Rapport de stage

Stage effectué du 7 Aout au 7 Septembre par :

-Michel-Ange DAGRAIN

-Carla M. ST JUSTE

-Sandy THELEMAQUE

2012

Carla

Hewlett-Packard

17-Sep-12



Nous adressons nos remerciements au groupe women peer to peer network pour nous avoir permis d’effectuer ce stage. Un remerciement particulier à notre maitre de stage **Mme Tayana ETIENNE** pour son support et son encadrement.

Nous remercions **Vania CHARLES** et **Marguerite SAINT-LOUIS** qui ont bien voulu nous transmettre pendant ces quatre semaines leurs compétences et leurs savoirs faires.

Enfin, Nous remercions le staff de Solutions S.A de nous avoir chaleureusement accueilli.

**SOMMAIRE**

Introduction……………………………………………………………………………………………………………………………………3

Presentation de “women peer to peer”……………………………………………………………………..……………….4

Présentation du programme de stage…………………………………………………………………………………………4

Les différentes étapes a la réalisation du projet…………………………………………………………………..5

Conclusion…………………………………………………………………………………………………………………………………………22



Le stage constitue un élément indispensable à notre formation. C’est un contact qui permet à l’étudiant, de vivre la réalité socio-économique des entreprises.

Logées a Solutions S.A du 7 Août au 7 Septembre, ce stage à Women Peer to Peer network nous a permis de tester nos capacités d’analyse et de synthèse et de faire une comparaison entre la théorie et la pratique. Grace à ce stage nous avons été en mesure de tester et d’améliorer nos connaissances, de voir de près le déroulement des travaux informatiques, de vivre la réalité professionnelle, d’affronter l’aspect pratique des problèmes rencontrés.

Women peer to peer network étant une organisation dont l’objectif première est d’encourager les femmes dans la technologie nous avait donnée comme mission de développer un prototype d’application pour l’enregistrement des numéros de téléphone des membres de FONKOZE.

Afin d’exposer de manière efficace ces 4 semaines de stage nous présenterons dans les pages suivantes women peer to peer network qui est l’organisation pour lequel on a effectué le stage ainsi que le projet sur lequel nous avons été amené à travailler. On exposera ensuite les différentes étapes à la réalisation du projet et les difficultés rencontrées et pour terminer on fera un résumé du travail réalisé et on partagera nos impressions sur ce stage.

**Presentation de women peer to peer**

Women « peer to peer » est un programme dirigé par Rachel Levantal dont l’objectif principal est la conception et la réalisation d’un réseau de communication pour les femmes haïtiennes dans le commerce. Pour mettre sur pied ce réseau des femmes haïtiennes dans la technologie seront utilisés.

**Présentation du programme de stage**

L’objectif du stage est de développer et d’implémenter un prototype d’application d’enregistrement des numéros de téléphone des membres de FONKOZE. Ce travail devait être réalisé en 4 semaines allant du 6 Aout au 7 septembre 2012 suivant un horaire de travail de 5h par jour du lundi au vendredi.

* Planification

1. Première semaine :

Durant cette semaine on devait effectuer des recherches sur les outils qu’on aura à utiliser pour effectuer le travail. A la fin de la semaine un rapport de synthèse sur les recherches effectuées devrait être livré.

1. Deuxième semaine :

-Visite à FONKOZE

-Modélisation de la base de données

-Choix et apprentissage des technologies a utilisé.

1. Troisième semaine :

-Développement du prototype

1. Quatrième semaine :

-développement du prototype et formation.

**Les différentes étapes à la réalisation du projet :**

1. **Recherches et Expériences**

Avant de commencer à implémenter l’application nous avons commencé par faire des recherches. Durant le stage, à la fin de la première semaine on avait rédigé un rapport mettant en exergue les grands points de nos recherches qui ont été effectuées principalement sur Frontline SMS, Google table de fusion, Python…Dans le cadre de la réalisation du projet les recherches sur Frontline SMS ont été d’une grande utilité puisque nous avons utilisé cette application pour la collecte des SMS, delà l’intérêt à partager dans ce rapport nos expériences avec cette plate-forme.

