Plagiarism Detection - CS5500 Spring 2018 Project

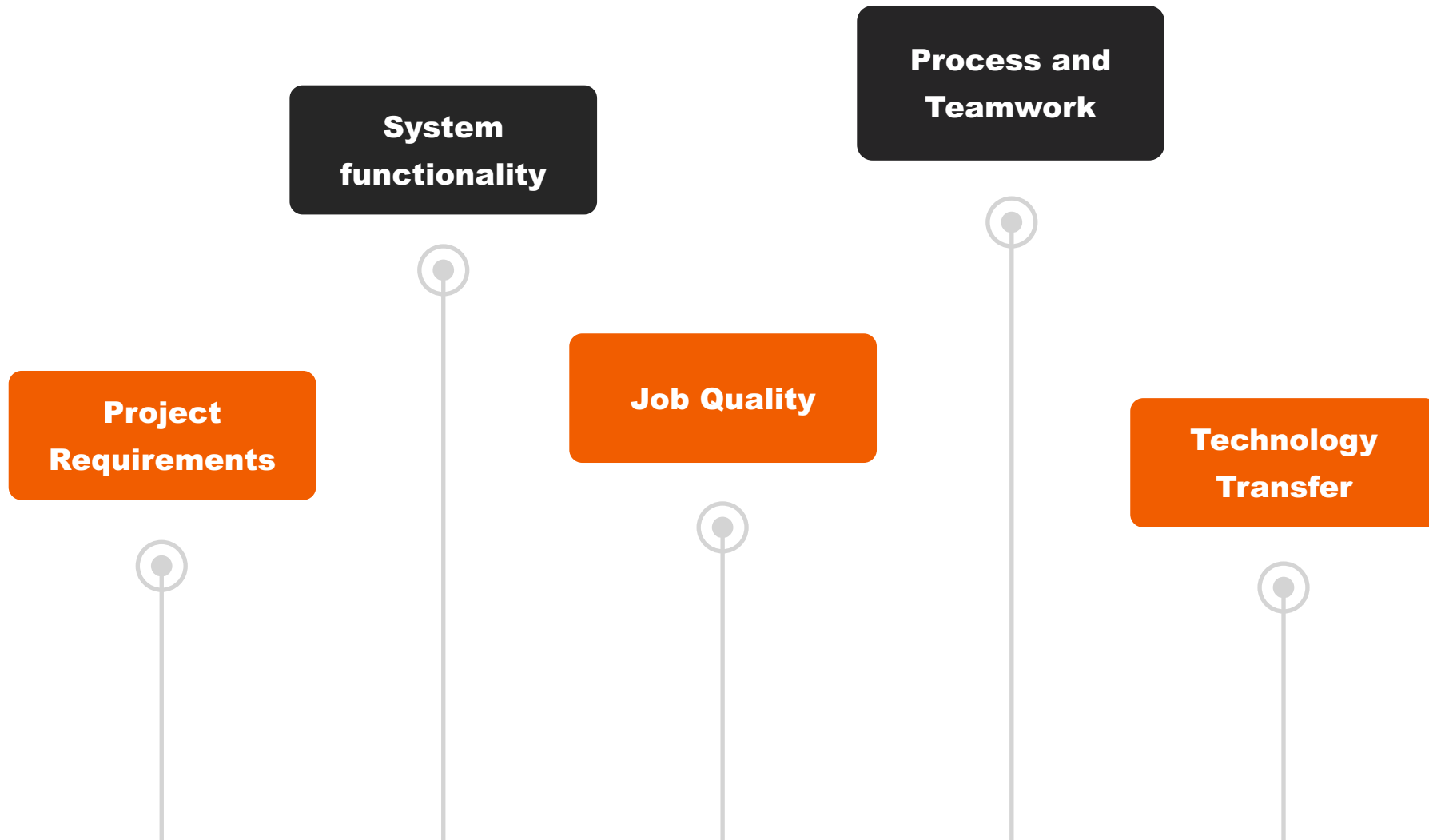**Professor: Mike Weintraub**

Team: Team – 105

Team Members: Naga Sai Anirudh Upadhyayula
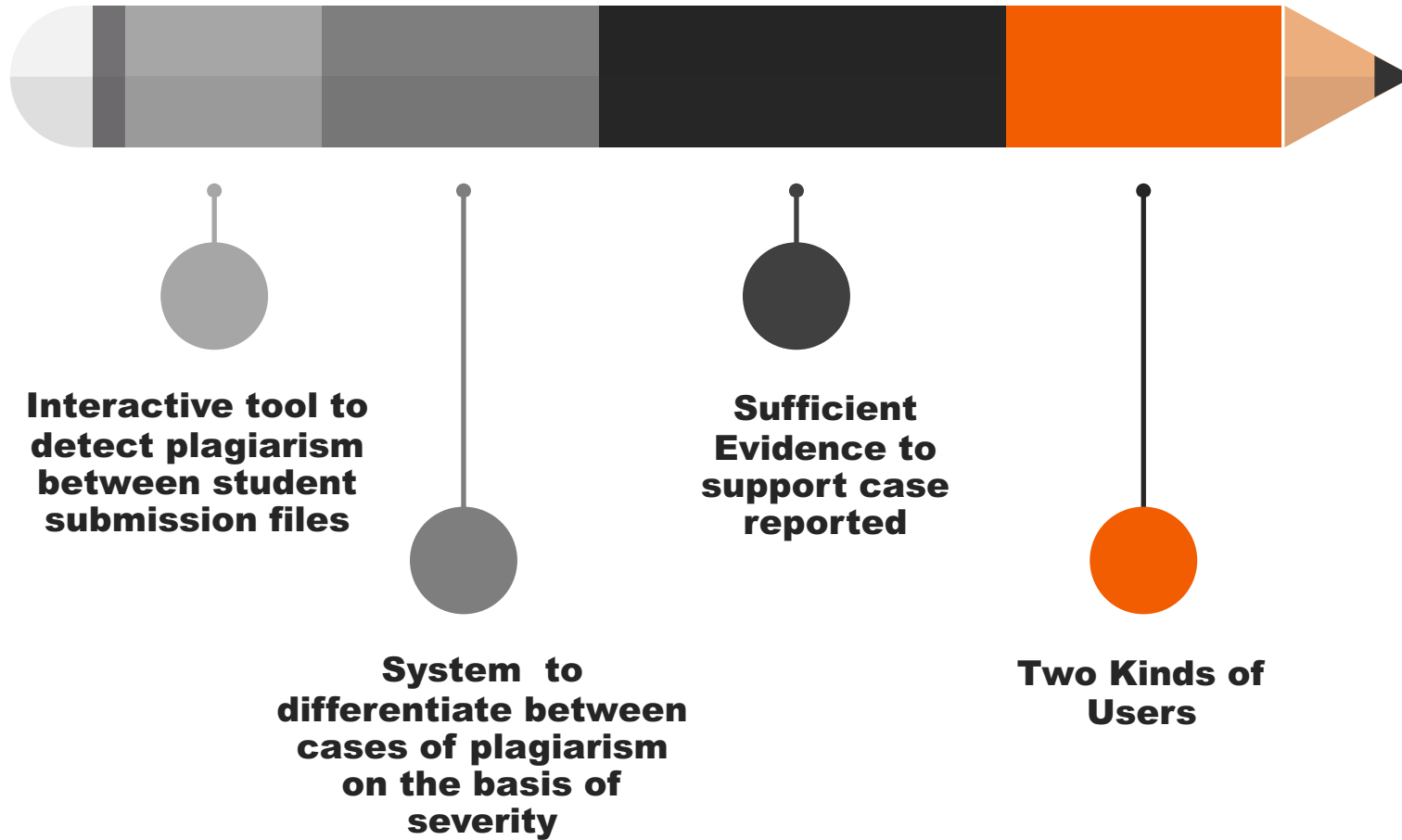
Nirupama Sharma

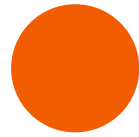**Aunsh Chaudhari**

Presentation Agenda
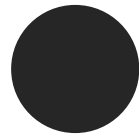
Project Requirements

System functionality

Job Quality

Process and Teamwork

Technology Transfer

Project Requirements

Interactive tool to detect plagiarism between student submission files

System to differentiate between cases of plagiarism on the basis of severity

Sufficient Evidence to support case reported

Two Kinds of Users

System Functionality

Login for multiple users

Github Integration

Plagiarism Detection Algorithm Implementation

Visualization of Plagiarism Statistics With Submission Metadata

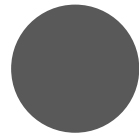Evidence of Plagiarism Through Highlighted Code

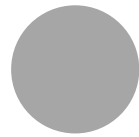Downloading Plagiarism Case Report

Plagiarism Scenarios

● Variable Name Change

● Function swapping

● Code Refactoring

● Files having the same author – possibly an exact match

```python
# 3sum.py
import collections


class Solution(object):
    def threeSum(self, nums):
        """
        :type nums: List[int]
        :rtype: List[List[int]]
        """
        nums, result, i = sorted(nums), [], 0
        while i < len(nums) - 2:
            if i == 0 or nums[i] != nums[i - 1]:
                j, k = i + 1, len(nums) - 1
                while j < k:
                    if nums[i] + nums[j] + nums[k] < 0:
                        j += 1
                    elif nums[i] + nums[j] + nums[k] > 0:
                        k -= 1
                    else:
                        result.append([nums[i], nums[j], nums[k]])
                        j, k = j + 1, k - 1
                        while j < k and nums[j] == nums[j - 1]:
                            j += 1
                        while j < k and nums[k] == nums[k + 1]:
                            k -= 1
            i += 1
        return result

    def threeSum2(self, nums):
        """
        :type nums: List[int]
        :rtype: List[List[int]]
        """
        d = collections.Counter(nums)
        nums_2 = [x[0] for x in d.items() if x[1] > 1]
        nums_new = sorted([x[0] for x in d.items()])
        rtn = [[0, 0, 0]] if d[0] >= 3 else []
        for i, j in enumerate(nums_new):
            if j <= 0:
