/\*\*

\*

\*/

package in.gov.mgov;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.UnsupportedEncodingException;

import java.security.KeyManagementException;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import java.util.ArrayList;

import java.util.List;

import java.util.StringTokenizer;

import javax.net.ssl.SSLContext;

import org.apache.http.HttpResponse;

import org.apache.http.NameValuePair;

import org.apache.http.client.ClientProtocolException;

import org.apache.http.client.HttpClient;

import org.apache.http.client.entity.UrlEncodedFormEntity;

import org.apache.http.client.methods.HttpPost;

import org.apache.http.conn.scheme.Scheme;

import org.apache.http.conn.ssl.SSLSocketFactory;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.message.BasicNameValuePair;

/\*\*

\* @author Mobile Seva < msdp@cdac.in >

\* <p>Kindly add require Jar files to run Java client</p><p>

\* Apache commons-codec-1.9

\* Apache commons-httpclient-3.1

\* Apache commons-logging-1.2

\* @see <!-- <a href="https://mgov.gov.in/doc/RequiredJars.zip">Download required Jar files</a> -->

\*/

public class SMSServices {

/\*\*

\* Send Single text SMS

\* @param username : Department Login User Name

\* @param password : Department Login Password

\* @param message : Message e.g. 'Welcome to mobile Seva'

\* @param senderId : Department allocated SenderID

\* @param mobileNumber : Single Mobile Number e.g. '99XXXXXXX'

\* @param secureKey : Department key generated by login to services portal

\* @param templateid : 12 to 19 digit id for each template content

\* @return {@link String} response from Mobile Seva Gateway e.g. '402,MsgID = 150620161466003974245msdgsms'

\* @see <!-- <a href="https://mgov.gov.in/msdp\_sms\_push.jsp">Return types code details</a> -->

\*

\*/

public String sendSingleSMS(String username, String password , String message , String senderId, String mobileNumber,String secureKey, String templateid)

{

String responseString = "";

SSLSocketFactory sf=null;

SSLContext context=null;

String encryptedPassword;

try {

//context=SSLContext.getInstance("TLSv1.1"); // Use this line for Java version 6

context=SSLContext.getInstance("TLSv1.2"); // Use this line for Java version 7 and above

context.init(null, null, null);

sf=new SSLSocketFactory(context, SSLSocketFactory.STRICT\_HOSTNAME\_VERIFIER);

Scheme scheme=new Scheme("https",443,sf);

HttpClient client=new DefaultHttpClient();

client.getConnectionManager().getSchemeRegistry().register(scheme);

HttpPost post=new HttpPost("https://msdgweb.mgov.gov.in/esms/sendsmsrequestDLT");

encryptedPassword = MD5(password);

String genratedhashKey = hashGenerator(username, senderId, message, secureKey);

List<NameValuePair> nameValuePairs=new ArrayList<NameValuePair>(1);

nameValuePairs.add(new BasicNameValuePair("mobileno", mobileNumber));

nameValuePairs.add(new BasicNameValuePair("senderid", senderId));

nameValuePairs.add(new BasicNameValuePair("content", message));

nameValuePairs.add(new BasicNameValuePair("smsservicetype", "singlemsg"));

nameValuePairs.add(new BasicNameValuePair("username", username));

nameValuePairs.add(new BasicNameValuePair("password", encryptedPassword));

nameValuePairs.add(new BasicNameValuePair("key", genratedhashKey));

nameValuePairs.add(new BasicNameValuePair("templateid", templateid));

post.setEntity(new UrlEncodedFormEntity(nameValuePairs));

HttpResponse response=client.execute(post);

BufferedReader bf=new BufferedReader(new InputStreamReader(response.getEntity().getContent()));

String line="";

while((line=bf.readLine())!=null){

responseString = responseString+line;

}

System.out.println(responseString);

} catch (NoSuchAlgorithmException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (KeyManagementException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (UnsupportedEncodingException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ClientProtocolException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return responseString;

}

/\*\*

\* Send Bulk text SMS

\* @param username : Department Login User Name

\* @param password : Department Login Password

\* @param message : Message e.g. 'Welcome to mobile Seva'

\* @param senderId : Department allocated SenderID

\* @param mobileNumber : Bulk Mobile Number with comma separated e.g. '99XXXXXXX,99XXXXXXXX'

\* @param secureKey : Department key generated by login to services portal

\* @param templateid : 12 to 19 digit id for each template content

\* @return {@link String} response from Mobile Seva Gateway e.g. '402,MsgID = 150620161466003974245msdgsms'

