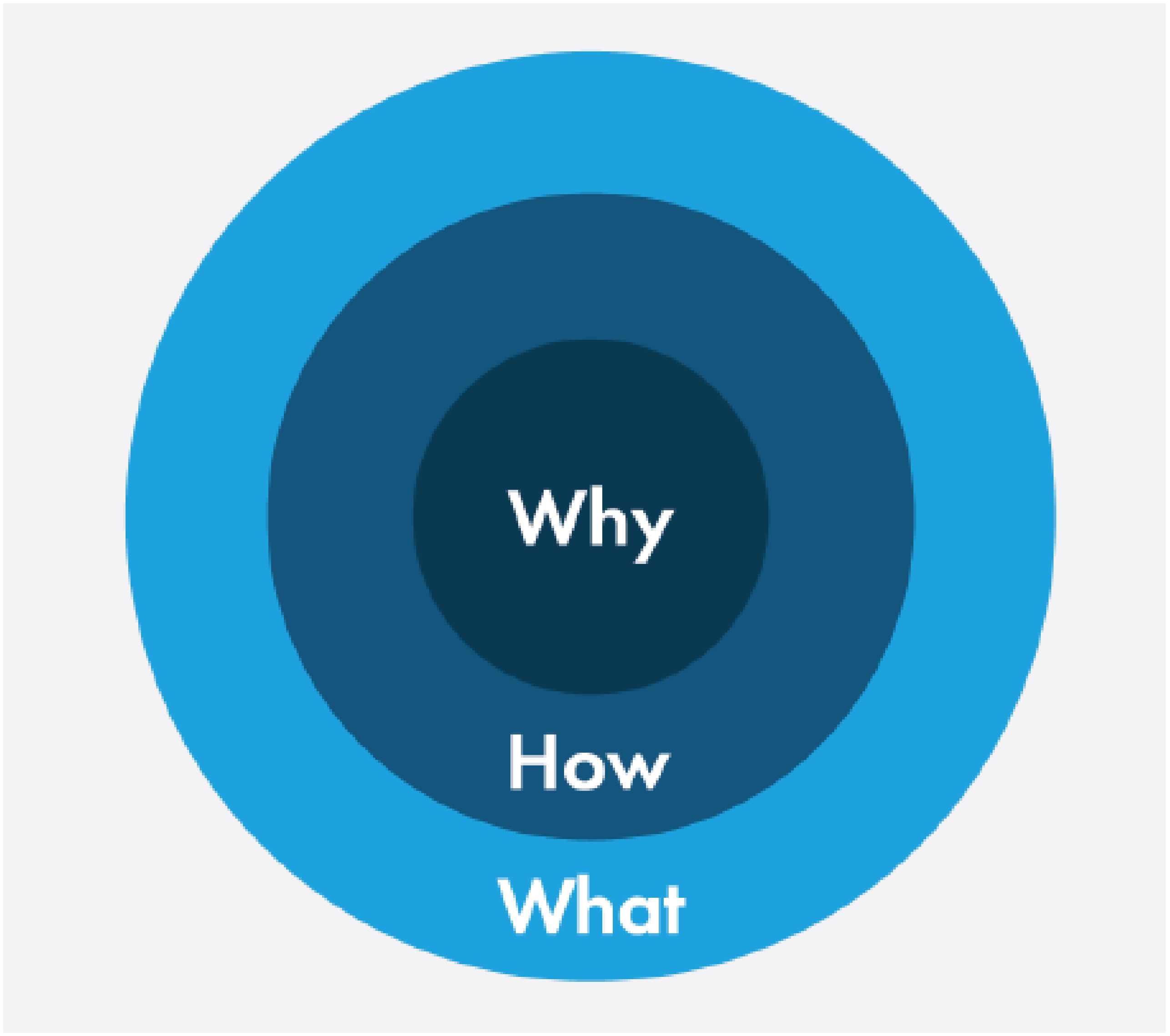
HEINZ NIXDORF INSTITUT
UNIVERSITÄT PADERBORN

Integration of Multiple Static Analysis Tools in a Single Interface

- G. S. Varma

- Supervisors:
  - Prof. Dr. Eric Bodden
  - Dr.-Ing. Ben Hermann


Why
How
What

Problem Statement

- How to integrate the results of multiple static analysis tools

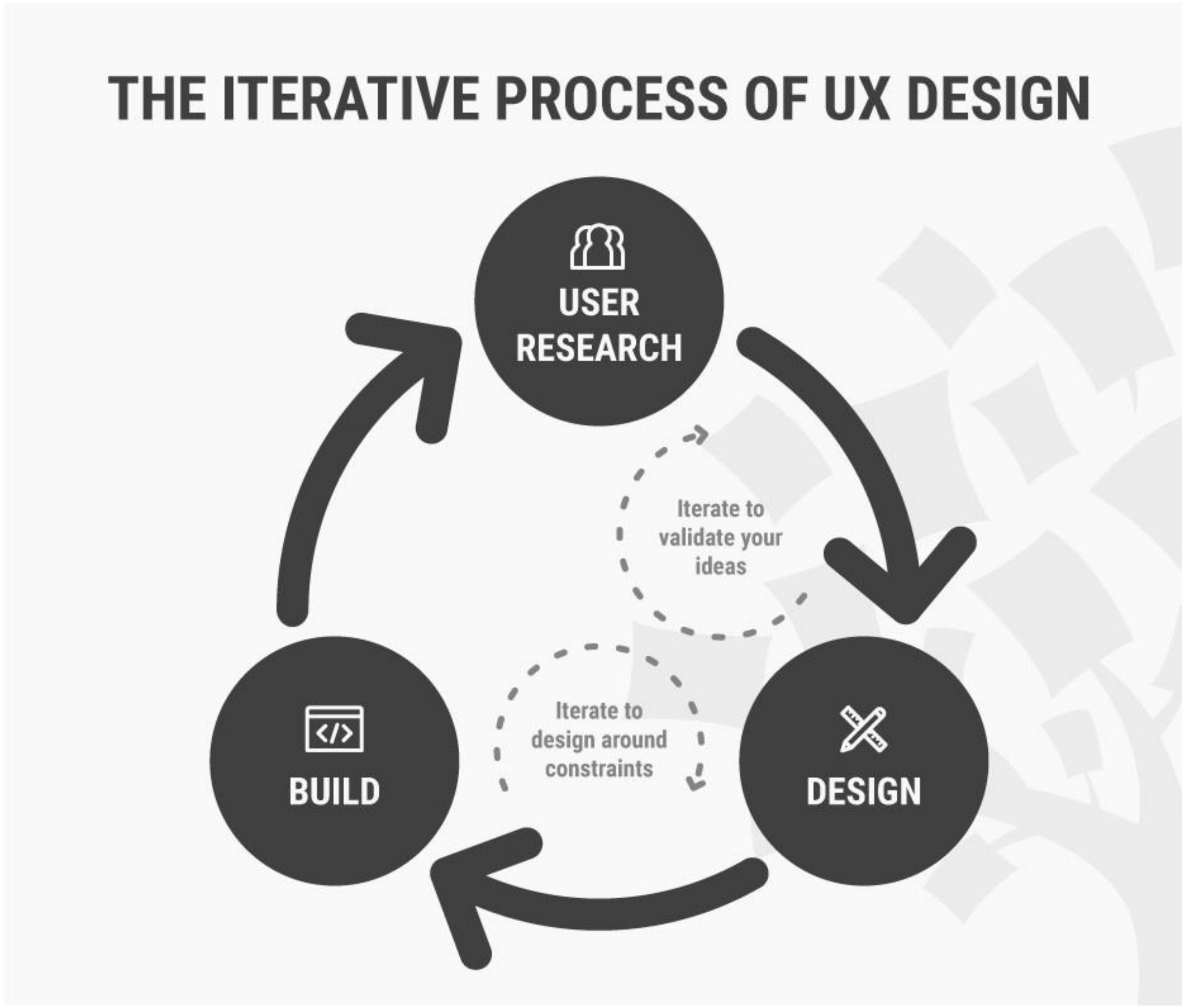
  in a unified user interface?