                numss2 = nums_new[i + 1:]
                for x, y in enumerate(numss2):
                    if 0 - j - y in [j, y] and 0 - j - y in nums_2:
                        if sorted([j, y, 0 - j - y]) not in rtn:
                            rtn.append(sorted([j, y, 0 - j - y]))
                    if 0 - j - y not in [j, y] and 0 - j - y in nums_new:
                        if sorted([j, y, 0 - j - y]) not in rtn:
                            rtn.append(sorted([j, y, 0 - j - y]))
        return rtn


if __name__ == '__main__':
    result = Solution().threeSum([-1, 0, 1, 2, -1, -4])
    print result
```

```python
# 3sum -variable.py
import collections


class Solution(object):
    def threeSum(self, a):
        """
        :type a: List[int]
        :rtype: List[List[int]]
        """
        a, result, n = sorted(a), [], 0
        while n < len(a) - 2:
            if n == 0 or a[n] != a[n - 1]:
                j, k = n + 1, len(a) - 1
                while j < k:
                    if a[n] + a[j] + a[k] < 0:
                        j += 1
                    elif a[n] + a[j] + a[k] > 0:
                        k -= 1
                    else:
                        result.append([a[n], a[j], a[k]])
                        j, k = j + 1, k - 1
                        while j < k and a[j] == a[j - 1]:
                            j += 1
                        while j < k and a[k] == a[k + 1]:
                            k -= 1
            n += 1
        return result

    def threeSum2(self, b):
        """
        :type b: List[int]
        :rtype: List[List[int]]
        """
        d = collections.Counter(b)
        nums_2 = [z[0] for z in d.items() if z[1] > 1]
        nums_new = sorted([z[0] for z in d.items()])
        rtn = [[0, 0, 0]] if d[0] >= 3 else []
        for k, j in enumerate(nums_new):
            if j <= 0:
                numss2 = nums_new[i + 1:]
                for x, y in enumerate(numss2):
                    if 0 - j - y in [j, y] and 0 - j - y in nums_2:
                        if sorted([j, y, 0 - j - y]) not in rtn:
                            rtn.append(sorted([j, y, 0 - j - y]))
                    if 0 - j - y not in [j, y] and 0 - j - y in nums_new:
                        if sorted([j, y, 0 - j - y]) not in rtn:
                            rtn.append(sorted([j, y, 0 - j - y]))
        return rtn


if __name__ == '__main__':
    result = Solution().threeSum([-1, 0, 1, 2, -1, -4])
    print result
```