* 1. Expériences avec Frontline SMS

Sur le site de Frontline SMS deux versions de l’application étaient disponibles : la version 1.6.16.3 et la version…... Nous avons téléchargé et expérimenté les deux. Vue la simplicité d’utilisation de la première version par rapport à la deuxième nous avons opté pour la première version qui en plus paraissait plus complète. Comme modem nous a avons utilisé celui distribué par NATCOM, le E173Eu-1. Nous avons testé l’envoie et la réception des sms à travers Frontline SMS à une personne et à un groupe de personne, expérimenté l’option de définition de mots clé et de création de commande externe fournie par l’application. Nous avons également expérimenté les exportations et les importations de données sur Frontline SMS. En effet, on a pu remarquer qu’on pouvait exporter tous les SMS reçus et sauvegarder dans la base de Frontline SMS dans une table Excel. Nous avons testé la connexion de Frontline SMS avec une base de H2 et également une base de MySQL.

* 1. Connexion de Frontline avec une base de MySQL

Nos premières connexions de Frontline SMS avec des bases de MySQL qu’on avait créé juste pour le test ont générées des anomalies dans le fonctionnement de Frontline SMS. En effet, après avoir établie la connexion de ce dernier avec une quelconque base de MySQL, 21 tables ont été créés automatiquement dans la base en question et l’application s’est également fermée automatiquement. C’est au redémarrage de l’application que les problèmes ont surgit, tantôt l’application ne veut pas se redémarrer, tantôt elle essaie de se redémarrer mais génère par la suite des erreurs. Ceci a rendu Frontline SMS inutilisable durant quelques jours vu que le problème persistait même après la désinstallation et la réinstallation de l’application. C’est ce qui nous a poussés à connecter Frontline SMS à une base de H2 pour ensuite établir une connexion entre cette base et la base de MySQL qui sera utilisée par notre application pour stocker les nouveaux numéros. Cependant après l’installation de SQLyog Entreprise établir une connexion directe entre Frontline SMS et une base de MySQL semble bien se passer donc nous avons une autre version de l’application où uniquement la base de MySQL est utilisée puisque les données de Frontline qu’on désire traiter sont stockées également sur cette base.

1. **Rencontre à FONKOZE**

**FONKOZE est une organisation à but non lucratif, composée de 60000 membres et de 250000 comptes d’épargne, dont la mission principale est d’aider les femmes haïtiennes pauvres a améliorer leur mode de vie grâce a un crédit et a une formation leur permettant de mieux gérer leur business.**

**A FONKOZE on utilise un logiciel bancaire très fiable « eMerge » et plusieurs petites bases de données de MS Access et des tables MS Excel, contenant chacune des informations différentes. Aucune technique systématique de mise à jour des différentes bases n’est encore opérationnelle, entre autre ils aimeraient interconnecter ces petites bases de données entre elles.**

* **Besoins des responsables de FONKOZE**

**Avant, lors de l’enregistrement des informations concernant un nouveau membre de FONKOZE dans une base de données, le numéro de téléphone n’était pas une donnée obligatoire, Maintenant, du fait que tout le monde, indépendamment de son niveau économique, soit en mesure de posséder un téléphone, les responsables pensent qu’il serait intéressant d’avoir cette information afin de pouvoir contacter directement un membre sans passer par le chef de centre. Pour combler ce besoin c’est-a-dire obtenir le numéro de téléphone d’un membre les responsables de FONKOZE ont fait appel a la technologie. Les responsables aurait aimé que de n’importe quel téléphone portable un membre puisse envoyer un SMS pour mettre à jour son numéro de téléphone dans la base. Delà, une plate-forme SMS, en l’occurrence Frontline SMS sera utilisée pour recueillir les SMS reçus et une application sera implémentée pour effectuer des traitements sur ses SMS et sauvegarder le numéro de téléphone reçu dans une base de MySQL crée principalement à cet effet. Les responsables ont souligné que l’application implémentée devrait avoir sa propre base de données qu’ils pourront ensuite connecter avec d’autres bases existantes.**

1. **Implémentation de l’application**

Pour commencer nous allons presenter le format que le client de FONKOZE doit utiliser pour envoyer par SMS son nouveau numéro de téléphone portable. En gros, il aura la structure qui suit: Mot-clé[espace]IDmembre[espace] Numéro. Le mot-clé utilisé sera « fk » pour FONKOZE. Toutes les variations sont prises en compte par l’application, c’est-à-dire que «FK », « fK » et « Fk » sont également admissibles.

