Assignment No.: B5

1. TITLE

Write a web application using Scala/ Python/ Java /HTML5 to check the plagiarism in the given text paragraph written/ copied in the text box. Give software Modelling, Design, UML and Test cases for the same using Analysis Modelling (Static Modelling, Object Modelling, Dynamic Modelling).

2. PREREQUISITES

- 64-bit Fedora or equivalent OS with 64-bit Intel-5/i7
- Java 1.7.0

3. OBJECTIVE

- To implement the logic for Check the Plagiarism in the given text.
- Understand the Meaning of Software modeling using COMET.

4. MATHEMATICAL MODELS

Let, S be the System Such that,
A={ S, E, I,O, F, DD, NDD, success, failure }
Where,
S= Start state,
E= End State,
I= Set of Input
O= Set of Out put
F =Set of Function
DD=Deterministic Data
NDD=Non Deterministic Data
Success Case: It is the case when all the inputs are given by system are entered correctly. Failure Case: It is the case when the input does not match the validation Criteria.

5. THEORY

Observations of plagiarism behavior in practice reveal a number of commonly found methods for illegitimate text usage, which can briefly be summarized as follows. Copy & Paste (c & p) plagiarism specifies the act of taking over parts or the entirety of a text verbatim from another author. Disguised plagiarism includes practices intended to mask literally copied segments. Undue paraphrasing defines the intentional rewriting of foreign thoughts, in the vocabulary and

style of the plagiarist without giving due credit in order to conceal the original source. Translated plagiarism is the manual or automated conversion of content from one language to another intended to cover its origin. Idea plagiarism encompasses the usage of a broader foreign concept without appropriate source acknowledgement. An Example is the appropriation of research approaches, methods, experimental setups, argumentative structures, background sources etc.

COMET

COMET is a highly iterative object-oriented software development method that addresses the requirements, analysis, and design modeling phases of the object-oriented development life cycle. The functional requirements of the system are defined in terms of actors and use cases. Each use case defines a sequence of interactions between one or more actors and the system. A use case can be viewed at various levels of detail. In a *requirements* model, the functional requirements of the system are defined in terms of actors and use cases. In an *analysis* model, the use case is realized to describe the objects that participate in the use case, and their interactions. In the *design* model.

Test Case Id	**Test case**	**Test case Description**	**Status**
1	Title Test	Title test passed	
2	Fileone input	Fileone input Passed	
3	Filetwo input	Filetwo input Passed	
4	ButtonCheck clicked	ButtonCheck passed	

Java Scanner class

To Check the Plagiarism in the given text first of all perform string compare operation also need to understand the operation related to scanner classes. There are various ways to read input from the keyboard, the java.util.Scanner class is one of them.

- The **Java Scanner** class breaks the input into tokens using a delimiter that is whitespace bydefault. It provides many methods to read and parse various primitive values.
- Java Scanner class is widely used to parse text for string and primitive types using regular expression.
- Java Scanner class extends Object class and implements Iterator and Closeable interfaces.

java.io.PrintStream class: The PrintStream class provides methods to write data to another stream. The PrintStream class automatically flushes the data so there is no need to call flush() method. Moreover, its methods don't throw IOException.

Stream: A stream is a sequence of data.In Java a stream is composed of bytes. It's called a stream because it's like a stream of water that continues to flow. In java, 3 streams are created for us automatically. All these streams are attached with console.

1) System.out: standard output stream
2) System.in: standard input stream
3) System.err: standard error stream

OutputStream
Java application uses an output stream to write data to a destination, it may be a file,an array,peripheral device or socket.

InputStream:
Java application uses an input stream to read data from a source, it may be a file,an array,peripheral device or socket.

Output Stream class: OutputStream class is an abstract class.It is the superclass of all classes representing an output stream of bytes. An output stream accepts output bytes and sends them to some sink.

Reading data from keyboard: There are many ways to read data from the keyboard. For example:

- InputStreamReader
- Console
- Scanner
- DataInputStream etc.

InputStreamReader class: InputStreamReader class can be used to read data from keyboard.It performs two tasks:

- connects to input stream of keyboard
- converts the byte-oriented stream into character-oriented stream

The Java Console class is be used to get input from console. It provides methods to read text and password. If you read password using Console class, it will not be displayed to the user. The java.io.Console class is attached with system console internally.

6. CONCLUSION

A web application is created to check the plagiarism in the given text paragraph written/ copied in the text box.

File: index.jsp

```
<%--
   Document   : index
   Created on : 9 Mar, 2016, 11:06:22 AM
   Author     : @@
--%>

<%@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>
   </head>
   <body>
       <form action="Plagarism">

      Enter the first input <textarea name="File1" rows="5" cols="10">
      </textarea><br>
      Enter the second input<textarea name="File2" rows="5" cols="10">
      </textarea><br>
      <input type="submit" value="Check Plagarism" name="btn"/>
         </form>
   </body>
</html>
```

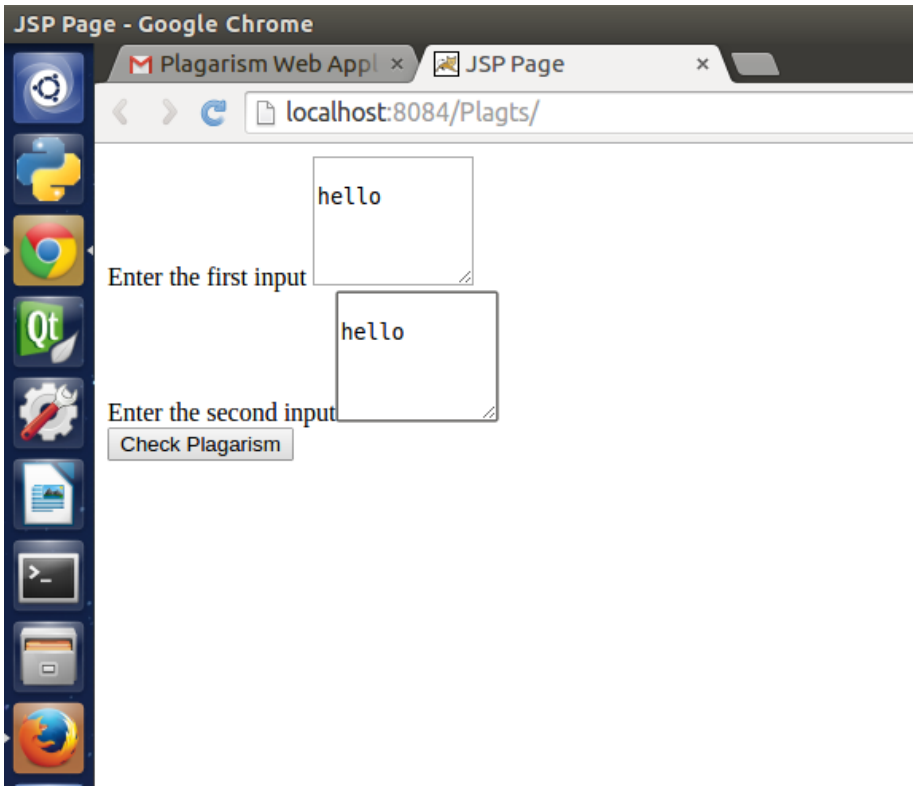
File: Plagarism.java

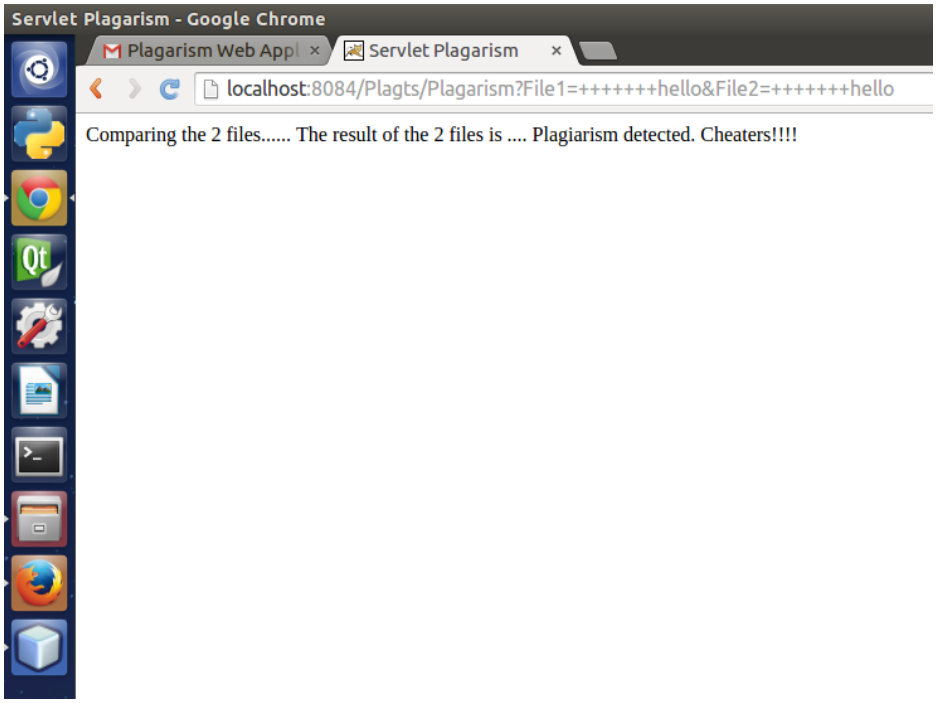
```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class Plagarism extends HttpServlet
{

   protected void processRequest(HttpServletRequest request, HttpServletResponse response)
           throws ServletException, IOException {
       response.setContentType("text/html;charset=UTF-8");
       PrintWriter out = response.getWriter();
       try {
           out.println("<!DOCTYPE html>");
           out.println("<html>");
           out.println("<head>");
           out.println("<title>Servlet Plagarism</title>");
           out.println("</head>");
           out.println("<body>");

           String fileOne = request.getParameter("File1");
           String fileTwo = request.getParameter("File2");
           out.println("Comparing the 2 files.....");
           out.println("The result of the 2 files is ....");
           if (compareStrings(fileOne,fileTwo)) {
               out.println("Plagiarism detected. Cheaters!!!!");
           } else {
               out.println("No plagiarism detected");
           }
```

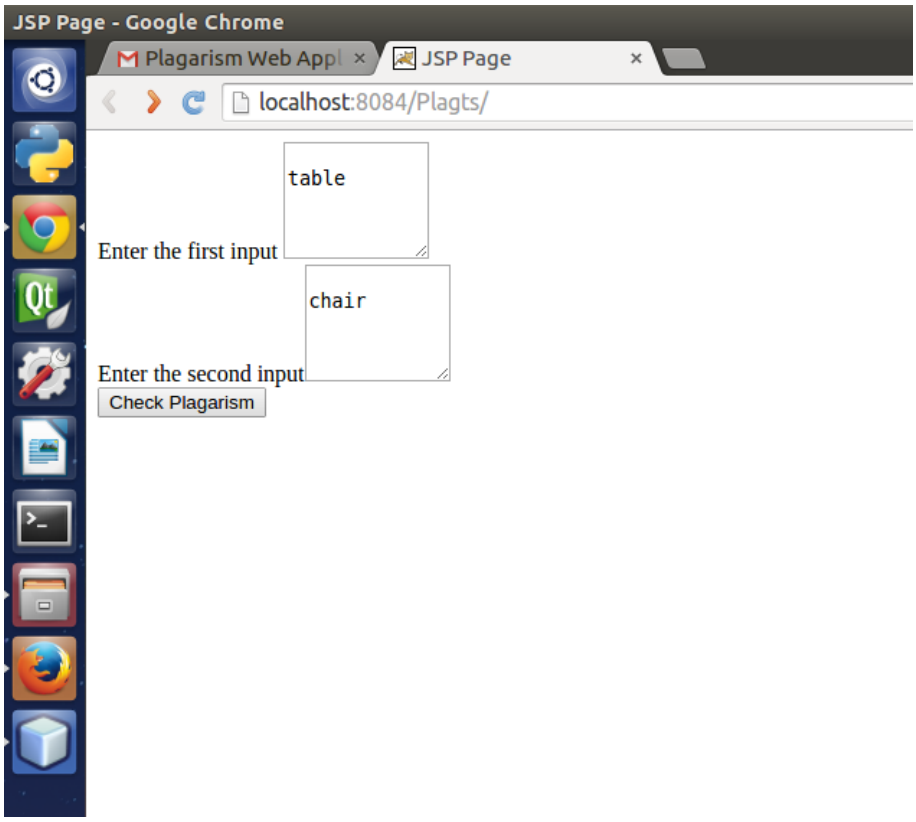
```
            out.println("</body>");
            out.println("</html>");
        } finally {
            out.close();
        }
    }

    public static boolean compareStrings(String a,String b){
        boolean checkForPlagiarism = false;
        String[] piecesA = a.split("\\s");
        String[] piecesB = b.split("\\s");

        int count1 = 0;
        int count2 = 0;

        for (int counter = 0; counter <= piecesA.length - 1; counter++){
            for(int counter2 = 0; counter2<= piecesB.length - 1; counter2++){
                if(piecesA[counter].equals(piecesB[counter2])){
                count1++;
                }
            }
        }

        for(int counter=0;counter<=piecesA.length-1;counter++){
            for(int counter2 = 0; counter2 < piecesB.length; counter2++){
                if(piecesA[counter].equals(piecesB[counter2])){
                    count2++;
                }
            }
        }

        if((count1/(int)piecesA.length)*100>=65 && (count2/(int)piecesB.length)*100>=65){
            checkForPlagiarism=true;
            }
        return checkForPlagiarism;
        }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign
on the left to edit the code.">
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

        @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

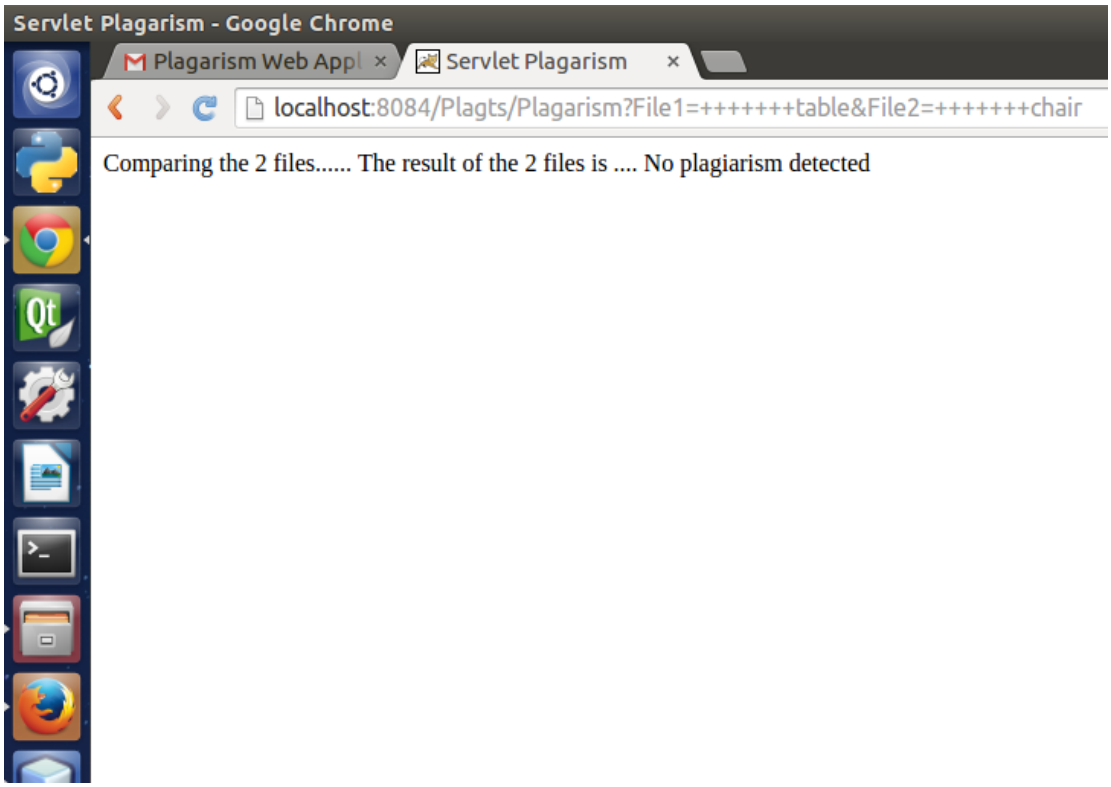
File: mytext3.java

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;

public class mytest3
{


   public static void main(String[] args) {