Quick Example of Results For Variable Change Scenario

Logged Issues – Backlog and Completed

Completed Issues

Sprint 1

Key	Summary	Issue Type	Priority	Status	Story Points (-)
CS105-1 *	Setting up development environment	Story	↑ Medium	CLOSED	-
CS105-2 *	Basic log In functionality for application	Story	↑ Medium	CLOSED	-
CS105-3 *	Generating Abstract Syntax Tree for File	Story	↑ Medium	CLOSED	-
CS105-4 *	First Comparison Strategy - Basic Algorithm	Story	↑ Medium	CLOSED	-
CS105-5 *	Creating UI as wireframes	Story	↑ Medium	CLOSED	-
CS105-6 *	Setting up Jenkins pipeline	Story	↑ Medium	CLOSED	-
CS105-7 *	Second comparison strategy between ASTs	Story	↑ Medium	CLOSED	-
CS105-8 *	Integrate SonarQube into Jenkins pipeline	Story	↑ Medium	CLOSED	-
CS105-9 *	UI beyond wireframes	Story	↑ Medium	CLOSED	-

Completed Issues

Sprint 2

Key	Summary	Issue Type	Priority	Status	Story Points (12)
CS105-10	Employ a sophisticated comparison strategy	Story	↑ Medium	CLOSED	1
CS105-11	Use other comparison strategies	Story	↑ Medium	CLOSED	2
CS105-12	Report results of each comparison strategy	Task	↑ Medium	CLOSED	3
CS105-14	Ensure system logs activity	Story	↑ Medium	CLOSED	1
CS105-16	Perform the comparison against multiple files used in two submissions	Story	↑ Medium	CLOSED	1
CS105-17	Performs the comparison against multiple submissions	Story	↑ Medium	CLOSED	2
CS105-18	UI for login	Story	↑ Medium	CLOSED	1
CS105-21	Strategy pattern to allow strategy on demand	Task	↑ Medium	CLOSED	1
CS105-24 *	Cloning Git repositories	Task	↑ Medium	CLOSED	-
CS105-25 *	Pulling changes to created Git repository	Task	↑ Medium	CLOSED	-
CS105-26 *	Longest Common Subsequence	Task	↑ Medium	CLOSED	-
CS105-27 *	Comparison Strategy - Using metadata of file	Story	↑ Medium	CLOSED	-
CS105-28 *	Make Sonarqube analysis and code private to team	Task	↑ Medium	CLOSED	-
CS105-30 *	Edit Distance String Comparison	Task	↑ Medium	CLOSED	-
CS105-31 *	System sends an email on exception	Story	↑ Medium	CLOSED	-
CS105-32 *	Backend - Test and Case classes	Task	↑ Medium	CLOSED	-
CS105-33 *	Sequence Detection - improvement of Basic Algorithm	Task	↑ Medium	CLOSED	-