Tous les membres de FONKOZE possède un identifiant unique qui permet de les identifier. Ce numéro ayant le format 711+6 autres chiffre+01 devra suivre le mot-clé et précéder le nouveau numéro.

Comme mentionné plus haut à cause des problèmes rencontrés avec la connexion de Frontline SMS avec une base de MySQL nous avions opté pour une connexion directe avec une base de H2. Nous avons utilisé Notepad++ et phpMyAdmin pour la réalisation du prototype.

* Présentation des codes source

import java.sql.DriverManager;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.awt.event.\*;

import java.io.\*;

class ImportDriver{

private String link="";

private String username="";

private String passwd="";

private Connection connect;

private Statement statement;

public ImportDriver(String link, String username, String passwd){

this.link=link;

this.username=username;

this.passwd=passwd;

}

public boolean verifyConnection(){

try{

Class.forName("org.h2.Driver").newInstance();

this.connect=DriverManager.getConnection("jdbc:h2:file:"+this.link, this.username, this.passwd);

this.statement=this.connect.createStatement(ResultSet.TYPE\_SCROLL\_SENSITIVE, ResultSet.CONCUR\_UPDATABLE);

return true;

}

catch(SQLException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

catch(ClassNotFoundException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

catch(InstantiationException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

catch(IllegalAccessException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

return false;

}

public ResultSet executer(String chaineSQL){

try{

ResultSet rs=this.statement.executeQuery(chaineSQL);

return rs;

}catch(SQLException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

return null;

}

public void close(){

try{

this.statement.close();

this.connect.close();

}catch(SQLException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

}

class ImportDriver1{

private String link="";

private String username="";

private String passwd="";

private Connection connect;

private Statement statement;

public ImportDriver1(String link, String username, String passwd){

this.link=link;

this.username=username;

this.passwd=passwd;}

public boolean verifyConnection(){

try{

Class.forName("com.mysql.jdbc.Driver").newInstance();

this.connect=DriverManager.getConnection("jdbc:mysql:"+this.link,this.username, this.passwd)

this.statement=this.connect.createStatement(ResultSet.TYPE\_SCROLL\_SENSITIVE, ResultSet.CONCUR\_UPDATABLE);

return true;

}catch(SQLException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

catch(ClassNotFoundException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

catch(InstantiationException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

catch(IllegalAccessException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

return false;

}

public ResultSet executer(String chaineSQL){

try{

ResultSet rs=this.statement.executeQuery(chaineSQL);

return rs;

}catch(SQLException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);

}

return null;

}

public void close(){

try{

this.statement.close();

this.connect.close();

}catch(SQLException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

public class FontlineSms2{

public static ResultSet rs,result;

private static String resourcePath;

public static ImportDriver1 ImD1;

public static ImportDriver ImD;

public static void main(String args[]){

System.out.println(getConfigDirectoryPath());

loadFonkozeBase();

loadSmsBase();

}

public static void loadSmsBase(){

ImD=new ImportDriver(getConfigDirectoryPath(), "", "");

if(ImD.verifyConnection()){

System.out.println("Connected to database fontlineSms");

try{

rs=ImD.executer("SELECT \* FROM message");

if(rs !=null){

while(rs.next()){

System.out.println("id: "+rs.getInt("id"));

System.out.println("Date: "+rs.getInt("Date"));

System.out.println("Expediteur: "+rs.getString("senderMsisdn"));

System.out.println("Message: "+rs.getString("textContent")+"\n\n");

String chaine=rs.getString("textContent");

String chaineTab[]=chaine.split(" ");

if (chaineTab.length!=0){

for(int i = 0; i<chaineTab.length; i++){

System.out.println("chaine("+i+")="+chaineTab[i]);

}

if(result !=null){

while(result.next()){

if (result.getString("numero\_membre")==chaineTab[0]){

result.moveToInsertRow();

result.updateString("telephone",chaineTab[1]);

result.insertRow();

break;

}

}

}

}

else{ System.out.println("the database is empty");}

}

}

else{

System.out.println("the database is empty");

}

}catch(SQLException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);

}

}else{

System.out.println("Connection to message failed !!!");

}

}

public static void loadFonkozeBase(){

ImD1 = new ImportDriver1("//localhost/fonkoze","root","");

if(ImD1.verifyConnection()){

try{

result=ImD1.executer("SELECT \* FROM membre");

if(result !=null){

while(result.next()){

System.out.println("id: "+result.getInt("id"));

