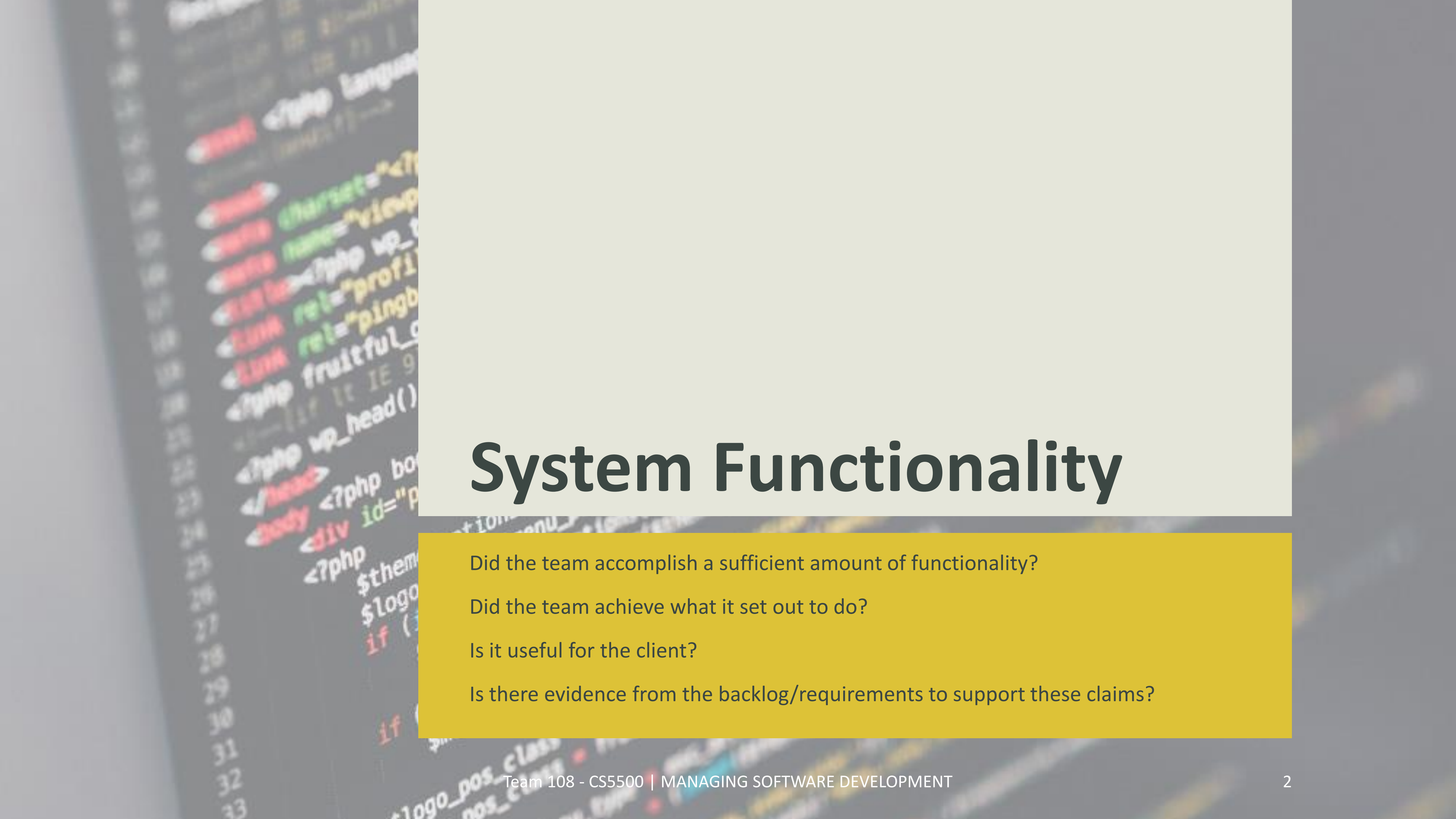
Plagiarism Detector

Team 108



System Functionality

Did the team accomplish a sufficient amount of functionality?

