**UNIVERSITE DE TUNIS EL MANAR**



**FACULTE DES SCIENCES DE TUNIS**

**DEPARTEMENT D’INFORMATIQUE**

Rapport Système Réparties

**Projet réalisé par :**

Dagbouj Hatem ( G02, CIN : 09927829)

(El) Ghoul Saber ( G01, CIN : 14754340)

**Année Universitaire 2016 – 2017**

**Plan**

1. Conceptions.
2. Quelques captures d’écran.
3. Paquetage.
4. Codes Source.

1. Conceptions.

RMI

Serveur UDP

Host : 127.0.0.1

Port : 2345

UDP

JAX-WS



Serveur Proxy

Host : 127.0.0.1

Port : 6000



Serveur JAX-WS

Host : 127.0.0.1

Port : 5000

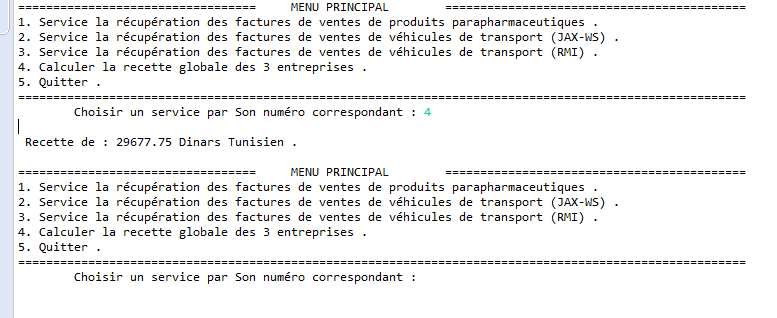
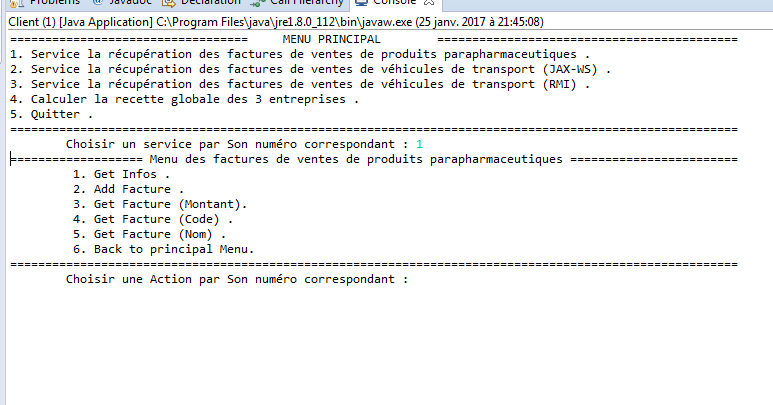


Serveur RMI

Host :127.0.0.1

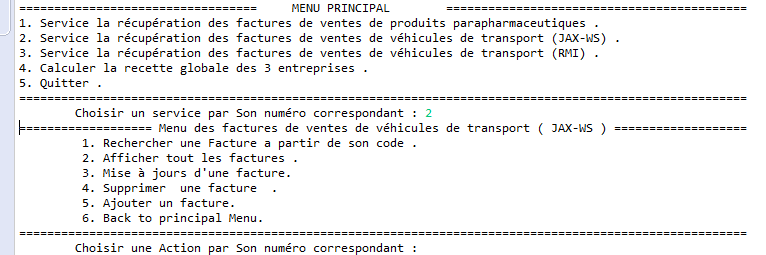
Port : 1099

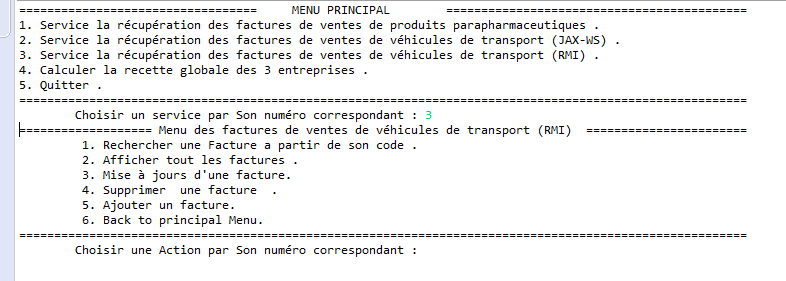
1. Quelques captures d’écran.

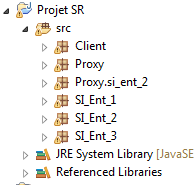
* Recette de 3 Entreprise :
* Menu 1ere Entreprise :

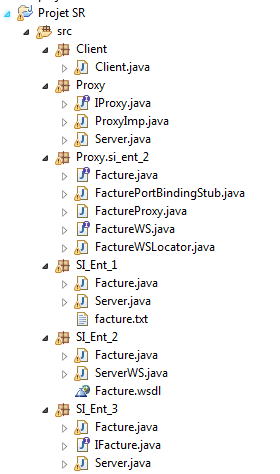
–––––

* Menu 2eme entreprise :



* Menu 3eme entreprise :

1. Paquetage.



1. Codes Source.

* Client.java

**package** Client;

**import** java.net.MalformedURLException;

**import** java.rmi.Naming;

**import** java.rmi.NotBoundException;

**import** java.rmi.RemoteException;

**import** java.util.Scanner;

**import** Proxy.IProxy;

**public** **class** Client {

**public** **static** **void** main(String[] args)

{

**int** menuSelection = 0; // choix nemu 1

**int** choix =0 ; // choix menu 2

Scanner in=**new** Scanner(System.***in***);

**while** (menuSelection != 5)

{

**if** (menuSelection==0)

// 1ere menu

menuSelection = *mainMenu*();

// if 5, exit program

**if** (menuSelection == 5) {

System.***out***.println("Quitting.");

System.*exit*(0);

}

// 2eme menu

choix =*sousMenu*(menuSelection);

/\*-------------------------------------------------------------------------

\* Récuperer l instance distant

\*/

**try** {

IProxy stub= (IProxy) Naming.*lookup*("rmi://localhost:6000/P");

//-------------------------------------------------------------------------

**if**(choix==6)

{

// ne faire rien pour revenir a menu précedant

menuSelection=0;

}

**else**

{

// SI entreprise 1

**if**(menuSelection==1)

{

**if**(choix==1)

{ // Get Infos

System.***out***.println(" --------------- Liste des Factures --------");

System.***out***.println(stub.getInfos());

System.***out***.println(" --------------------- Fin Liste -----------");

}

**if**(choix==2)

{ // Add Facture

in.nextLine();

System.***out***.println(" --------------- info du facture ----------");