Sprint 3

Completed Issues

Key	Summary	Issue Type	Priority	Status	Story Points (2)
CS105-15 *	Provide usage statistics	Story	↑ Medium	CLOSED	-
CS105-19 *	UI for showing similar lines between two files	Task	↑ Medium	CLOSED	1
CS105-20 *	UI for test results - comparison strategies	Story	↑ Medium	CLOSED	1
CS105-22 *	Compute overall score using weighted polynomial function	Story	↑ Medium	CLOSED	-
CS105-34 *	Training Observed Results of Strategies according to MOSS data	Story	↑ Medium	CLOSED	-
CS105-35 *	Refactor, integrate strategies and tests	Task	↑ Medium	CLOSED	-
CS105-36 *	Create candidate branch for HW5 with packaged version	Story	↑ Medium	CLOSED	-
CS105-39 *	Test issue- dummy	Bug	↑ Medium	CLOSED	-
CS105-40 *	Test Issue	Story	↑ Medium	CLOSED	-
CS105-41 *	Include line numbers in plagiarism result	Story	↑ Medium	CLOSED	-
CS105-43 *	Testcases giving errors for pythonpasrsers.Python3Parsers testcases, 1 for Login	Bug	↑ Medium	CLOSED	-
CS105-45 *	Javadoc errors in ASTParser, TreeTraverse, Instructor, PlagiarismResult, ASTStrategy, EditDistance, LCS	Task	↑ Medium	CLOSED	-
CS105-46 *	refactoring Antlr code	Story	↑ Medium	CLOSED	-
CS105-47 *	Updating references to show appropriate plagiarism result objects on page redirect	Task	↑ Medium	CLOSED	-
CS105-49 *	Connect to MOSS	Task	↑ Medium	CLOSED	-
CS105-50 *	create six sets of test projects	Story	↑ Medium	CLOSED	-
CS105-51 *	Test set for author strategy	Story	↑ Medium	CLOSED	-
CS105-52 *	Store files on AWS from Git	Task	↑ Medium	CLOSED	-
CS105-53 *	Configure AWS Elasticbeanstalk instances for storing files and directories	Task	↑ Medium	CLOSED	-
CS105-54 *	Integrate Moss with weighted polynomial, new tests and folder structure	Task	↑ Medium	CLOSED	-
CS105-55 *	UI for usage statistics	Task	↑ Medium	CLOSED	-

Steps followed by team to ensure Code Quality

Unit and Functional Tests

SonarLint on local system

Peer Code Reviews

Jenkins

SonarQube
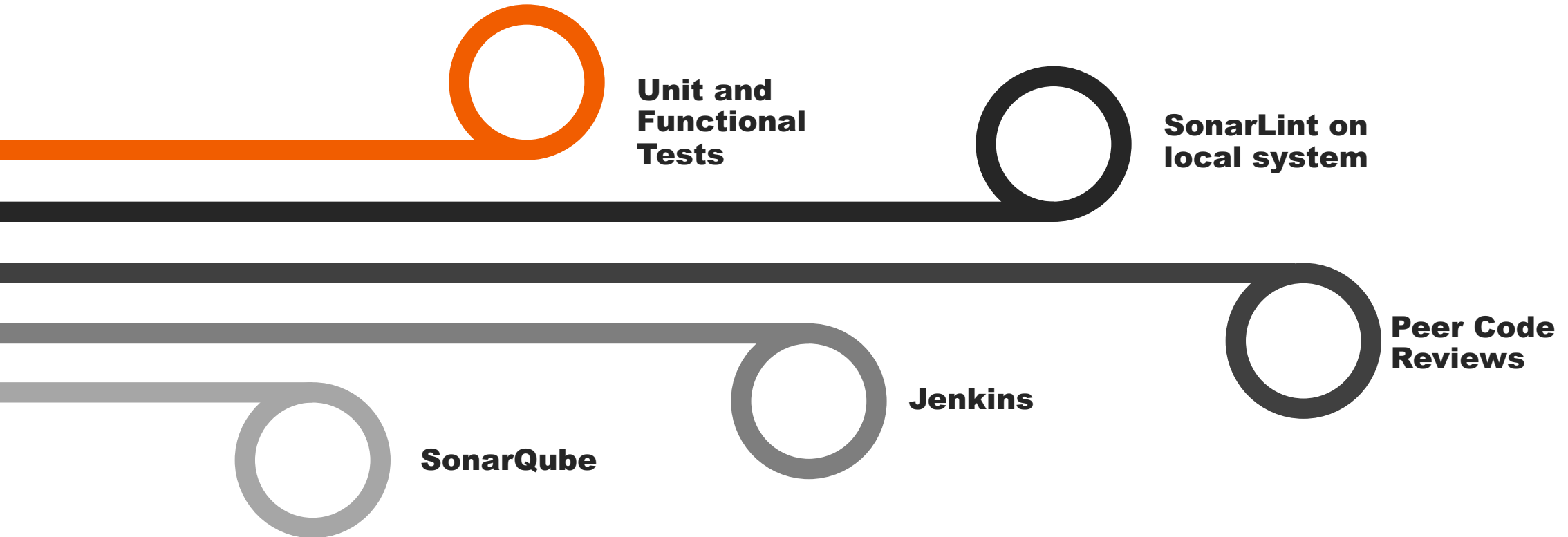
68 Unit Tests

Achieved 85.9% code coverage

Focused on pulling logic out of controllers into services to test them

Kept a lookout for bugs, vulnerabilities and suggestions from SonarLint

SonarQube Report

Q Search for projects, sub-projects and files...   ?   T

☆ ⬜ **cs5500-sp2018-team105** Private   April 20, 2018 9:15 AM   Version 0.0.1-SNAPSHOT

🏠   Issues   Measures   Code   Activity

Quality Gate   Passed

Properties file for plagiarism check

Bugs & Vulnerabilities

| | | Leak Period: since previous version |
| | | started a month ago |

0 A   0 A   0 A   0 A
🐛 Bugs   🔒 Vulnerabilities 🐛 New Bugs   🔒 New Vulnerabilities

S 1.8k   Java ▬ 1.6k
Lines of Code   XML ▎ 148

🏷 No tags

Activity

Code Smells

6h A   26   6h A   26
Debt   ⊕ Code Smells   New Debt   ⊕ New Code Smells
started a month ago

April 20, 2018
0.0.1-SNAPSHOT

April 20, 2018
Project Analyzed

Coverage

85.9%   68   86.3%
Coverage   Unit Tests   Coverage on
                             779 New Lines to Cover

April 20, 2018
Project Analyzed

Show More

Quality Gate
(Default) SonarQube way

Duplications

2.9%   2   2.0%
Duplications Duplicated Blocks Duplications on
                             4.3k New Lines

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

Key
Team-105

How did we as a team improve performance and code quality?