  - 3 Research Questions

Research Questions

- RQ 1: How to display results of the same codebase from different analysis tools?
- RQ 2: What feedback works to know that bug fixing is on-going?
- RQ 3: How to carry traceability of bug fixing?

Research Methodology


THE ITERATIVE PROCESS OF UX DESIGN
USER RESEARCH
DESIGN
BUILD
Iterate to validate your ideas
Iterate to design around constraints

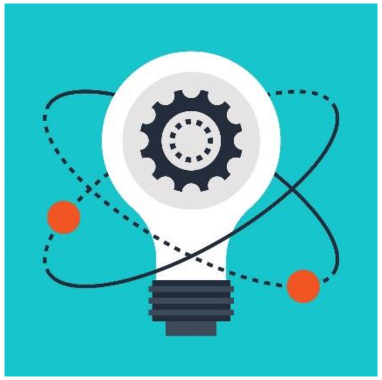
❖ How to Change Your Career from Graphic Design to UX Design. url: https://www.interaction-design.org/literature/article/how-to-change-your-career-fromgraphic-design-to-ux-design.

Our Approaches

- Software Engineering disciplines:
    - Complex datasets
    - Compiler reporting
    - Continuous integration
    - Refactoring tools
    - Issue tracker
    - Stack Overflow
    - Gamification
    - Usability Engineering

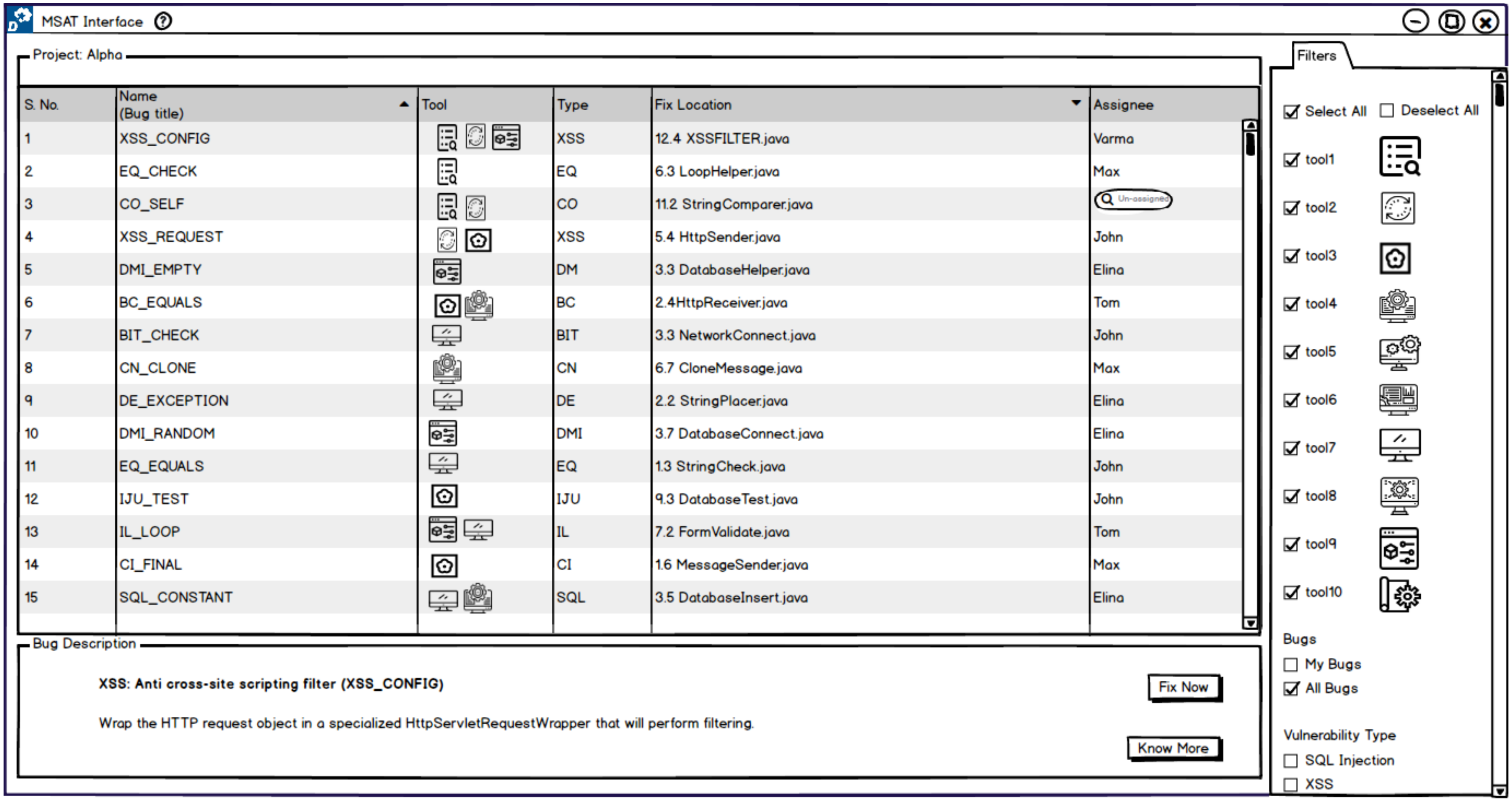
Evaluation

- Experimental Design
  - Recruit Test Users
  - Order of evaluation altered
  - Perform Tasks (Metric 1 – Task Success)
  - Likert Scale (Metric 2 – Perceived Usability)
  - Usability inspection methods: Cognitive Walkthrough



❖ Rensis Likert. “A technique for the measurement of attitudes.” In: Archives of psychology (1932).