System.***out***.print("\tDonner le Code :");**int** code=in.nextInt();in.nextLine();

System.***out***.print("\tDonner le Nom :");String nom =in.nextLine();

System.***out***.print("\tDonner le Montant :");**double** montant=in.nextDouble();

System.***out***.println(stub.addFacture(code, nom, montant));

System.***out***.println(" --------------------- Fin Ajout -----------");

}

**if**(choix==3)

{ // Get Facture (Montant)

System.***out***.print("\tDonner le Montant :");

**double** montant=in.nextDouble();

System.***out***.println(stub.getInfo(montant));

}

**if**(choix==4)

{ //Get Facture (Code)

System.***out***.print("\tDonner le Code :");

**int** code=in.nextInt();

System.***out***.println(stub.getInfo(code));

}

**if**(choix==5)

{ // Get Facture (Nom)

System.***out***.print("\tDonner le Nom :");

String nom=in.nextLine();

System.***out***.println(stub.getInfo(nom));

}

}

// SI entreprise 2

**if**(menuSelection==2)

{

**if**(choix==1)

{ //Rechercher une Facture a partir de son code

System.***out***.print("\n\tDonner le code du facture :");

**int** Code=in.nextInt();

System.***out***.println("\n"+stub.findFacture2(Code));

}

**if**(choix==2)

{ // Afficher tout les factures

System.***out***.println("\n"+stub.findFactures2());

}

**if**(choix==3)

{ // Mise à jours d'une facture

System.***out***.print("\n\tDonner le code du facture :");

**int** Code=in.nextInt();

**int** numero\_att=0;

**while** ((numero\_att <= 0) || (numero\_att > 5))

{

System.***out***.print("\n\t 1. Modifier le Nom du Client :");

System.***out***.print("\n\t 2. Modifier le Prénom du Client :");

System.***out***.print("\n\t 3. Modifier la Marque de Voiture :");

System.***out***.print("\n\t 4. Modifier le Numéro immatriculation :");

System.***out***.println("\n\t 5. Modifier le Prix :");

System.***out***.print("\n\t Choisir un attribut par Son numéro correspondant :");

numero\_att=in.nextInt();

}

in.nextLine();

System.***out***.print("\n\t Donner la valeur souhaitée :");

String value=in.nextLine();

**if**(stub.updateFacture2(Code, numero\_att, value))

System.***out***.println("\n\t Modification avec Succée .\n");

**else**

System.***out***.println("\n\t Echec de Modification .\n");

}

**if**(choix==4)

{ //Supprimer une facture

System.***out***.print("\n\tDonner le code du facture :");

**int** Code=in.nextInt();

**if** (stub.deleteFacture2(Code))

System.***out***.println("\n\t Suppression avec Succée .\n");

**else**

System.***out***.println("\n\t Echec de Suppression .\n");

}

**if**(choix==5)

{ // Ajouter un facture

in.nextLine();

System.***out***.print("\n\t Donner le Nom du Client :"); String NomC=in.nextLine();

System.***out***.print("\t Donner le Prénom du Client :"); String PrenomC=in.nextLine();

System.***out***.print("\t Donner la Marque de Voiture :"); String marque=in.nextLine();

System.***out***.print("\t Donner le Numéro immatriculation :"); String num\_immat=in.nextLine();

System.***out***.print("\t Donner le Prix :"); String prix=in.nextLine();

**if**( stub.addFacture2(NomC, PrenomC, marque, num\_immat, prix))

System.***out***.println("\n\t Ajout avec Succée .\n");

**else**

System.***out***.println("\n\t Echec lors de l'ajout .\n");

}

}

// SI entreprise 3

**if**(menuSelection==3)

{

**if**(choix==1)

{ // Rechercher une Facture a partir de son code

System.***out***.print("\n\tDonner le code du facture :");

**int** Code=in.nextInt(); System.***out***.println("\n"+stub.findFacture(Code));

}

**if**(choix==2)

{ // Afficher tout les factures System.***out***.println("\n"+stub.findFacture());

}

**if**(choix==3)

{ // Mise à jours d'une facture

System.***out***.print("\n\tDonner le code du facture :");

**int** Code=in.nextInt();

**int** numero\_att=0;

**while** ((numero\_att <= 0) || (numero\_att > 5))

{

System.***out***.print("\n\t 1. Modifier le Nom du Client :");

System.***out***.print("\n\t 2. Modifier le Prénom du Client :");

System.***out***.print("\n\t 3. Modifier la Marque de Voiture :");

System.***out***.print("\n\t 4. Modifier le Numéro immatriculation :");

System.***out***.println("\n\t 5. Modifier le Prix :");

System.***out***.print("\n\t Choisir un attribut par Son numéro correspondant :");

numero\_att=in.nextInt();

}

in.nextLine();

System.***out***.print("\n\t Donner la valeur souhaitée :");

String value=in.nextLine();

**if**(stub.updateFacture(Code, numero\_att, value))

System.***out***.println("\n\t Modification avec Succée .\n");

**else**

System.***out***.println("\n\t Echec de Modification .\n");

}

**if**(choix==4)

{ //Supprimer une facture

System.***out***.print("\n\tDonner le code du facture :");

**int** Code=in.nextInt();

**if** (stub.deleteFacture(Code))

System.***out***.println("\n\t Suppression avec Succée .\n");

**else**

System.***out***.println("\n\t Echec de Suppression .\n");

}

**if**(choix==5)

{ // Ajouter un facture

in.nextLine();

System.***out***.print("\n\t Donner le Nom du Client :"); String NomC=in.nextLine();

System.***out***.print("\t Donner le Prénom du Client :"); String PrenomC=in.nextLine();

System.***out***.print("\t Donner la Marque de Voiture :"); String marque=in.nextLine();

System.***out***.print("\t Donner le Numéro immatriculation :"); String num\_immat=in.nextLine();

System.***out***.print("\t Donner le Prix :"); String prix=in.nextLine();

**if**( stub.addFacture(NomC, PrenomC, marque, num\_immat, prix))

System.***out***.println("\n\t Ajout avec Succée .\n");

**else**

System.***out***.println("\n\t Echec lors de l'ajout .\n");

}

}

// Claculer la recette des 3 entreprise

**if**(menuSelection==4)

{

System.***out***.println("\n Recette de : "+stub.calculRecette()+" Dinars Tunisien .\n");

// revenir a menu principal

menuSelection=0;

}

//------------------------------------------ end Catch ----------------------------------------

}

} **catch** (MalformedURLException | RemoteException | NotBoundException e)

{

System.***out***.println(e.getMessage());

}

}

in.close();