System.out.println("NumeroMembre: "+result.getString("numero\_membre"));

System.out.println("Nom: "+result.getString("nom"));

System.out.println("Prenom: "+result.getString("prenom"));

System.out.println("Centre: "+result.getString("prenom"));

System.out.println("Telephone: "+result.getString("telephone"));

System.out.println("Adresse: "+result.getString("adresse")+"\n");

}

}

else{

System.out.println("the database is empty")}

}catch(SQLException ex){

Logger.getLogger(ImportDriver.class.getName()).log(Level.SEVERE, null, ex);}

else{

System.out.println("Connection failed !!!"); }

}

}

public static String getUserHome() {

return System.getProperty("user.home")+File.separatorChar+"AppData"+File.separatorChar+"Roaming"+File.separatorChar+"Microsoft"+File.separatorChar+"Windows"+File.separatorChar+"FrontlineSMS" + File.separatorChar;

}

public synchronized static String getConfigDirectoryPath() {

if(resourcePath == null) {

resourcePath = getUserHome();

}

if(resourcePath.charAt(resourcePath.length()-1) != File.separatorChar) {

resourcePath += File.separatorChar;

}

return resourcePath + "frontlinesms\_h2\_db";

}

}

Une fois que le problème de la connexion de Frontline SMS avec une base de MySQL a été résolu, nous avons opté pour l’utilisation d’une seule base au lieu de deux. L’une des tables générée par Frontline SMS (la table *message*) dans la base de MySQL *smsapp* crée sera utilisée par cette dernière pour stocker les SMS reçus. Pour cela il faut s’assurer qu’à tout moment l’application c’est-à-dire Frontline SMS est en mode « running », qu’elle reconnaisse le modem qui sera utilisé pour la réception des SMS, et qu’elle est en permanence connectée avec la base smsapp.

Lorsqu’un SMS est reçu par Frontline SMS, son contenu est sauvegardé dans le champ *textContent* de la table *message*. Dans cette même table on trouve dans le champ *date*, qui contient la date à laquelle le SMS a été livré. L’application que nous avons implémentée prendra le contenu du SMS reçu effectuera des tests sur lui pour vérifier s’il est valide c’est-à-dire que le mot clé, le numéro de membre et le format du numéro de téléphone sont corrects. Si le message reçu est invalide l’application ignore le message, dans le cas contraire elle vérifiera si ce numéro n’avait pas été déjà enregistré dans la base. Si le numéro était actif il ignore la requête du client qui voulait faire une mise à jour, si le numéro n’était pas dans la base mais le client avait fait dans le passé enregistrer un numéro, l’application placera l’ancien numéro dans une autre table (la table archive), effacera l’ancien de la table updatenum qui est la table qui contient les numéros de téléphone actif et y ajoute le nouveau numéro. Une présentation des codes sources de l’application donnera une idée plus claire sur la façon dont elle procède pour la mise à jour des numéros de téléphone. En dehors de la classe principale nous avons deux autres classes qui sont la classe *MySQLcli* et la classe *refresh*. Dans les lignes qui suivent de plus amples détails seront donnés sur les fonctions de ses classes.

* Présentation de la classe MySQLCli

Comme IDE nous avons utilisé NetBeans 7.1.2 et comme SGBD nous avons utilisé SQLyog Enterprise.

package repertoire\_telephonique;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

public class MySQLCli {

private String dbURL = "";

private String user = "";

private String password = "";

private java.sql.Connection dbConnect = null;

private java.sql.Statement dbStatement = null;

private int countrow;

//setter

public void setlastrow (int lr)

{this.countrow=lr;}

/\*\*

\* Constructeur

\* @param url

\* @param user

\* @param password

\*/

public MySQLCli(String url, String user, String password)

{

this.dbURL = url;

this.user = user;

this.password = password;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Méthodes\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* Connecter à la base de données

\* @return false en cas d'échec

\*/

public Boolean connect()

{

try {

Class.forName("com.mysql.jdbc.Driver").newInstance(); this.dbConnect=DriverManager.getConnection("jdbc:mysql:"+this.dbURL,this.user,this.password)

this.dbStatement = this.dbConnect.createStatement();

return true

}

catch(SQLException ex)

