Team 10

Jonathan Chery
Harshil Jain
Gaurav Jain
Rahul Sinha

System Functionality

- Team 10 goal was to create a secure messaging system that allows users to message each other, create and join Groups, update and delete Groups, and perform functionalities that allows one-to-one messaging and group messaging to be simple and secure.
- By the end of the 6-weeks, team 10 was able to accomplish 68% of what they sought out to do.
- 68% allows the client to:
    - create an account
    - log into their account
    - search users/groups
    - broadcast a message
    - message other users privately
    - create/update/delete a group
    - join a group
    - add/invite/remove users and moderators to or from a group
    - change their status
    - notify users of their status and requests

System Functionality

- Team 10 had created **65 Issues**
  - *18 Functional Requirements*
  - *32 Non-Functional Requirement*
  - *15 Miscellaneous (Documentation, UML, Bug fixes)*
- Team 10 had **successfully** completed **49 Issues**
  - *18 Functional Requirements*
  - *16 Non-Functional Requirements*
  - *15 Miscellaneous (Documentation, UML, Bug fixes)*
- Team 10 had **unsuccessfully** completed **16 Issues**
  - *0 Functional Requirements*
  - *16 Non-Functional Requirements*
  - *0 Miscellaneous (Documentation, UML, Bug fixes)*