Analysis View



MSAT Interface

Project: Alpha

S. No.	Name (Bug title)	Tool	Type	Fix Location	Assignee
1	XSS_CONFIG		XSS	12.4 XSSFILTER.java	Varma
2	EQ_CHECK		EQ	6.3 LoopHelper.java	Max
3	CO_SELF		CO	11.2 StringComparer.java	Un-assigned
4	XSS_REQUEST		XSS	5.4 HttpSender.java	John
5	DMI_EMPTY		DM	3.3 DatabaseHelper.java	Elina
6	BC_EQUALS		BC	2.4HttpReceiver.java	Tom
7	BIT_CHECK		BIT	3.3 NetworkConnect.java	John
8	CN_CLONE		CN	6.7 CloneMessage.java	Max
9	DE_EXCEPTION		DE	2.2 StringPlacer.java	Elina
10	DMI_RANDOM		DMI	3.7 DatabaseConnect.java	Elina
11	EQ_EQUALS		EQ	1.3 StringCheck.java	John
12	IJU_TEST		IJU	9.3 DatabaseTest.java	John
13	IL_LOOP		IL	7.2 FormValidate.java	Tom
14	CI_FINAL		CI	1.6 MessageSender.java	Max
15	SQL_CONSTANT		SQL	3.5 DatabaseInsert.java	Elina

Bug Description

XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized HttpServletRequestWrapper that will perform filtering.

Fix Now

Know More

Filters

☑ Select All ☐ Deselect All

☑ tool1

☑ tool2

☑ tool3

☑ tool4

☑ tool5

☑ tool6

☑ tool7

☑ tool8

☑ tool9

☑ tool10

Bugs

☐ My Bugs

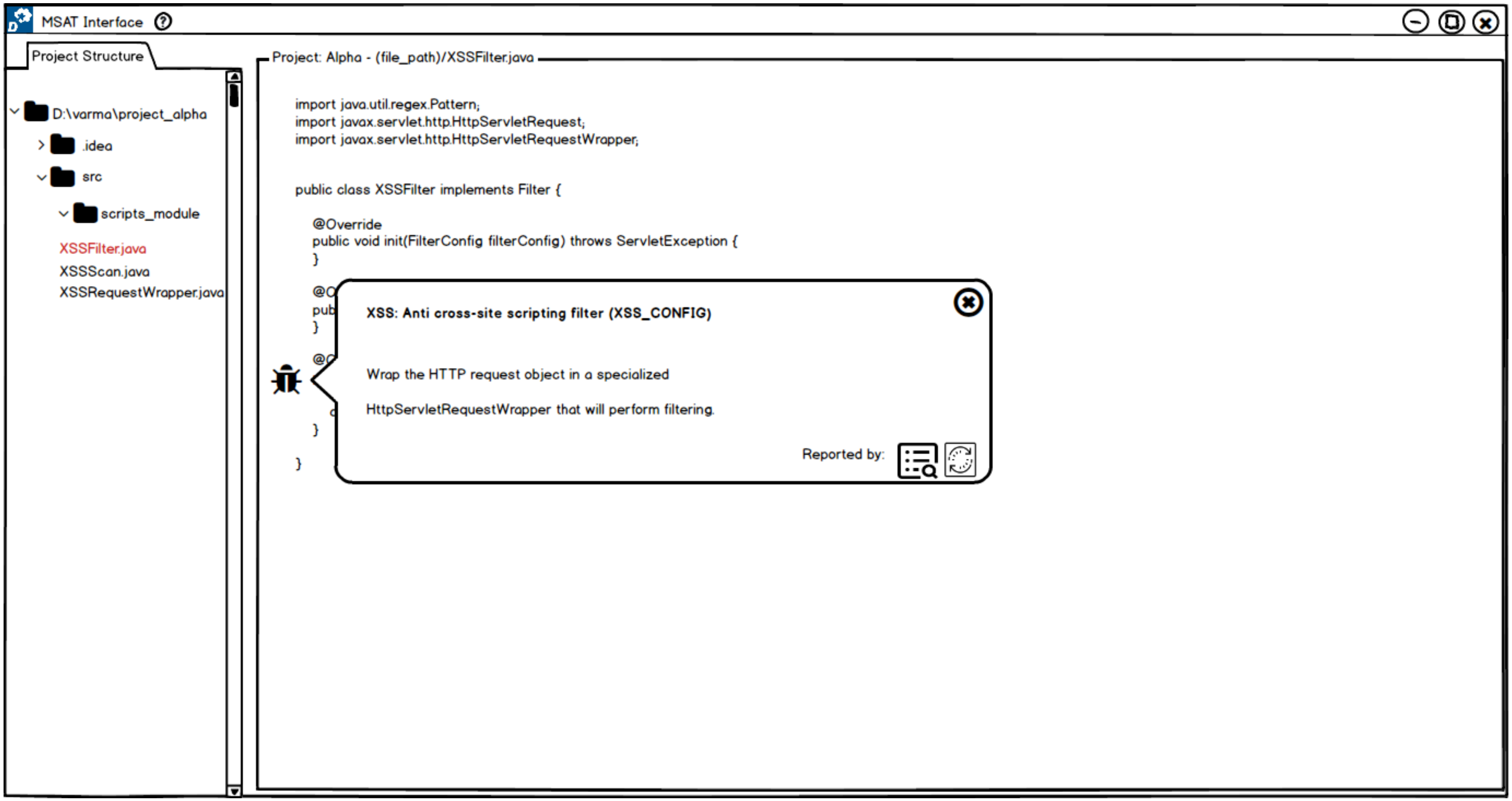
☑ All Bugs

Vulnerability Type

☐ SQL Injection

☐ XSS

Code View



```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;


public class XSSFilter implements Filter {

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
    }

    @O
    pub
    }

    @O


    }

}
```

XSS: Anti cross-site scripting filter (XSS_CONFIG)

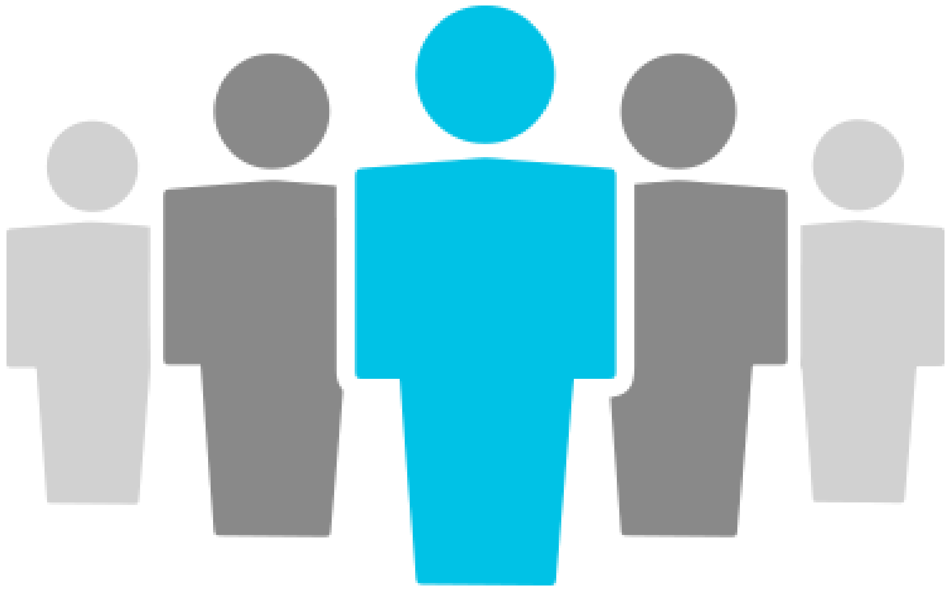
Wrap the HTTP request object in a specialized

HttpServletRequestWrapper that will perform filtering.