{ Logger.getLogger(MySQLCli.class.getName()).log(Level.SEVERE,null,ex);}

catch(ClassNotFoundException ex)

{ Logger.getLogger(MySQLCli.class.getName()).log(Level.SEVERE, null, ex);}

catch(InstantiationException ex)

{ Logger.getLogger(MySQLCli.class.getName()).log(Level.SEVERE, null, ex);}

catch(IllegalAccessException ex)

{ Logger.getLogger(MySQLCli.class.getName()).log(Level.SEVERE, null, ex);}

return false;

}

/\*\*

\* Executer une requete SQL

\* @param sql

\* @return resultat de la requete

\*/

public ResultSet exec(String sql)

{

try {

ResultSet rs = this.dbStatement.executeQuery(sql);

return rs;

}

catch (SQLException ex)

{ Logger.getLogger(MySQLCli.class.getName()).log(Level.SEVERE, null, ex);}

return null;

}

/\*\*

\* Executer une requête SQL

\* @param sql

\* @return le numero de la ligne

\*/

public int exec1(String sql)

{

try {

int rowcount = this.dbStatement.executeUpdate(sql);

return rowcount

}

catch(SQLException ex)

{Logger.getLogger(MySQLCli.class.getName()).log(Level.SEVERE, null, ex);}

return 0;

}

/\*\*

\* Fermer la connexion au serveur de donnees

\*/

public void close()

{

try {

this.dbStatement.close();

this.dbConnect.close();

}

catch (SQLException ex)

{Logger.getLogger(MySQLCli.class.getName()).log(Level.SEVERE, null, ex);

}

}

Cette classe contient quatre méthodes, deux qui permettent de se connecter et de se déconnecter respectivement à une base de MySQL et deux autres pour l’exécution des requêtes SQL.

* Présentation de la classe refresh

package repertoire\_telephonique;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Date;

public class Refresh {

private MySQLCli M;

public Refresh (MySQLCli m)

{

this.M=m;

}

public int CheckAdd() throws SQLException

{

int NbreAjout=0;

ResultSet R=M.exec("SELECT value FROM variable WHERE ID=1");

if(R!=null && R.absolute(1))

{

int x=R.getInt("Value");

ResultSet RS=M.exec("SELECT COUNT(\*)FROM message");

If (RS!=null && RS.absolute(1))

{

int y=RS.getInt("COUNT(\*)");

if(x<y)

{

NbreAjout=y-x;

}

}

}

return NbreAjout;

}

public void treatment () throws SQLException

{

ResultSet R=M.exec("SELECT value FROM variable WHERE ID=1");

if(R!=null && R.absolute(1))

{

int lastposition=R.getInt("Value");

ResultSet Re=M.exec("SELECT \* FROM message");

if (Re!=null && Re.absolute(lastposition))

{

while (Re.next())

{

String textcontent=Re.getString("textContent");

String[] tabsplitmessage=textcontent.split(" ");

if (tabsplitmessage.length==3)

{

boolean keyvalid=Checkkey(tabsplitmessage[0]);

boolean IDvalid=CheckID(tabsplitmessage[1]);

boolean numvalid=Checknum(tabsplitmessage[2]);

if(keyvalid==true && IDvalid==true && numvalid==true)

{

String ID=tabsplitmessage[1];

String num=tabsplitmessage[2];

Date date=Re.getDate("date");

ResultSet Res= M.exec("SELECT IDmember FROM updatetab WHERE IDmember=tabsplitmessage[1]");

if(Res==null)

{

int countrow = M.exec1("INSERT INTO updatetab (IDmember, NewNumber,Dateupdate) VALUES('" + ID + "','" + num+ "','" + date + "'");

lastposition++;

}

if (Res! =null &&! Res.getString ("NewNumber").equals (tabsplitmessage [2]))

{

String IDMember=Res.getString ("IDmember");

String OldNumber=Res.getString("NewNumber");

Date DateEnable=Res.getDate("Dateupdate");

Date Date Disable=date;

int countrowar=M.exec1("INSERT INTO archive (IDMember,OldNumber,DateEnable,DateDisable)VALUE ('"+IDMember+"','"+OldNumber+"','"+DateEnable+"','"+DateDisable+"')");

int countrowup=M.exec1("DELETE FROM updatetab WHERE IDmember=IDMember"); int countrow = M.exec1("INSERT INTO updatetab (IDmember, NewNumber, Dateupdate) VALUES('" + ID + "','" + num+ "','" + date + "'");

lastposition++;