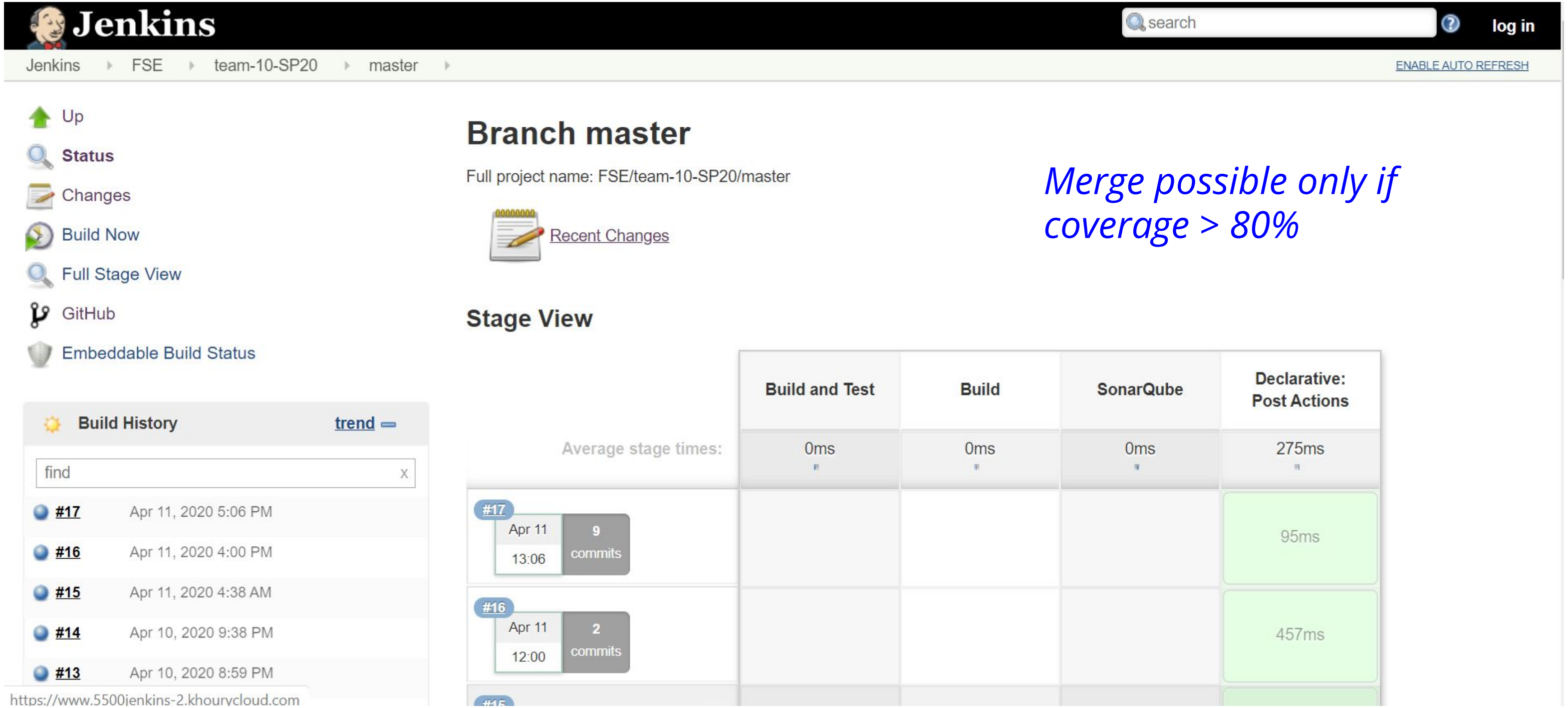
Results

1. The team accomplished 68% of functionalities.
2. The 68% consist of 100% of functional requirements and 50% of non-functional requirements.
3. The team delivered a MVP to the client that does C.R.U.D functionalities for Users, Groups, and Messages defined with the SRS documentation.

Job Quality

- Team 10 received **83.3% Coverage** on SonarQube.
- Team 10 produced **125 Unit Tests** and received **100% success** on those test.
- Team 10 received **85.6% line coverage** on SonarQube.
- Team 10 received **68.2 condition coverage** on SonarQube.

Job Quality



Branch master

Full project name: FSE/team-10-SP20/master

Stage View

	Build and Test	Build	SonarQube	Declarative: Post Actions
Average stage times:	0ms	0ms	0ms	275ms
#17 Apr 11 13:06 9 commits				95ms
#16 Apr 11 12:00 2 commits				457ms

Build History

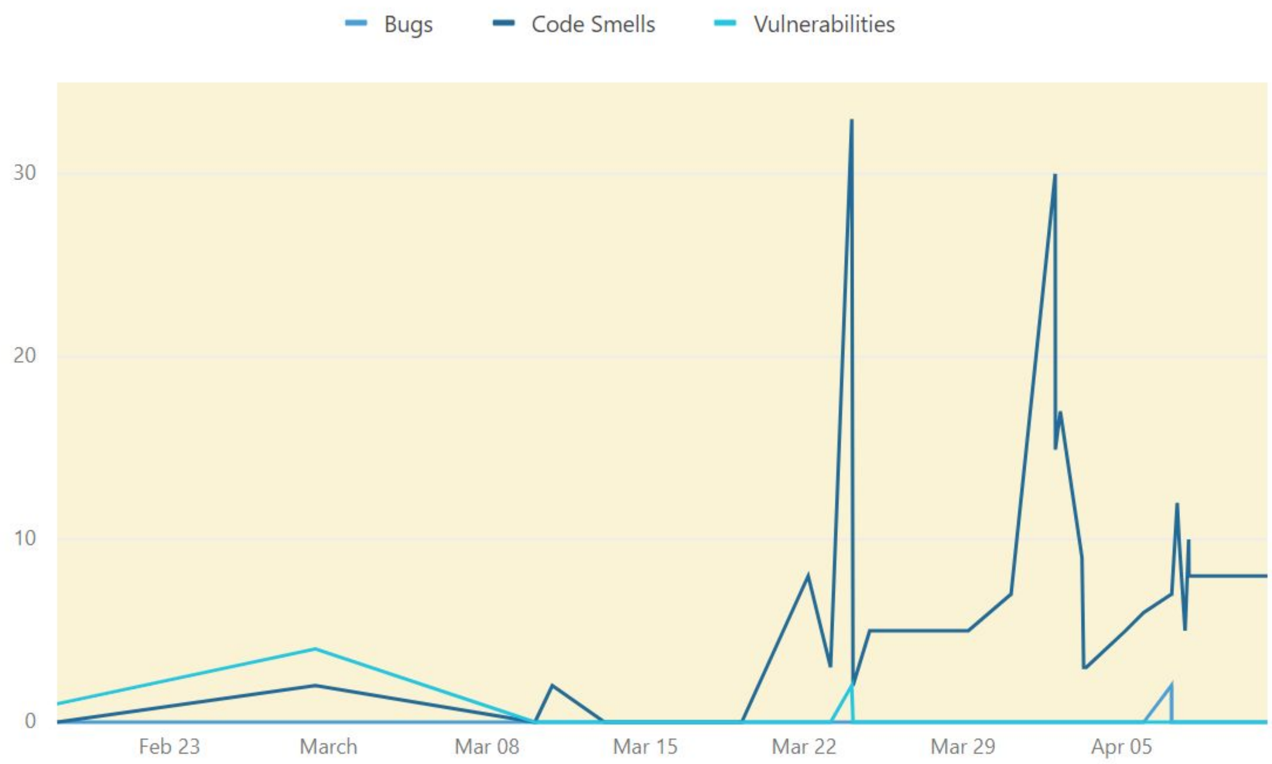
#17	Apr 11, 2020 5:06 PM
#16	Apr 11, 2020 4:00 PM
#15	Apr 11, 2020 4:38 AM
#14	Apr 10, 2020 9:38 PM
#13	Apr 10, 2020 8:59 PM

