**CODE:**

**DBConnection.java**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package DatabaseConnectivity;

import java.sql.Connection;

import java.sql.DriverManager;

/\*\*

\*

\* @author java2

\*/

public class Dbconn {

public static Connection getConnection() {

Connection con = null;

try {

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

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/blockchain", "root", "root");

} catch (Exception ex) {

ex.printStackTrace();

}

return con;

}

}

File**uploadaction:**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package FileUpload;

import DatabaseConnectivity.Dbconn;

import java.io.ByteArrayInputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.PrintWriter;

import java.nio.charset.StandardCharsets;

import java.sql.Connection;

import java.sql.PreparedStatement;

import javax.servlet.ServletException;

import javax.servlet.annotation.MultipartConfig;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import javax.servlet.http.Part;

/\*\*

\*

\* @author Ramu Maloth

\*/

@MultipartConfig(maxFileSize = 16177215)

public class fileUploadaction extends HttpServlet {

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

String filename = request.getParameter("filename");

String atrbname = request.getParameter("atrbname");

String cloudname = request.getParameter("cloudname");

String contents = request.getParameter("contents");

String macKey = request.getParameter("macKey");

InputStream inputStream = null;

InputStream cipherData = null;

Part filePart = request.getPart("file");

// String filename = getFileName(filePart);

if (filePart != null) {

// prints out some information for debugging

System.out.println(filePart.getName());

System.out.println(filePart.getSize());

System.out.println(filePart.getContentType());

// obtains input stream of the upload file

inputStream = filePart.getInputStream();

}

HttpSession hs = request.getSession();

String name = hs.getAttribute("uname").toString();

System.out.println("name:"+name);

String email = hs.getAttribute("email").toString();

System.out.println("email:"+email);

cipherData = new ByteArrayInputStream(contents.getBytes(StandardCharsets.UTF\_8));

Connection con = null;

PreparedStatement ps = null;

java.sql.Date sqlDate = new java.sql.Date(new java.util.Date().getTime());

try {

con = Dbconn.getConnection();

String query = "insert into datafiles(name,email,filename,attributename,cloudname,mackey,cipherdata,cdate) values(?,?,?,?,?,?,?,?)";

ps = con.prepareStatement(query);

ps.setString(1,name);

ps.setString(2, email);

ps.setString(3, filename);

ps.setString(4, atrbname);

ps.setString(5, cloudname);

ps.setString(6, macKey);

ps.setBinaryStream(7, cipherData);

//ps.setBinaryStream(8, inputStream);

ps.setDate(8, sqlDate);

int no = ps.executeUpdate();

if(no > 0){

response.sendRedirect("fileupload.jsp?msg=success");

}else{

response.sendRedirect("fileupload.jsp?msg=faild");

}

} catch (Exception e) {

System.out.println("Error at DBA File upload "+e.getMessage());

}finally{

try {

inputStream.close();

cipherData.close();

ps.close();

con.close();

} catch (Exception e) {

}

}

}

/\*\*

\* Returns a short description of the servlet.

\*

\* @return a String containing servlet description

\*/

@Override

public String getServletInfo() {

return "Short description";

}// </editor-fold>

}

**download:**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package com.app.action;

import com.app.algorithms.AESImple;

import com.app.utility.DBConnectionn;

import com.app.wrapper.ECCEncoder;

import com.app.wrapper.GetKeysFromServer;

import com.app.wrapper.StoreDownloadHistory;

import java.io.IOException;

import java.io.OutputStream;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import javax.servlet.ServletContext;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

/\*\*

\*

\* @author Meghana Reddy

\*/

public class DODownloadAction extends HttpServlet {

/\*\*

\* Processes requests for both HTTP <code>GET</code> and <code>POST</code>

\* methods.

\*

\* @param request servlet request

\* @param response servlet response

\* @throws ServletException if a servlet-specific error occurs

\* @throws IOException if an I/O error occurs

\*/

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

int id = Integer.parseInt(request.getParameter("id"));

GetKeysFromServer gkfs = new GetKeysFromServer();

String privKey = gkfs.getPrivateKey(id);

String pubkey = gkfs.getPublicKey(id);

String key = ECCEncoder.binaryToText(privKey);

String initVector = ECCEncoder.binaryToText(pubkey);

System.out.println("Decryption Key "+key);

System.out.println("InitVector Key "+initVector);

String fileName = null;

String fileData = null;

String plainData = null;

PreparedStatement ps = null;

ResultSet rs = null;