}

if (Res! =null && Res.getString("NewNumber").equals(tabsplitmessage[2]))

{

lastposition++;

}

} else {lastposition++ ;}

} else {lastposition++ ;}

}

int up=M.exec1 ("UPDATE variable SET value=lastposition WHERE ID=1");

}

}

}

/\*\*

\* Check if the keyword is valid

\* @param s

\* @return state of the keyword

\*/

public static boolean Checkkey (String s)

{

if ("fk".equals(s)||"Fk".equals(s)||"FK".equals(s)||"fK".equals(s))

{

return true;

}

else

{

return false;

}

}

/\*\*

\* Check if the IDmember is correct

\* @param s

\* @return state of the IDMember

\*/

public static boolean CheckID (String s)

{ String [] tab=s.split ("");

if(tab.length==12 && "7".equals(tab[1]) && "0".equals(tab[2]) && "5".equals(tab[3]) &&"0".equals(tab[10])&& "1".equals(tab[11]))

{ System.out.println("ID correct");

return true;

}

else

{ System.out.println("ID incorrect");

return false;

}

}

/\*\*

\* Check if the number's format is correct

\* @param s

\* @return state of the number's format

\*/

public static boolean Checknum(String s)

{

String [] tab= s.split ("");

if (tab.length==9)

{

System.out.println("Numero correct");

return true;

}

else

{

System.out.println("Numero incorrect");

return false;

}

}

}

Le rôle de cette classe est d’effectuer les traitements sur les messages reçus et de faire des mises à jour. Elle contient cinq (5) méthodes :

- checkAdd() permet de vérifier s’il y a des mises à jour c’est-à-dire vérifie s’il n’y a pas des SMS envoyés par les clients de FONKOZE qui n’ont pas encore été traités. Cette méthode retourne un « int » qui est le nombre de message qui a été ajouté dans la base par Frontline SMS depuis la dernière mise à jour. Une table sera utilisée pour sauvegarder l’ID de la dernière ligne parcouru dans la table message, a noter que l’ID de la dernière ligne de la table correspond au nombre de ligne que contient la table.

–checkkey(String s) teste la validité du mot-clé est correct. Le mot-clé qui doit être utilisé est « fk » et tous les formats sont admis.

-ckeckID(String s) teste la validité du numéro de membre.

-checknum(String s) teste si le numéro envoyé est valide.

-treatment() effectue les traitements sur les SMS reçus. Elle divise le message en trois parties : le mot-clé, l’identifiant et le numéro. Elle fait appel à d’autres méthodes pour s’assurer que ces derniers sont valides. Si le message est valide, on vérifie si le membre en question n’avait pas déjà au préalable un numéro enregistré dans la base. Delà, plusieurs cas peuvent se presenter :

-Le nouveau numéro était déjà dans la base : Dans ce cas on ignore la requête du client.

-Le client avait déjà un numéro dans la base mais qui est diffèrent de celui qu’il veut ajouter : Dans ce cas, l’ancien numéro sera ajouté dans la table archive et effacé de la table updatetab et le nouveau numéro sera inséré dans la table updatetab.

-le client n’avait pas de numéro dans la base : Dans ce cas on insère le nouveau numéro du client dans la table updatetab.

A chaque fois qu’un numéro sera enregistré dans la base la date à laquelle la mise à jour a été faite sera également enregistrée.

* La classe principale

package repertoire\_telephonique;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

public class Repertoire\_telephonique {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

MySQLCli mysqlCli = new MySQLCli("//localhost/smsapp", "root", "");

if (mysqlCli.connect())

{

System.out.println("Connected to database smsapp");

Refresh R=new Refresh(mysqlCli);

try {

int NumberAdd=R.CheckAdd();

if(NumberAdd!=0)

{

R.treatment();

}

} catch (SQLException ex) { Logger.getLogger(Repertoire\_telephonique.class.getName()).log(Level.SEVERE, null, ex);

}

}

Ce stage fut pour nous une expérience très positives tant au niveau professionnel que relationnel.

Durant ces quatre semaines on a pu observer le fonctionnement de service d’une entreprise.

Le stage nous a donné gout à la programmation et nous a permis de nous évaluer.