Did the team achieve what it set out to do?

Is it useful for the client?

Is there evidence from the backlog/requirements to support these claims?

Base Expectations

- Login feature to log into the application using the registered username and password.
- User can upload either two submission files or multiple files in two submission folders to compare against each other.
- The application offers two on-demand sophisticated strategies to detect plagiarism: Longest Common Subsequence algorithm and Needleman-Wunsch algorithm
- The application logs user activity in rolling log files and logs errors in error files and send a mail to the admin with the stack trace of the error.
- The application displays user statistics of the number of plagiarism detection cases run and system status.

Stretch Expectations

- The user can upload multiple submission folders having multiple files to compare against each other.
- The application offers a weighted combination of Longest Common Subsequence algorithm and Needleman–Wunsch algorithm where the user can adjust the weights using a slider.
- The application offers another weighted combination of Longest Common Subsequence algorithm and Needleman–Wunsch algorithm where the weights are trained using Stanford’s MOSS as the benchmark.
- The application allows users to download the results of their comparison in an excel format or pdf format.

Useful to the client

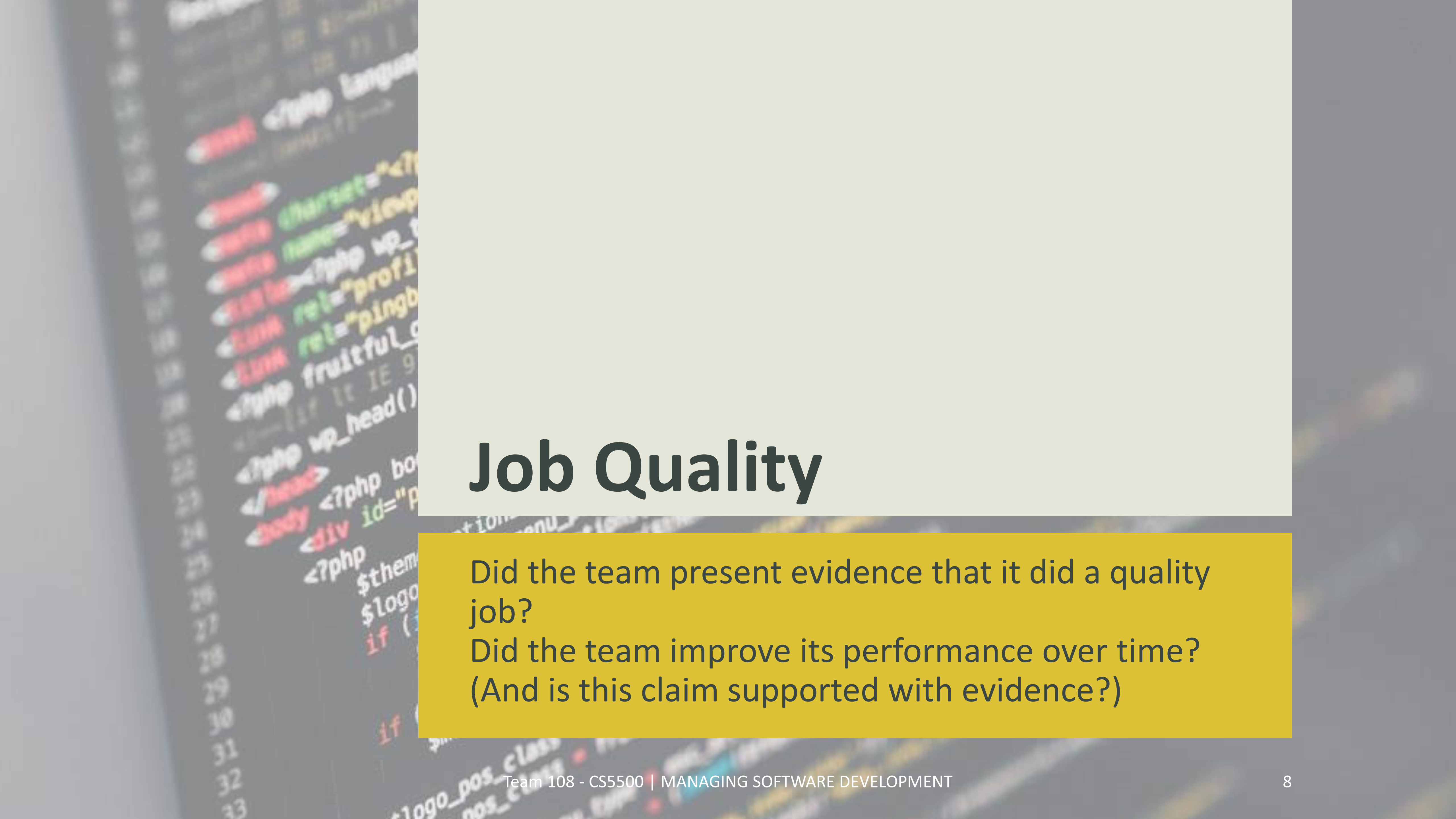
- Sadly student plagiarism is a critical problem at almost every academic institution.
- Some institutions employ the use of text-matching software to check a student’s work against other submitted work or materials on the internet.
- This might be effective for essays and textual solutions but is not very effect in detecting plagiarism in source codes.
- There are different scenarios where two or more people produce different versions of the same code, one version is derived from another by either using copy-then-modify technique, or moving around blocks of code or splitting a block of code into multiple blocks.
- Text-matching software fails to detect such instances of plagiarism. This application however looks beyond textual differences and is able to detect lexical and syntactic differences. The application can also detect instances where a copied code is split across multiple files.