\* @see <!-- <a href="https://mgov.gov.in/msdp\_sms\_push.jsp">Return types code details</a> -->

\*

\*/

public String sendBulkSMS(String username, String password , String message , String senderId, String mobileNumber, String secureKey, String templateid){

String responseString = "";

SSLSocketFactory sf=null;

SSLContext context=null;

String encryptedPassword;

try {

//context=SSLContext.getInstance("TLSv1.1"); // Use this line for Java version 6

context=SSLContext.getInstance("TLSv1.2"); // Use this line for Java version 7 and above

context.init(null, null, null);

sf=new SSLSocketFactory(context, SSLSocketFactory.STRICT\_HOSTNAME\_VERIFIER);

Scheme scheme=new Scheme("https",443,sf);

HttpClient client=new DefaultHttpClient();

client.getConnectionManager().getSchemeRegistry().register(scheme);

HttpPost post=new HttpPost("https://msdgweb.mgov.gov.in/esms/sendsmsrequestDLT");

encryptedPassword = MD5(password);

String genratedhashKey = hashGenerator(username, senderId, message, secureKey);

List<NameValuePair> nameValuePairs=new ArrayList<NameValuePair>(1);

nameValuePairs.add(new BasicNameValuePair("bulkmobno", mobileNumber));

nameValuePairs.add(new BasicNameValuePair("senderid", senderId));

nameValuePairs.add(new BasicNameValuePair("content", message));

nameValuePairs.add(new BasicNameValuePair("smsservicetype", "bulkmsg"));

nameValuePairs.add(new BasicNameValuePair("username", username));

nameValuePairs.add(new BasicNameValuePair("password", encryptedPassword));

nameValuePairs.add(new BasicNameValuePair("key", genratedhashKey));

nameValuePairs.add(new BasicNameValuePair("templateid", templateid));

post.setEntity(new UrlEncodedFormEntity(nameValuePairs));

HttpResponse response=client.execute(post);

BufferedReader bf=new BufferedReader(new InputStreamReader(response.getEntity().getContent()));

String line="";

while((line=bf.readLine())!=null){

responseString = responseString+line;

}

System.out.println(responseString);

} catch (NoSuchAlgorithmException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (KeyManagementException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (UnsupportedEncodingException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ClientProtocolException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return responseString;

}

/\*\*

\* Send Unicode text SMS

\* @param username : Department Login User Name

\* @param password : Department Login Password

\* @param message : Unicode Message e.g. 'à¤µà¤¿à¤•à¤¾à¤¸ à¤†à¤£à¤¿ à¤ªà¥à¤°à¤—à¤¤ à¤¸à¤‚à¤—à¤£à¤¨ à¤•à¥‡à¤‚à¤¦à¥à¤° à¤®à¤§à¥à¤¯à¥‡ à¤¸à¥à¤µà¤¾à¤—à¤¤ à¤†à¤¹à¥‡'

\* @param senderId : Department allocated SenderID

\* @param mobileNumber : Bulk Mobile Number with comma separated e.g. '99XXXXXXX,99XXXXXXXX'

\* @param secureKey : Department key generated by login to services portal

\* @param templateid : 12 to 19 digit id for each template content

\* @return {@link String} response from Mobile Seva Gateway e.g. '402,MsgID = 150620161466003974245msdgsms'

\* @see <!-- <a href="https://mgov.gov.in/msdp\_sms\_push.jsp">Return types code details</a> -->

\*

\*/

public String sendUnicodeSMS(String username, String password , String message , String senderId, String mobileNumber,String secureKey, String templateid){

String finalmessage = "";

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

char ch = message.charAt(i);

int j = (int) ch;

String sss = "&#"+j+";";

finalmessage = finalmessage+sss;

}

String responseString = "";

SSLSocketFactory sf=null;

SSLContext context=null;

String encryptedPassword;

try {

//context=SSLContext.getInstance("TLSv1.1"); // Use this line for Java version 6

context=SSLContext.getInstance("TLSv1.2"); // Use this line for Java version 7 and above

context.init(null, null, null);

sf=new SSLSocketFactory(context, SSLSocketFactory.STRICT\_HOSTNAME\_VERIFIER);

Scheme scheme=new Scheme("https",443,sf);

HttpClient client=new DefaultHttpClient();

client.getConnectionManager().getSchemeRegistry().register(scheme);

HttpPost post=new HttpPost("https://msdgweb.mgov.gov.in/esms/sendsmsrequestDLT");

encryptedPassword = MD5(password);

