission.ACCESS_FINE_LOCATION},
MY_PERMISSIONS_REQUEST_LOCATION);


          } else {
              // No explanation needed, we can request the permission.
              ActivityCompat.requestPermissions(this, new String[]
{Manifest.permission.ACCESS_FINE_LOCATION},
MY_PERMISSIONS_REQUEST_LOCATION);
          }
          return false;
      } else {
          return true;
      }
   }

   @Override
   public void onRequestPermissionsResult(int requestCode, String permissions[], int[]
grantResults) {
      switch (requestCode) {
          case MY_PERMISSIONS_REQUEST_LOCATION: {
              // If request is cancelled, the result arrays are empty.
              if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {

                  // permission was granted. Do the
                  // contacts-related task you need to do.
                  if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {

                      if (mGoogleApiClient == null) {
                          buildGoogleApiClient();
                      }
                      googleMap.setMyLocationEnabled(true);
                  }
```

```
            } else {
                Toast.makeText(this, "Location Permission has been denied, can not search the
places you want.", Toast.LENGTH_LONG).show();
            }
            return;
        }
    }
}

@Override
public void onConnected(@Nullable Bundle bundle) {
    startLocationUpdates();
}

@Override
public void onConnectionSuspended(int i) {

}

@Override
public void onConnectionFailed(@NonNull ConnectionResult connectionResult) {
    Toast.makeText(this,"Could not connect google api",Toast.LENGTH_LONG).show();
}


protected void startLocationUpdates() {
    mLocationRequest = new LocationRequest();
    mLocationRequest.setInterval(1000);
    mLocationRequest.setFastestInterval(1000);
    mLocationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
    if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
        LocationServices.FusedLocationApi.requestLocationUpdates(mGoogleApiClient,
mLocationRequest, this);
    }
}

@Override
public void onLocationChanged(Location location) {
    if(location!=null){
        this.location = location;
        if(!btnHospitalFind.isEnabled()){
            LatLng latLng = new LatLng(location.getLatitude(),location.getLongitude());
            googleMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
            googleMap.animateCamera(CameraUpdateFactory.zoomTo(15));
```

```
                btnRestorentFind.setEnabled(true);
                btnHospitalFind.setEnabled(true);
            }
        }
    }
}
```