Write more unit tests

Refactor methods and reduce cognitive complexity

Closer inspection of code through peer reviews

Better error handling

Client – Developer communication

Improved logging
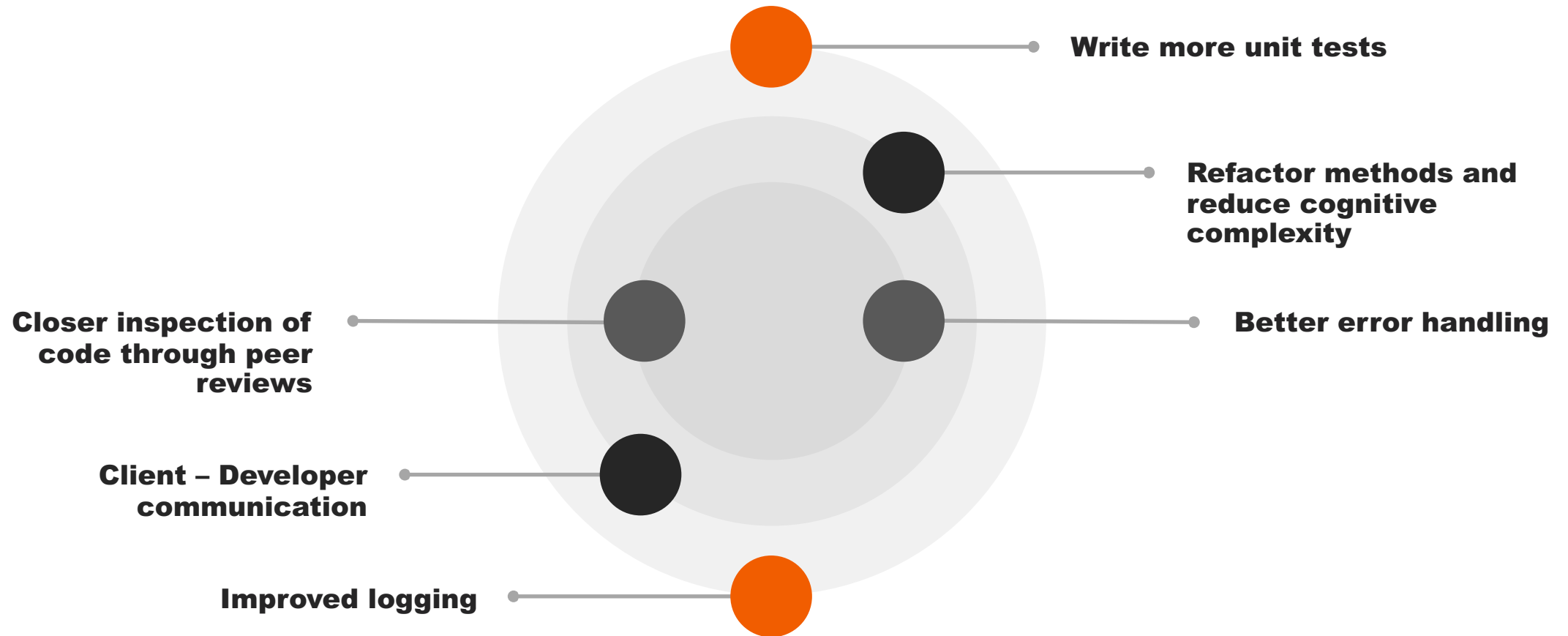
SonarQube Report History – Improvement Over Time
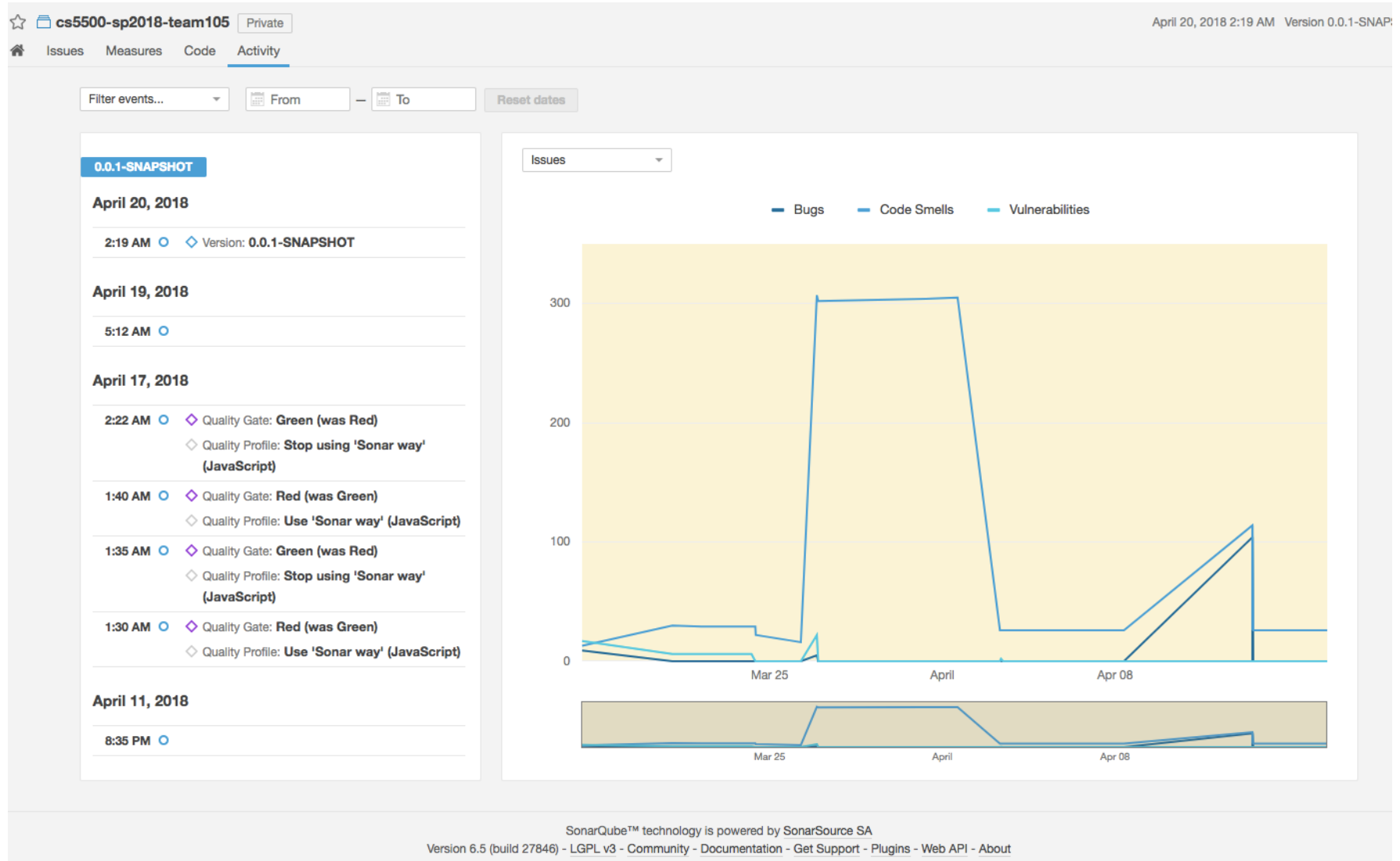
Process