String genratedhashKey = hashGenerator(username, senderId, finalmessage, secureKey);

List<NameValuePair> nameValuePairs=new ArrayList<NameValuePair>(1);

nameValuePairs.add(new BasicNameValuePair("bulkmobno", mobileNumber));

nameValuePairs.add(new BasicNameValuePair("senderid", senderId));

nameValuePairs.add(new BasicNameValuePair("content", finalmessage));

nameValuePairs.add(new BasicNameValuePair("smsservicetype", "unicodemsg"));

nameValuePairs.add(new BasicNameValuePair("username", username));

nameValuePairs.add(new BasicNameValuePair("password", encryptedPassword));

nameValuePairs.add(new BasicNameValuePair("key", genratedhashKey));

nameValuePairs.add(new BasicNameValuePair("templateid", templateid));

post.setEntity(new UrlEncodedFormEntity(nameValuePairs));

HttpResponse response=client.execute(post);

BufferedReader bf=new BufferedReader(new InputStreamReader(response.getEntity().getContent()));

String line="";

while((line=bf.readLine())!=null){

responseString = responseString+line;

}

System.out.println(responseString);

} catch (NoSuchAlgorithmException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (KeyManagementException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (UnsupportedEncodingException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ClientProtocolException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return responseString;

}

/\*\*

\* Send Single OTP text SMS

\*</p><p> Use only in case of OTP related message</p><p>

\* Messages other than OTP will not be delivered to the users

\* @param username : Department Login User Name

\* @param password : Department Login Password

\* @param message : Message e.g. 'Welcome to mobile Seva'

\* @param senderId : Department allocated SenderID

\* @param mobileNumber : Single Mobile Number e.g. '99XXXXXXX'

\* @param secureKey : Department key generated by login to services portal

\* @param templateid : 12 to 19 digit id for each template content

\* @return {@link String} response from Mobile Seva Gateway e.g. '402,MsgID = 150620161466003974245msdgsms'

\* @see <!-- <a href="https://mgov.gov.in/msdp\_sms\_push.jsp">Return types code details</a> -->

\*

\*/

public String sendOtpSMS(String username, String password , String message , String senderId, String mobileNumber,String secureKey, String templateid){

String responseString = "";

SSLSocketFactory sf=null;

SSLContext context=null;

String encryptedPassword;

try {

//context=SSLContext.getInstance("TLSv1.1"); // Use this line for Java version 6

context=SSLContext.getInstance("TLSv1.2"); // Use this line for Java version 7 and above

context.init(null, null, null);

sf=new SSLSocketFactory(context, SSLSocketFactory.STRICT\_HOSTNAME\_VERIFIER);

Scheme scheme=new Scheme("https",443,sf);

HttpClient client=new DefaultHttpClient();

client.getConnectionManager().getSchemeRegistry().register(scheme);

HttpPost post=new HttpPost("https://msdgweb.mgov.gov.in/esms/sendsmsrequestDLT");

encryptedPassword = MD5(password);

String genratedhashKey = hashGenerator(username, senderId, message, secureKey);

List<NameValuePair> nameValuePairs=new ArrayList<NameValuePair>(1);

nameValuePairs.add(new BasicNameValuePair("mobileno", mobileNumber));

nameValuePairs.add(new BasicNameValuePair("senderid", senderId));

nameValuePairs.add(new BasicNameValuePair("content", message));

nameValuePairs.add(new BasicNameValuePair("smsservicetype", "otpmsg"));

nameValuePairs.add(new BasicNameValuePair("username", username));

nameValuePairs.add(new BasicNameValuePair("password", encryptedPassword));

nameValuePairs.add(new BasicNameValuePair("key", genratedhashKey));

nameValuePairs.add(new BasicNameValuePair("templateid", templateid));

post.setEntity(new UrlEncodedFormEntity(nameValuePairs));

HttpResponse response=client.execute(post);

BufferedReader bf=new BufferedReader(new InputStreamReader(response.getEntity().getContent()));

String line="";

while((line=bf.readLine())!=null){

responseString = responseString+line;

}

System.out.println(responseString);

} catch (NoSuchAlgorithmException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (KeyManagementException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (UnsupportedEncodingException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ClientProtocolException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return responseString;