}

//----------------------------- End Main ---------------------------------------------------

**public** **static** **int** mainMenu() {

**int** menuSelection = 0;

// loop (and prompt again) until the user's input is an integer between 1 and 8

**while** ((menuSelection <= 0) || (menuSelection > 5)) {

System.***out***.println("================================== MENU PRINCIPAL ===========================================");

System.***out***.println( "1. Service la récupération des factures de ventes de produits parapharmaceutiques .\n"

+"2. Service la récupération des factures de ventes de véhicules de transport (JAX-WS) .\n"

+"3. Service la récupération des factures de ventes de véhicules de transport (RMI) .\n"

+"4. Calculer la recette globale des 3 entreprises .\n"

+"5. Quitter ." );

System.***out***.println("========================================================================================================");

System.***out***.print("\tChoisir un service par Son numéro correspondant : ");

Scanner sc = **new** Scanner(System.***in***);

**if** (sc.hasNextInt()) menuSelection = sc.nextInt();

}

**return** menuSelection;

}

**public** **static** **int** sousMenu(**int** menuSelection ) {

**int** choix =0;

Scanner sc = **new** Scanner(System.***in***);

**if**(menuSelection==1)

{

**while** ((choix <= 0) || (choix > 6)) {

System.***out***.println("=================== Menu des factures de ventes de produits parapharmaceutiques ========================");

System.***out***.println( "\t 1. Get Infos . \n"

+"\t 2. Add Facture .\n"

+"\t 3. Get Facture (Montant).\n"

+"\t 4. Get Facture (Code) .\n"

+"\t 5. Get Facture (Nom) .\n"

+"\t 6. Back to principal Menu. " );

System.***out***.println("========================================================================================================");

System.***out***.print("\tChoisir une Action par Son numéro correspondant : ");

**if** (sc.hasNextInt()) choix = sc.nextInt();

}

}

**if**(menuSelection==2)

{

**while** ((choix <= 0) || (choix > 6)) {

System.***out***.println("=================== Menu des factures de ventes de véhicules de transport ( JAX-WS ) ===================");

System.***out***.println( "\t 1. Rechercher une Facture a partir de son code . \n"

+"\t 2. Afficher tout les factures .\n"

+"\t 3. Mise à jours d'une facture.\n"

+"\t 4. Supprimer une facture .\n"

+"\t 5. Ajouter un facture.\n"

+"\t 6. Back to principal Menu. " );

System.***out***.println("========================================================================================================");

System.***out***.print("\tChoisir une Action par Son numéro correspondant : ");

**if** (sc.hasNextInt()) choix = sc.nextInt();

}

}

**if**(menuSelection==3)

{

**while** ((choix <= 0) || (choix > 6)) {

System.***out***.println("=================== Menu des factures de ventes de véhicules de transport (RMI) =======================");

System.***out***.println( "\t 1. Rechercher une Facture a partir de son code . \n"

+"\t 2. Afficher tout les factures .\n"

+"\t 3. Mise à jours d'une facture.\n"

+"\t 4. Supprimer une facture .\n"

+"\t 5. Ajouter un facture.\n"

+"\t 6. Back to principal Menu. " );

System.***out***.println("========================================================================================================");

System.***out***.print("\tChoisir une Action par Son numéro correspondant : ");

**if** (sc.hasNextInt()) choix = sc.nextInt();

}

}

**return** choix;

}

}

**IProxy.java**

**package** Proxy;

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.FileInputStream;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.InputStreamReader;

**import** java.io.StringWriter;

**import** java.rmi.Remote;

**import** java.rmi.RemoteException;

**import** javax.jws.WebParam;

**public** **interface** IProxy **extends** Remote

{

/\*

\* 1ere entreprise

\*/

// ajouter un facture

String addFacture(**int** code,String nom , **double** montant) **throws** RemoteException;

// listter tous les facture

String getInfos()**throws** RemoteException;

// rechercher une facture par son code

String getInfo(**int** code ) **throws** RemoteException;

// rechercher une facture par son : Nom

String getInfo(String nom )**throws** RemoteException;

// rechercher une facture par son : Montant

String getInfo(**double** montant )**throws** RemoteException;

// Claculer la recette de le 1ere entreprise

**double** getReccete() **throws** RemoteException;

/\*

\* 2eme entreprise

\*/

// ajouter une facture

**boolean** addFacture2(String NomC, String PrenomC, String marque, String num\_immat, String prix) **throws** RemoteException;

// listter tous les facture

String findFactures2() **throws** RemoteException;

// rechercher une facture par son code

String findFacture2(**int** Code) **throws** RemoteException;

// mise à jours d'une facture

**boolean** updateFacture2(**int** Code, **int** numero\_att , String value) **throws** RemoteException;

// supprimer une facture

**boolean** deleteFacture2(**int** Code) **throws** RemoteException;

/\*

\* 3eme entreprise

\*/

// ajouter un facture

**boolean** addFacture(String NomC, String PrenomC, String marque ,

String num\_immat , String prix) **throws** RemoteException ;

//rechercher une Facture a partir de son code

String findFacture(**int** Code )**throws** RemoteException;

//afficher tout les facture

String findFacture( )**throws** RemoteException;

// mise à jours d'une facture

**boolean** updateFacture(**int** Code , **int** numero\_att , String value)**throws** RemoteException;

// supprimer une facture

**boolean** deleteFacture(**int** Code)**throws** RemoteException;

// Calculer la recette de 2 eme & eme entreprise

**double** calculRecette() **throws** RemoteException;

}

**ProxyImp.java**

**package** Proxy;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.Inet4Address;

**import** java.net.InetAddress;

**import** java.net.Socket;

**import** java.rmi.Naming;

**import** java.rmi.RemoteException;

**import** java.rmi.server.UnicastRemoteObject;

**import** java.util.concurrent.Callable;

**import** java.util.concurrent.atomic.AtomicLong;

**import** SI\_Ent\_3.IFacture;

**import** Proxy.si\_ent\_2.\*;

**public** **class** ProxyImp **extends** UnicastRemoteObject **implements** IProxy {

//public static FactureWS facture;

**public** **static** String *hostName* = "localhost";

Socket socket = **null**;

**public** **static** Facture *stub2*; // stub entreprise 2

**public** **static** IFacture *stub*; // stub entreprise 3

**protected** ProxyImp() **throws** RemoteException

{

**super**();

**try**

{

// serveur enteprise 2

FactureWS facture = **new** FactureWSLocator();

*stub2*= facture.getFacturePort();

// serveur entreprise 3

*stub* = (IFacture) Naming.*lookup*("rmi://localhost:1099/F");

