

PROJECT TITLE

Submitted in partial fulfillment of the requirements

of the degree of

Bachelor of Engineering

by

Deven Bhalerao

Roll No. 05

Supervisor:

Prof. Prasad Padalkar



Department of Information Technology

Don Bosco Institute of Technology

2015-2016

AFFILIATED TO

UNIVERSITY OF MUMBAI

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

CERTIFICATE

This is to certify that the project entitled “**NAME OF PROJECT**” is a bonafide work of

Deven Bhalerao	05
Manish Jain	28
Vishal Jha	29
Kunal Naik	45

submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **Undergraduate** in **Bachelor of Information Technology**

Date: / /

(Prof. Prasad Padalkar)
Supervisor

(Prof. Shiv Negi)
HOD, IT Department

(Dr. Prasanna Nambiar)
Principal

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

Project Report Approval for B.E.

This project report entitled “**NAME OF PROJECT**“ by **Author Name** is approved for the degree of **Bachelor of Engineering in Information Technology**

(Examiner's Name and Signature)

1. _____

2. _____

(Supervisor's Name and Signature)

1. _____

(Chairman)

1. _____

Date:

Place:

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(_____)
(Signature)

Deven Bhalerao

Date:

ABSTRACT

Your abstract, paragraph 1.

Your abstract, paragraph 2.

Keywords: keyword1, keyword2

Contents

1	Introduction	2
1.1	Problem Statement	2
1.2	Scope of the Project	2
1.3	Current Scenario	2
1.4	Need for the Proposed System	2
1.5	Summary of the Results / Task completed	3
2	Review of Literature	4
2.1	Summary of the investigation in the published papers	4
2.2	Comparison between the tools / methods / algorithms	5
2.3	Algorithm(s) with example	6
3	Analysis and Design	7
3.1	Methodology / Procedure adopted	7
3.2	Analysis	7
3.2.1	Software / System Requirement Specification - IEEE format	7
3.3	Proposed System	7
3.3.1	Hardware / Software requirements	7
3.3.2	Design Details	8
3.3.3	Implementation Plan	8
4	Results and Discussion	9
4.1	Result of Comparison for various levels for Execution time	9
5	Conclusion	11

List of Figures

List of Tables

2.1	Compression of four source code detection tools with its characteristics, function and technique	5
2.2	Comparison between different metrics : Structural Metrics and Similarity Metrics	5

Chapter 1

Introduction

1.1 Problem Statement

To develop a tool for detecting plagiarism in software source code using the machine learning algorithms.

1.2 Scope of the Project

- For the first level of implementation , tool will be on a local machine for checking plagiarism.
- The source and target program should be in Java.

1.3 Current Scenario

1.4 Need for the Proposed System

The Role of Plagiarism detection in Education, Role of Plagiarism Detection in Software, Industry especially copyright of software.

	MOSS	SHERLOCK	JPLAG	CODEMATCH	
Cost	Free	Open Source	Free	It is a Commercial T	
Safety	Sign-in Required	Executes at Local machine	Sign-in Required	Executes at Local m	
Service	Internet	Standalone	Web-Service	Standalone	
Algorithm	Winowing	Token Matching	Greedy String Tiling	String Matching	
Speed	Fast	More files requires more time	Fast	More files requires r	

1.5 Summary of the Results / Task completed

Study of different Research Papers, Thesis on Source Code Plagiarism. Survey on different source code plagiarism tools. Implementation upto Level 3 Plagiarism (Rule-based).

Chapter 2

Review of Literature

2.1 Summary of the investigation in the published papers

- **Computer Algorithms for Plagiarism Detection**

Described the various levels of plagiarism and the different metrics (approaches) like Halstead metric, ACCUSE metric, FORTRAN programs, etc. This paper also compared different algorithms based on the software metrics used. The different levels of plagiarism defined in this paper will form the foundation of our project and we'll be looking to detect plagiarism based on these levels.

- **A Comparison of Similarity Techniques for Detecting Source Code Plagiarism**

This paper outlines different modern approaches to software similarity measurement. Algorithms like Levenshtein edit distance, Tree edit distance and graph edit distance were discussed. Knowledge of different approaches was gained and this will help us choose our approach.

- **An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis**

Latent Semantic Analysis, a statistical approach to detecting similarity is analysed in-depth in this paper. This paper also has an exhaustive description of what constitutes Plagiarism which includes surveys and research papers. Deeper understanding of Plagiarism was made possible by this research paper.

- **Plagiarism Detection in Java Code**

This thesis gives a step-by-step procedure on how to detect plagiarism in java source code using the Levenshtein Edit distance algorithm. It uses

different normalization techniques and demonstrates these techniques using examples. Normalization techniques shown in this thesis will be used in our project.

• Source Code Plagiarism Detection ‘SCPDet’: A Review

In this paper author describes the real meaning of source code plagiarism and then described the different source code plagiarism detection tools and compared its function, characteristics and technique. In the last phase, authors discussed the different research papers and compared in tabular form with its technique, method, characteristics, functionality and its result.