UX Design Cycle 1

UX 1

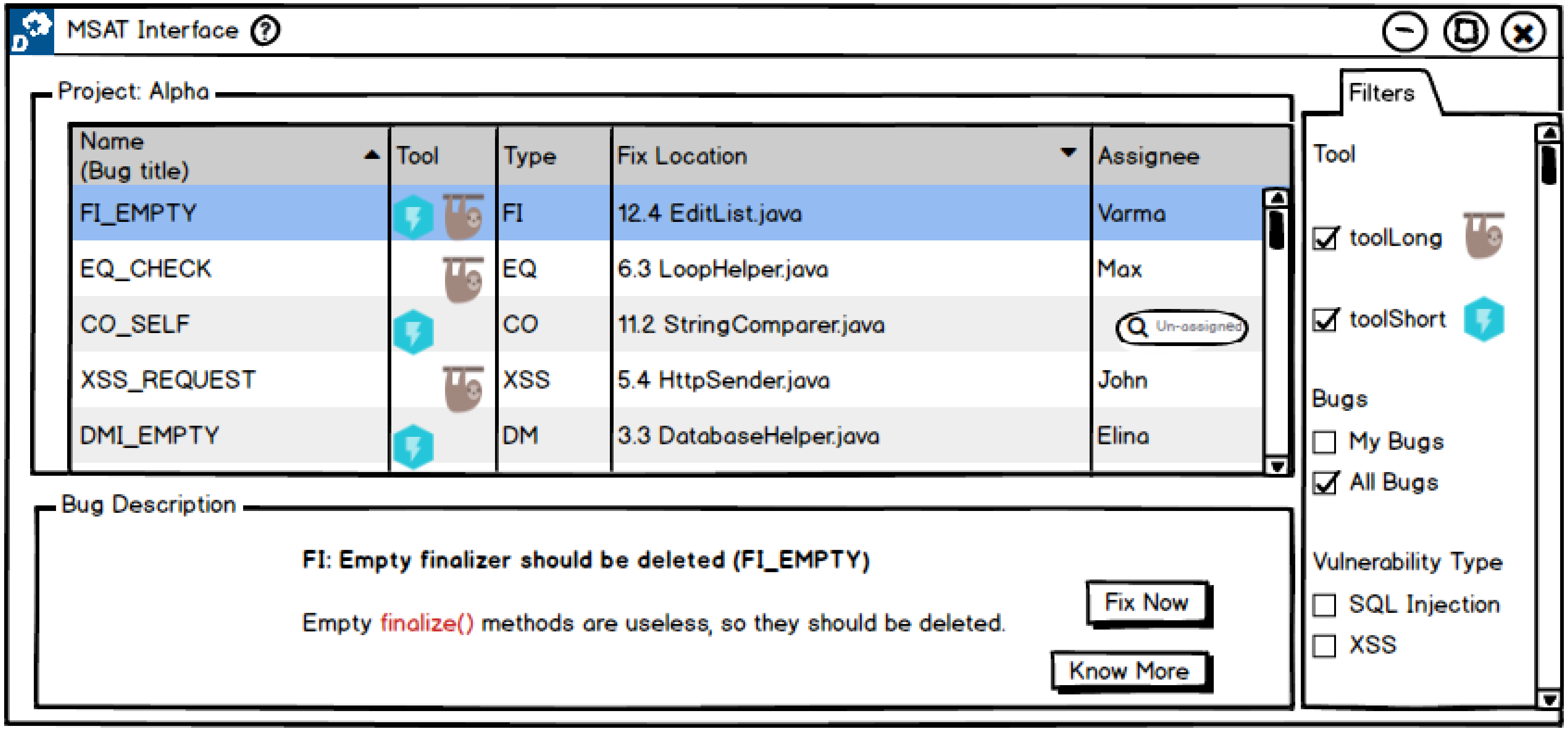
- Users: 5
- Sub research questions: 9
- Each session: ~ 90 minutes



UX 1

	Analysis View
RQ 1 (display)	• Tags • Single List • Separate List • Statistics Screen
RQ 2 (feedback)	• Animated Icons • Progress Bar • Popup
RQ 3 (trace)	• Numbers

[RQ 1.1] Does a separate list or single list help the user to identify the common bug?



Name (Bug title)	Tool	Type	Fix Location	Assignee
FI_EMPTY	toolShort, toolLong	FI	12.4 EditList.java	Varma
EQ_CHECK	toolLong	EQ	6.3 LoopHelper.java	Max
CO_SELF	toolShort	CO	11.2 StringComparer.java	Un-assigned
XSS_REQUEST	toolLong	XSS	5.4 HttpSender.java	John
DMI_EMPTY	toolShort	DM	3.3 DatabaseHelper.java	Elina

Bug Description

FI: Empty finalizer should be deleted (FI_EMPTY)

Empty finalize() methods are useless, so they should be deleted.