}

**catch** ( Exception e)

{

System.***out***.println(e.getMessage());

}

}

/\*

\* 1ere entreprise

\*/

@Override

**public** String addFacture(**int** code, String nom, **double** montant) **throws** RemoteException {

**try** {

//on initialise la connexion

DatagramSocket client =**new** DatagramSocket();

//on crée notre Datgramme

InetAddress adresse= Inet4Address.*getByName*("127.0.0.1");

**byte**[] buffer\_commande= **new** String("1").getBytes() ;

DatagramPacket packet\_commande= **new** DatagramPacket(buffer\_commande , buffer\_commande.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_commande);

// envoiyer aux serveur

client.send(packet\_commande);

//1ere argument

**byte**[] buffer\_arg1= **new** String(String.*valueOf*(code)).getBytes() ;

DatagramPacket packet\_arg1= **new** DatagramPacket(buffer\_arg1 , buffer\_arg1.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_arg1);

// envoiyer aux serveur

client.send(packet\_arg1);

//2eme argument

**byte**[] buffer\_arg2= **new** String(nom).getBytes() ;

DatagramPacket packet\_arg2= **new** DatagramPacket(buffer\_arg2 , buffer\_arg2.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_arg2);

// envoiyer aux serveur

client.send(packet\_arg2);

//3eme argument

**byte**[] buffer\_arg3= **new** String(String.*valueOf*(montant)).getBytes() ;

DatagramPacket packet\_arg3= **new** DatagramPacket(buffer\_arg3 , buffer\_arg3.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_arg3);

// envoiyer aux serveur

client.send(packet\_arg3);

//récupére de la reponce du serveur

**byte** [] buffer\_responce = **new** **byte**[1024];

DatagramPacket packet\_responce= **new** DatagramPacket(buffer\_responce , buffer\_responce.length , adresse ,2345);

client.receive(packet\_responce);

**return**(**new** String(buffer\_responce).trim());

} **catch** (Exception e) {

// affichage du exception

System.***out***.println(e.getMessage());

}

**return** **null**;

}

@Override

**public** String getInfos() **throws** RemoteException {

**try** {

//on initialise la connexion

DatagramSocket client =**new** DatagramSocket();

//on crée notre Datgramme

InetAddress adresse= Inet4Address.*getByName*("127.0.0.1");

**byte**[] buffer\_commande= **new** String("2").getBytes() ;

DatagramPacket packet\_commande= **new** DatagramPacket(buffer\_commande , buffer\_commande.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_commande);

// envoiyer aux serveur

client.send(packet\_commande);

//récupére de la reponce du serveur

**byte** [] buffer\_responce = **new** **byte**[1024];

DatagramPacket packet\_responce= **new** DatagramPacket(buffer\_responce , buffer\_responce.length , adresse ,2345);

client.receive(packet\_responce);

**return**(**new** String(buffer\_responce).trim());

} **catch** (Exception e) {

// affichage du exception

System.***out***.println(e.getMessage());

}

**return** **null** ;

}

@Override

**public** String getInfo(**int** code) **throws** RemoteException {

**try** {

//on initialise la connexion

DatagramSocket client =**new** DatagramSocket();

//on crée notre Datgramme

InetAddress adresse= Inet4Address.*getByName*("127.0.0.1");

**byte**[] buffer\_commande= **new** String("3").getBytes() ;

DatagramPacket packet\_commande= **new** DatagramPacket(buffer\_commande , buffer\_commande.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_commande);

// envoiyer aux serveur

client.send(packet\_commande);

**byte**[] buffer\_arg1= **new** String(String.*valueOf*(code)).getBytes() ;

DatagramPacket packet\_arg1= **new** DatagramPacket(buffer\_arg1 , buffer\_arg1.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_arg1);

// envoiyer aux serveur

client.send(packet\_arg1);

//récupére de la reponce du serveur

**byte** [] buffer\_responce = **new** **byte**[1024];

DatagramPacket packet\_responce= **new** DatagramPacket(buffer\_responce , buffer\_responce.length , adresse ,2345);

client.receive(packet\_responce);

**return**(**new** String(buffer\_responce).trim());

} **catch** (Exception e) {

// affichage du exception

System.***out***.println(e.getMessage());

}

**return** **null**;

}

@Override

**public** String getInfo(String nom) **throws** RemoteException {

**try** {

//on initialise la connexion

DatagramSocket client =**new** DatagramSocket();

//on crée notre Datgramme

InetAddress adresse= Inet4Address.*getByName*("127.0.0.1");

**byte**[] buffer\_commande= **new** String("4").getBytes() ;

DatagramPacket packet\_commande= **new** DatagramPacket(buffer\_commande , buffer\_commande.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_commande);

// envoiyer aux serveur

client.send(packet\_commande);

**byte**[] buffer\_arg1= **new** String(nom).getBytes() ;

DatagramPacket packet\_arg1= **new** DatagramPacket(buffer\_arg1 , buffer\_arg1.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_arg1);

// envoiyer aux serveur

client.send(packet\_arg1);

//récupére de la reponce du serveur

**byte** [] buffer\_responce = **new** **byte**[1024];

DatagramPacket packet\_responce= **new** DatagramPacket(buffer\_responce , buffer\_responce.length , adresse ,2345);

client.receive(packet\_responce);

**return**(**new** String(buffer\_responce).trim());

} **catch** (Exception e) {

// affichage du exception

System.***out***.println(e.getMessage());

}

**return** **null**;

}

@Override

**public** String getInfo(**double** montant) **throws** RemoteException {

**try** {

//on initialise la connexion

DatagramSocket client =**new** DatagramSocket();

//on crée notre Datgramme

InetAddress adresse= Inet4Address.*getByName*("127.0.0.1");

**byte**[] buffer\_commande= **new** String("5").getBytes() ;

DatagramPacket packet\_commande= **new** DatagramPacket(buffer\_commande , buffer\_commande.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_commande);

// envoiyer aux serveur

client.send(packet\_commande);

**byte**[] buffer\_arg1= **new** String(String.*valueOf*(montant)).getBytes() ;

DatagramPacket packet\_arg1= **new** DatagramPacket(buffer\_arg1 , buffer\_arg1.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_arg1);

// envoiyer aux serveur

client.send(packet\_arg1);

//récupére de la reponce du serveur

**byte** [] buffer\_responce = **new** **byte**[1024];

DatagramPacket packet\_responce= **new** DatagramPacket(buffer\_responce , buffer\_responce.length , adresse ,2345);

client.receive(packet\_responce);

**return**(**new** String(buffer\_responce).trim());

} **catch** (Exception e) {

// affichage du exception

System.***out***.println(e.getMessage());