2.2 Comparison between the tools / methods / algorithms

Table 2.1: Comparison of four source code detection tools with its characteristics, function and technique

Tools	JPlag	SIM	MOSS	Plaggie
Open Source Tools/Paid	NO	YES	NO	YES
Local/online tool	Web	Local	Web	Local
Code Submit/File	Submit Code	Submit File	Submit Code	Submit Code
Lang. Support	6	5	23	1
Expandability	No	Yes	No	No
Founded in Year	1996	1989	1994	2002
Founded By	Guido Malpohl	Dick Grune	Aiken	Ahtiaine
Technique	Greedy String Tiling and Optimization and Tokenization	Flax lexical analyzer	Winnowing technique	Greedy String Tiling and Tokenization

Table 2.2: Comparison between different metrics : Structural Metrics and Similarity Metrics

Structural metrics	Similarity metrics
Structural metrics – no. of variables, no. of keywords, no. of loops, no. of comment lines	Similarity metrics – no. of characters per line, no of code lines, no. of blank lines
Structural metrics represent information about programming constructs and elements used in the code.	Similarity metrics are indicative of the style used in programming and are effective in detecting plagiarism.
Lots of rudimentary plagiarism detection algorithms like Halstead use only structural metrics and are ineffective for larger programs.	Modern approaches include algorithms that use a combination of structural and similarity metrics to detect plagiarism and are highly effective.

2.3 Algorithm(s) with example

Give the pseudo code / algorithm along with explanation. Analysis of algorithm on the basis of parameters like time complexity , space complexity, etc are expected.

Chapter 3

Analysis and Design

3.1 Methodology / Procedure adopted

Describe on the development methodology / model you would use. (E.g. Agile method or Iterative Model)

How you intend manage the weekly meetings ?

How do you intend to monitor and measure the progress of the project?

3.2 Analysis

Based on the requirements gathered, how was the feasibility study of the project carried out?

If any requirements, were modified why they were modified?

3.2.1 Software / System Requirement Specification - IEEE format

3.3 Proposed System

Give the details of your proposed system and architecture Advantage of the proposed system over the existing system

3.3.1 Hardware / Software requirements

Development Hardware / Software requirements

Deployment Hardware / Software requirements

3.3.2 Design Details

Different UML diagrams as per the project requirement (For e.g. Use Case Diagram)