}

/\*\*

\* Send Single Unicode OTP text SMS

\* @param username : Department Login User Name

\* @param password : Department Login Password

\* @param message : Unicode Message e.g. 'à¤µà¤¿à¤•à¤¾à¤¸ à¤†à¤£à¤¿ à¤ªà¥à¤°à¤—à¤¤ à¤¸à¤‚à¤—à¤£à¤¨ à¤•à¥‡à¤‚à¤¦à¥à¤° à¤®à¤§à¥à¤¯à¥‡ à¤¸à¥à¤µà¤¾à¤—à¤¤ à¤†à¤¹à¥‡'

\* @param senderId : Department allocated SenderID

\* @param mobileNumber : Bulk Mobile Number with comma separated e.g. '99XXXXXXX,99XXXXXXXX'

\* @param secureKey : Department key generated by login to services portal

\* @param templateid : 12 to 19 digit id for each template content

\* @return {@link String} response from Mobile Seva Gateway e.g. '402,MsgID = 150620161466003974245msdgsms'

\* @see <!-- <a href="https://mgov.gov.in/msdp\_sms\_push.jsp">Return types code details</a> -->

\*

\*/

public String sendUnicodeOtpSMS(String username, String password , String message , String senderId, String mobileNumber,String secureKey, String templateid){

String finalmessage = "";

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

char ch = message.charAt(i);

int j = (int) ch;

String sss = "&#"+j+";";

finalmessage = finalmessage+sss;

}

String responseString = "";

SSLSocketFactory sf=null;

SSLContext context=null;

String encryptedPassword;

try {

//context=SSLContext.getInstance("TLSv1.1"); // Use this line for Java version 6

context=SSLContext.getInstance("TLSv1.2"); // Use this line for Java version 7 and above

context.init(null, null, null);

sf=new SSLSocketFactory(context, SSLSocketFactory.STRICT\_HOSTNAME\_VERIFIER);

Scheme scheme=new Scheme("https",443,sf);

HttpClient client=new DefaultHttpClient();

client.getConnectionManager().getSchemeRegistry().register(scheme);

HttpPost post=new HttpPost("https://msdgweb.mgov.gov.in/esms/sendsmsrequestDLT");

encryptedPassword = MD5(password);

String genratedhashKey = hashGenerator(username, senderId, finalmessage, secureKey);

List<NameValuePair> nameValuePairs=new ArrayList<NameValuePair>(1);

nameValuePairs.add(new BasicNameValuePair("mobileno", mobileNumber));

nameValuePairs.add(new BasicNameValuePair("senderid", senderId));

nameValuePairs.add(new BasicNameValuePair("content", finalmessage));

nameValuePairs.add(new BasicNameValuePair("smsservicetype", "unicodeotpmsg"));

nameValuePairs.add(new BasicNameValuePair("username", username));

nameValuePairs.add(new BasicNameValuePair("password", encryptedPassword));

nameValuePairs.add(new BasicNameValuePair("key", genratedhashKey));

nameValuePairs.add(new BasicNameValuePair("templateid", templateid));

post.setEntity(new UrlEncodedFormEntity(nameValuePairs));

HttpResponse response=client.execute(post);

BufferedReader bf=new BufferedReader(new InputStreamReader(response.getEntity().getContent()));

String line="";

while((line=bf.readLine())!=null){

responseString = responseString+line;

}

System.out.println(responseString);

} catch (NoSuchAlgorithmException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (KeyManagementException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (UnsupportedEncodingException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ClientProtocolException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return responseString;

}

protected String hashGenerator(String userName, String senderId, String content, String secureKey) {

// TODO Auto-generated method stub

StringBuffer finalString=new StringBuffer();

finalString.append(userName.trim()).append(senderId.trim()).append(content.trim()).append(secureKey.trim());

// logger.info("Parameters for SHA-512 : "+finalString);

String hashGen=finalString.toString();

StringBuffer sb = null;

MessageDigest md;

try {

md = MessageDigest.getInstance("SHA-512");

md.update(hashGen.getBytes());

byte byteData[] = md.digest();

//convert the byte to hex format method 1

sb = new StringBuffer();

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

sb.append(Integer.toString((byteData[i] & 0xff) + 0x100, 16).substring(1));

}

} catch (NoSuchAlgorithmException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return sb.toString();

}

/\*\*

\* Get units of the unicode message

\* @param message e.g. 'à¤µà¤¿à¤•à¤¾à¤¸ à¤†à¤£à¤¿ à¤ªà¥à¤°à¤—à¤¤ à¤¸à¤‚à¤—à¤£à¤¨ à¤•à¥‡à¤‚à¤¦à¥à¤° à¤®à¤§à¥à¤¯à¥‡ à¤¸à¥à¤µà¤¾à¤—à¤¤ à¤†à¤¹à¥‡'