Jira Tickets

- Sprint 1
  - CS108-1 Creating AST
  - CS108-2 Project Setup
  - CS108-3 Proof of Concept for comparing two strings
  - CS108-4 Creating Login
- Sprint 2
  - CS108-18 Add features to the comparison module
  - CS108-20 Add Database connectivity
  - CS108-22 Logging Activity of the system
  - CS108-24 Code Upload functionality
  - CS108-26 Levenshtein Distance
  - CS108-30 AST based comparison
  - CS108-31 User Login
  - CS108-32 Test report functionality
  - CS108-33 Weighted Polynomial Implementation and Moss Trained Model

Jira Tickets

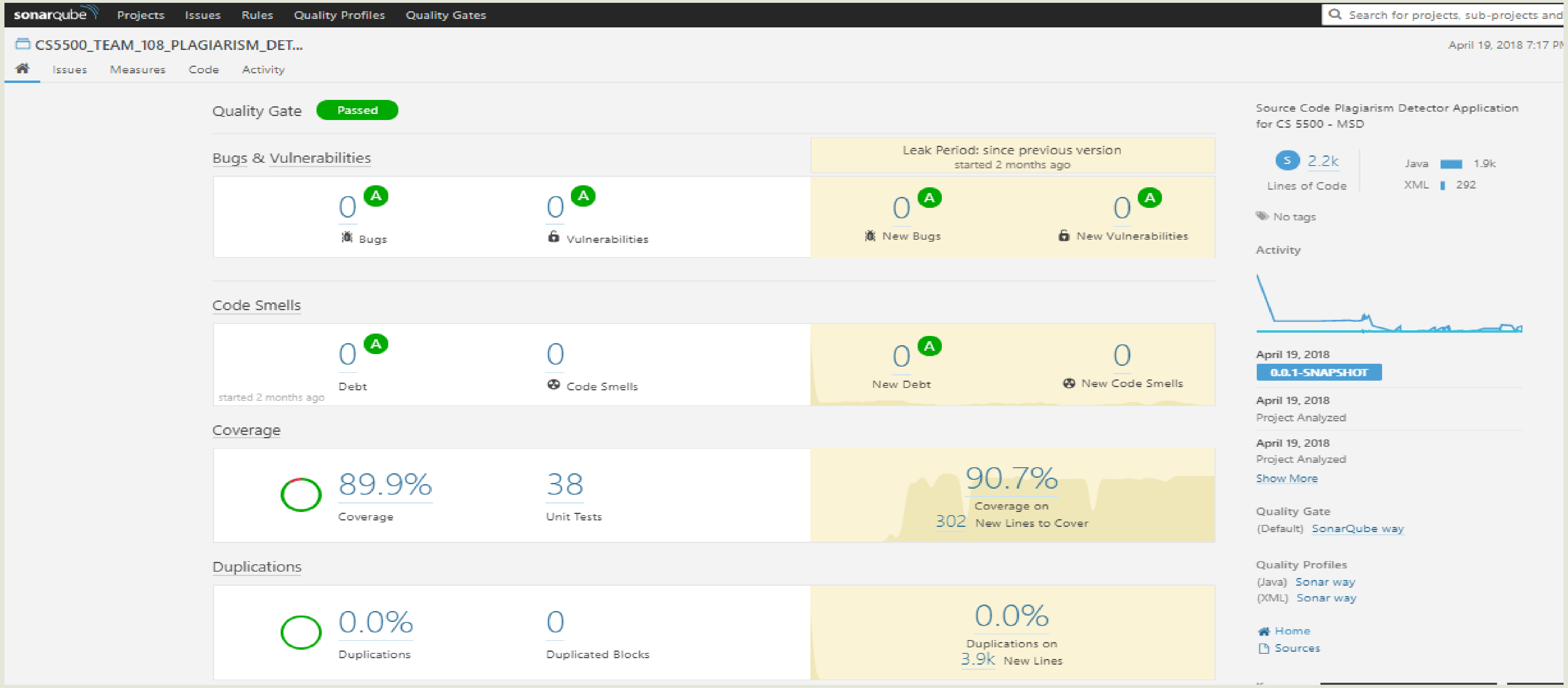
- Sprint 3
  - CS108-71   Sprint 3 Refactor
  - CS108-170 Functionality to Generate report
- Bug fixes
  - CS108-44 Password not encrypted
  - CS108-66 Incorrect login message overlaps with the content
  - CS108-78 Read values from config files
  - CS108-81 No user session management
  - CS108-93 No Navigations buttons
  - CS108-109 Does not validate c file input
  - CS108-130 No way to go to home page



Job Quality

Did the team present evidence that it did a quality job?
Did the team improve its performance over time?
(And is this claim supported with evidence?)

SonarQube Report



sonarqube Projects Issues Rules Quality Profiles Quality Gates Search for projects, sub-projects and

