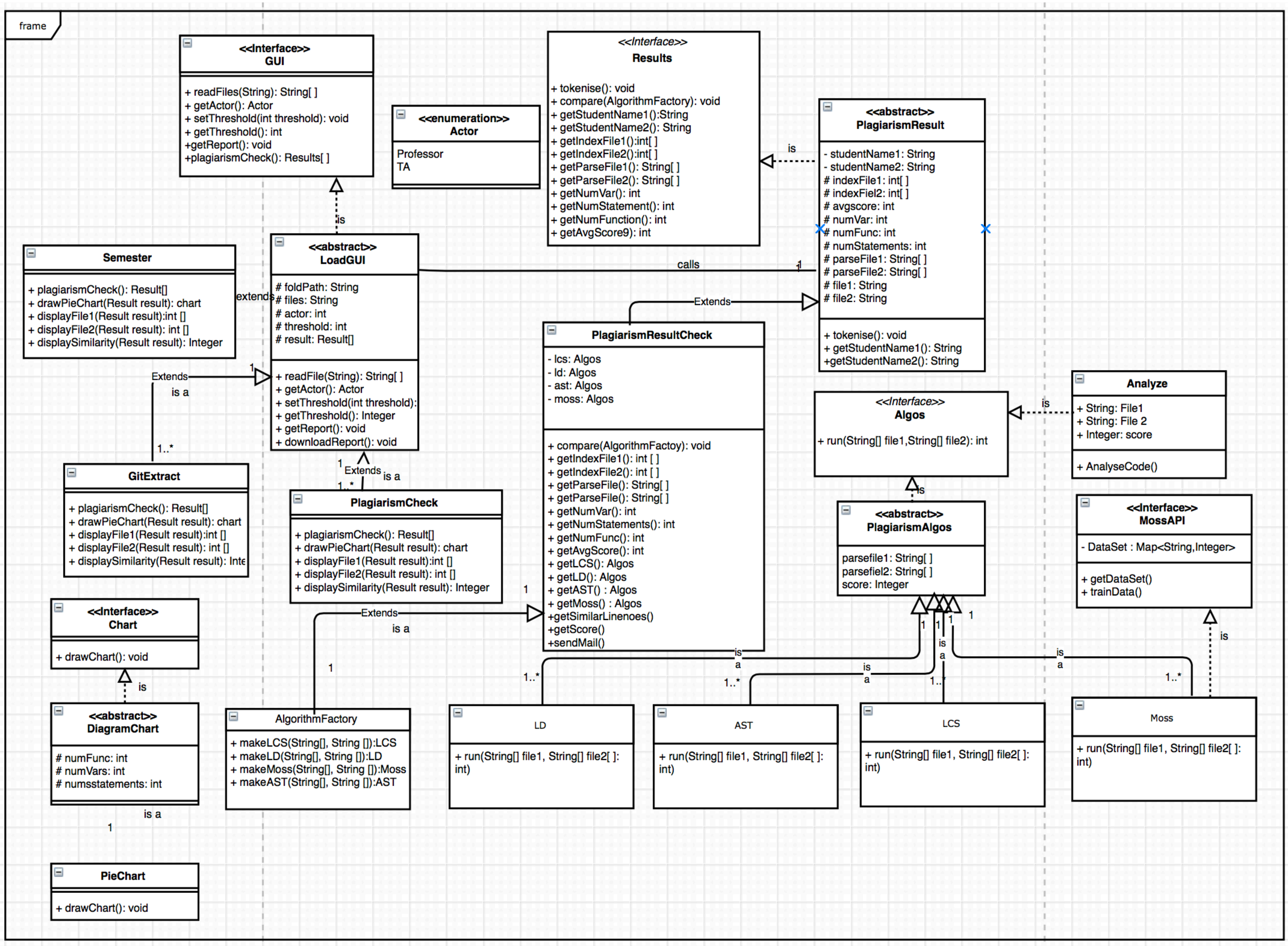
frame
<<Interface>>
GUI
+ readFiles(String): String[ ]
+ getActor(): Actor
+ setThreshold(int threshold): void
+ getThreshold(): int
+getReport(): void
+plagiarismCheck(): Results[ ]
<<enumeration>>
Actor
Professor
TA
<<Interface>>
Results
+ tokenise(): void
+ compare(AlgorithmFactory): void
+ getStudentName1():String
+ getStudentName2(): String
+ getIndexFile1():int[ ]
+ getIndexFile2():int[ ]
+ getParseFile1(): String[ ]
+ getParseFile2(): String[ ]
+ getNumVar(): int
+ getNumStatement(): int
+ getNumFunction(): int
+ getAvgScore9): int
is
<<abstract>>
PlagiarismResult
- studentName1: String
- studentName2: String
# indexFile1: int[ ]
# indexFiel2: int[ ]
# avgscore: int
# numVar: int
# numFunc: int
# numStatements: int
# parseFile1: String[ ]
# parseFile2: String[ ]
# file1: String
# file2: String
+ tokenise(): void
+ getStudentName1(): String
+getStudentName2(): String
is
Semester
+ plagiarismCheck(): Result[]
+ drawPieChart(Result result): chart
+ displayFile1(Result result):int []
+ displayFile2(Result result): int []
+ displaySimilarity(Result result): Integer
extends
<<abstract>>
LoadGUI
# foldPath: String
# files: String
# actor: int
# threshold: int
# result: Result[]
+ readFile(String): String[ ]
+ getActor(): Actor
+ setThreshold(int threshold):
+ getThreshold(): Integer
+ getReport(): void
+ downloadReport(): void
calls
1
Extends
Extends
is a
1
1..*
PlagiarismResultCheck
- lcs: Algos
- ld: Algos
- ast: Algos
- moss: Algos
+ compare(AlgorithmFactoy): void
+ getIndexFile1(): int [ ]
+ getIndexFile2(): int [ ]
+ getParseFile(): String[ ]
+ getParseFile(): String[ ]
+ getNumVar(): int
+ getNumStatements(): int
+ getNumFunc(): int
+ getAvgScore(): int
+ getLCS(): Algos
+ getLD(): Algos