}

**return** **null**;

}

@Override

**public** **double** getReccete() **throws** RemoteException {

**try** {

//on initialise la connexion

DatagramSocket client =**new** DatagramSocket();

//on crée notre Datgramme

InetAddress adresse= Inet4Address.*getByName*("127.0.0.1");

**byte**[] buffer\_commande= **new** String("6").getBytes() ;

DatagramPacket packet\_commande= **new** DatagramPacket(buffer\_commande , buffer\_commande.length , adresse ,2345);

// mettres les donnée dans le Datagramme à envoiyé

packet\_commande.setData(buffer\_commande);

// envoiyer aux serveur

client.send(packet\_commande);

//récupére de la reponce du serveur

**byte** [] buffer\_responce = **new** **byte**[1024];

DatagramPacket packet\_responce= **new** DatagramPacket(buffer\_responce , buffer\_responce.length , adresse ,2345);

client.receive(packet\_responce);

**return** Double.*parseDouble*(**new** String(buffer\_responce).trim());

} **catch** (Exception e) {

// affichage du exception

System.***out***.println(e.getMessage());

}

**return** 0;

}

/\*

\* 2eme entreprise

\*/

// ajouter une facture

@Override

**public** **boolean** addFacture2(String NomC, String PrenomC, String marque, String num\_immat, String prix)**throws** RemoteException {

**return** *stub2*.addFacture(NomC, PrenomC, marque, num\_immat, prix);

}

// listter tous les facture

@Override

**public** String findFactures2() **throws** RemoteException {

**return** *stub2*.afficher();

}

// rechercher une facture par son code

@Override

**public** String findFacture2(**int** Code) **throws** RemoteException {

**return** *stub2*.findFacture(Code);

}

// mise à jours d'une facture

@Override

**public** **boolean** updateFacture2(**int** Code, **int** numero\_att, String value) **throws** RemoteException {

**return** *stub2*.updateFacture(Code, numero\_att, value);

}

// supprimer une facture

@Override

**public** **boolean** deleteFacture2(**int** Code) **throws** RemoteException {

**return** *stub2*.deleteFacture(Code);

}

// 3eme entreprise

@Override

**public** **boolean** addFacture(String NomC, String PrenomC, String marque, String num\_immat, String prix) **throws** RemoteException {

**return** *stub*.addFacture(NomC, PrenomC, marque, num\_immat, prix);

}

@Override

**public** String findFacture(**int** Code) **throws** RemoteException {

**return** *stub*.findFacture(Code);

}

@Override

**public** String findFacture() **throws** RemoteException {

**return** *stub*.findFacture();

}

@Override

**public** **boolean** updateFacture(**int** Code, **int** numero\_att, String value) **throws** RemoteException {

**return** *stub*.updateFacture(Code, numero\_att, value);

}

@Override

**public** **boolean** deleteFacture(**int** Code) **throws** RemoteException {

**return** *stub*.deleteFacture(Code);

}

@Override

**public** **double** calculRecette() **throws** RemoteException {

**double** montantSE1 = getReccete();

**return** (*stub*.calculRecette()+montantSE1);

}

}

**Proxy : server.java**

**package** Proxy;

**import** java.rmi.Naming;

**import** java.rmi.registry.LocateRegistry;

**import** java.rmi.registry.Registry;

**import** SI\_Ent\_3.Facture;

**public** **class** Server {

**public** **static** **void** main(String[] args)

{

**try** {

//System.setProperty("java.rmi.server.hostname","localhost");

Registry reg= LocateRegistry.*createRegistry*(6000);

ProxyImp p=**new** ProxyImp();

Naming.*rebind*("rmi://localhost:6000/P", p);

System.***out***.println("\t Proxy : Attente des invocations des clients sur le port 6000 ");

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

}

}

}

SI\_Ent1

Facture.java

**package** SI\_Ent\_1;

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.io.InputStreamReader;

**import** java.io.StringWriter;

**import** java.util.Scanner;

**import** java.util.StringTokenizer;

**import** java.util.Vector;

**public** **class** Facture {

// Constructor ---> ajouter un facture

/\* code --> 3 Octets

\* nom --> 10 Octets

\* montant --> 5 Octets

\*/

**public** **static** String addFacture(**int** code,String nom , **double** montant) {

BufferedWriter bw = **null**;

FileWriter fw = **null**;

**try** {

String ligne = "Code:"+*getFixedLengthString*(String.*valueOf*(code),3)+"Nom:"+*getFixedLengthString*(String.*valueOf*(nom),10)+"Montant:"+*getFixedLengthString*(String.*valueOf*(montant),5)+"\n";

fw = **new** FileWriter("C:\\Users\\acer\\work\_space\\Projet SR\\src\\SI\_Ent\_1\\facture.txt", **true** );

bw = **new** BufferedWriter(fw);

bw.write(ligne);

**return** " Successfully added\n";

} **catch** (IOException e) { System.***out***.println(e.getMessage()); **return** " Error \n";}

**finally** {

**try** {

bw.close();

fw.close();

}

**catch** (IOException e) { System.***out***.println(e.getMessage()); }

}

}

// listter tous les facture

**public** **static** String getInfos()

{ StringWriter out = **new** StringWriter();

**try**{

InputStream flux=**new** FileInputStream("C:\\Users\\acer\\work\_space\\Projet SR\\src\\SI\_Ent\_1\\facture.txt");

InputStreamReader lecture=**new** InputStreamReader(flux);

BufferedReader buff=**new** BufferedReader(lecture);

String ligne;

**while** ((ligne=buff.readLine())!=**null**)

{

out.write(ligne+"\n");

}

buff.close();

}

**catch** (Exception e){

System.***out***.println(e.toString());

}

**return** out.toString();

}

// rechercher une facture par son code

**public** **static** String getInfo(**int** code )

{

StringWriter out = **new** StringWriter();

**int** code\_ligne;

**try**{

InputStream flux=**new** FileInputStream("C:\\Users\\acer\\work\_space\\Projet SR\\src\\SI\_Ent\_1\\facture.txt");

InputStreamReader lecture=**new** InputStreamReader(flux);

BufferedReader buff=**new** BufferedReader(lecture);

String ligne;

**while** ((ligne=buff.readLine())!=**null**)

{

// tester sur le code

code\_ligne=Integer.*parseInt*(ligne.substring(5,8).trim());

**if**(code\_ligne==code) out.write(ligne+"\n");

}

buff.close();

}

**catch** (Exception e){

System.***out***.println(e.toString());

}

**return** out.toString();

}

// rechercher une facture par son : Nom

**public** **static** String getInfo(String nom )

{

StringWriter out = **new** StringWriter();