https://www.5500jenkins-2.khourycloud.com

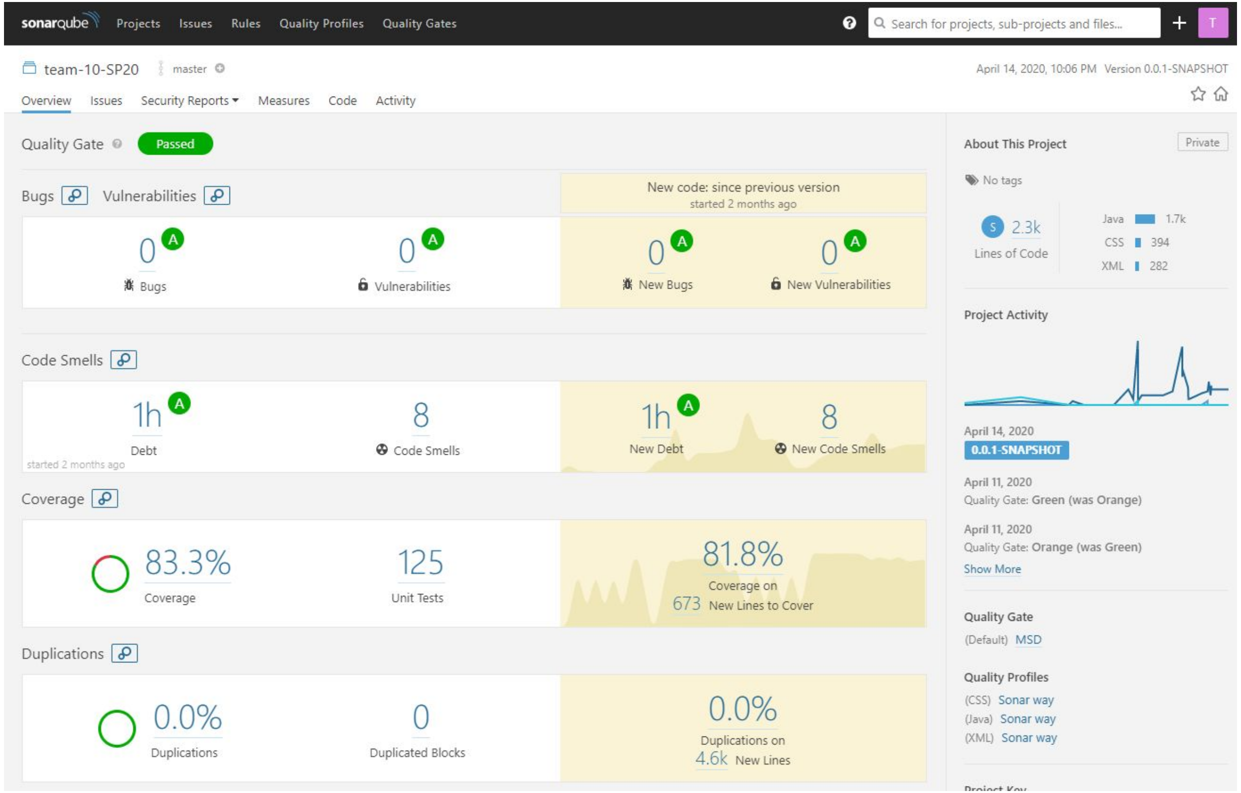
Merge possible only if coverage > 80%

CICD Model

Job Quality


Bugs
Code Smells
Vulnerabilities
30
20
10
0
Feb 23
March
Mar 08
Mar 15
Mar 22
Mar 29
Apr 05