+ getAST() : Algos
+ getMoss() : Algos
+getSimilarLinenoes()
+getScore()
+sendMail()
GitExtract
+ plagiarismCheck(): Result[]
+ drawPieChart(Result result): chart
+ displayFile1(Result result):int []
+ displayFile2(Result result): int []
+ displaySimilarity(Result result): Inte
1
Extends
is a
1..*
PlagiarismCheck
+ plagiarismCheck(): Result[]
+ drawPieChart(Result result): chart
+ displayFile1(Result result):int []
+ displayFile2(Result result): int []
+ displaySimilarity(Result result): Integer
<<Interface>>
Algos
+ run(String[] file1,String[] file2): int
is
Analyze
+ String: File1
+ String: File 2
+ Integer: score
+ AnalyseCode()
is
<<abstract>>
PlagiarismAlgos
parsefile1: String[ ]
parsefiel2: String[ ]
score: Integer
<<Interface>>
MossAPI
- DataSet : Map<String,Integer>
+ getDataSet()
+ trainData()
is
<<Interface>>
Chart
+ drawChart(): void
is
<<abstract>>
DiagramChart
# numFunc: int
# numVars: int
# numsstatements: int
is a
1
PieChart
+ drawChart(): void
Extends
is a
1
1
AlgorithmFactory
+ makeLCS(String[], String []):LCS
+ makeLD(String[], String []):LD
+ makeMoss(String[], String []):Moss
+ makeAST(String[], String []):AST
1
1
1
1
is
a
is
a
is
a
is
a
1..*
1..*
1..
1..*
LD
+ run(String[] file1, String[] file2[ ]: int)
AST
+ run(String[] file1, String[] file2[ ]: int)
LCS
+ run(String[] file1, String[] file2[ ]: int)
Moss
+ run(String[] file1, String[] file2[ ]: int)

Future Developments:

The above diagram is extended UML for project with future developments.

1. Extract Repository of all assignments and projects from GitHub.
2. Use Machine Learning to train dataset and set the weights of polynomial function.
3. Compare files across semesters.
4. Analyze the code as soon as it is uploaded such that relevant algorithm is automatically selected.