Managing Software Development (CS5500)

Team number: 209

Members:

1. Akhilesh Hegde
2. Elavazhagan Sethuraman
3. Varsha Muroor Rao
4. Vinayakaram Nagasubramanian

Agenda

- Introduction
- Development process
- Product quality
- Futuristic scope
- Statistics

Introduction

- A slack-like chat application with additional features.
- Starter code
  - Prattle server
  - Broadcasting client
- Incremental application development by prioritizing requirements.

Core entities/functionalities

- Users
- Groups
- Private Messaging
- Group Messaging
- Security
  - Authentication
  - Encryption

Features

- Forward messages
- Track messages
- Timeout messages
- Pending messages
- Chat History
- Translation (using IBM Watson’s API)
- Filtering flagged terms
- Reply to sub-group
- Delete messages
- Wire-tapping

Tools and Frameworks

- Jenkins – Automated building tool
- GitHub – Version Control
- JIRA – Task Management
- AWS – Deployment server
- Java – Application Development
- Mockito – Structural Testing
- MySQL – Persistence
- SonarQube – Quality Control
- Slack – Communication Channel

Development process

- Agile methodology
- Three different perspectives:
  - Client Layer
  - Server Layer
  - Persistence Layer
- Created backlogs
  - Story points for prioritization
  - Sub tasks for fast paced development
- Pick up tasks from product backlogs for each sprint

Implementation of ideas

- Individual research of methodologies
- Brainstorming with the team
- Feature Development
- Unit Testing
- Pull request
- Peer review of code
- Merge

Evolution of product

- Sprint specific development branch for each sprint
- Daily standup through SlackBot
- Semi weekly team meet ups
- Pair programming
- Collaboration during application integration
- Periodic feedback from the Client

Code Robustness

- Relevant usage of design patterns
  - Singleton, Bridge, Factory method, Observer
- Decoupled components in each layer
- Utility Components (especially for tests)
- Constants files for common access

Optimization

- Computational burden shifted to persistence layer
- Error handling / Fall back control
- Well abstracted and encapsulated code
- Refactoring as and when necessary

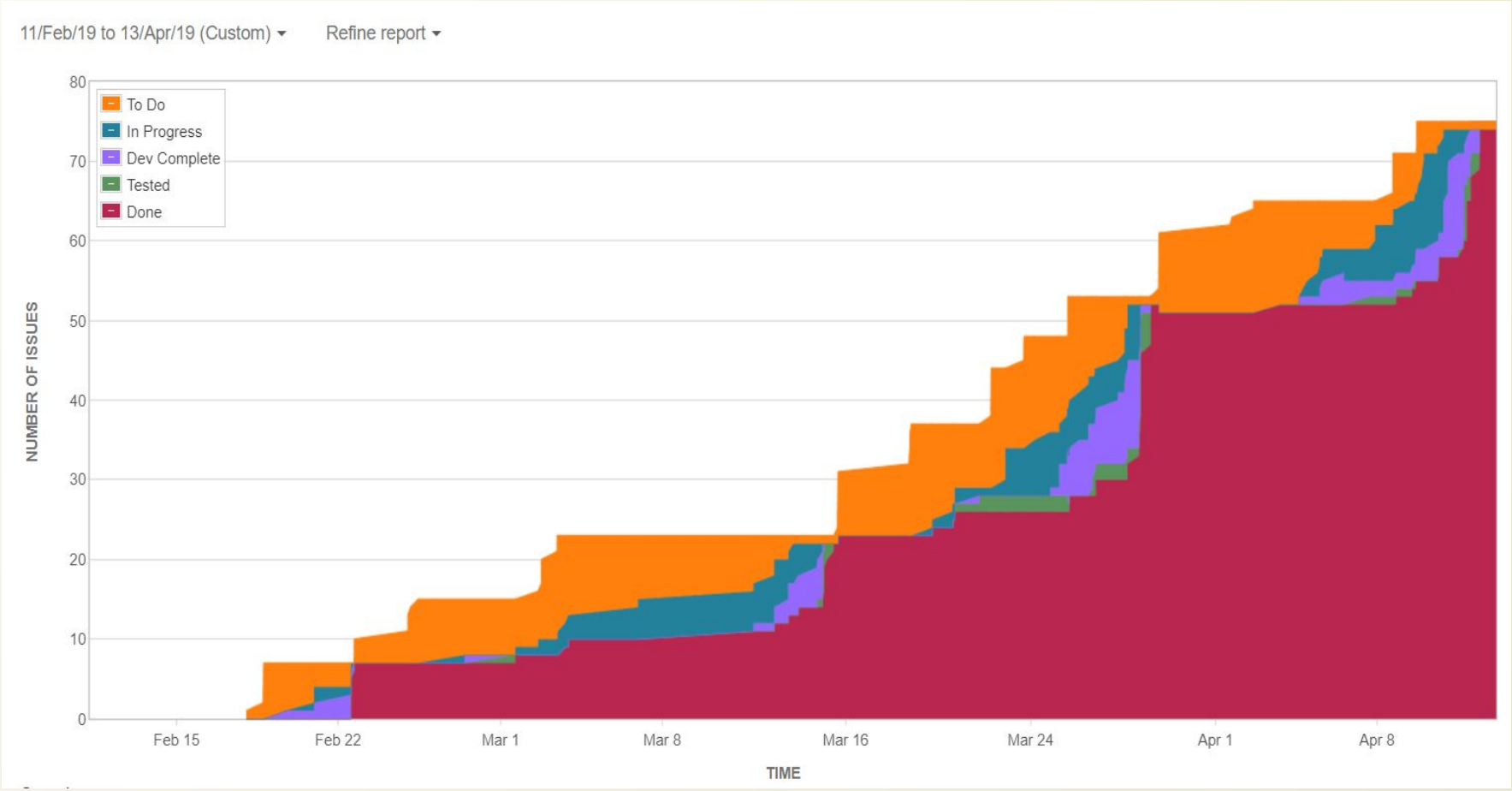
Testing

- **Functional testing**
  - Performed after integration of each feature
- **Structural testing**
  - Powered by Mockito and Reflection
  - Proper set up and tear down to ensure consistency
  - Succinct adequate tests with the aid of utility methods

