**Aim**: Team One, over the course of three phases, will develop an interactive interface for instructors, teaching assistants, and administrators to oversee the detection of plagiarism. In this summary, the reader will find the expected input for the project, an explanation of the project’s interface, expected users of this project, and the underlying plagiarism detection methods that will be employed.

**Constraints**: While this project could be expanded in the future, the scope of this iteration is constrained as described below.

* This project is intended to be used to detect plagiarism in programs written in Java.
* Only single file programs will be considered for plagiarism at this time.
* Comments will not be taken into consideration when detecting plagiarism.

**Logistics**: Our plagiarism detection software will be written with a combination of Java, HTML, and JavaScript. The portion of the project that will detect plagiarism and manage underlying data storage and retrieval will be written in Java. The project’s data will be stored in a SQLite database. Users of this project will interact with this project by means of a web interface that will require a secure logon.

**Implementation Plan**: This project has three intended types of users:

* Instructors

Instructors can create, modify, delete courses and assignments related to the course. Instructors can also assign and remove TAs from his courses.

* Teaching assistants (TAs)

TAs can upload the assignment submissions to the courses that they are associated. After uploading submissions, they can run plagiarism detection on the submissions. They can review the results of their courses’ plagiarism detection. These actions can be performed by the Instructors as well.

* Administrators (admin).

Admin users handle the creation and deletion of users of this application.

Our approach for detecting plagiarism between any two files borrows heavily from the Rojas Method1 and consists of the three phases described below.

* Preprocessing: Removal of unnecessary components of a program, such as comments.
* Recording: Ordering functions based on character count.
* Running a comparison against the restructured submissions: Results in a score based on percent match between two given submissions. Submission pairs with a score above a given threshold will be flagged as potential instances of plagiarism.

JavaParser2 toolset is used for processing and modification of submissions.

**Schedule**: Our project is scheduled to be fully implemented by the end of the Northeastern’s Fall 2017 semester. The project will be implemented in three phases: Phase A: design, Phase B: documentation, and Phase C: implementation. All deliverables will be stored in Team One’s GitHub repository3.