3.3.3 Implementation Plan

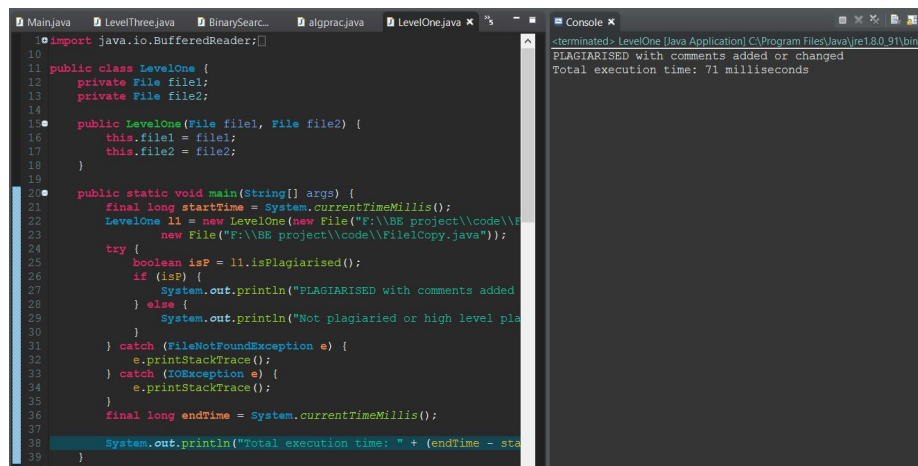
Timeline chart is for Next semester

Chapter 4

Results and Discussion

4.1 Result of Comparison for various levels for Execution time

- Level One Plagiarism Detection.



```

10 import java.io.BufferedReader;
11 public class LevelOne {
12     private File file1;
13     private File file2;
14
15     public LevelOne(File file1, File file2) {
16         this.file1 = file1;
17         this.file2 = file2;
18     }
19
20     public static void main(String[] args) {
21         final long startTime = System.currentTimeMillis();
22         LevelOne l1 = new LevelOne(new File("F:\\BE project\\code\\V\\
23             new File("F:\\BE project\\code\\FileCopy.java"));
24         try {
25             boolean isP = l1.isPlagiarised();
26             if (isP) {
27                 System.out.println("PLAGIARISED with comments added
28             } else {
29                 System.out.println("Not plagiarised or high level pla
30             }
31         } catch (FileNotFoundException e) {
32             e.printStackTrace();
33         } catch (IOException e) {
34             e.printStackTrace();
35         }
36         final long endTime = System.currentTimeMillis();
37         System.out.println("Total execution time: " + (endTime - sta
38     }
39 }

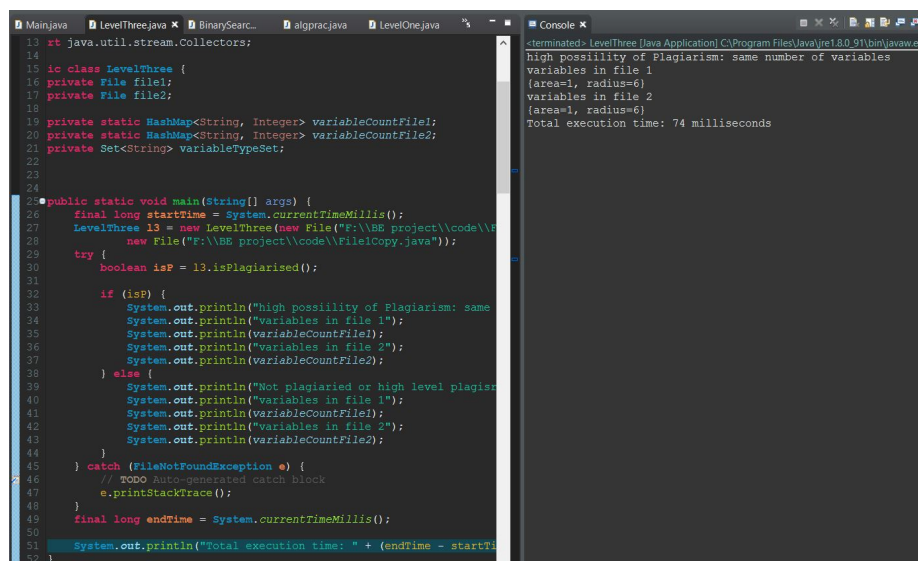
```

```

<terminated> LevelOne [Java Application] C:\Program Files\Java\jre1.8.0_91\bin\javaw.exe
PLAGIARISED with comments added or changed
Total execution time: 71 milliseconds

```

- Level Three Plagiarism Detection.



```

13 import java.util.stream.Collectors;
14
15 public class LevelThree {
16     private File file1;
17     private File file2;
18
19     private static HashMap<String, Integer> variableCountFile1;
20     private static HashMap<String, Integer> variableCountFile2;
21     private Set<String> variableTypeSet;
22
23
24
25     public static void main(String[] args) {
26         final long startTime = System.currentTimeMillis();
27         LevelThree l3 = new LevelThree(new File("F:\\BE project\\code\\V\\
28             new File("F:\\BE project\\code\\FileCopy.java"));
29         try {
30             boolean isP = l3.isPlagiarised();
31
32             if (isP) {
33                 System.out.println("high possibility of Plagiarism: same
34                 System.out.println("variables in file 1");
35                 System.out.println(variableCountFile1);
36                 System.out.println("variables in file 2");
37                 System.out.println(variableCountFile2);
38             } else {
39                 System.out.println("Not plagiarised or high level plagia
40                 System.out.println(variableCountFile1);
41                 System.out.println("variables in file 2");
42                 System.out.println(variableCountFile2);
43             }
44         } catch (FileNotFoundException e) {
45             // TODO Auto-generated catch block
46             e.printStackTrace();
47         }
48         final long endTime = System.currentTimeMillis();
49         System.out.println("Total execution time: " + (endTime - startTi
50     }
51 }
52 }

```

```

<terminated> LevelThree [Java Application] C:\Program Files\Java\jre1.8.0_91\bin\javaw.exe
high possibility of Plagiarism: same number of variables
variables in file 1
(area=1, radius=6)
variables in file 2
(area=1, radius=6)
Total execution time: 74 milliseconds

```


- Result from JPlag.

```
F:\BE project\other software>java -jar jplag-2.11.8.jar -l java17 -r "F:\BE project\other software\result" -s "F:\BE project\other software\source"
Language accepted: Java1.7 Parser
Command line: -l java17 -r F:\BE project\other software\result -s F:\BE project\other software\source
Initialize ok
2 submissions
2 submissions parsed successfully!
0 parser errors!

Comparing File1.java-File1Copy.java: 100.0
Writing results to: F:\BE project\other software\result
```

- Result from MOSS.

Moss Results

Thu Oct 13 10:45:16 PDT 2016

Options -l java -m 3

[[How to Read the Results](#) | [Tips](#) | [FAQ](#) | [Contact](#) | [Submission Scripts](#) | [Credits](#)]

File 1	File 2	Lines Matched
File1.java (97%)	File1Copy.java (97%)	77

Any errors encountered during this query are listed below.

Chapter 5

Conclusion

Summary of the entire report

Appendix - I

Data Sheet(s) - Electronic component

Installation Procedure - Development Software

References

- [1] Zhi Zhou, Member, IEEE, Gonzalo R. Arce, Fellow, IEEE, and Giovanni Di Crescenzo; *Halftone Visual Cryptography*; IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 15, NO. 8, AUGUST 2006
- [2] HTML 5 <http://en.wikipedia.org/wiki/HTML5> , last modified on 6 October 2014

Acknowledgements

Parargraph 1 of you acknowledgement

Parargraph 2 of you acknowledgement

(_____)
(Signature)

(_____)
(Name of Student and Roll No.)

Date: