**Project Coding:**

**Index Page:**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>JSP Page</title>

<script>

</script>

<style>

\*{

margin: 0px;

padding: 0px;

}

.header{

background-image: url(img/j3.jpg);

width: 100%;

height: 500px;

background-size: 100% 500px;

animation-name: jaha;

animation-delay: 3s;

animation-duration: 10s;

animation-timing-function: ease-in-out;

animation-iteration-count: infinite;

animation-direction: alternate;

}

@keyframes jaha{

0% {

background-image: url(img/j1.jpg); background-size: 100% 500px;

}

45% {

background-image: url(img/j2.jpg); background-size: 100% 500px;

}

55% {

background-image: url(img/j5.jpg); background-size: 100% 500px;

}

100% {

background-image: url(img/j4.jpg); background-size: 100% 500px;

}

}

.nav{

width:100%;

height:80px;

background-color: #ffffff;

position: fixed;

}

.nav ul{

padding-top:15px;

margin-left:60%;

}

.nav ul li,a{

list-style-type: none;

float: left;

margin-left:5%;

text-decoration: none;

text-align: center;

color: #000;

font-family: Gill Sans, sans-serif;

font-size: 20PX;

padding-top: 20PX;

}

.nav a:hover{

color: #269abc;

}

.con1{

width:100%;

height:700px;

margin: auto;

}

.con1 h1,h2{

text-align: center;

color: #31b0d5;

margin-top:50px;

}

.con1 h3{

text-align: center;

color: gray;

font-family: Courier;

}

.con{

width:70%;

height: 300px;

margin: auto;

}

.con2{

width:100%;

height: 150px;

margin:auto;

}

.con2 .service-box{

width:360px;

height: 120px;

background-color: #000;

text-align: center;

color: #ffffff;

}

.con2 .service{

margin-top: 40px;

margin-left:5%;

float: left;

}

.con2 .service-box :hover{

background-color: #31b0d5;

}

.our\_serive{

width: 100%;

height: 400px;

background-image:url("img/pattern.jpg");

}

.our\_serive .service-img img{

width:100%;

height:150px;

}

.our\_serive .single-service{

width:20%;

float: left;

margin-left:60px;

}

.our\_serive .single-service h3{

text-align: center;

font-family: fantasy;

color: white;

}

.footer{

width:100%;

height: 100px;

}

.main1{

width: 100%;

background-color: gray;

}

</style>

</head>

<body>

<div class="nav">

<ul>

<li><a href="#">HOME</a></li>

<li><a href="#">CLIENT</a></li>

<li><a href="#">SERVICES</a></li>

<li><a href="#">ABOUT</a></li>

</ul>

</div>

<div class="header"></div>

<div class="main1">

</div>

<div class="con1" >

<h1 style="font-size:70px;font-family: Trebuchet MS">ABOUT US</h1>

<h3 style="width: 80%;margin: auto">The practice of using a network of remote servers hosted on the Internet to store, manage, and process data.</h3>

<div class="con2">

<div class="service">

<div class="about-img">

<img src="img/about1.jpg" alt="">

</div>

<div class="service-box">

<h3>Infrastructure as a Service (IaaS)</h3>

<p>Iaas is a form of cloud computing that provides virtualized computing resources and networks over the Internet.</p>

</div>

</div>

<div class="service">

<div class="about-img">

<img class="img-responsive" src="img/about2.jpg" alt="">

</div>

<div class="service-box">

<h3>Platform as a Service (PaaS)</h3>

<p> PaaS provides a platform and environment to allow developers to build applications and services over the internet</p>

</div>

</div>

<div class="service">

<div class="about-img">

<img class="img-responsive" src="img/about3.jpg" alt="">

</div>

<div class="service-box">

<h3>Software as a Service (SaaS)</h3>

<p>Saas is a software licensing and delivery model in which software is licensed on a subscription basis and is centrally hosted</p>

</div>

</div>

</div>

</div>

<div class="our\_serive">

<h2 style="color: #ffffff;padding-top:50px;font-size: 70px;font-family: Trebuchet MS">OUR SERVICES</h2>

<div class="single-service">

<div class="single-service-img">

<div class="service-img">

<img src="img/s1.jpg" alt="">

</div>

</div>

<h3>Public cloud</h3>

</div>

<div class="single-service">

<div class="single-service-img">

<div class="service-img">

<img class="img-responsive" src="img/s2.jpg" alt="">

</div>

</div>

<h3>Private Cloud</h3>

</div>

<div class="single-service">

<div class="single-service-img">

<div class="service-img">

<img class="img-responsive" src="img/s3.png" alt="">

</div>

</div>

<h3>Hybrid Cloud</h3>

</div>

<div class="single-service">

<div class="single-service-img">

<div class="service-img">

<img class="img-responsive" src="img/s4.png" alt="">

</div>

</div>

<h3>Community Cloud</h3>

</div>

<%

if(request.getParameter("mgs")!=null){

if(request.getParameter("mgs").equals("added")){

%>

<script>

alert("Successfully Registered");

</script>

<%

}

if(request.getParameter("mgs").equals("failed")){

%>

<script>

alert("login Failed");

</script>

<%

}

}

%>

</div>

<div class="footer">

<p style="padding-top: 50px;">&copy;CloudComputing.com All right reserved</p>

</div>

</body>

</html>

**File page:**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<%@page import="java.io.\*" %>

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">

<head>

<title>CloudSecureErasureCode</title>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<style type="text/css">

.basic

{

height:200px;

}

.basic1

{

width:378px;

height:71px;

margin-top:50px;

}

</style>

<!-- JS -->

<script type="text/javascript" src="js/jquery\_1.3.2.js"></script>

<script type="text/javascript" src="js/jqueryui.js"></script>

<script type="text/javascript" src="js/easing.js"></script>

<script type="text/javascript" src="js/jquery.cycle.all.js"></script>

<script type="text/javascript" src="js/custom.js"></script>

<!-- ENDS JS -->

<!-- superfish -->

<link rel="stylesheet" type="text/css" media="screen" href="css/superfish-custom.css" />

<script type="text/javascript" src="js/superfish-1.4.8/js/hoverIntent.js"></script>

<script type="text/javascript" src="js/superfish-1.4.8/js/superfish.js"></script>

<!-- ENDS superfish -->

<!-- CSS -->

<link rel="stylesheet" href="css/style.css" type="text/css" media="screen" />

<link rel="stylesheet" href="css/spring.css" type="text/css" media="screen" />

<!--[if IE 6]>

<link rel="stylesheet" type="text/css" media="screen" href="css/ie-hacks.css" />

<script type="text/javascript" src="js/DD\_belatedPNG.js"></script>

<script>/\* EXAMPLE \*/ DD\_belatedPNG.fix('\*');</script>

<![endif]-->

<!-- ENDS CSS -->

<!-- Cufon -->

<script src="js/cufon-yui.js" type="text/javascript"></script>

<script src="js/bebas\_400.font.js" type="text/javascript"></script>

<script type="text/javascript">Cufon.replace('.custom', { fontFamily: 'bebas', hover: true });</script>

<!-- /Cufon -->

<style type="text/css">

<!--

.style3 {font-size: 1000%}

.style6 {

color: #0066CC;

font-family: Geneva, Arial, Helvetica, sans-serif;

}

.style7 {

color: #0099FF;

font-family: Georgia, "Times New Roman", Times, serif;

}

.style8 {color: #990000}

body,td,th {

color: #993300;

}

-->

</style>

</head>

<body>

<!-- WRAPPER -->

<div id="wrapper">

<!-- navigation -->

<ul id="nav" class="sf-menu">

<li class="custom"><a href="clientlogin.jsp">Client Login </a></li>

<li class="custom"><a href="adminlogin.jsp">Admin Login </a></li>

<li class="custom"><a href="index.jsp">Home</a></li>

<span class="style3"></span>

</ul>

<!-- ENDS navigation -->

<!-- HEADER -->

<div id="header"><a href="index.jsp"><img src="img/logo.png" alt="" id="logo" /></a><img src="img/nav-arrow.png" alt="" id="arrow" class="arrow-home" />