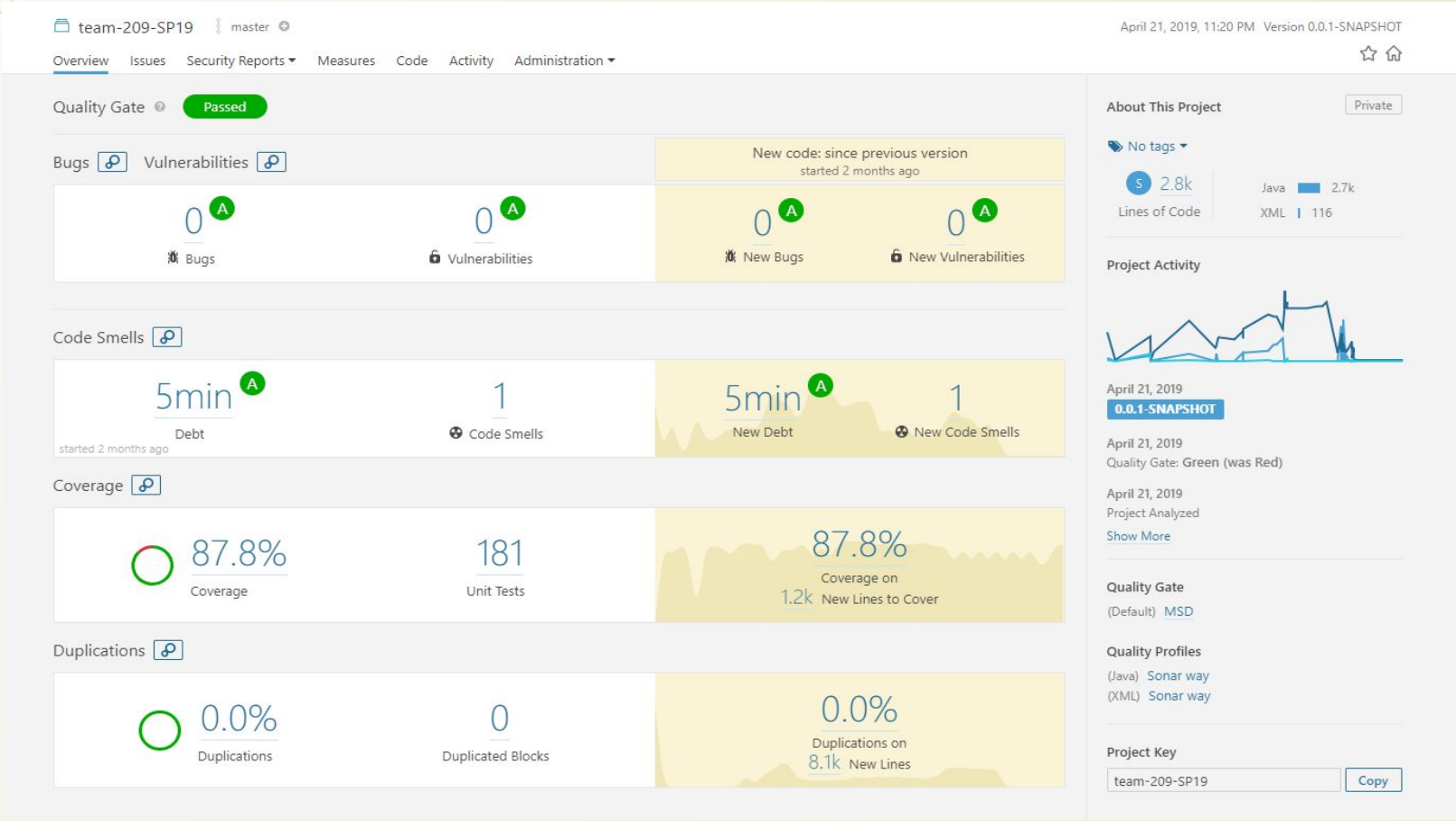
Challenges

- Testing the server code
  - Lot of private methods and network connections using ports
  - Used Mockito and reflections
- Support future requirements to extend functionalities
  - Continuous refactoring- POJOS
- Server and client communication
  - Standardizing conventions
- Deployment to AWS
  - Triaged and included the dependencies while running the jar

Statistics


11/Feb/19 to 13/Apr/19 (Custom)
Refine report
To Do
In Progress
Dev Complete
Tested
Done
NUMBER OF ISSUES
0
10
20
30
40
50
60
70
80
Feb 15
Feb 22
Mar 1
Mar 8
Mar 16
Mar 24
Apr 1
Apr 8
TIME

SonarQube metrics



Scope for improvement/ Future work

- Minimize the database and server interactions
  - use caching mechanisms
- Implement a web interface to enrich user experience
- Personalized profile
- Enable multimedia support
  - Images, Videos, Emojis and Files

Thank You