CS5500_TEAM_108_PLAGIARISM_DET... April 19, 2018 7:17 PM

Issues Measures Code Activity

Quality Gate Passed

Bugs & Vulnerabilities

Leak Period: since previous version
started 2 months ago

0 A Bugs	0 A Vulnerabilities	0 A New Bugs	0 A New Vulnerabilities

Code Smells

0 A Debt started 2 months ago	0 Code Smells	0 A New Debt	0 New Code Smells

Coverage

89.9% Coverage	38 Unit Tests	90.7% Coverage on 302 New Lines to Cover

Duplications

0.0% Duplications	0 Duplicated Blocks	0.0% Duplications on 3.9k New Lines

Source Code Plagiarism Detector Application for CS 5500 - MSD

S 2.2k Lines of Code

Java	1.9k
XML	292

No tags

Activity

April 19, 2018
0.0.1-SNAPSHOT

April 19, 2018
Project Analyzed

April 19, 2018
Project Analyzed

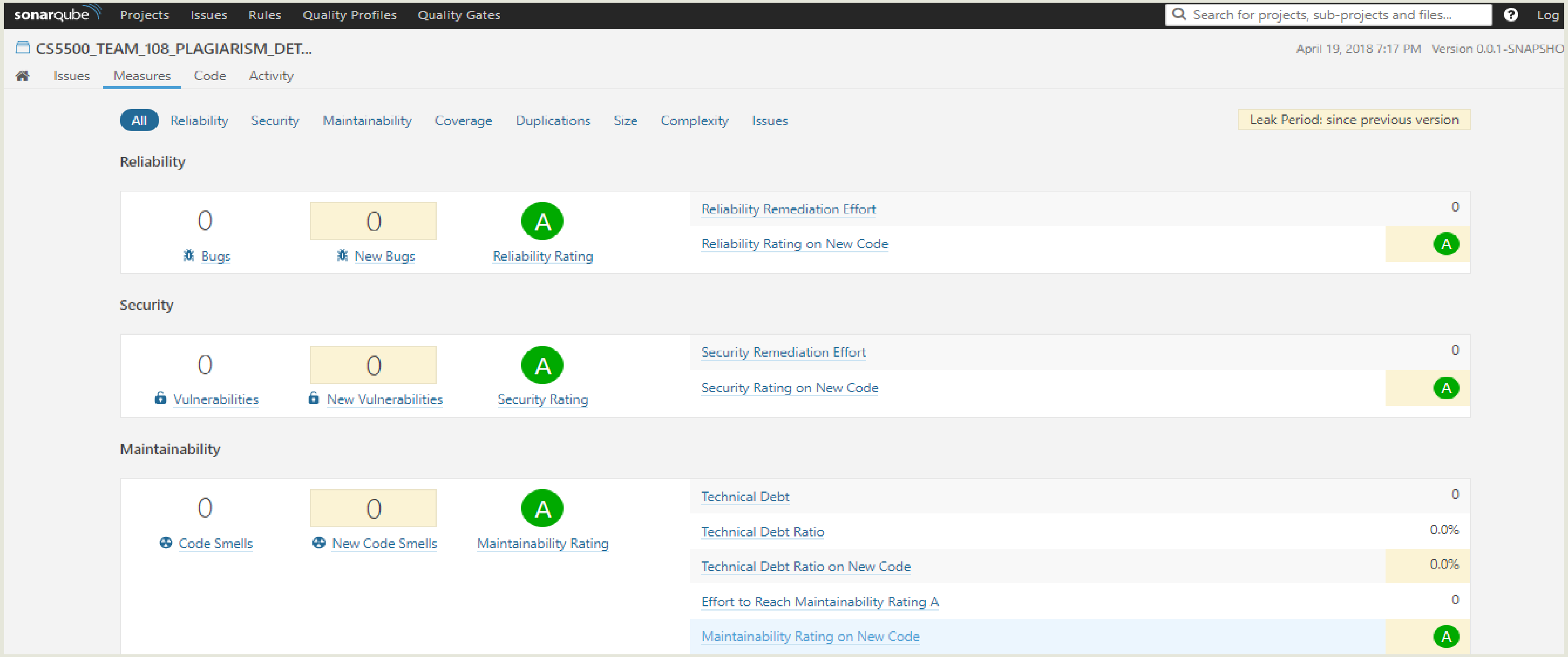
Show More

Quality Gate
(Default) SonarQube way

Quality Profiles
(Java) Sonar way
(XML) Sonar way

Home
Sources

Continued…



sonarqube Projects Issues Rules Quality Profiles Quality Gates Search for projects, sub-projects and files... Log

CS5500_TEAM_108_PLAGIARISM_DET... April 19, 2018 7:17 PM Version 0.0.1-SNAPSHO

Issues Measures Code Activity

All Reliability Security Maintainability Coverage Duplications Size Complexity Issues

Leak Period: since previous version

Reliability

0 Bugs	0 New Bugs	A Reliability Rating

Reliability Remediation Effort	0
Reliability Rating on New Code	A

Security

0 Vulnerabilities	0 New Vulnerabilities	A Security Rating

Security Remediation Effort	0
Security Rating on New Code	A

Maintainability

0 Code Smells	0 New Code Smells	A Maintainability Rating

Technical Debt	0
Technical Debt Ratio	0.0%
Technical Debt Ratio on New Code	0.0%
Effort to Reach Maintainability Rating A	0
Maintainability Rating on New Code	A

Bugs

- 105 bugs filed by testers, a vast majority of these bugs are now fixed.
- Large number of bugs filed were for features not implemented, deprecated features, along with duplicates.
- Significant issues with authentication, comparison module were identified and resolved.
- Helped identify UI changes the make the interface more intuitive.