<form action="#" method="post" id="search">

<p>

<input type="text" onfocus="defaultInput(this)" onblur="clearInput(this)" name="keyword" id="keyword" value="Search..." />

</p>

<p>

<input type="submit" id="go" value="" />

</p>

<div class="clear"></div>

</form>

</div>

<!-- ENDS HEADER -->

<!-- MAIN -->

<div id="main">

<!-- slideshow -->

<div id="slideshow">

<ul id="slides">

<li><img src="slides/01.jpg" alt="" /></li>

<li><img src="slides/02.jpg" alt="" /></li>

<li><img src="slides/03.jpg" alt="" /></li>

<li><img src="slides/04.jpg" alt="" /></li>

</ul>

<span></span> <a href="#"><img src="img/prev-slide.png" alt="" id="prev" /></a> <a href="#"><img src="img/next-slide.png" alt="" id="next" /></a> </div>

<div class="holder">

</div>

<div class="block">

</div>

<div class="block last">

</div>

<div class="basic">

<%

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

if ( file == null) {

out.print(" File name is missing ");

}

else {

out.print(" file Name is: " + file );

}

%>

</div>

</div>

</div>

>

</div>

<div id="footer"></div>

<script type="text/javascript">Cufon.now();</script>

<!-- ENDS start cufon -->

</body>

</html>

**Update Page:**

<%@page import="javax.mail.\*"%>

<%@page import="javax.mail.internet.\*"%>

<%@page import="javax.servlet.http.\*"%>

<%@page import="javax.servlet.\*"%>

<%@page import="javax.mail.Session"%>

<%@page import="javax.mail.Message"%>

<%@page import="javax.mail.internet.InternetAddress"%>

<%@page import="javax.mail.internet.MimeMessage"%>

<%@page import="java.math.BigInteger"%>

<%@page import="java.security.MessageDigest"%>

<%@page import="javax.crypto.\*"%>

<%@page import="javax.crypto.spec.\*"%>

<%@page import="java.util.\*"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@page import="com.oreilly.servlet.MultipartRequest"%>

<%@page import="java.io.BufferedWriter"%>

<%@page import="java.io.FileWriter"%>

<%@page import="java.io.File"%>

<%@page import="java.util.Random"%>

<%@page import="java.io.InputStreamReader"%>

<%@page import="java.io.BufferedReader"%>

<%@page import="java.io.DataInputStream"%>

<%@page import="java.io.FileInputStream"%>

<%@page import="java.sql.\*"%>

<%@page import="java.io.\*"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>JSP Page</title>

</head>

<body>

<% int a; %>

<%=a=(int) (Math.random() \* 1000) %>

<%

try

{

//String share=request.getParameter("shar");

// String kword=request.getParameter("fname");

// String kword=request.getParameter("fname");

String tkn1="";

String tkn2="";

String tkn3="";

String flname=null;

String name=null;

String shared\_users="";

String paramname=null;

String share=null;

String s1 = null;

String s2 = null;

String s3 = null;

int ii=0;

ArrayList list = new ArrayList();

String fname=null;

int f1=0;

File file1=null;

ServletContext context = getServletContext();

String dirName =context.getRealPath("/OFile/");

session.setAttribute("orginalfiledir", dirName);

String dirName1 =context.getRealPath("/split1/");

String dirName2 =context.getRealPath("/split2/");

String dirName3 =context.getRealPath("/split3/");

String dirNameE1 =context.getRealPath("/splitE1/");

String dirNameE2 =context.getRealPath("/splitE2/");

String dirNameE3 =context.getRealPath("/splitE3/");

MultipartRequest multi = new MultipartRequest(request, dirName,40 \* 1024 \* 1024); // 10MB

Enumeration params = multi.getParameterNames();

while (params.hasMoreElements())

{

paramname = (String) params.nextElement();

if(paramname.equalsIgnoreCase("fname"))

{

name=multi.getParameter(paramname);

}

if(paramname.equalsIgnoreCase("sharusers"))

{

shared\_users=multi.getParameter(paramname);

}

if(paramname.equalsIgnoreCase("shar"))

{

share=multi.getParameter(paramname);

}

}

Enumeration files = multi.getFileNames();

while (files.hasMoreElements())

{

fname = (String) files.nextElement();

if(fname.equals("d1"))

{

fname = null;

}

if(fname != null)

{

f1 = 1;

flname = multi.getFilesystemName(fname);

out.println("cp1");

String fPath = context.getRealPath("/OFile/"+flname);

out.println("cp2");

file1 = new File(fPath);

out.println("cp3");

FileInputStream fs = new FileInputStream(file1);

System.out.println("FileInputStream fs "+fs);

list.add(fs);

}

}

String ff = request.getParameter("file");

if (flname != null) {

String ss = flname.toString();

System.out.println(flname);

FileInputStream fis = new FileInputStream(dirName+"/"+flname);

String ffmt = "";

String sss = ss.replace('.', ' ');

String ffs[] = sss.split(" ");

String fn = ffs[0];

int fln = ffs.length;

if (fln > 0) {

ffmt = ffs[1];

System.out.println(ffmt);

}

DataInputStream dis = new DataInputStream(fis);

BufferedReader br = new BufferedReader(new InputStreamReader(dis));

BufferedReader br\_1 = new BufferedReader(new FileReader(dirName+"/"+flname));

String brr;

int line\_count=0;

while ((brr = br.readLine()) != null)

{

line\_count++;

}

int line\_div=line\_count/3;

System.out.println(line\_div);

boolean sp1=true;

boolean sp2=false;

boolean sp3=false;

int sp\_end=1;

int sp\_end\_add=0;

int splt\_count=0;

StringBuilder sbuild=new StringBuilder();

String sline;

if(line\_count>2)

{

while((sline=br\_1.readLine())!=null)

{

if(sp1)

{

System.out.println("sp1 : "+sline);

sbuild.append(sline);

sbuild.append(System.getProperty("line.separator"));

if(sp\_end==line\_div)

{

//----------------------SPLIT\_1-START---------------------------

splt\_count++;

s1 = fn + splt\_count + "." + ffmt;

File ff1 = new File(dirName1+"/"+fn+splt\_count+ "." +ffmt);

FileWriter fw = new FileWriter(ff1);

BufferedWriter bw = new BufferedWriter(fw);

bw.write(sbuild.toString());

bw.flush();

fw.close();

bw.close();

sbuild.setLength(0);

//-----------------SPLIT\_1-END--------------------

//-------------------------------------Token-1 value creation Starts--------------------------------------

FileReader fReader;

// String tkn1="";

try

{

fReader = new FileReader(dirName1+"/"+fn+splt\_count+ "." +ffmt);

BufferedReader reader = new BufferedReader(fReader);

TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

String cursor; //

String content = "";

int lines = 0;

int words = 0;

int chars = 0;

while((cursor = reader.readLine()) != null){

// count lines

lines += 1;

content += cursor;

// count words

String []\_words = cursor.split(" ");

for( String w : \_words)

{

Integer frequency = frequencyMap.get(w);

if(frequency == null){

frequency = 0;

}

frequencyMap.put(w, frequency + 1);

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

String[] ws=w.split("(?!^)");

for(String wsc : ws)

{

Integer frequency1 = frequencyMap1.get(wsc);

if(frequency1 == null){

frequency1 = 0;

}

frequencyMap1.put(wsc, frequency1 + 1);

}

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

words++;

}

}

System.out.println(frequencyMap1);

Iterator itr1=frequencyMap1.entrySet().iterator();

while(itr1.hasNext())

{

Map.Entry pairs=(Map.Entry) itr1.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn1=tkn1+pairs.getValue();

}

System.out.println(frequencyMap);

Iterator itr=frequencyMap.entrySet().iterator();

// String tkn="";

while(itr.hasNext())

{

Map.Entry pairs=(Map.Entry) itr.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn1=tkn1+pairs.getValue();

}

chars = content.length();

System.out.println("File test.txt has ");

System.out.println(chars + " Characters,");

System.out.println(words + " words and " + lines + " lines.");

tkn1=tkn1+lines+words+chars;

System.out.println("B-TOKEN-1 IS "+tkn1);

MessageDigest msdn=MessageDigest.getInstance("MD5");

msdn.update(tkn1.getBytes(),0,tkn1.length());

tkn1=new BigInteger(1,msdn.digest()).toString(16);

System.out.println("TOKEN-1 IS "+tkn1);

} catch (FileNotFoundException ex) {

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

System.out.println("File not found!");

} catch (IOException ex) {

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

System.out.println("An error has occured: " + ex.getMessage());

}

//-------------------------------------Token-1 value creation ENDS--------------------------------------

//--------------------- Encrypt------------------------

FileInputStream fise = new FileInputStream(dirName1+"/"+fn+splt\_count+ "." +ffmt);

FileOutputStream fos = new FileOutputStream(dirNameE1+"/"+fn+splt\_count+ "." +ffmt);

// InputStream is=fise;

// OutputStream os=fos;

String key="Murali123";

int mode=Cipher.ENCRYPT\_MODE;

DESKeySpec dks = new DESKeySpec(key.getBytes());

SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

SecretKey desKey = skf.generateSecret(dks);

Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

if (mode == Cipher.ENCRYPT\_MODE) {

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

CipherInputStream cis = new CipherInputStream(fise, cipher);

//doCopy(cis, os);

byte[] bytes = new byte[64];

int numBytes;

while ((numBytes = cis.read(bytes)) != -1) {

fos.write(bytes, 0, numBytes);

}

fos.flush();

fos.close();

cis.close();

}

//--------------------- END-Encrypt------------------------

sp1=false;

sp2=true;

sp\_end\_add=sp\_end;

}

}

else if(sp2)

{

System.out.println("sp2 :"+sline);

sbuild.append(sline);

sbuild.append(System.getProperty("line.separator"));

if(sp\_end==(sp\_end\_add+line\_div))

{

//----------------------SPLIT\_2-START---------------------------

splt\_count++;

s2 = fn + splt\_count + "." + ffmt;

File ff1 = new File(dirName2+"/"+fn+splt\_count+ "." +ffmt);

FileWriter fw = new FileWriter(ff1);

BufferedWriter bw = new BufferedWriter(fw);

bw.write(sbuild.toString());

bw.flush();

fw.close();

bw.close();

sbuild.setLength(0);

//----------------------SPLIT\_2-END---------------------------

//-------------------------------------Token-2 value creation STARTS--------------------------------------

FileReader fReader;

// String tkn2="";

try

{

fReader = new FileReader(dirName2+"/"+fn+splt\_count+ "." +ffmt);

BufferedReader reader = new BufferedReader(fReader);

TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

String cursor; //

String content = "";

int lines = 0;

int words = 0;

int chars = 0;

while((cursor = reader.readLine()) != null){

// count lines

lines += 1;

content += cursor;

// count words

String []\_words = cursor.split(" ");

for( String w : \_words)

{

Integer frequency = frequencyMap.get(w);

if(frequency == null){

frequency = 0;

}

frequencyMap.put(w, frequency + 1);

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

String[] ws=w.split("(?!^)");

for(String wsc : ws)

{

Integer frequency1 = frequencyMap1.get(wsc);

if(frequency1 == null){

frequency1 = 0;

}

frequencyMap1.put(wsc, frequency1 + 1);

}

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

words++;

}

}

System.out.println(frequencyMap1);

Iterator itr1=frequencyMap1.entrySet().iterator();

while(itr1.hasNext())

{

Map.Entry pairs=(Map.Entry) itr1.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn2=tkn2+pairs.getValue();

}

System.out.println(frequencyMap);

Iterator itr=frequencyMap.entrySet().iterator();

// String tkn="";

while(itr.hasNext())

{

Map.Entry pairs=(Map.Entry) itr.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn2=tkn2+pairs.getValue();

}

chars = content.length();

System.out.println("File test.txt has ");

System.out.println(chars + " Characters,");

System.out.println(words + " words and " + lines + " lines.");

tkn2=tkn2+lines+words+chars;

System.out.println("B-TOKEN-2 IS "+tkn2);

MessageDigest msdn=MessageDigest.getInstance("MD5");

msdn.update(tkn2.getBytes(),0,tkn2.length());

tkn2=new BigInteger(1,msdn.digest()).toString(16);

System.out.println("TOKEN-2 IS "+tkn2);

} catch (FileNotFoundException ex) {

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

System.out.println("File not found!");

} catch (IOException ex) {

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

System.out.println("An error has occured: " + ex.getMessage());

}

//-------------------------------------Token-2 value creation ENDS--------------------------------------

//--------------------- Encrypt------------------------

FileInputStream fise = new FileInputStream(dirName2+"/"+fn+splt\_count+ "." +ffmt);

FileOutputStream fos = new FileOutputStream(dirNameE2+"/"+fn+splt\_count+ "." +ffmt);

// InputStream is=fise;

// OutputStream os=fos;

String key="Murali123";

int mode=Cipher.ENCRYPT\_MODE;

DESKeySpec dks = new DESKeySpec(key.getBytes());

SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

SecretKey desKey = skf.generateSecret(dks);

Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

if (mode == Cipher.ENCRYPT\_MODE) {

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

CipherInputStream cis = new CipherInputStream(fise, cipher);

//doCopy(cis, os);

byte[] bytes = new byte[64];

int numBytes;

while ((numBytes = cis.read(bytes)) != -1) {

fos.write(bytes, 0, numBytes);

}

fos.flush();

fos.close();

cis.close();

}

//--------------------- END-Encrypt------------------------

sp2=false;

sp3=true;

sp\_end\_add=sp\_end;

}

}

else if(sp3)

{

System.out.println("sp3 :"+sline);

sbuild.append(sline);

sbuild.append(System.getProperty("line.separator"));

if(sp\_end==(line\_count))

{

//----------------------SPLIT\_3-START---------------------------

splt\_count++;

s3 = fn + splt\_count + "." + ffmt;

File ff1 = new File(dirName3+"/"+fn+splt\_count+ "." +ffmt);

FileWriter fw = new FileWriter(ff1);

BufferedWriter bw = new BufferedWriter(fw);

bw.write(sbuild.toString());

bw.flush();

fw.close();

bw.close();

sbuild.setLength(0);

//----------------------SPLIT\_3-END---------------------------

//-------------------------------------Token-3 value creation STARTS--------------------------------------

FileReader fReader;

try

{

fReader = new FileReader(dirName3+"/"+fn+splt\_count+ "." +ffmt);

BufferedReader reader = new BufferedReader(fReader);

TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

String cursor; //

String content = "";

int lines = 0;

int words = 0;

int chars = 0;

while((cursor = reader.readLine()) != null){

// count lines

lines += 1;

content += cursor;

// count words

String []\_words = cursor.split(" ");

for( String w : \_words)

{

Integer frequency = frequencyMap.get(w);

if(frequency == null){

frequency = 0;

}

frequencyMap.put(w, frequency + 1);

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

String[] ws=w.split("(?!^)");

for(String wsc : ws)

{

Integer frequency1 = frequencyMap1.get(wsc);

if(frequency1 == null){

frequency1 = 0;

}

frequencyMap1.put(wsc, frequency1 + 1);

}

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

words++;

}

}

System.out.println(frequencyMap1);

Iterator itr1=frequencyMap1.entrySet().iterator();

while(itr1.hasNext())

{

Map.Entry pairs=(Map.Entry) itr1.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn3=tkn3+pairs.getValue();

}

System.out.println(frequencyMap);

Iterator itr=frequencyMap.entrySet().iterator();

// String tkn="";

while(itr.hasNext())

{

Map.Entry pairs=(Map.Entry) itr.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn3=tkn3+pairs.getValue();

}

chars = content.length();

System.out.println("File test.txt has ");

System.out.println(chars + " Characters,");

System.out.println(words + " words and " + lines + " lines.");

tkn3=tkn3+lines+words+chars;

System.out.println("B-TOKEN-3 IS "+tkn3);

MessageDigest msdn=MessageDigest.getInstance("MD5");

msdn.update(tkn3.getBytes(),0,tkn3.length());

tkn3=new BigInteger(1,msdn.digest()).toString(16);

System.out.println("TOKEN-3 IS "+tkn3);

} catch (FileNotFoundException ex) {

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

System.out.println("File not found!");