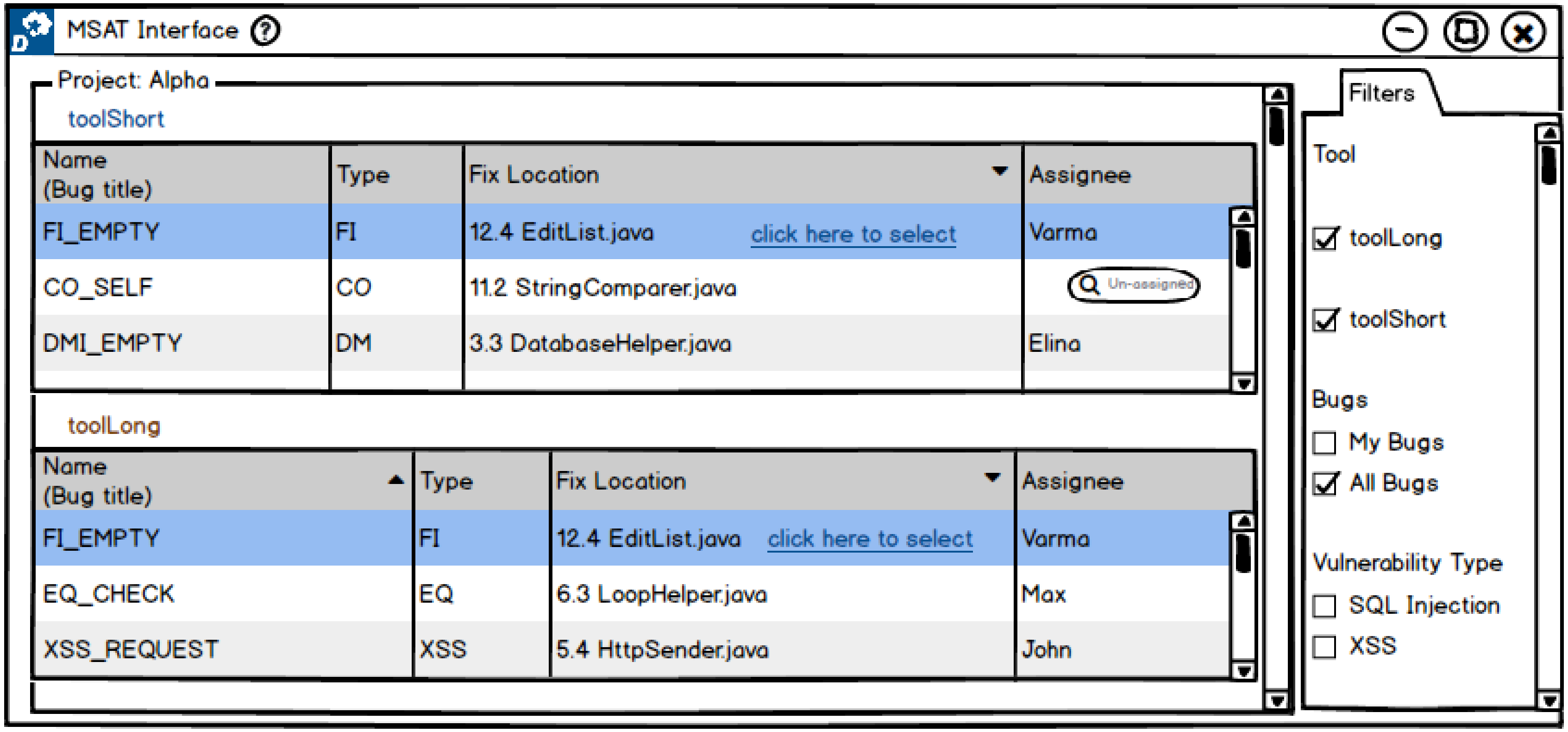
Tool
- [✓] toolLong
- [✓] toolShort

Bugs
- [ ] My Bugs
- [✓] All Bugs

Vulnerability Type
- [ ] SQL Injection
- [ ] XSS

- Single List

[RQ 1.1] Does a separate list or single list help the user to identify the common bug?



toolShort

Name (Bug title)	Type	Fix Location		Assignee
FI_EMPTY	FI	12.4 EditList.java	click here to select	Varma
CO_SELF	CO	11.2 StringComparer.java		Un-assigned
DMI_EMPTY	DM	3.3 DatabaseHelper.java		Elina

toolLong

Name (Bug title)	Type	Fix Location		Assignee
FI_EMPTY	FI	12.4 EditList.java	click here to select	Varma
EQ_CHECK	EQ	6.3 LoopHelper.java		Max
XSS_REQUEST	XSS	5.4 HttpSender.java		John

Tool
- ✓ toolLong
- ✓ toolShort

Bugs
- ☐ My Bugs
- ✓ All Bugs

Vulnerability Type
- ☐ SQL Injection
- ☐ XSS

- Seperate List

[RQ 1.1] Does a separate list or single list help the user to identify the common bug?

- Results:

	Single List	Separate List
Task Success	100 %	100 %
Usability	8	7.6
Votes	2	3
		✓

  - Separate List – “more effective when using more tools”

UX 1 - Findings

	Analysis View
RQ 1 (display)	• Tags • Single List • Separate List ✓ • Statistics Screen ✓
RQ 2 (feedback)	• Animated Icons ✓ • Progress Bar ✓ • Popup ✓
RQ 3 (trace)	• Numbers ✓

UX 1 – Lessons

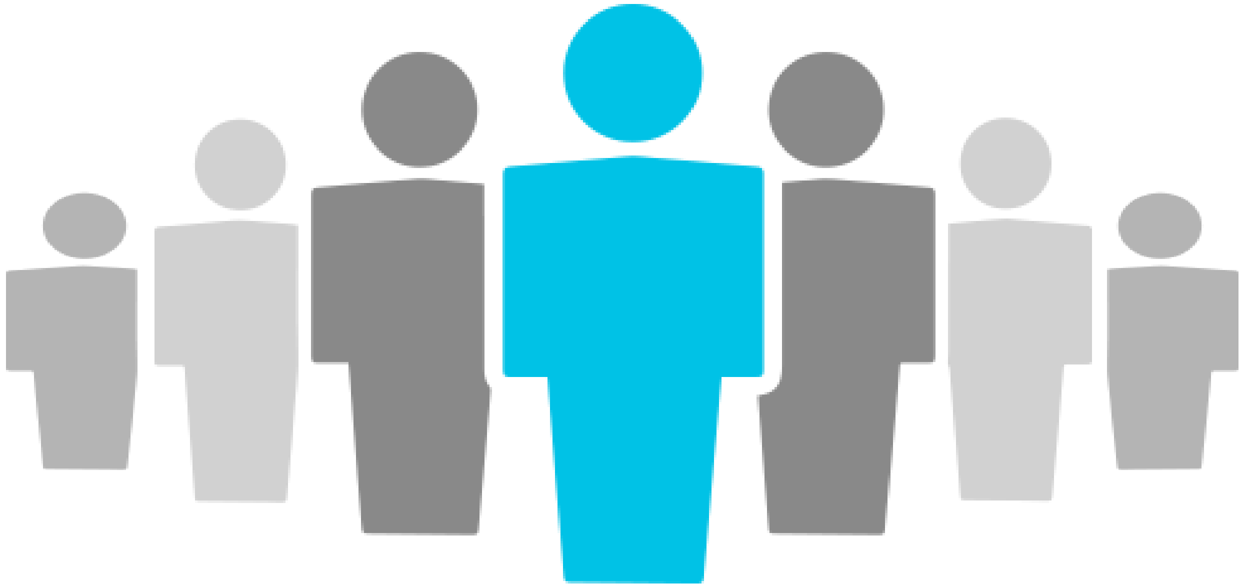
- Analysis View ✓

- Improvisations for next UX cycle:
  - Increase code base
  - Volume of bugs (+ scroll)
  - Integrate more tools
  - Code view perspective
  - + new sub RQ’s