Job Quality



team-10-SP20 master

April 14, 2020, 10:06 PM Version 0.0.1-SNAPSHOT

Quality Gate Passed

Bugs Vulnerabilities

New code: since previous version
started 2 months ago

0 A Bugs

0 A Vulnerabilities

0 A New Bugs

0 A New Vulnerabilities

Code Smells

1h A Debt
started 2 months ago

8 Code Smells

1h A New Debt

8 New Code Smells

Coverage

83.3% Coverage

125 Unit Tests

81.8% Coverage on 673 New Lines to Cover

Duplications

0.0% Duplications

0 Duplicated Blocks

0.0% Duplications on 4.6k New Lines

About This Project Private

No tags

S 2.3k Lines of Code

Java	1.7k
CSS	394
XML	282

Project Activity

April 14, 2020
0.0.1-SNAPSHOT

April 11, 2020
Quality Gate: Green (was Orange)

April 11, 2020
Quality Gate: Orange (was Green)

Show More

Quality Gate
(Default) MSD

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

Project Key

Process and Teamwork

- Team 10’s **challenges**
  - *Communication*
  - *Project Management*
  - *Time Management*
- Team 10’s **successes**
  - *Improved on Communication through the use of communicate tools (Google Meet, Slack)*
  - *Improves on Project Management through the use of integrating (Github, Jenkins) into Slack messaging.*
  - *Added Jenkins master restriction onto another branch to ensure code quality is sufficient.*
- Team 10’s **shortcomings**
  - *Did not implement more than 85% of the non-requirements.*
  - *2/4 Sprints hindered our success to deliver 100% of the product.*

Process and Teamwork

1. The team had a few minor bumps in the road however, the team managed to improve on areas that hindered their success.
2. The team managed to deliver 100% of the functional requirements but, suffered in the non-functional requirements.
3. The team was able to improve their teamwork throughout the progress of the project.

Technology Used

- Team 10 used HTML/CSS/Javascript for UI functionalities
  - *The team did not want to use too many new or advance technologies to fulfil the purpose on the client-side.*
  - *Team members had a better understanding of the basics of HTML, CSS, & Javascript.*

- Team 10 used WebSocket
  - *The team used the JAVA API for Websockets that provides bi-directional, full-duplex, real-time client/server communications.*
  - *Because it runs over TCP, it also provides a low-latency low-level communication and reduces the overhead of each message.*

- Team 10 used Relational SQL Database
  - *The team used Java Persistence API for accessing, managing and persisting data between the Java classes and a relational database hosted on AWS*
- Team 10 used Mockito for testing
  - *Mockito provided a way to test the services and controllers of the application without touching the database.*

Technology Transfer

- Easy and Secure sign up and login process.
- The application enables users to message each other effectively .
- Secure messaging with end to end encryption of messages.
- Efficient group chat functionality.
- Message Persistence : Messages stick around forever in the database and are not lost.
- Group moderator can manage group effectively.
- Simple to search for and follow other users and see their status updates in feed for a good social experience.
- The government has special privilege to wiretap any person of interest and monitor their activity on the system.

Next Steps

- Create a better and visually appealing user interface to provide a better User experience.
- Enhancement to the status feed. A separate feed just to see status updates of the users being followed.
- Content filtering would be a great addition. Inappropriate , abusive messages can be flagged and blocked.
- Enabling users to send multimedia messages such as audio, video, images and GIF.
- Enabling users to view the other users who are online and toggle their online/offline status.
- Enhancement to types of messages, such as private messages and poll messages.
- Allow the application to run on mobile platforms such as Android and iOS.