} catch (IOException ex) {

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

System.out.println("An error has occured: " + ex.getMessage());

}

//-------------------------------------Token-3 value creation ENDS--------------------------------------

//--------------------- Encrypt------------------------

FileInputStream fise = new FileInputStream(dirName3+"/"+fn+splt\_count+ "." +ffmt);

FileOutputStream fos = new FileOutputStream(dirNameE3+"/"+fn+splt\_count+ "." +ffmt);

// InputStream is=fise;

// OutputStream os=fos;

String key="Murali123";

int mode=Cipher.ENCRYPT\_MODE;

DESKeySpec dks = new DESKeySpec(key.getBytes());

SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

SecretKey desKey = skf.generateSecret(dks);

Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

if (mode == Cipher.ENCRYPT\_MODE) {

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

CipherInputStream cis = new CipherInputStream(fise, cipher);

//doCopy(cis, os);

byte[] bytes = new byte[64];

int numBytes;

while ((numBytes = cis.read(bytes)) != -1) {

fos.write(bytes, 0, numBytes);

}

fos.flush();

fos.close();

cis.close();

}

//--------------------- END-Encrypt------------------------

sp2=false;

sp3=false;

}

}

sp\_end++;

}

}

else

{

ArrayList<String> als=new ArrayList<String>();

while((sline=br\_1.readLine())!= null)

{

String[] sln=sline.split(" ");

for(String sn : sln)

{

als.add(sn);

}

}

int al=als.size();

if(al>2)

{

int la=al/3;

for(String ll : als)

{

if(sp1)

{

System.out.println("sp1 : "+ll);

sbuild.append(ll);

if(sp\_end==la)

{

//----------------------SPLIT\_1-START---------------------------

splt\_count++;

File ff1 = new File(dirName1+"/"+fn+splt\_count+ "." +ffmt);

FileWriter fw = new FileWriter(ff1);

BufferedWriter bw = new BufferedWriter(fw);

bw.write(sbuild.toString());

bw.flush();

fw.close();

bw.close();

sbuild.setLength(0);

//-----------------SPLIT\_1-END--------------------

//-------------------------------------Token-1 value creation Starts--------------------------------------

FileReader fReader;

// String tkn1="";

try

{

fReader = new FileReader(dirName1+"/"+fn+splt\_count+ "." +ffmt);

BufferedReader reader = new BufferedReader(fReader);

TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

String cursor; //

String content = "";

int lines = 0;

int words = 0;

int chars = 0;

while((cursor = reader.readLine()) != null){

// count lines

lines += 1;

content += cursor;

// count words

String []\_words = cursor.split(" ");

for( String w : \_words)

{

Integer frequency = frequencyMap.get(w);

if(frequency == null){

frequency = 0;

}

frequencyMap.put(w, frequency + 1);

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

String[] ws=w.split("(?!^)");

for(String wsc : ws)

{

Integer frequency1 = frequencyMap1.get(wsc);

if(frequency1 == null){

frequency1 = 0;

}

frequencyMap1.put(wsc, frequency1 + 1);

}

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

words++;

}

}

System.out.println(frequencyMap1);

Iterator itr1=frequencyMap1.entrySet().iterator();

while(itr1.hasNext())

{

Map.Entry pairs=(Map.Entry) itr1.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn1=tkn1+pairs.getValue();

}

System.out.println(frequencyMap);

Iterator itr=frequencyMap.entrySet().iterator();

// String tkn="";

while(itr.hasNext())

{

Map.Entry pairs=(Map.Entry) itr.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn1=tkn1+pairs.getValue();

}

chars = content.length();

System.out.println("File test.txt has ");

System.out.println(chars + " Characters,");

System.out.println(words + " words and " + lines + " lines.");

tkn1=tkn1+lines+words+chars;

System.out.println("B-TOKEN-1 IS "+tkn1);

MessageDigest msdn=MessageDigest.getInstance("MD5");

msdn.update(tkn1.getBytes(),0,tkn1.length());

tkn1=new BigInteger(1,msdn.digest()).toString(16);

System.out.println("TOKEN-1 IS "+tkn1);

} catch (FileNotFoundException ex) {

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

System.out.println("File not found!");

} catch (IOException ex) {

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

System.out.println("An error has occured: " + ex.getMessage());

}

//-------------------------------------Token-1 value creation ENDS--------------------------------------

//--------------------- Encrypt------------------------

FileInputStream fise = new FileInputStream(dirName1+"/"+fn+splt\_count+ "." +ffmt);

FileOutputStream fos = new FileOutputStream(dirNameE1+"/"+fn+splt\_count+ "." +ffmt);

// InputStream is=fise;

// OutputStream os=fos;

String key="Murali123";

int mode=Cipher.ENCRYPT\_MODE;

DESKeySpec dks = new DESKeySpec(key.getBytes());

SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

SecretKey desKey = skf.generateSecret(dks);

Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

if (mode == Cipher.ENCRYPT\_MODE) {

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

CipherInputStream cis = new CipherInputStream(fise, cipher);

//doCopy(cis, os);

byte[] bytes = new byte[64];

int numBytes;

while ((numBytes = cis.read(bytes)) != -1) {

fos.write(bytes, 0, numBytes);

}

fos.flush();

fos.close();

cis.close();

}

//--------------------- END-Encrypt------------------------

sp1=false;

sp2=true;

sp\_end\_add=sp\_end;

}

}

else if(sp2)

{

System.out.println("sp2 :"+ll);

sbuild.append(ll);

if(sp\_end==(sp\_end\_add+la))

{

//----------------------SPLIT\_2-START---------------------------

splt\_count++;

File ff1 = new File(dirName1+"/"+fn+splt\_count+ "." +ffmt);

FileWriter fw = new FileWriter(ff1);

BufferedWriter bw = new BufferedWriter(fw);

bw.write(sbuild.toString());

bw.flush();

fw.close();

bw.close();

sbuild.setLength(0);

//----------------------SPLIT\_2-END---------------------------

//-------------------------------------Token-2 value creation STARTS--------------------------------------

FileReader fReader;

// String tkn2="";

try

{

fReader = new FileReader(dirName2+"/"+fn+splt\_count+ "." +ffmt);

BufferedReader reader = new BufferedReader(fReader);

TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

String cursor; //

String content = "";

int lines = 0;

int words = 0;

int chars = 0;

while((cursor = reader.readLine()) != null){

// count lines

lines += 1;

content += cursor;

// count words

String []\_words = cursor.split(" ");

for( String w : \_words)

{

Integer frequency = frequencyMap.get(w);

if(frequency == null){

frequency = 0;

}

frequencyMap.put(w, frequency + 1);

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

String[] ws=w.split("(?!^)");

for(String wsc : ws)

{

Integer frequency1 = frequencyMap1.get(wsc);

if(frequency1 == null){

frequency1 = 0;

}

frequencyMap1.put(wsc, frequency1 + 1);

}

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

words++;

}

}

System.out.println(frequencyMap1);

Iterator itr1=frequencyMap1.entrySet().iterator();

while(itr1.hasNext())

{

Map.Entry pairs=(Map.Entry) itr1.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn2=tkn2+pairs.getValue();

}

System.out.println(frequencyMap);

Iterator itr=frequencyMap.entrySet().iterator();

// String tkn="";

while(itr.hasNext())

{

Map.Entry pairs=(Map.Entry) itr.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn2=tkn2+pairs.getValue();

}

chars = content.length();

System.out.println("File test.txt has ");

System.out.println(chars + " Characters,");

System.out.println(words + " words and " + lines + " lines.");

tkn2=tkn2+lines+words+chars;

System.out.println("B-TOKEN-2 IS "+tkn2);

MessageDigest msdn=MessageDigest.getInstance("MD5");

msdn.update(tkn2.getBytes(),0,tkn2.length());

tkn2=new BigInteger(1,msdn.digest()).toString(16);

System.out.println("TOKEN-2 IS "+tkn2);

} catch (FileNotFoundException ex) {

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

System.out.println("File not found!");

} catch (IOException ex) {

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

System.out.println("An error has occured: " + ex.getMessage());

}

//-------------------------------------Token-2 value creation ENDS--------------------------------------