- Development process was divided into 3 sprints

- Priority list of requirements was placed in the backlog

- Face to face meetings to discuss roadblocks and design decisions

- Pulled in tickets and created sub tasks as and when needed through sprint

- Feedback through Sprint Review with TA

Task Division and Focus

Sprint 1

initial set up of development environment
- **Maven project structure**
- **Jenkins Github integration**
- **SonarQube**

Abstract Syntax Trees
- **Generation from code files using ANTLR library**
- **Sophisticated Comparison Strategy**

Login
- **Basic registration, login and database integration**
- **UI for the same**

Sprint 2

■ Improved sophisticated strategy

■ Other strategies – String comparison and Metadata

■ Strategy Design Pattern

■ Github Integration

■ UI

■ Logging

Sprint 3

Weighted Polynomial Strategy

MOSS Training

Display Usage Statistics and System Status

Reporting Plagiarism Results and Metadata on UI

Highlighting Common Code in Plagiarism Case

Validating plagiarism detection against six test sets

Automating Development Process and Environment Setup

Smart commits on JIRA

Jenkins run on all pull request submissions

Github informs team of PRs

Sonarqube configured in Jenkins pipeline

Jenkins informs team of build failure via email

Maven project deployed as webapp on AWS
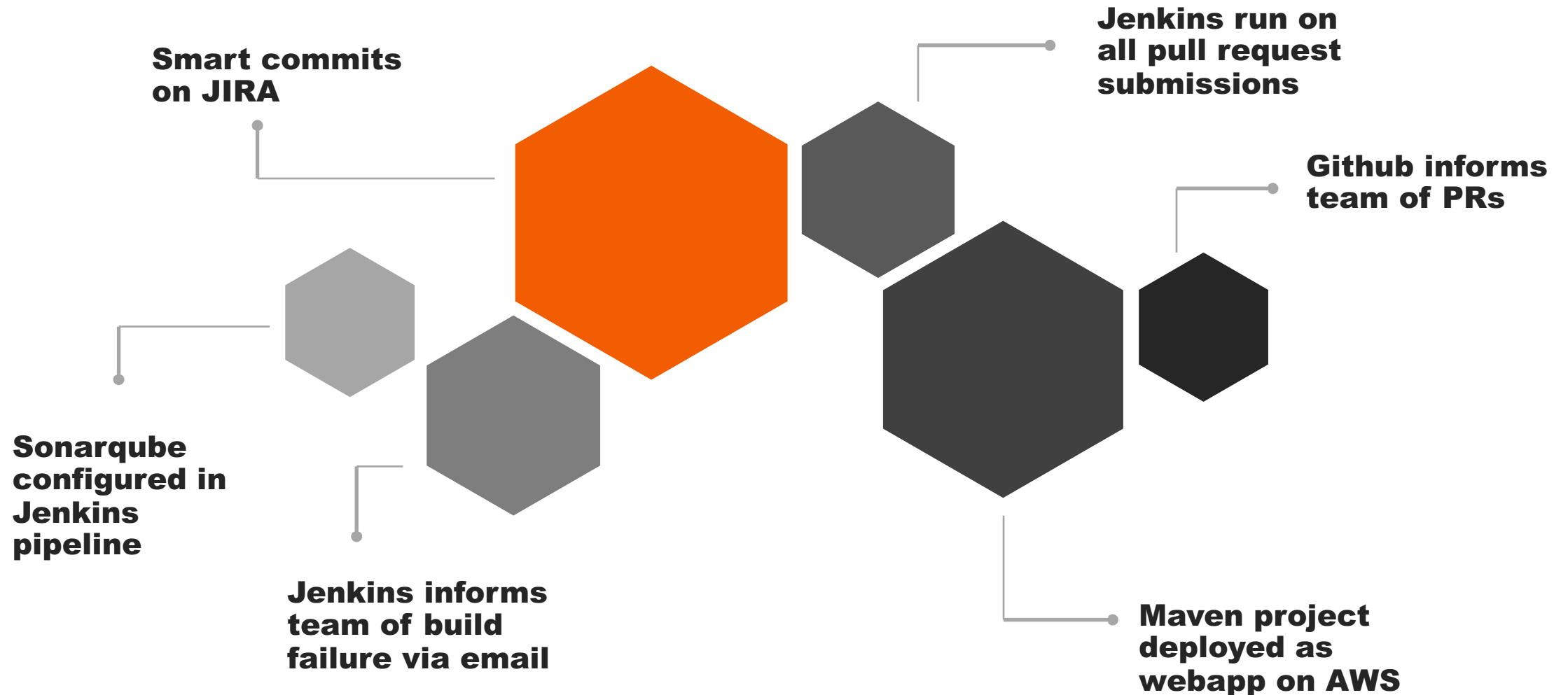
Steps followed by team to push code to Master

Push completed code from local branch to remote branch

Create a pull request

At least one approval required

All 3 checks on Jenkins must pass

Add more commits by pushing to the **UIFix** branch on **cs5500/team-105**.

Review required — Show all reviewers
At least one approved review is required by reviewers with write access. Learn more.

Some checks haven't completed yet — Hide all checks
2 pending and 1 successful checks

continuous-integration/jenkins/pr-head — This commit is being built — Required — Details

continuous-integration/jenkins/pr-merge — This commit is being built — Required — Details