       WebDriver driver = new FirefoxDriver();
       String baseUrl = "http://localhost:8084/plagraism/";
       driver.get(baseUrl);
       String expected = "JSP Page";
       String actual = "";
       driver.manage().window().maximize();
       actual = driver.getTitle();
       if (actual.equals(expected)) {
           System.out.println("Title test passed");
       } else {
           System.out.println("Title test failed");}
           WebElement text=driver.findElement(By.name("File1"));
           text.sendKeys("hello");
           WebElement text1=driver.findElement(By.name("File2"));
           text1.sendKeys("hiee");

           WebElement btn=driver.findElement(By.name("btn"));
           btn.click();
           System.out.println(" test script sucessful");
           driver.close();

   }
}
```


JSP Page - Google Chrome
Plagarism Web Appl
JSP Page
localhost:8084/Plagts/
hello
Enter the first input
hello
Enter the second input
Check Plagarism


Servlet Plagarism - Google Chrome
Plagarism Web Appl
Servlet Plagarism
localhost:8084/Plagts/Plagarism?File1=+++++++hello&File2=+++++++hello
Comparing the 2 files..... The result of the 2 files is .... Plagiarism detected. Cheaters!!!!


JSP Page - Google Chrome
Plagarism Web Appl
JSP Page
localhost:8084/Plagts/
table
Enter the first input
chair
Enter the second input
Check Plagarism


Servlet Plagarism - Google Chrome
Plagarism Web Appl
Servlet Plagarism
localhost:8084/Plagts/Plagarism?File1=+++++++table&File2=+++++++chair
Comparing the 2 files..... The result of the 2 files is .... No plagiarism detected