//--------------------- Encrypt------------------------

FileInputStream fise = new FileInputStream(dirName2+"/"+fn+splt\_count+ "." +ffmt);

FileOutputStream fos = new FileOutputStream(dirNameE2+"/"+fn+splt\_count+ "." +ffmt);

// InputStream is=fise;

// OutputStream os=fos;

String key="Murali123";

int mode=Cipher.ENCRYPT\_MODE;

DESKeySpec dks = new DESKeySpec(key.getBytes());

SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

SecretKey desKey = skf.generateSecret(dks);

Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

if (mode == Cipher.ENCRYPT\_MODE) {

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

CipherInputStream cis = new CipherInputStream(fise, cipher);

//doCopy(cis, os);

byte[] bytes = new byte[64];

int numBytes;

while ((numBytes = cis.read(bytes)) != -1) {

fos.write(bytes, 0, numBytes);

}

fos.flush();

fos.close();

cis.close();

}

//--------------------- END-Encrypt------------------------

sp2=false;

sp3=true;

sp\_end\_add=sp\_end;

}

}

else if(sp3)

{

System.out.println("sp3 :"+ll);

sbuild.append(ll);

if(sp\_end==(line\_count-sp\_end\_add))

{

//----------------------SPLIT\_3-START---------------------------

splt\_count++;

File ff1 = new File(dirName1+"/"+fn+splt\_count+ "." +ffmt);

FileWriter fw = new FileWriter(ff1);

BufferedWriter bw = new BufferedWriter(fw);

bw.write(sbuild.toString());

bw.flush();

fw.close();

bw.close();

sbuild.setLength(0);

//----------------------SPLIT\_3-END---------------------------

//-------------------------------------Token-3 value creation STARTS--------------------------------------

FileReader fReader;

try

{

fReader = new FileReader(dirName3+"/"+fn+splt\_count+ "." +ffmt);

BufferedReader reader = new BufferedReader(fReader);

TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

String cursor; //

String content = "";

int lines = 0;

int words = 0;

int chars = 0;

while((cursor = reader.readLine()) != null){

// count lines

lines += 1;

content += cursor;

// count words

String []\_words = cursor.split(" ");

for( String w : \_words)

{

Integer frequency = frequencyMap.get(w);

if(frequency == null){

frequency = 0;

}

frequencyMap.put(w, frequency + 1);

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

String[] ws=w.split("(?!^)");

for(String wsc : ws)

{

Integer frequency1 = frequencyMap1.get(wsc);

if(frequency1 == null){

frequency1 = 0;

}

frequencyMap1.put(wsc, frequency1 + 1);

}

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

words++;

}

}

System.out.println(frequencyMap1);

Iterator itr1=frequencyMap1.entrySet().iterator();

while(itr1.hasNext())

{

Map.Entry pairs=(Map.Entry) itr1.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn3=tkn3+pairs.getValue();

}

System.out.println(frequencyMap);

Iterator itr=frequencyMap.entrySet().iterator();

// String tkn="";

while(itr.hasNext())

{

Map.Entry pairs=(Map.Entry) itr.next();

System.out.println(pairs.getKey() + "----" + pairs.getValue());

tkn3=tkn3+pairs.getValue();

}

chars = content.length();

System.out.println("File test.txt has ");

System.out.println(chars + " Characters,");

System.out.println(words + " words and " + lines + " lines.");

tkn3=tkn3+lines+words+chars;

System.out.println("B-TOKEN-3 IS "+tkn3);

MessageDigest msdn=MessageDigest.getInstance("MD5");

msdn.update(tkn3.getBytes(),0,tkn3.length());

tkn3=new BigInteger(1,msdn.digest()).toString(16);

System.out.println("TOKEN-3 IS "+tkn3);

} catch (FileNotFoundException ex) {

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

System.out.println("File not found!");

} catch (IOException ex) {

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

System.out.println("An error has occured: " + ex.getMessage());

}

//-------------------------------------Token-3 value creation ENDS--------------------------------------

//--------------------- Encrypt------------------------

FileInputStream fise = new FileInputStream(dirName3+"/"+fn+splt\_count+ "." +ffmt);

FileOutputStream fos = new FileOutputStream(dirNameE3+"/"+fn+splt\_count+ "." +ffmt);

// InputStream is=fise;

// OutputStream os=fos;

String key="Murali123";

int mode=Cipher.ENCRYPT\_MODE;

DESKeySpec dks = new DESKeySpec(key.getBytes());

SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

SecretKey desKey = skf.generateSecret(dks);

Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

if (mode == Cipher.ENCRYPT\_MODE) {

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

CipherInputStream cis = new CipherInputStream(fise, cipher);

//doCopy(cis, os);

byte[] bytes = new byte[64];

int numBytes;

while ((numBytes = cis.read(bytes)) != -1) {

fos.write(bytes, 0, numBytes);

}

fos.flush();

fos.close();

cis.close();

}

//--------------------- END-Encrypt------------------------

sp2=false;

sp3=false;

}

}

sp\_end++;

}

}

}

// StringBuilder sb = new StringBuilder();

// int chr\_cunt=0;

// while ((brr = br.readLine()) != null) {

// String[] brl = brr.split("(?!^)");

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

// sb.append(brl[i]);

// chr\_cunt+=brl.length;

// }

// }

// int si = sb.length() / 2;

// int si = ((chr\_cunt+50) / 3);

// ii=si;

//out.println(si);

// System.out.println("SI " + si);

// ------------- split the file............

// char cr[] = new char[si];

// StringBuffer fid = new StringBuffer();

//

// Random rm = new Random();

//

// int fc = 0;

// FileInputStream fr = new FileInputStream(dirName+"/"+flname);

//

// DataInputStream dis1 = new DataInputStream(fr);

// BufferedReader br1 = new BufferedReader(new InputStreamReader(dis1));

//

// for (int j = 0; br1.read(cr) != -1; j++) {

// StringBuffer sb1 = new StringBuffer();

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

// sb1.append(cr[i]);

// }

// if (j == 0) {

//

// s1 = fn + (j + 1) + "." + ffmt;

//

//

// File ff1 = new File(dirName1+"/"+fn+(j + 1)+ "." +ffmt);

// FileWriter fw = new FileWriter(ff1);

// BufferedWriter bw = new BufferedWriter(fw);

// bw.write(cr);

// bw.flush();

// fw.close();

// bw.close();

//-------------------------------------Token-1 value creation Starts--------------------------------------

// FileReader fReader;

// String tkn1="";

// try

//{

//fReader = new FileReader(dirName1+"/"+fn+(j + 1)+ "." +ffmt);

// BufferedReader reader = new BufferedReader(fReader);

//

// TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

// TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

//

// String cursor; //

// String content = "";

// int lines = 0;

// int words = 0;

// int chars = 0;

// while((cursor = reader.readLine()) != null){

// // count lines

// lines += 1;

// content += cursor;

//

// // count words

// String []\_words = cursor.split(" ");

// for( String w : \_words)

// {

// Integer frequency = frequencyMap.get(w);

// if(frequency == null){

// frequency = 0;

// }

// frequencyMap.put(w, frequency + 1);

//

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

// String[] ws=w.split("(?!^)");

// for(String wsc : ws)

// {

// Integer frequency1 = frequencyMap1.get(wsc);

// if(frequency1 == null){

// frequency1 = 0;

// }

// frequencyMap1.put(wsc, frequency1 + 1);

// }

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

// words++;

// }

//

// }

// System.out.println(frequencyMap1);

// Iterator itr1=frequencyMap1.entrySet().iterator();

//

// while(itr1.hasNext())

// {

// Map.Entry pairs=(Map.Entry) itr1.next();

// System.out.println(pairs.getKey() + "----" + pairs.getValue());

// tkn1=tkn1+pairs.getValue();

// }

// System.out.println(frequencyMap);

// Iterator itr=frequencyMap.entrySet().iterator();

// // String tkn="";

// while(itr.hasNext())

// {

// Map.Entry pairs=(Map.Entry) itr.next();

// System.out.println(pairs.getKey() + "----" + pairs.getValue());

// tkn1=tkn1+pairs.getValue();

// }

// chars = content.length();

//

// System.out.println("File test.txt has ");

// System.out.println(chars + " Characters,");