✓ continuous-integration/jenkins/branch — This commit looks good — Required — Details

Merging is blocked — Update branch
Merging can be performed automatically with one approved review.

Merge pull request ▾ You can also open this in GitHub Desktop or view command line instructions.

Jenkins Builds

Branches

Pull Requests

Master

Configure
Scan Repository Now
Scan Repository Log
Repository Events
Delete Repository
People
Build History
Project Relationship
Check File Fingerprint
Open Blue Ocean
GitHub
Pipeline Syntax
Credentials

Build Queue
No builds in the queue.

Build Executor Status
master

Team repo for Team 105 (Section 1, Team 5)

Branches (35) Pull Requests (4)

S	W	Name ↓	Last Success	Last Failure	Last Duration	Fav
		AlgoTests	N/A	N/A	N/A	
		ASTAlgorithm	N/A	N/A	N/A	
		AstAlgorithm	N/A	N/A	N/A	
		backend	N/A	N/A	N/A	
		candidate	16 days - #1	N/A	3 min 25 sec	
		ConfigBranch	13 days - #8	13 days - #6	5 min 16 sec	
		DisplayComparedFiles	13 days - #2	N/A	3 min 30 sec	
		DisplayPlagiarism	N/A	N/A	N/A	
		Email2	21 hr - #2	21 hr - #1	4 min 58 sec	
		finalBranch	2 days 23 hr - #4	3 days 0 hr - #3	5 min 15 sec	
		GITClones-branch	N/A	N/A	N/A	
		GITIntegration	N/A	N/A	N/A	
		ImprovingUI	13 days - #1	N/A	4 min 58 sec	
		IntegratingStrategies	22 days - #1	N/A	2 min 43 sec	
		loggerBranch	N/A	N/A	N/A	
		LoggerImplementation	N/A	26 days - #8	2 min 10 sec	
		master	3 days 0 hr - #93	13 days - #86	9 min 48 sec	
		MOSS	14 days - #5	14 days - #7	3 min 0 sec	
		NewAlgo	16 days - #4	15 days - #5	5 min 25 sec	
		NewIntegration-GIT	22 days - #1	N/A	2 min 43 sec	

Team-105-JenkinsJob

Team repo for Team 105 (Section 1, Team 5)

Branches (35) Pull Requests (4)

S	W	Name ↓	Last Success	Last Failure	Last Duration	Fav
		PR-57-head	2 days 21 hr - #1	N/A	5 min 14 sec	
		PR-57-merge	2 days 21 hr - #1	N/A	10 min	
		PR-58-head	21 hr - #2	21 hr - #1	9 min 57 sec	
		PR-58-merge	21 hr - #2	21 hr - #1	15 min	

Icon: S M L

Legend RSS for all RSS for failures RSS for just latest builds

Up
Status
Changes
Build Now
View Configuration
Full Stage View
SonarQube
Open Blue Ocean
GitHub
Pipeline Syntax