UX Design Cycle 2

UX 2

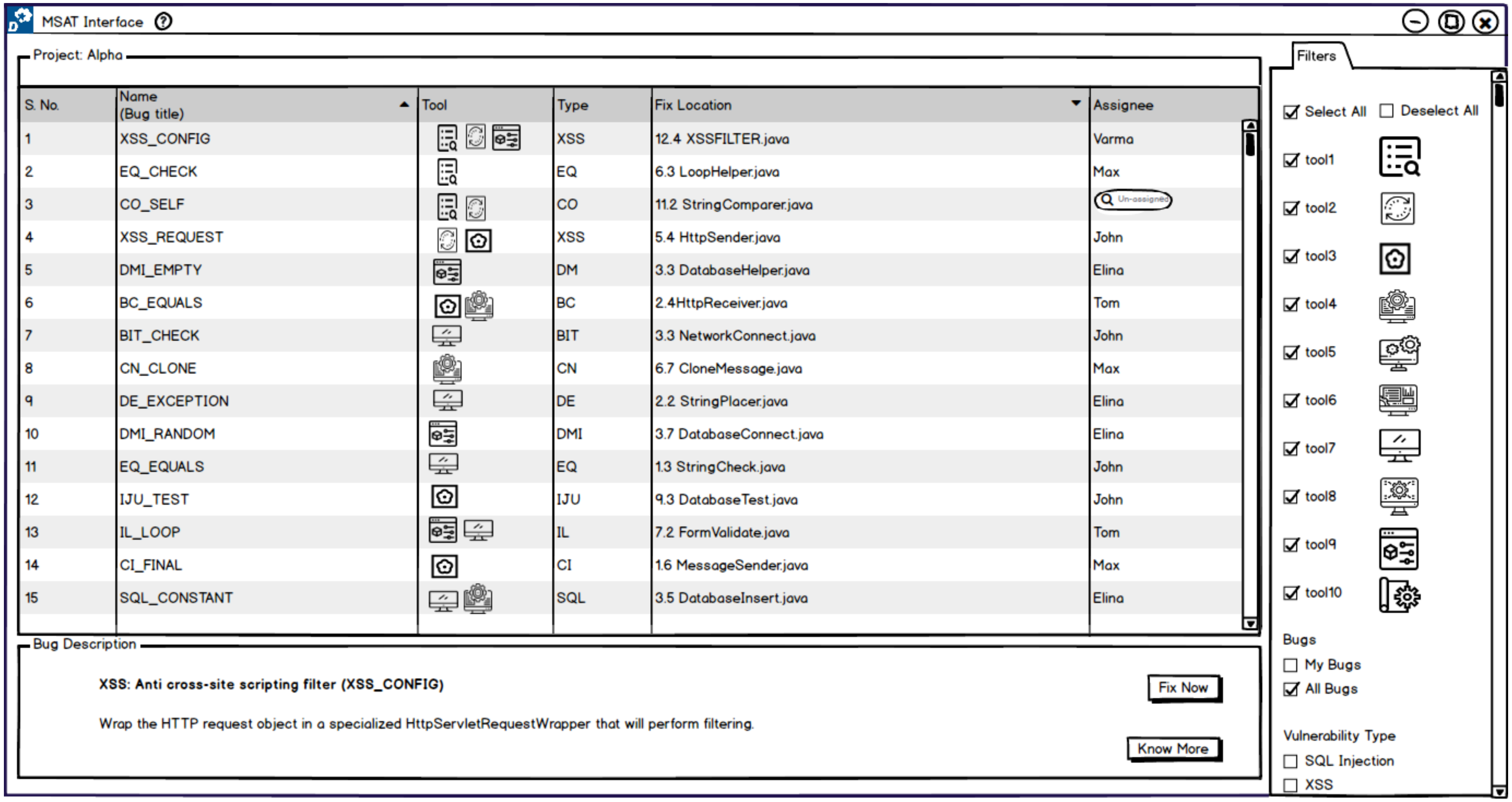
- Users: 7
- Sub research questions: 9
- Each session: ~ 105 minutes



UX 2

	Analysis View	**Code View**
RQ 1 (display)	• Tags • Single List • Separate List	• Multiple Icons • Single Icon
RQ 2 (feedback)	• Animated Icons • Progress Bar • Popup	• Toasts (alerts) • Spinner (status)
RQ 3 (trace)	• Numbers • Adjectives	• Before/After

[RQ 1.1] Does a separate list or single list help the user to identify the common bug?



Project: Alpha

S. No.	Name (Bug title)	Tool	Type	Fix Location	Assignee
1	XSS_CONFIG		XSS	12.4 XSSFILTER.java	Varma
2	EQ_CHECK		EQ	6.3 LoopHelper.java	Max
3	CO_SELF		CO	11.2 StringComparer.java	Un-assigned
4	XSS_REQUEST		XSS	5.4 HttpSender.java	John
5	DMI_EMPTY		DM	3.3 DatabaseHelper.java	Elina
6	BC_EQUALS		BC	2.4HttpReceiver.java	Tom
7	BIT_CHECK		BIT	3.3 NetworkConnect.java	John
8	CN_CLONE		CN	6.7 CloneMessage.java	Max
9	DE_EXCEPTION		DE	2.2 StringPlacer.java	Elina
10	DMI_RANDOM		DMI	3.7 DatabaseConnect.java	Elina
11	EQ_EQUALS		EQ	1.3 StringCheck.java	John
12	IJU_TEST		IJU	9.3 DatabaseTest.java	John
13	IL_LOOP		IL	7.2 FormValidate.java	Tom
14	CI_FINAL		CI	1.6 MessageSender.java	Max
15	SQL_CONSTANT		SQL	3.5 DatabaseInsert.java	Elina

Bug Description

XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized HttpServletRequestWrapper that will perform filtering.

☑ Select All ☐ Deselect All

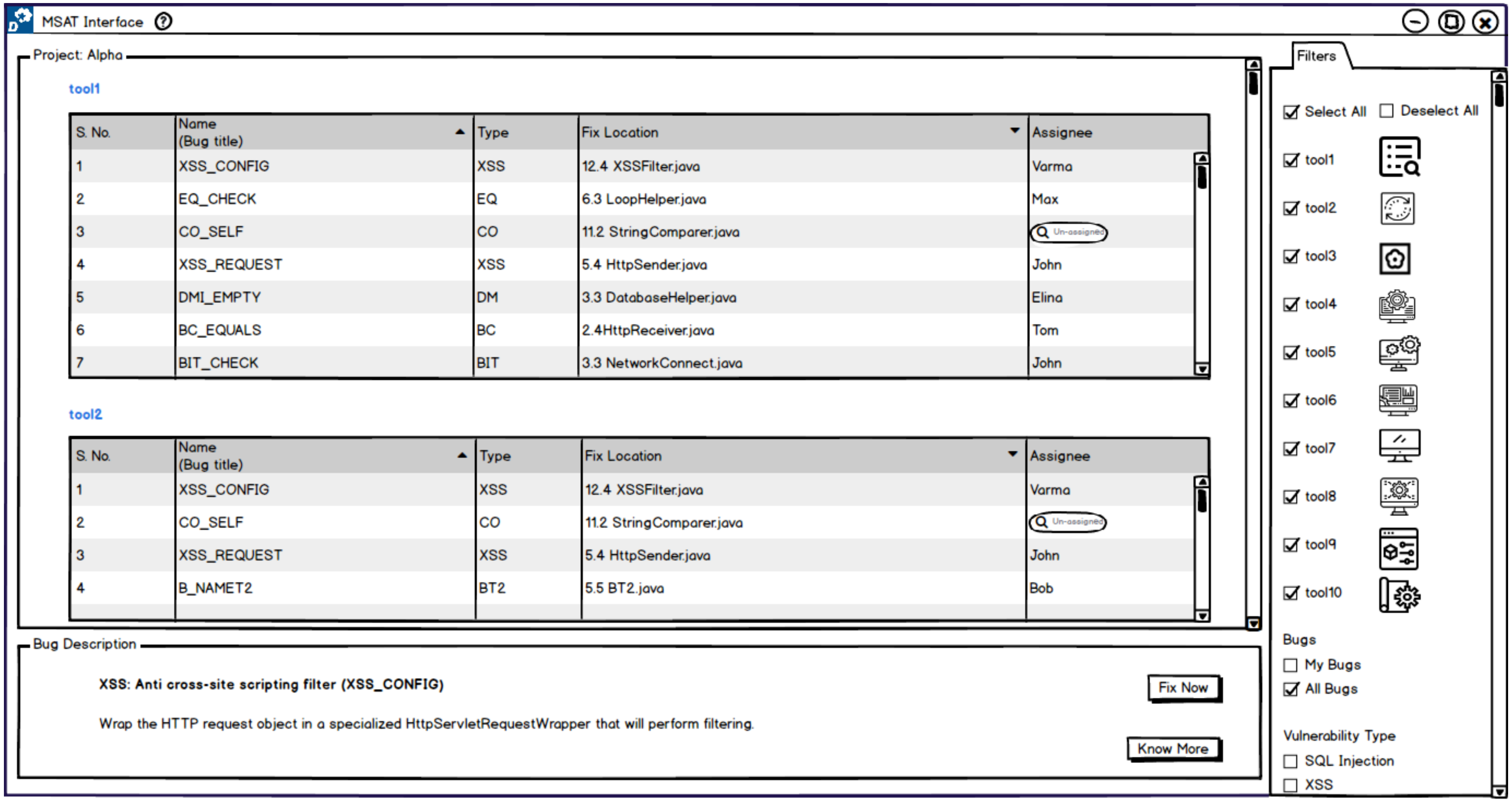
☑ tool1
☑ tool2
☑ tool3
☑ tool4
☑ tool5
☑ tool6
☑ tool7
☑ tool8
☑ tool9
☑ tool10