// System.out.println(words + " words and " + lines + " lines.");

// tkn1=tkn1+lines+words+chars;

// System.out.println("B-TOKEN-1 IS "+tkn1);

// MessageDigest msdn=MessageDigest.getInstance("MD5");

//msdn.update(tkn1.getBytes(),0,tkn1.length());

// tkn1=new BigInteger(1,msdn.digest()).toString(16);

//

// System.out.println("TOKEN-1 IS "+tkn1);

//

// } catch (FileNotFoundException ex) {

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

// System.out.println("File not found!");

// } catch (IOException ex) {

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

// System.out.println("An error has occured: " + ex.getMessage());

// }

//

//

//

////-------------------------------------Token-1 value creation ENDS--------------------------------------

// //--------------------- Encrypt------------------------

//

// FileInputStream fise = new FileInputStream(dirName1+"/"+fn+(j + 1)+ "." +ffmt);

// FileOutputStream fos = new FileOutputStream(dirNameE1+"/"+fn+(j + 1)+ "." +ffmt);

// // InputStream is=fise;

// // OutputStream os=fos;

// String key="Murali123";

//

// int mode=Cipher.ENCRYPT\_MODE;

// DESKeySpec dks = new DESKeySpec(key.getBytes());

// SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

// SecretKey desKey = skf.generateSecret(dks);

// Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

// if (mode == Cipher.ENCRYPT\_MODE) {

// cipher.init(Cipher.ENCRYPT\_MODE, desKey);

// CipherInputStream cis = new CipherInputStream(fise, cipher);

// //doCopy(cis, os);

//

// byte[] bytes = new byte[64];

// int numBytes;

// while ((numBytes = cis.read(bytes)) != -1) {

// fos.write(bytes, 0, numBytes);

// }

// fos.flush();

// fos.close();

// cis.close();

// }

//

//

//

// //--------------------- END-Encrypt------------------------

// } else if (j == 1) {

//

//

//

// s2 = fn + (j + 1) + "." + ffmt;

//

//

// File ff1 = new File(dirName2+"/" + fn + (j + 1) + "." + ffmt);

// FileWriter fw = new FileWriter(ff1);

// BufferedWriter bw = new BufferedWriter(fw);

// bw.write(cr);

// bw.flush();

// fw.close();

// bw.close();

//

////-------------------------------------Token-2 value creation STARTS--------------------------------------

// FileReader fReader;

//// String tkn2="";

// try

//{

//fReader = new FileReader(dirName2+"/"+fn+(j + 1)+ "." +ffmt);

// BufferedReader reader = new BufferedReader(fReader);

//

// TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

// TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

//

// String cursor; //

// String content = "";

// int lines = 0;

// int words = 0;

// int chars = 0;

// while((cursor = reader.readLine()) != null){

// // count lines

// lines += 1;

// content += cursor;

//

// // count words

// String []\_words = cursor.split(" ");

// for( String w : \_words)

// {

// Integer frequency = frequencyMap.get(w);

// if(frequency == null){

// frequency = 0;

// }

// frequencyMap.put(w, frequency + 1);

//

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

// String[] ws=w.split("(?!^)");

// for(String wsc : ws)

// {

// Integer frequency1 = frequencyMap1.get(wsc);

// if(frequency1 == null){

// frequency1 = 0;

// }

// frequencyMap1.put(wsc, frequency1 + 1);

// }

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

// words++;

// }

//

// }

// System.out.println(frequencyMap1);

// Iterator itr1=frequencyMap1.entrySet().iterator();

//

// while(itr1.hasNext())

// {

// Map.Entry pairs=(Map.Entry) itr1.next();

// System.out.println(pairs.getKey() + "----" + pairs.getValue());

// tkn2=tkn2+pairs.getValue();

// }

// System.out.println(frequencyMap);

// Iterator itr=frequencyMap.entrySet().iterator();

// // String tkn="";

// while(itr.hasNext())

// {

// Map.Entry pairs=(Map.Entry) itr.next();

// System.out.println(pairs.getKey() + "----" + pairs.getValue());

// tkn2=tkn2+pairs.getValue();

// }

// chars = content.length();

//

// System.out.println("File test.txt has ");

// System.out.println(chars + " Characters,");

// System.out.println(words + " words and " + lines + " lines.");

// tkn2=tkn2+lines+words+chars;

// System.out.println("B-TOKEN-2 IS "+tkn2);

// MessageDigest msdn=MessageDigest.getInstance("MD5");

//msdn.update(tkn2.getBytes(),0,tkn2.length());

// tkn2=new BigInteger(1,msdn.digest()).toString(16);

// System.out.println("TOKEN-2 IS "+tkn2);

//

// } catch (FileNotFoundException ex) {

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

// System.out.println("File not found!");

// } catch (IOException ex) {

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

// System.out.println("An error has occured: " + ex.getMessage());

// }

//

//

//

////-------------------------------------Token-2 value creation ENDS--------------------------------------

//

// //--------------------- Encrypt------------------------

//

// FileInputStream fise = new FileInputStream(dirName2+"/"+fn+(j + 1)+ "." +ffmt);

// FileOutputStream fos = new FileOutputStream(dirNameE2+"/"+fn+(j + 1)+ "." +ffmt);

// // InputStream is=fise;

// // OutputStream os=fos;

// String key="Murali123";

//

// int mode=Cipher.ENCRYPT\_MODE;

// DESKeySpec dks = new DESKeySpec(key.getBytes());

// SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

// SecretKey desKey = skf.generateSecret(dks);

// Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

// if (mode == Cipher.ENCRYPT\_MODE) {

// cipher.init(Cipher.ENCRYPT\_MODE, desKey);

// CipherInputStream cis = new CipherInputStream(fise, cipher);

// //doCopy(cis, os);

//

// byte[] bytes = new byte[64];

// int numBytes;

// while ((numBytes = cis.read(bytes)) != -1) {

// fos.write(bytes, 0, numBytes);

// }

// fos.flush();

// fos.close();

// cis.close();

// }

//

//

//

// //--------------------- END-Encrypt------------------------

//

//

//

// } else {

//

//

// s3 = fn + (j + 1) + "." + ffmt;

//

//

//

// File ff1 = new File(dirName3+"/" + fn + (j + 1) + "." + ffmt);

// FileWriter fw = new FileWriter(ff1);

// BufferedWriter bw = new BufferedWriter(fw);

// bw.write(cr);

// bw.flush();

// fw.close();

// bw.close();

//

////-------------------------------------Token-3 value creation STARTS--------------------------------------

// FileReader fReader;

//

// try

//{

//fReader = new FileReader(dirName3+"/"+fn+(j + 1)+ "." +ffmt);

// BufferedReader reader = new BufferedReader(fReader);

//

// TreeMap<String, Integer> frequencyMap = new TreeMap<String, Integer>();

// TreeMap<String, Integer> frequencyMap1 = new TreeMap<String, Integer>();

//

// String cursor; //

// String content = "";

// int lines = 0;

// int words = 0;

// int chars = 0;

// while((cursor = reader.readLine()) != null){

// // count lines

// lines += 1;

// content += cursor;

//

// // count words

// String []\_words = cursor.split(" ");

// for( String w : \_words)

// {

// Integer frequency = frequencyMap.get(w);

// if(frequency == null){

// frequency = 0;

// }

// frequencyMap.put(w, frequency + 1);

//

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

// String[] ws=w.split("(?!^)");

// for(String wsc : ws)

// {

// Integer frequency1 = frequencyMap1.get(wsc);

// if(frequency1 == null){

// frequency1 = 0;

// }

// frequencyMap1.put(wsc, frequency1 + 1);

// }

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

// words++;

// }

//

// }

// System.out.println(frequencyMap1);

// Iterator itr1=frequencyMap1.entrySet().iterator();

//

// while(itr1.hasNext())

// {

// Map.Entry pairs=(Map.Entry) itr1.next();

// System.out.println(pairs.getKey() + "----" + pairs.getValue());

// tkn3=tkn3+pairs.getValue();

// }

// System.out.println(frequencyMap);

// Iterator itr=frequencyMap.entrySet().iterator();

// // String tkn="";

// while(itr.hasNext())