Branch master

Full project name: Team-105-JenkinsJob/master

Recent Changes

Stage View

	Declarative: Checkout SCM	Declarative: Agent Setup	Build	Test	SonarQube	Quality
Average stage times: (Average full run time: ~6min 48s)	4s	778ms	1min 16s	45s	1min 48s	27s
#96 Apr 20, 2018 02:15 6 commits	10s	753ms	1min 31s	57s		
3min 56s						
#93 Apr 17, 2018 01:26 3 commits	1s	609ms	1min 17s	1min 3s	1min 50s	32s
#90 Apr 06, 2018 23:33 9 commits	1s	759ms	1min 33s	1min 11s	1min 52s	32s

Build History trend

find

#96	Apr 20, 2018 6:15 AM
#93	Apr 17, 2018 5:26 AM
#90	Apr 7, 2018 3:33 AM
#88	Apr 7, 2018 12:29 AM
#87	Apr 6, 2018 11:54 PM

Teamwork

Effective Communication

Division of Tasks on JIRA

Pair Programming

Effective peer code review for better code quality

Mutual Design Decisions

Challenges faced by Team and how did we work around them?

Project Structure – SonarQube

Lack of skills/knowledge of front end technologies like Angular/REACT

Understanding of ANTLR jar for language specific AST Generation

Adapting to the practice of Test Driven Development

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

All client requirements have been completed

Test Driven Development – Rigorously tested by developers

Major bugs in the system discovered by testers of the system were triaged, assigned to developers and fixed systematically
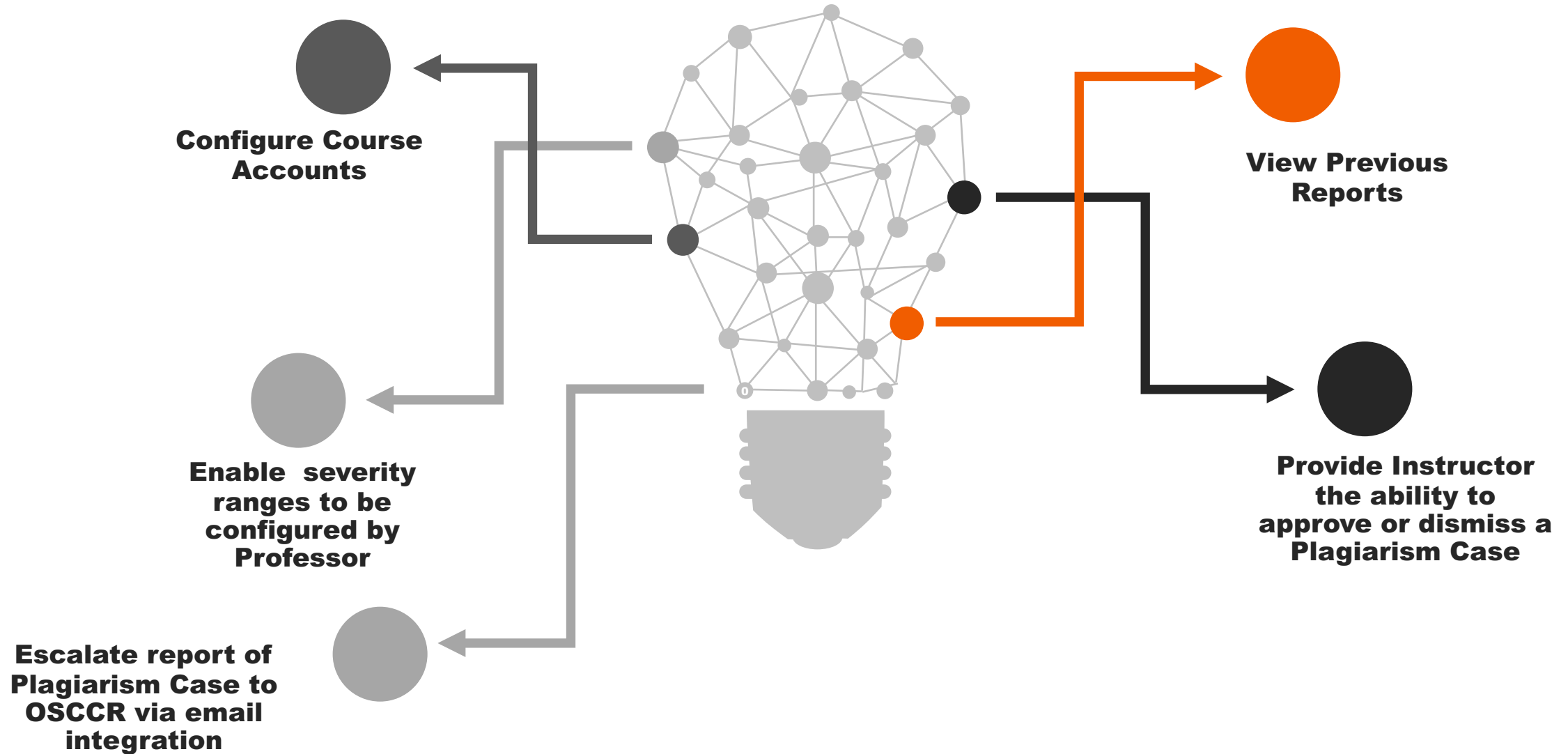
Validated plagiarism detection accuracy against six test sets portraying different scenarios

UI and UX - User friendly and easily navigable

Future Scope

Configure Course Accounts

View Previous Reports

Enable severity ranges to be configured by Professor

Provide Instructor the ability to approve or dismiss a Plagiarism Case

Escalate report of Plagiarism Case to OSCCR via email integration

Thank You!