HttpSession hs = request.getSession();

try(Connection con = DBConnectionn.getConnection()) {

String sqlQuery = "select cipherdata,filename from datafiles where id =?";

ps = con.prepareStatement(sqlQuery);

ps.setInt(1, id);

rs = ps.executeQuery();

if(rs.next()){

fileName = rs.getString("filename");

fileData = rs.getString("cipherdata");

plainData = AESImple.decrypt(fileData, key, initVector);

String ownername = hs.getAttribute("username").toString();

StoreDownloadHistory.storeDownloads(id,ownername,fileName,key,initVector);

}

} catch (Exception e) {

}

ServletContext context = getServletContext();

int BUFFER\_SIZE = 4096;

// sets MIME type for the file download

String mimeType = context.getMimeType(fileName);

if (mimeType == null) {

mimeType = "application/octet-stream";

}

// set content properties and header attributes for the response

response.setContentType(mimeType);

response.setContentLength(plainData.length());

String headerKey = "Content-Disposition";

String headerValue = String.format("attachment; filename=\"%s\"", fileName);

response.setHeader(headerKey, headerValue);

// writes the file to the client

OutputStream outStream = response.getOutputStream();

byte[] buffer = new byte[BUFFER\_SIZE];

int bytesRead = -1;

outStream.write(plainData.getBytes());

}

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">

/\*\*

\* Handles the HTTP <code>GET</code> method.

\*

\* @param request servlet request

\* @param response servlet response

\* @throws ServletException if a servlet-specific error occurs

\* @throws IOException if an I/O error occurs

\*/

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

/\*\*

\* Handles the HTTP <code>POST</code> method.

\*

\* @param request servlet request

\* @param response servlet response

\* @throws ServletException if a servlet-specific error occurs

\* @throws IOException if an I/O error occurs

\*/

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

/\*\*

\* Returns a short description of the servlet.

\*

\* @return a String containing servlet description

\*/

@Override

public String getServletInfo() {

return "Short description";

}// </editor-fold>

}

**EncryptionAlgorithm:**

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package EncryptionDecryption;

import java.math.BigInteger;

import java.security.MessageDigest;

import java.security.spec.KeySpec;

import javax.crypto.Cipher;

import javax.crypto.SecretKey;

import javax.crypto.SecretKeyFactory;

import javax.crypto.spec.DESedeKeySpec;

import org.apache.commons.codec.binary.Base64;

/\*\*

\*

\* @author Ramu Maloth

\*/

public class EncryptionAlgoritham {

private static final String UNICODE\_FORMAT = "UTF8";

public static final String DESEDE\_ENCRYPTION\_SCHEME =

"DESede";

private KeySpec ks;

private SecretKeyFactory skf;

private Cipher cipher;

byte[] arrayBytes;

private String myEncryptionKey;

private String myEncryptionScheme;

SecretKey key;

public EncryptionAlgoritham() throws Exception {

myEncryptionKey = "ThisIsSpartaThisIsSparta";

myEncryptionScheme = DESEDE\_ENCRYPTION\_SCHEME;

arrayBytes = myEncryptionKey.getBytes(UNICODE\_FORMAT);

ks = new DESedeKeySpec(arrayBytes);

skf = SecretKeyFactory.getInstance(myEncryptionScheme);

cipher = Cipher.getInstance(myEncryptionScheme);

key = skf.generateSecret(ks);

}

public String encrypt(String unencryptedString) {

String encryptedString = null;

try {

cipher.init(Cipher.ENCRYPT\_MODE, key);

byte[] plainText =

unencryptedString.getBytes(UNICODE\_FORMAT);

byte[] encryptedText = cipher.doFinal(plainText);

encryptedString = new String(Base64.encodeBase64(encryptedText));

} catch (Exception e) {

e.printStackTrace();

}

return encryptedString;

}

public String decrypt(String encryptedString) {

String decryptedText = null;

try {

cipher.init(Cipher.DECRYPT\_MODE, key);

byte[] encryptedText = Base64.decodeBase64(encryptedString);

byte[] plainText = cipher.doFinal(encryptedText);

decryptedText = new String(plainText);

} catch (Exception e) {

e.printStackTrace();

}

return decryptedText;

}

public String generatemacKey() {

String mackey = null;

try {

MessageDigest md = MessageDigest.getInstance("SHA1");

BigInteger bi1 = new BigInteger(md.digest());

String spl1 = bi1.toString();

mackey = bi1.toString(16);

} catch (Exception e) {

System.out.println("Getting mac Key error "+e.getMessage());

}

return mackey;

}

}