// {

// Map.Entry pairs=(Map.Entry) itr.next();

// System.out.println(pairs.getKey() + "----" + pairs.getValue());

// tkn3=tkn3+pairs.getValue();

// }

// chars = content.length();

//

// System.out.println("File test.txt has ");

// System.out.println(chars + " Characters,");

// System.out.println(words + " words and " + lines + " lines.");

// tkn3=tkn3+lines+words+chars;

// System.out.println("B-TOKEN-3 IS "+tkn3);

// MessageDigest msdn=MessageDigest.getInstance("MD5");

//msdn.update(tkn3.getBytes(),0,tkn3.length());

// tkn3=new BigInteger(1,msdn.digest()).toString(16);

// System.out.println("TOKEN-3 IS "+tkn3);

//

// } catch (FileNotFoundException ex) {

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

// System.out.println("File not found!");

// } catch (IOException ex) {

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

// System.out.println("An error has occured: " + ex.getMessage());

// }

//

//

//

////-------------------------------------Token-3 value creation ENDS--------------------------------------

//

//

// //--------------------- Encrypt------------------------

//

// FileInputStream fise = new FileInputStream(dirName3+"/"+fn+(j + 1)+ "." +ffmt);

// FileOutputStream fos = new FileOutputStream(dirNameE3+"/"+fn+(j + 1)+ "." +ffmt);

// // InputStream is=fise;

// // OutputStream os=fos;

// String key="Murali123";

//

// int mode=Cipher.ENCRYPT\_MODE;

// DESKeySpec dks = new DESKeySpec(key.getBytes());

// SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

// SecretKey desKey = skf.generateSecret(dks);

// Cipher cipher = Cipher.getInstance("DES"); // DES/ECB/PKCS5Padding for SunJCE

// if (mode == Cipher.ENCRYPT\_MODE) {

// cipher.init(Cipher.ENCRYPT\_MODE, desKey);

// CipherInputStream cis = new CipherInputStream(fise, cipher);

// //doCopy(cis, os);

//

// byte[] bytes = new byte[64];

// int numBytes;

// while ((numBytes = cis.read(bytes)) != -1) {

// fos.write(bytes, 0, numBytes);

// }

// fos.flush();

// fos.close();

// cis.close();

// }

//

//

//

// //--------------------- END-Encrypt------------------------

// }

//

// }

// }

}

String usrnme = session.getAttribute("usern").toString();

int k = usrnme.length();

// String f = request.getParameter("fname");

String f=name;

int l = f.length();

// String loc = request.getParameter("file");

String loc=flname;

String m = Integer.toString(k);

String n = Integer.toString(l);

String skey = (m+n);

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

//Connection conn = DriverManager.getConnection("jdbc:mysql://ec2-50-19-213-178.compute-1.amazonaws.com:3306/group","group","group");

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cloudsecure","root","admin");

// Connection conn = DriverManager.getConnection("jdbc:mysql://mysql92594-SCloud.j.layershift.co.uk/cloudsecure","root","AuaQhD3I3l");

Statement st = conn.createStatement();

st.executeUpdate("insert upload(name,split1,split2,split3,token1,token2,token3,filename,secretkey,orfilocation,counter,shareduser,sival) "

+ "values('"+usrnme+"','"+s1+"','"+s2+"','"+s3+"','"+tkn1+"','"+tkn2+"','"+tkn3+"','"+f+"','"+skey+"','"+loc+"','0','"+shared\_users+"',"+ii+")");

// response.sendRedirect("upanddown.jsp");

if(share.equals("YES")){

String s=shared\_users;

boolean debug = true;

Properties props = new Properties();

props.put("mail.smtp.host", "smtp.gmail.com");

props.put("mail.smtp.auth", "true");

props.put("mail.debug", "true");

props.put("mail.smtp.port", "465");

props.put("mail.smtp.socketFactory.port", "465");

props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");

props.put("mail.smtp.socketFactory.fallback", "false");

Session session2 = Session.getInstance(props,new javax.mail.Authenticator() {

protected PasswordAuthentication getPasswordAuthentication()

{

return new PasswordAuthentication("mailfromcloud844@gmail.com","mailfromcloud844");

}

});

String fkword=f.toUpperCase();

String message=usrnme+", had shared a file with you and the keyword(s) is "+fkword;

System.out.println(message);

String[] too=s.split(",");

session2.setDebug(debug);

Message msg = new MimeMessage(session2);

InternetAddress addressFrom = new InternetAddress("mailfromcloud844@gmail.com");

msg.setFrom(addressFrom);

InternetAddress[] addressTo = new InternetAddress[too.length];

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

addressTo[i] = new InternetAddress(too[i]);

}

msg.setRecipients(Message.RecipientType.TO, addressTo);

// Setting the Subject and Content Type

msg.setSubject("Shared File Keyword");

msg.setContent(message, "text/plain");

Transport.send(msg);

System.out.println("Successfully Sent");

} } catch (Exception e) {

System.out.println(e);

}

response.sendRedirect("upload.jsp?msm=suc");

%>

</body>

</html>

**Java Code**

**Mail:**

/\*

\* 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.

\*/

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.util.Properties;

import java.util.Random;

import javax.mail.Message;

import javax.mail.PasswordAuthentication;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

import javax.servlet.http.HttpSession;

/\*\*

\*

\* @author ADMIN

\*/

public class Mail21 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");

PrintWriter out = response.getWriter();

ResultSet rs2;

String s1=null;

String s2=null;

try {

Random rd=new Random();

int rd1=rd.nextInt(8);

int rd2=rd.nextInt(8);

int rd3=rd.nextInt(8);

int rd4=rd.nextInt(8);

String rendom\_key=""+rd1+rd2+rd3+rd4;

rendom\_key=rendom\_key.trim();

HttpSession session2=request.getSession(true);

String dflm=request.getParameter("dfln");

session2.setAttribute("dfln", dflm);

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

// Connection conn = DriverManager.getConnection("jdbc:mysql://ec2-50-19-213-178.compute-1.amazonaws.com:3306/group","group","group");

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cloudsecure","root","admin");

// Connection conn = DriverManager.getConnection("jdbc:mysql://mysql92594-SCloud.j.layershift.co.uk/cloudsecure","root","AuaQhD3I3l");

Statement st=conn.createStatement();

Statement st1=conn.createStatement();

Statement st2=conn.createStatement();

//Random Key Creation

String filename="";

String Orfilocation="";

String sql=("select \* from upload where Orfilocation='"+dflm+"' ");

ResultSet rs=st.executeQuery(sql);

while(rs.next())

{

s2=rs.getString("name");

filename=rs.getString("filename");

Orfilocation=rs.getString("Orfilocation");

// s1=rs.getString("secretkey");

}

s1=rendom\_key;

st1.executeUpdate("update upload set secretkey='"+rendom\_key+"' where Orfilocation='"+dflm+"' and name='"+s2+"'");

System.out.println(rendom\_key);

//String message = request.getParameter("k");

//String s = request.getParameter("m");

boolean debug = true;

Properties props = new Properties();

props.put("mail.smtp.host", "smtp.gmail.com");

props.put("mail.smtp.port", "587");

props.put("mail.smtp.auth", "true");

props.put("mail.smtp.starttls.enable", "true");

Session session = Session.getInstance(props, new javax.mail.Authenticator() {

protected PasswordAuthentication getPasswordAuthentication() {

return new PasswordAuthentication("mailfromcloud4", "Corona1!@#");

}

});

//String[] too=s2.split(",");

MimeMessage msg = new MimeMessage(session);

msg.setFrom(new InternetAddress("mailfromcloud4@gmail.com"));

msg.addRecipient(Message.RecipientType.TO, new InternetAddress(s2));

msg.setSubject("Your Securet Key from Admin");

String msge="The secret key of "+Orfilocation+" is "+s1+".";

msg.setContent(msge, "text/plain");

//msg.setContent(message, "text/plain");

//msg.setText(message);

// InternetAddress[] addressTo = new InternetAddress[too.length];

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

// addressTo[i] = new InternetAddress(too[i]);

// }

// msg.setRecipients(Message.RecipientType.TO, addressTo);

Transport.send(msg);

response.sendRedirect("secretkey.jsp");