Bugs
☐ My Bugs
☑ All Bugs

Vulnerability Type
☐ SQL Injection
☐ XSS

- Single List

[RQ 1.1] Does a separate list or single list help the user to identify the common bug?



Project: Alpha

tool1

S. No.	Name (Bug title)	Type	Fix Location	Assignee
1	XSS_CONFIG	XSS	12.4 XSSFilter.java	Varma
2	EQ_CHECK	EQ	6.3 LoopHelper.java	Max
3	CO_SELF	CO	11.2 StringComparer.java	Un-assigned
4	XSS_REQUEST	XSS	5.4 HttpSender.java	John
5	DMI_EMPTY	DM	3.3 DatabaseHelper.java	Elina
6	BC_EQUALS	BC	2.4HttpReceiver.java	Tom
7	BIT_CHECK	BIT	3.3 NetworkConnect.java	John

tool2

S. No.	Name (Bug title)	Type	Fix Location	Assignee
1	XSS_CONFIG	XSS	12.4 XSSFilter.java	Varma
2	CO_SELF	CO	11.2 StringComparer.java	Un-assigned
3	XSS_REQUEST	XSS	5.4 HttpSender.java	John
4	B_NAMET2	BT2	5.5 BT2.java	Bob

Bug Description

XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized HttpServletRequestWrapper that will perform filtering.

[✓] Select All ☐ Deselect All

[✓] tool1
[✓] tool2
[✓] tool3
[✓] tool4
[✓] tool5
[✓] tool6
[✓] tool7
[✓] tool8
[✓] tool9
[✓] tool10

Bugs
☐ My Bugs
[✓] All Bugs

Vulnerability Type
☐ SQL Injection
☐ XSS

- Separate List

[RQ 1.1] Does a separate list or single list help the user to identify the common bug?

- Results:

	Single List	Separate List
Task Success	71.43 %	42.85 %
Usability	8.14	5.43
Votes	5	0

✓

  - Single List – “effortless to perceive, more user friendly”

UX 2 - Findings

	Analysis View	**Code View**
RQ 1 (display)	• Tags • Single List ✓ • Separate List	• Multiple Icons • Single Icon ✓
RQ 2 (feedback)	• Animated Icons ✓ • Progress Bar ✓ • Popup ✓	• Toasts (alerts) ✓ • Spinner (status)
RQ 3 (trace)	• Numbers • Adjectives ✓	• Before/After ✓

UX 2 – Lessons

- Analysis View
- Code View
- UX 1 Scalability

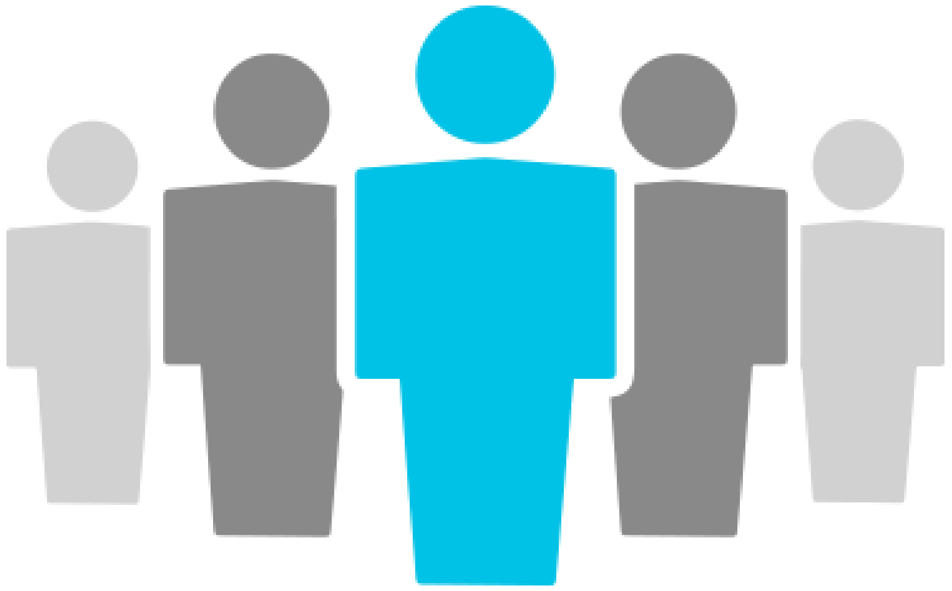
✓

- Improvisations for next UX cycle:
  - UX 2 Scalability
  - + new sub RQ’s

UX Design Cycle 3

UX 3

- Users: 5
- Sub research questions: 13
- Each session: ~ 120 minutes



UX 3

	Analysis View	Code View
RQ 1 (display)		• Next • List view • Bug icons • Table view • Vertical view • Horizontal view • Similar boxes • Similar list
RQ 2 (feedback)	• Animated Icons • Progress Bar • Popup	• Toasts (alerts) • Spinner (status)
RQ 3 (trace)		• Before/After • Table view

[RQ 2.1]

How usable are each feedback functionality compared to the scenario of using unified UI to native UIs?

- Evaluation Setup: 3 native UI tools for a JavaScript project.

  - CLI – ESLint
  - IDE – SonarLint
  - WEB - SonarQube

❖ ESLint – The pluggable linting utility for JavaScript and JSX. url: https://eslint.org/

❖ Sonarlint - Fix issues before they exist. url: https://www.sonarlint.org/

❖ Sonarqube - Code Quality and Security . url: https://www.sonarqube.org/

[RQ 2.1]

How usable are each feedback functionality compared to the scenario of using unified UI to native UIs?

Result	MSAT-UI	Native UI
Animated Icons	8.4	0
Progress Bar	7.6	0.8
Pending Status Popup	7.8	0
Alerts	9.2	0
Status	8.8	4

✓

Almost all users agreed the ideas being novel and hardly present with native UIs.