Team Progress

- Our performance, as a team, was consistent throughout the sprints.
- As the expectations increased, we played to the strengths of the team and collaborated to cope with the requirements and implement features in time.
- Basic functionality implemented in sprint 1, crucial features implemented in sprints 2 and 3
- Feedback from the sprint reviews were incorporated, improving the deliverables throughout the development cycle.

Process and Teamwork

Did the team work as a team?

Did the team use process well or was the project mis/mal-organized?

Was the team able to automate the build, test, and promote processes?

Did the team recognize short-comings? If so, how well did they work around the challenges?

Did the team work as a team?

Together Everyone Achieves More

- The TEAM managed to achieve this objective by producing work beyond a single members scope.
- The TEAM had meetings to pitch in ideas and select what everyone liked.
- The TEAM grew as a unit by teaching each other our areas of expertise.
- The TEAM Resolved disagreements in a professional manner by discussing opinions.

So **YES,**

Our TEAM did work as a TEAM.

Did the team use process well or was the project miss/mal-organized?

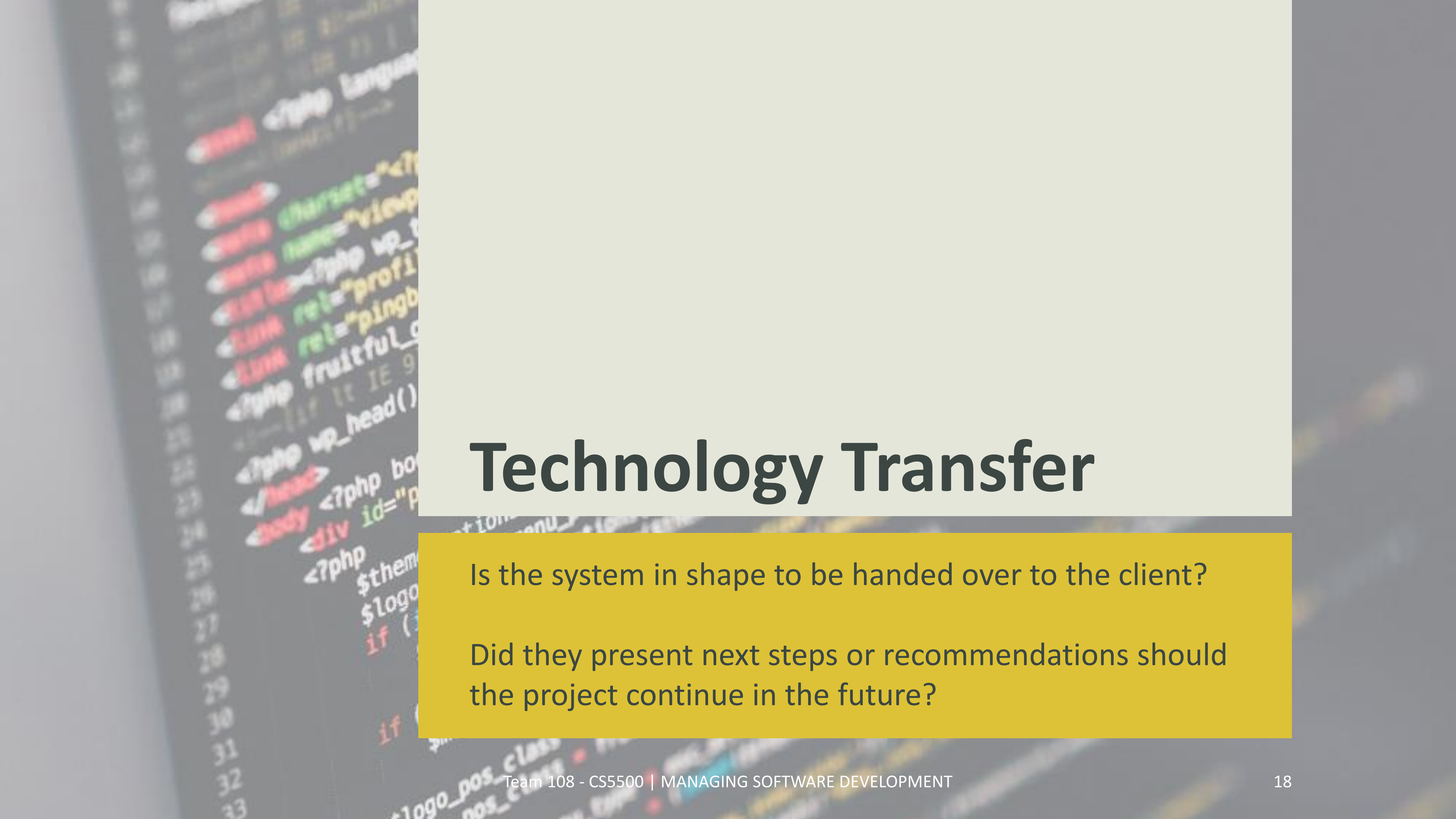
- The sprint’s tasks were well decided and assigned using Jira.
- We individually expanded each task into sub tasks and decided a passing criteria for them to be complete.
- The subtasks were complete only after testing the module and smart commits kept track of progress.
- We met before sprint deadline and verified correctness of one others code.
- So the Team used the Process well, and the process helped the team function better.