} catch (Exception e) {

out.println(e);

} finally {

out.close();

}

}

// <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>

}

**Download code:**

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

import java.sql.\*;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.FileInputStream;

import javax.servlet.ServletContext;

import java.util.\*;

import javax.servlet.http.HttpSession;

import java.io.\*;

import javax.crypto.\*;

import javax.crypto.spec.\*;

import javax.servlet.ServletException;

/\*\*

\*

\* @author uniq

\*/

@WebServlet(name="down1", urlPatterns={"/down1"})

public class down1 extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

PrintWriter out = response.getWriter();

HttpSession session = request.getSession(true);

int counter=0;

String s1=null;

String s2=null;

String s3=null;

//String ofile=null;

String key=request.getParameter("e");

String file\_owner="";

String urname=session.getAttribute("usern").toString();

try {

// Connection conn = DriverManager.getConnection("jdbc:mysql://mysql91468-CloudSecure.j.layershift.co.uk/cloudsecure","root","admin");

String url="jdbc:mysql://localhost:3306/cloudsecure";

// String url="jdbc:mysql://mysql92594-SCloud.j.layershift.co.uk/cloudsecure";

ServletContext context = getServletContext();

String dirName =context.getRealPath("/DFile/");

Connection c=null;

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

c=DriverManager.getConnection(url, "root", "admin");

Statement st=c.createStatement();

String fln=session.getAttribute("dfln").toString();

ResultSet rs=st.executeQuery("select \* from upload where Orfilocation='"+fln+"' ");

String filename=null;

boolean f=false;

while(rs.next())

{

file\_owner=rs.getString("name");

filename=rs.getString("Orfilocation");

String uu=rs.getString("secretkey");

int conr=rs.getInt("counter");

s1=rs.getString("split1");

s2=rs.getString("split2");

s3=rs.getString("split3");

// ofile=rs.getString("Orfilocation");

if(key.equals(uu))

{

try {

String yes="YES";

String noo="NO";

Statement st1=c.createStatement();

Statement st2=c.createStatement();

Statement st3=c.createStatement();

ResultSet rss=st1.executeQuery("select \* from notify where owner='"+file\_owner+"' and suser='"+urname+"' and filenm='"+fln+"'");

boolean flg=false;

while(rss.next())

{

flg=true;

break;

}

if(flg)

{

st2.executeUpdate("update notify set fdownload='"+yes+"' where owner='"+file\_owner+"' and filenm='"+fln+"' and suser='"+urname+"'");

}

else

{

st3.executeUpdate("insert notify(owner,suser,filenm,fview,fedit,fdownload,atime) values('"+file\_owner+"','"+urname+"','"+filename+"','"+noo+"','"+noo+"','"+yes+"',now())");

}

//----------------------------Decrypt-----------------------------

int mode = Cipher.DECRYPT\_MODE;

String dirNameE1 =context.getRealPath("/splitE1/");

String dirNameE2 =context.getRealPath("/splitE2/");

String dirNameE3 =context.getRealPath("/splitE3/");

String dirNameDF =context.getRealPath("/DFile/");

String dirNameD1 =context.getRealPath("/splitD1/");

String dirNameD2 =context.getRealPath("/splitD2/");

String dirNameD3 =context.getRealPath("/splitD3/");

String keye="Murali123";

FileInputStream fis1 = new FileInputStream(dirNameE1+"/"+s1);

FileInputStream fis2 = new FileInputStream(dirNameE2+"/"+s2);

FileInputStream fis3 = new FileInputStream(dirNameE3+"/"+s3);

FileOutputStream fos1 = new FileOutputStream(dirNameD1+"/"+s1);

FileOutputStream fos2 = new FileOutputStream(dirNameD2+"/"+s2);

FileOutputStream fos3 = new FileOutputStream(dirNameD3+"/"+s3);

DESKeySpec dks = new DESKeySpec(keye.getBytes());

SecretKeyFactory skf = SecretKeyFactory.getInstance("DES");

SecretKey desKey = skf.generateSecret(dks);

Cipher cipher = Cipher.getInstance("DES");

if (mode == Cipher.DECRYPT\_MODE) {

//-----FILE-1-----------

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

CipherOutputStream cos = new CipherOutputStream(fos1, cipher);

//doCopy(is, cos);

byte[] bytes = new byte[64];

int numBytes;

while ((numBytes = fis1.read(bytes)) != -1) {

cos.write(bytes, 0, numBytes);

}

cos.flush();

cos.close();

fis1.close();

//-----FILE-2-----------

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

CipherOutputStream cos2 = new CipherOutputStream(fos2, cipher);

//doCopy(is, cos);

byte[] bytes2 = new byte[64];

int numBytes2;

while ((numBytes2 = fis2.read(bytes2)) != -1) {

cos2.write(bytes2, 0, numBytes2);

}

cos2.flush();

cos2.close();

fis2.close();

//-----FILE-3-----------

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

CipherOutputStream cos3 = new CipherOutputStream(fos3, cipher);

//doCopy(is, cos);

byte[] bytes3 = new byte[64];

int numBytes3;

while ((numBytes3 = fis3.read(bytes3)) != -1) {

cos3.write(bytes3, 0, numBytes3);

}

cos3.flush();

cos3.close();

fis3.close();

}

//----------------------------END Decrypt-----------------------------

//----------------------------File Join-------------------------------

FileInputStream ff1=new FileInputStream (dirNameD1+"/"+s1);

FileInputStream ff2=new FileInputStream (dirNameD2+"/"+s2);

FileInputStream ff3=new FileInputStream (dirNameD3+"/"+s3);

Vector<InputStream> inputStreams = new Vector<InputStream>();

inputStreams.add(ff1);

inputStreams.add(ff2);

inputStreams.add(ff3);

FileWriter fileWriter = new FileWriter(dirNameDF+"/"+filename);

PrintWriter out1 = new PrintWriter(fileWriter);

Enumeration<InputStream> enu = inputStreams.elements();

SequenceInputStream sis = new SequenceInputStream(enu);

int oneByte;

while ((oneByte = sis.read()) != -1) {

// out.println(oneByte);

//als.add(oneByte)));

out1.write(oneByte);

System.out.write(oneByte);

}

System.out.flush();

out1.flush();

out1.close();

fileWriter.close();

//----------------------------END File Join-------------------------------

//String filename1="";

//String filepath=rs.getString("resumename");

String filepath=dirName;

response.setContentType("APPLICATION/OCTET-STREAM");

response.setHeader("Content-Disposition", "attachment;name=\""+filename+"\"");

FileInputStream fileInputStream=new FileInputStream(dirName+"/"+filename);

int i;

while((i=fileInputStream.read())!=-1)

{

System.out.println(i);

out.write(i);

}

fileInputStream.close();

out.close();

} catch (Exception e) {

System.out.println(e);

}

}

else if(conr>2)

{

String str=session.getAttribute("usern").toString();

out.println("user "+str);

ResultSet rslt=st.executeQuery("select \* from attacker where uname='"+str+"'");

int flag=1;

while(rslt.next())

{

flag=0;

int ab=rslt.getInt("counter");

if(ab>2)

{

st.executeUpdate("update register set counter='0' where mail='"+str+"'");

st.executeUpdate("update upload set counter='0'");

response.sendRedirect("index.jsp?blok=blk");

}

else

{

ab=ab+1;

st.executeUpdate("update attacker set counter='"+ab+"'");

st.executeUpdate("update upload set counter='0'");

System.out.println("Sorry No More Access2");

response.sendRedirect("index.jsp?blok=alrt");

}

}

if(flag==1)

{

st.executeUpdate("insert attacker(uname,counter) values('"+str+"','1')");

st.executeUpdate("update upload set counter='0'");

System.out.println("Sorry No More Access1");

response.sendRedirect("index.jsp?blok=alrt");

}

out.println("TEST");

}

else

{

conr=conr+1;

st.executeUpdate("update upload set counter='"+conr+"' where Orfilocation='"+fln+"'");

response.sendRedirect("secretkey.jsp?blok=wrng");

}

}

}

catch(Exception e)

{

}

// out.println("your file succefully downloded");

//response.sendRedirect("download.jsp");

}

// <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>

}