\* @return int message unit

\* \*\*/

public int getUnicodeTextMessageUnit(String message) {

String charInUnicode = "";

int msgUnit = 1;

int msgLen = 0;

String unicodeMessgae = "";

String finalmessage = null;

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

char ch = message.charAt(i);

int j = (int) ch;

String sss = "&#"+j+";";

finalmessage = finalmessage+sss;

}

StringTokenizer st = new StringTokenizer(finalmessage, " ");

while (st.hasMoreElements()) {

String str1 = (String) st.nextElement();

StringTokenizer dd = new StringTokenizer(str1, ";");

while (dd.hasMoreElements()) {

charInUnicode = (String) dd.nextElement();

if (charInUnicode.startsWith("&#")) {

StringTokenizer df = new StringTokenizer(

charInUnicode, "&#");

while (df.hasMoreElements()) {

String kk = (String) df.nextElement();

unicodeMessgae = unicodeMessgae + "," + kk;

msgLen = msgLen+1;

}

} else {

if(charInUnicode.contains("&#")){

StringTokenizer st1 = new StringTokenizer(charInUnicode, "&#");

while (st1.hasMoreElements()) {

String kk = (String) st1.nextElement();

for (int i1 = 0; i1 < kk.length(); ++i1) {

char c = kk.charAt(i1);

int j = (int) c;

unicodeMessgae = unicodeMessgae + "," + j;

msgLen = msgLen+1;

}

String uni = st1.nextToken();

unicodeMessgae = unicodeMessgae + "," + uni;

msgLen = msgLen+1;

}

}

else{

for (int i1 = 0; i1 < charInUnicode.length(); ++i1) {

char c = charInUnicode.charAt(i1);

int j = (int) c;

unicodeMessgae = unicodeMessgae + "," + j;

msgLen = msgLen+1;

}

}

}

}

unicodeMessgae = unicodeMessgae + " ";

}

if (msgLen > 70) {

msgUnit = 2;

if (msgLen > 134) {

msgUnit = 3;

if (msgLen > 201) {

msgUnit = 4;

if (msgLen > 268) {

msgUnit = 5;

if (msgLen > 335) {

msgUnit = 6;

if (msgLen > 402) {

msgUnit = 7;

if (msgLen> 469) {

msgUnit = 8;

if (msgLen > 536) {

msgUnit = 9;

if (msgLen > 603) {

msgUnit = 10;

}

}

}

}

}

}

}

}

}else{

msgUnit = 1;

}

return msgUnit;

}

/\*\*

\* Get units of the text message

\* @param message e.g. 'Welcome to Mobile Seva'

\* @return int message unit

\* \*\*/

public int getNormalTextMessageUnit(String message) {

int msgUnit = 1;

if (message.length() > 160) {

msgUnit = 2;

if (message.length() > 306) {

msgUnit = 3;

}

if (message.length() > 459) {

msgUnit = 4;

}

if (message.length() > 612) {

msgUnit = 5;

}

if (message.length() > 765) {

msgUnit = 6;

}

if (message.length() > 918) {

msgUnit = 7;

}

if (message.length() > 1071) {

msgUnit = 8;

}

if (message.length() > 1224) {

msgUnit = 9;

}

if (message.length() > 1377) {

msgUnit = 10;

}

}else{

msgUnit = 1;

}

return msgUnit;

}

/\*\*\*\*

\* Method to convert Normal Plain Text Password to MD5 encrypted password

\*\*\*/

private static String MD5(String text) throws NoSuchAlgorithmException, UnsupportedEncodingException

{

MessageDigest md;

md = MessageDigest.getInstance("SHA-1");

byte[] md5 = new byte[64];

md.update(text.getBytes("iso-8859-1"), 0, text.length());

md5 = md.digest();

return convertedToHex(md5);

}

private static String convertedToHex(byte[] data)

{

StringBuffer buf = new StringBuffer();

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

{

int halfOfByte = (data[i] >>> 4) & 0x0F;

int twoHalfBytes = 0;

do

{

if ((0 <= halfOfByte) && (halfOfByte <= 9))

{

buf.append( (char) ('0' + halfOfByte) );

}

else

{

buf.append( (char) ('a' + (halfOfByte - 10)) );

}

halfOfByte = data[i] & 0x0F;

} while(twoHalfBytes++ < 1);

}

return buf.toString();

}

}