Was the team able to automate the build, test, and promote processes?

- The team used Jenkins for Continuous integration.
- Testing was a part of the Jenkins pipeline
- The configured J-Unit tests were executed in the build and the build only succeeded on passing those tests.
- On completion of building and testing automation the code went through a set of Quality and coverage checks using SonarQube.
- After the completions of these the steps the build is auto deployed onto AWS.

Did the team recognize short-comings? If so, how well did they work around the challenges?

- Few members of the team were new to Java and could not completely work on back-end logic.
  - To overcome this the members fluent in Java helped out by creating a skeleton for the task.
  - The fluent team members explained the process and pointed to resources that helped.
- Spring boot was a topic that nobody was experienced in, so creating endpoints for the back-end logic was a challenge.
  - We assigned two members to learn and create a proof on concept for creating endpoints with spring boot.
  - After successful proof of concepts we integrated the learnings to create end points for our back-end logic.



Technology Transfer

Is the system in shape to be handed over to the client?

Did they present next steps or recommendations should the project continue in the future?

Present System

- Current system is stable and working as expected
- Codebase is maintained in GitHub
- Code build and deployment is automated using Jenkins
- Test suits with coverage criteria are in place
- Code Quality is maintained using SonarQube
- AWS is used for deploying the application

System Logs and Error handling

- System logs the activity to logs maintained on the server
- Error handling is in place to recover from errors and exceptions
- Team is alerted of a system problem via email
- Jenkins build success/failures are notified to the team via email/slack messages

Maintenance & Future Steps

- System implemented using well know technologies like Java, Spring Boot, MySQL ANTLR in the backend and AngularJS, HTML/CSS on the front end
- Established frameworks/tools like Jenkins and AWS are used for build and deployment
- Easy to learn, pick up these technologies and continue the development
- Standard coding procedures are followed to the core
- Elaborate code installation and project structure details in GitHub README file
- Reference links to implementation/use cases are in place

Recommendations

- Technology stack is pretty solid and a good foundation for new development
- Comparison Strategies used could be improved
- Current ones i.e. LCS and NW fall short for some scenarios like type-4 plagiarism which involves complex syntactical similarities.
- Elaborate error handling at REST APIs response level could be added
- Front end code wasn’t tested much and could be improved

Thank You!