MSD
Final Presentation

Team-111
Xin Li
Gautam Baghel
Amritansh Tripathi
Veera Venkata Sasanka Uppu

Introduction

- Web based plagiarism detection tool.
- Plagiarism detection is commonly based on text, tokens or syntax trees.
- Proposed detection techniques are based on tokens generated by the AST and string similarity techniques.

Tools Used

- Programming language: Java
- Backend Framework: Spring Boot and JPA
- Frontend Frameworks: AngularJS, Bootstrap and HTML
- Testing Framework: Junit4
- Libraries: ANTLR, JCOCO, Javax.mail, Jsoup
- Database: H2
- Process management: Jira
- Version Control: Github
- Continuous Integration : Jenkins
- Quality Gate: SonarQube
- Communication tool: Slack

Algorithms used for plagiarism detection

Five different algorithms have been implemented

- Longest Common Subsequence with strings
- Longest Common Subsequence with AST nodes
- Jaccard Similarity
- Token edit distance
- Machine learning model

System Functionality and Features

- Ability to create a new user as an admin or an instructor
- Created user can login using the email id and password provided during registration
- User can upload multiple student submissions.
- System can detect plagiarism in multiple Python projects
- Edge cases like empty folder and non python files are handled
- Multiple comparison algorithms can be used to detect plagiarism
- Similarity score is provided for all the compared files

System Functionality and Features

- The systems uses weights generated using a machine learning model based on gradient descent and the MOSS as evaluation model.
- User can analyze the detected python files with side by side line comparison indicating where the system detected plagiarism.
- The System can show number of time each algorithm has run.
- The admin has the privilege to delete user
- The user interface is descriptive and user centric

Process and Teamwork

Before each sprint we had one hour sprint planning to align the workflow.

The sprint planning focused on following topics:

- List all the tasks we need to do
- Go through every task and discuss the job distribution.
- Set timeline for the sprint and point our tickets.

Daily 5 minutes check-in to discuss about the tasks each member has done yesterday and the tasks each member will do that day.

Double check the features and codes at the end of every sprint.

Used Jenkins to do the following things: Build project, Run test cases, Maintain code quality(SonarQube).

Process and Teamwork

Achievement:

- Finishing most of the requirements we set at the beginning of the project.
- Using the design patterns we learnt from the class.
- Following agile development process
- System was thoroughly tested

Major Problems during development:

- Jenkins server management (setting up and upgrading jenkins server).
- Implementing complex algorithms to detect various forms of plagiarism.
- User and file management system.

Job Quality

- Followed test driven design.
- Each module was thoroughly unit tested
- SonarQube quality gate always passed
- The code is extensively commeted for readability.

Technology Transfer

- The system can be deployed easily as a standalone application by following the readme document and the setup video
- Future scope includes
  - Sending email when plagiarism has been triggered in a submission
  - Generating a downloadable pdf report of the comparison result
  - Extending the system to other programming languages
  - Adding more comparison strategies

Thanks!