String nom\_ligne;

**try**{

InputStream flux=**new** FileInputStream("C:\\Users\\acer\\work\_space\\Projet SR\\src\\SI\_Ent\_1\\facture.txt");

InputStreamReader lecture=**new** InputStreamReader(flux);

BufferedReader buff=**new** BufferedReader(lecture);

String ligne;

**while** ((ligne=buff.readLine())!=**null**)

{

// tester sur le code

nom\_ligne=ligne.substring(12,22).trim();

**if**(nom\_ligne.equals(nom)) out.write(ligne+"\n");

}

buff.close();

}

**catch** (Exception e){

System.***out***.println(e.toString());

}

**return** out.toString();

}

// rechercher une facture par son : Montant

**public** **static** String getInfo(**double** montant )

{

StringWriter out = **new** StringWriter();

**double** montant\_ligne;

**try**{

InputStream flux=**new** FileInputStream("C:\\Users\\acer\\work\_space\\Projet SR\\src\\SI\_Ent\_1\\facture.txt");

InputStreamReader lecture=**new** InputStreamReader(flux);

BufferedReader buff=**new** BufferedReader(lecture);

String ligne;

**while** ((ligne=buff.readLine())!=**null**)

{

// tester sur le code

montant\_ligne=Double.*parseDouble*(ligne.substring(30).trim());

**if**(montant\_ligne==montant) out.write(ligne+"\n");

}

buff.close();

}

**catch** (Exception e){

System.***out***.println(e.toString());

}

**return** out.toString();

}

// Calculer la recette de 1ere entreprise

**public** **static** **double** getRecette()

{

**double** recette =0;

StringWriter out = **new** StringWriter();

**double** montant\_ligne;

**try**{

InputStream flux=**new** FileInputStream("C:\\Users\\acer\\work\_space\\Projet SR\\src\\SI\_Ent\_1\\facture.txt");

InputStreamReader lecture=**new** InputStreamReader(flux);

BufferedReader buff=**new** BufferedReader(lecture);

String ligne;

**while** ((ligne=buff.readLine())!=**null**)

{

// lire le montant d'une ligne

montant\_ligne=Double.*parseDouble*(ligne.substring(30).trim());

// ajouter montant à la recette

recette+=montant\_ligne;

}

buff.close();

}

**catch** (Exception e){

System.***out***.println(e.toString());

}

**return** recette ;

}

// Fixer la taille d'un champs Du fichier Plat

**public** **static** String getFixedLengthString(String pString, **int** length)

{

**if** (pString.length()==length) **return** pString;

**else** **if** (pString.length()<length)

{

String format = "%1$-"+String.*valueOf*(length)+"s";

String o = String.*format*(format, **new** String[]{pString}).replace(' ', ' ');

**return** o;

}

**else**

{

**return** pString.substring(0,length);

}

}

}

Udp : server.java

**package** SI\_Ent\_1;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.SocketException;

**public** **class** Server {

**public** **static** **void** main(String [] args )

{

**try** {

// création d'une connexion coté Serveur , en spécifiant un prort d'écoute

DatagramSocket server = **new** DatagramSocket(2345);

System.***out***.println(" UDP : Attente des Clients Sur le port : 2345");

// écoute

**while**(**true**)

{

// on s'occupe maintenant de l'objet paquet

**byte** [] buffer = **new** **byte**[1024];

DatagramPacket paquet = **new** DatagramPacket(buffer, buffer.length);

// récuperer le datagramme envoiyé par le client

server.receive(paquet);

// récuperer le contenu

String msg = **new** String(paquet.getData());

// afficher le contenue de "Commande " == MSG

**int** commande =Integer.*parseInt*(msg.trim());

String[] arguments = **null** ;

// lecture des arguments selon la commande

**switch** (commande )

{

**case** 1 :{ // addFacture - 3 arguments

arguments = **new** String[3];

// rénisialiser la taille du datagramme , pour les future datagramme

paquet.setLength(buffer.length);

// recive 1ere arguments

server.receive(paquet);

arguments[0]=**new** String(paquet.getData()).trim();

// rénisialiser la taille du datagramme , pour les future datagramme

paquet.setLength(buffer.length);

// recive 2eme arguments

server.receive(paquet);

arguments[1]=**new** String(paquet.getData()).trim();

// rénisialiser la taille du datagramme , pour les future datagramme

paquet.setLength(buffer.length);

// recive 3eme arguments

server.receive(paquet);

arguments[2]=**new** String(paquet.getData()).trim();

**break** ;

}

**case** 2 :{ // getInfo - 0 arguments

**break** ;

}

**case** 3 :{ // getInfo\_Code - 1 arguments

arguments = **new** String[1];

// rénisialiser la taille du datagramme , pour les future datagramme

paquet.setLength(buffer.length);

// recive 1ere arguments

server.receive(paquet);

arguments[0]=**new** String(paquet.getData()).trim();

**break** ;

}

**case** 4 :{ // getInfo\_Nom - 1 arguments

arguments = **new** String[1];

// rénisialiser la taille du datagramme , pour les future datagramme

paquet.setLength(buffer.length);

// recive 1ere arguments

server.receive(paquet);

arguments[0]=**new** String(paquet.getData()).trim();

**break** ;

}

**case** 5 :{ // getInfo\_Montant - 1 arguments

arguments = **new** String[1];

// rénisialiser la taille du datagramme , pour les future datagramme

paquet.setLength(buffer.length);

// recive 1ere arguments

server.receive(paquet);

arguments[0]=**new** String(paquet.getData()).trim();

**break** ;

}

**case** 6 :{ // getRecette - 0 arguments

**break** ;

}

}

// rénisialiser la taille du datagramme , pour les future datagramme

paquet.setLength(buffer.length);

// Preparer la Réponce du serveur

**byte**[] responce =*executeCommande*(commande,arguments).getBytes();

DatagramPacket dataResponce = **new** DatagramPacket(responce, responce.length,paquet.getAddress(),paquet.getPort());

// envoi la réponce a le client UDP

server.send(dataResponce);

}

} **catch** (Exception e) {

// afficher l'exception

System.***out***.println(e.getMessage());

}

}

// methode qui traite la demande du client et retournée le resultat

**public** **static** String executeCommande(**int** commande , String[] argument)

{

String result= **null** ;

Facture facture = **new** Facture();

// selon la commande on traite

**switch** (commande)

{

**case** 1 : {

result= **new** String(facture.*addFacture*(Integer.*parseInt*(argument[0]), argument[1], Double.*parseDouble*(argument[2]))) ;

**break** ;

}

**case** 2 : {

result= **new** String(facture.*getInfos*()) ; **break** ;

}

**case** 3 : {

result= **new** String(facture.*getInfo*(Integer.*parseInt*(argument[0]))) ;

**break** ;

}

**case** 4 : {

result= **new** String(facture.*getInfo*(argument[0])) ;

**break** ;

}

**case** 5 : {

result= **new** String(facture.*getInfo*( Double.*parseDouble*(argument[0]))) ;

**break** ;

}

**case** 6 : {

result= **new** String(String.*valueOf*(facture.*getRecette*())) ; **break** ;

}

}

**return** result;

}

}

Facture.txt 🡪 Fichier plat sur la forme :

Code:1 Nom:Adole Montant:500.0

Code:2 Nom:prod2 Montant:300.0

Code:3 Nom:prod3 Montant:400.0

Code:4 Nom:prod4 Montant:500.0

Code:5 Nom:prod5 Montant:60.05

Code:6 Nom:prod6 Montant:700.0

Code:7 Nom:prod7 Montant:800.0

Code:8 Nom:prod8 Montant:900.0

Code:9 Nom:prod9 Montant:1000.

Code:10 Nom:prod10 Montant:10.01

SI\_Ent2

Facture.java

**package** SI\_Ent\_2;

**import** java.io.StringWriter;

**import** java.rmi.RemoteException;

**import** java.sql.DriverManager;

**import** java.sql.ResultSet;

**import** javax.jws.WebMethod;

**import** javax.jws.WebParam;

**import** javax.jws.WebService;

**import** javax.xml.bind.annotation.XmlType;

@WebService (serviceName="FactureWS")

@XmlType(name="Facture", namespace="http://SI\_Ent\_2/")

**public** **class** Facture {

//rechercher une Facture a partir de son code

@WebMethod

**public** String findFacture(@WebParam(name="code") **int** Code)

{

StringWriter out = **new** StringWriter();

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="SELECT \* FROM `facture` WHERE `facture`.`id` = "+Code;;

ResultSet rs=st.executeQuery(query);

// test si le ResultSet est vide

**boolean** isEmpty=**true**;

// sinon

**while**(rs.next())

{ **if**(rs.getInt("id")==Code)

{ isEmpty=**false**;

out.append("ID :"+rs.getInt("id")+" ");

out.append("Nom :"+rs.getString("nomC")+" ");

out.append("Prenom :"+rs.getString("PrenomC")+" ");

out.append("Marque voiture :"+rs.getString("marqueV")+" ");

out.append("N° immatriculation :"+rs.getString("num\_immat")+" ");

out.append("Prix :"+rs.getString("prix")+"\n");

}

}

**if**(isEmpty) **return** " Aucune Facture correspond à le Code : "+String.*valueOf*(Code)+" .\n";

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

**return** " Erreur de Connexion .\n";

}

**return** out.toString();

}

//afficher tout les facture

@WebMethod (operationName="afficher")

**public** String findFactures()

{ StringWriter out = **new** StringWriter();

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost:3306/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="SELECT \* FROM `facture`";

ResultSet rs=st.executeQuery(query);

// test si le ResultSet est vide

**boolean** isEmpty=**true**;

// sinon

**while**(rs.next())

{ isEmpty=**false**;

out.append("ID :"+rs.getInt("id")+" ");

out.append("Nom :"+rs.getString("nomC")+" ");

out.append("Prenom :"+rs.getString("PrenomC")+" ");

out.append("Marque voiture :"+rs.getString("marqueV")+" ");

out.append("N° immatriculation :"+rs.getString("num\_immat")+" ");

out.append("Prix :"+rs.getString("prix")+"\n");

}

**if**(isEmpty) **return** " Aucune Facture .\n";

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

**return** " Erreur de Connexion .\n";

}

**return** out.toString();

}

// mise à jours d'une facture

@WebMethod

**public** **boolean** updateFacture(**int** Code, **int** numero\_att , String value)

{ String query = **null**;

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

**if**(numero\_att==1)

query="UPDATE `vente`.`facture` SET `nomC` = '"+value+"' WHERE `facture`.`id` = "+Code;

**else** **if**(numero\_att==2)

query="UPDATE `vente`.`facture` SET `PrenomC` = '"+value+"' WHERE `facture`.`id` = "+Code;

**else** **if**(numero\_att==3)

query="UPDATE `vente`.`facture` SET `marqueV` = '"+value+"' WHERE `facture`.`id` = "+Code;

**else** **if**(numero\_att==4)

query="UPDATE `vente`.`facture` SET `num\_immat` = '"+value+"' WHERE `facture`.`id` = "+Code;

**else**

query="UPDATE `vente`.`facture` SET `prix` = '"+value+"' WHERE `facture`.`id` = "+Code;

**if**( st.executeUpdate(query)>0) **return** **true**;

**else** **return** **false** ;

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

**return** **false**;

}

}

// supprimer une facture

@WebMethod

**public** **boolean** deleteFacture(**int** Code)

{

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="DELETE FROM `vente`.`facture` WHERE `facture`.`id` = "+Code;

**if**( st.executeUpdate(query)>0) **return** **true**;

**else** **return** **false** ;

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

**return** **false**;

}

}

// ajouter un facture

@WebMethod

**public** **boolean** addFacture(String NomC, String PrenomC, String marque, String num\_immat, String prix)

{

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="INSERT INTO `vente`.`facture` (`id`, `nomC`, `PrenomC`, `marqueV`, `num\_immat`, `prix`)"

+ " VALUES (NULL, '"+NomC+"', '"+PrenomC+"', '"+marque+"', '"+num\_immat+"', '"+prix+"')";

**if**( st.executeUpdate(query)>0) **return** **true**;

**else** **return** **false** ;

} **catch** (Exception e)

{ System.***out***.println(e.getMessage());

**return** **false**;

}

}

}

Web Service : server.java

**package** SI\_Ent\_2;

**import** javax.xml.ws.\*;

**import** javax.jws.\*;

**import** SI\_Ent\_2.Facture;

**public** **class** ServerWS {

**public** **static** **void** main(String[] args) {

**try** {

Endpoint.*publish*("http://localhost:5000/", **new** Facture());

System.***out***.println(" Web Service Publier sur le prot 5000 ...");

} **catch** (Exception e) {

System.***out***.println(e.getMessage());

}

}

}

SI\_Ent3

IFacture.java

**package** SI\_Ent\_3;

**import** java.rmi.Remote;

**import** java.rmi.RemoteException;

**public** **interface** IFacture **extends** Remote{

// ajouter un facture

**boolean** addFacture(String NomC, String PrenomC, String marque ,

String num\_immat , String prix) **throws** RemoteException ;

//rechercher une Facture a partir de son code

String findFacture(**int** Code )**throws** RemoteException;

//afficher tout les facture

String findFacture( )**throws** RemoteException;

// mise à jours d'une facture

**boolean** updateFacture(**int** Code , **int** numero\_att , String value)**throws** RemoteException;

// supprimer une facture

**boolean** deleteFacture(**int** Code)**throws** RemoteException;

// Calculer la recette de 2 eme & eme entreprise

**double** calculRecette() **throws** RemoteException;

}

Facture.java

**package** SI\_Ent\_3;

**import** java.io.PrintWriter;

**import** java.io.StringWriter;

**import** java.rmi.RemoteException;

**import** java.rmi.server.UnicastRemoteObject;

**import** java.sql.DriverManager;

**import** java.sql.ResultSet;

**public** **class** Facture **extends** UnicastRemoteObject **implements** IFacture {

// constructor

**public** Facture() **throws** RemoteException

{

**super**();

}

/\*

\* redefintition des methods(non-Javadoc)

\*

\*/

//-----------------------------------------------------------------------

//rechercher une Facture a partir de son code

@Override

**public** String findFacture(**int** Code) **throws** RemoteException

{

StringWriter out = **new** StringWriter();

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="SELECT \* FROM `facture` WHERE `facture`.`id` = "+Code;;

ResultSet rs=st.executeQuery(query);

// test si le ResultSet est vide

**boolean** isEmpty=**true**;

// sinon

**while**(rs.next())

{ **if**(rs.getInt("id")==Code)

{ isEmpty=**false**;

out.append("ID :"+rs.getInt("id")+" ");

out.append("Nom :"+rs.getString("nomC")+" ");

out.append("Prenom :"+rs.getString("PrenomC")+" ");

out.append("Marque voiture :"+rs.getString("marqueV")+" ");

out.append("N° immatriculation :"+rs.getString("num\_immat")+" ");

out.append("Prix :"+rs.getString("prix")+"\n");

}

}

**if**(isEmpty) **return** " Aucune Facture correspond à le Code : "+String.*valueOf*(Code)+" .\n";

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

**return** " Erreur de Connexion .\n";

}

**return** out.toString();

}

//afficher tout les facture

@Override

**public** String findFacture() **throws** RemoteException

{ StringWriter out = **new** StringWriter();

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="SELECT \* FROM `facture`";

ResultSet rs=st.executeQuery(query);

// test si le ResultSet est vide

**boolean** isEmpty=**true**;

// sinon

**while**(rs.next())

{ isEmpty=**false**;

out.append("ID :"+rs.getInt("id")+" ");

out.append("Nom :"+rs.getString("nomC")+" ");

out.append("Prenom :"+rs.getString("PrenomC")+" ");

out.append("Marque voiture :"+rs.getString("marqueV")+" ");

out.append("N° immatriculation :"+rs.getString("num\_immat")+" ");

out.append("Prix :"+rs.getString("prix")+"\n");

}

**if**(isEmpty) **return** " Aucune Facture .\n";

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

**return** " Erreur de Connexion .\n";

}

**return** out.toString();

}

// mise à jours d'une facture

@Override

**public** **boolean** updateFacture(**int** Code, **int** numero\_att , String value) **throws** RemoteException

{ String query = **null**;

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

**if**(numero\_att==1)

query="UPDATE `vente`.`facture` SET `nomC` = '"+value+"' WHERE `facture`.`id` = "+Code;

**else** **if**(numero\_att==2)

query="UPDATE `vente`.`facture` SET `PrenomC` = '"+value+"' WHERE `facture`.`id` = "+Code;

**else** **if**(numero\_att==3)

query="UPDATE `vente`.`facture` SET `marqueV` = '"+value+"' WHERE `facture`.`id` = "+Code;

**else** **if**(numero\_att==4)

query="UPDATE `vente`.`facture` SET `num\_immat` = '"+value+"' WHERE `facture`.`id` = "+Code;

**else**

query="UPDATE `vente`.`facture` SET `prix` = '"+value+"' WHERE `facture`.`id` = "+Code;

**if**( st.executeUpdate(query)>0) **return** **true**;

**else** **return** **false** ;

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

**return** **false**;

}

}

// supprimer une facture

@Override

**public** **boolean** deleteFacture(**int** Code) **throws** RemoteException

{

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="DELETE FROM `vente`.`facture` WHERE `facture`.`id` = "+Code;

**if**( st.executeUpdate(query)>0) **return** **true**;

**else** **return** **false** ;

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

**return** **false**;

}

}

// ajouter un facture

@Override

**public** **boolean** addFacture(String NomC, String PrenomC, String marque, String num\_immat, String prix)

**throws** RemoteException

{

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="INSERT INTO `vente`.`facture` (`id`, `nomC`, `PrenomC`, `marqueV`, `num\_immat`, `prix`)"

+ " VALUES (NULL, '"+NomC+"', '"+PrenomC+"', '"+marque+"', '"+num\_immat+"', '"+prix+"')";

**if**( st.executeUpdate(query)>0) **return** **true**;

**else** **return** **false** ;

} **catch** (Exception e)

{ System.***out***.println(e.getMessage());

**return** **false**;

}

}

// Calculer la recette de 2 eme & 3eme entreprise

@Override

**public** **double** calculRecette() **throws** RemoteException

{

**try**

{

java.sql.Connection connect= DriverManager.*getConnection*("jdbc:mysql://localhost/vente","root","");

java.sql.Statement st=connect.createStatement();

String query="SELECT SUM(`prix`) as recette FROM `facture`";

ResultSet r=st.executeQuery(query);

**if**( r.next())

{

**return** r.getDouble("recette") ;

}

**else**

**return** -1;

} **catch** (Exception e)

{ System.***out***.println(e.getMessage());

**return** -1;

}

}

}

Serveur Rmi : server.java

**package** SI\_Ent\_3;

**import** java.rmi.Naming;

**import** java.rmi.RemoteException;

**import** java.rmi.registry.LocateRegistry;

**import** java.rmi.registry.Registry;

**public** **class** Server {

**public** **static** **void** main(String[] args)

{

**try** {

//System.setProperty("java.rmi.server.hostname","localhost");

Registry reg= LocateRegistry.*createRegistry*(1099);

Facture f=**new** Facture();

Naming.*rebind*("rmi://localhost:1099/F", f);

System.***out***.println("\t Serveur RMI : Attente des invocations des clients sur le port 1099 ");

} **catch** (Exception e)

{

System.***out***.println(e.getMessage());

}

}

}