UX 3 - Findings

	Analysis View	**Code View**
RQ 1 (display)		• Next • List view ✓ • Bug icons • Table view ✓ • Vertical view • Horizontal view ✓ • Similar boxes • Similar list ✓
RQ 2 (feedback)	• Animated Icons • Progress Bar ✓ • Popup ✓	• Toasts (alerts) ✓ • Spinner (status)
RQ 3 (trace)		• Before/After • Table view ✓

Limitations

- Number of participants: > 5
    - UX 1 – 5
    - UX 2 – 7
    - UX 3 – 5

- Priming, Recency bias: Latin Square partition

- Closed Study, Design Tool

❖ Why You Only Need to Test with 5 Users. url: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users

❖ Balsamiq – Rapid, effective and fun wireframing software. url: https://balsamiq.com/

Findings

	Analysis View	Code View
RQ 1 (display)	• Single list	• List view • Single icon • Table view • Similar list
RQ 2 (feedback)	• Animated Icons • Progress Bar • Popup	• Toasts (alerts)
RQ 3 (trace)	• Adjectives	• Table view

Future Work

- Q. Would having tabs help scale the tools visibility with bugs results?
- Q. Do graphs help in understanding the bugs reported?

- RQ 4: How is teamwork facilitated in bug fixing in context of multiple tools?

… many more!

Summary

- Importance of Static Analysis tools
- Usage of Multiple Static Analysis tools

- Need for a single user interface for multiple tools

- This thesis work followed UX Design Cycle to achieve usable prototypes focussing on primary research questions such as,
  - How to display results of the same codebase from different analysis tools?
  - What feedback works to know that bug fixing is on-going?
  - How to carry traceability of bug fixing?

Backup Slides

Static Code Analysis

- Johnson et al.
- Christakis et al.

Usability Issues

- Habib et al.

- Brittany Johnson, Yoonki Song, Emerson Murphy-Hill, and Robert Bowdidge. 2013. Why don't software developers use static analysis tools to find bugs?. In *Proceedings of the 2013 International Conference on Software Engineering* (ICSE '13). IEEE Press, Piscataway, NJ, USA, 672-681.
- Maria Christakis and Christian Bird. 2016. What developers want and need from program analysis: an empirical study. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering* (ASE 2016). ACM, New York, NY, USA, 332-343. DOI: https://doi.org/10.1145/2970276.2970347
- Habib, A., & Pradel, M. (2018, September). How many of all bugs do we find? a study of static bug detectors. In *ASE* (pp. 317-328).

Multiple Tools

- Developers use multiple static analysis tools each having own coverage.

- Research trends:

  - Prioritise the bug warning alerts

    (Flynn et al.)

  - Merges 3 tools for Java to show warnings

    (Meng et al.)

❖ Lori Flynn, William Snavely, David Svoboda, Nathan VanHoudnos, Richard Qin, Jennifer Burns, David Zubrow, Robert Stoddard, and Guillermo Marce-Santurio. 2018. Prioritizing alerts from multiple static analysis tools, using classification models. In *Proceedings of the 1st International Workshop on Software Qualities and Their Dependencies* (SQUADE '18). ACM, New York, NY, USA, 13-20. DOI: https://doi.org/10.1145/3194095.3194100

❖ N. Meng, Q. Wang, Q. Wu and H. Mei, "An Approach to Merge Results of Multiple Static Analysis Tools (Short Paper)," *2008 The Eighth International Conference on Quality Software*, Oxford, 2008, pp. 169-174.doi: 10.1109/QSIC.2008.30

Multiple Tools

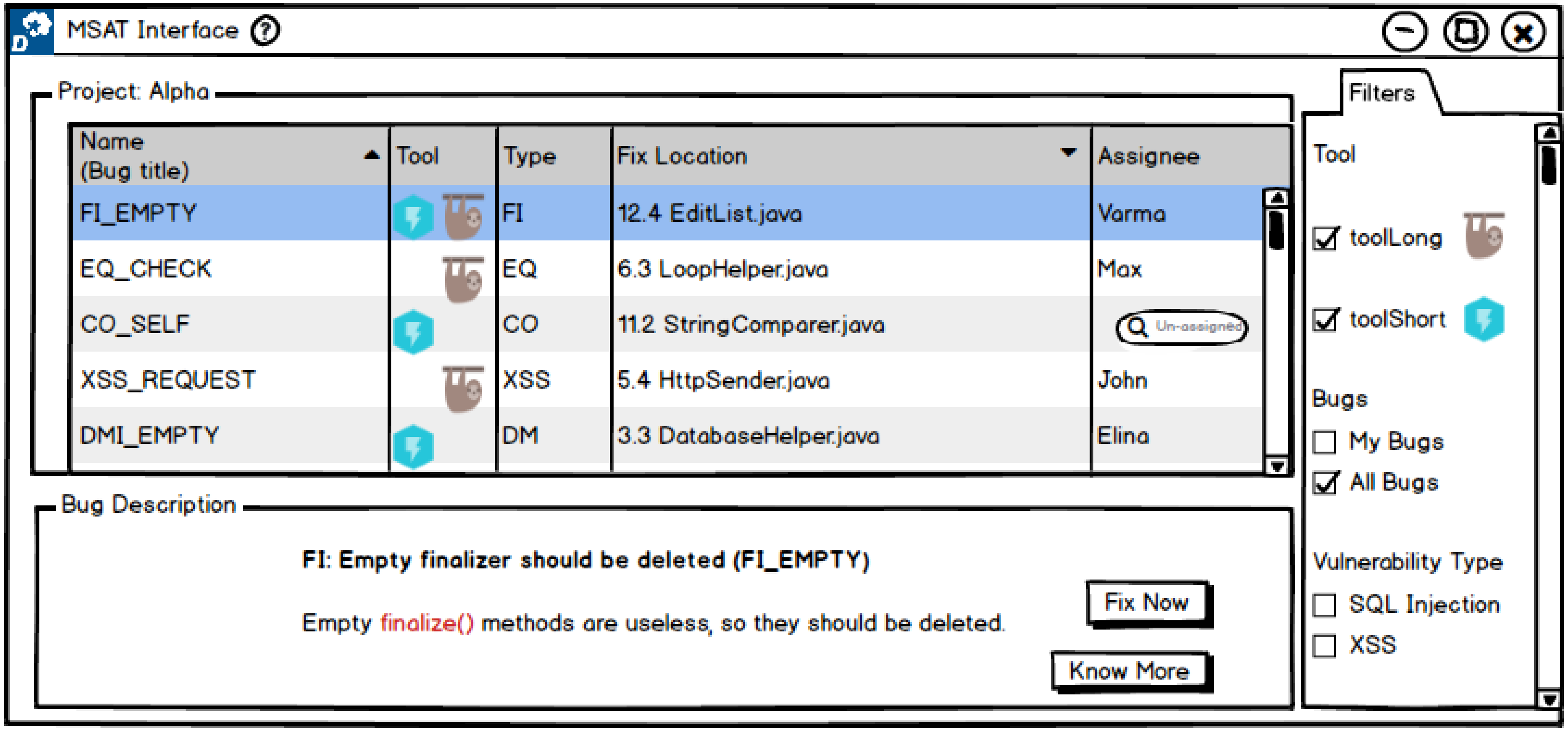
- Tricorder (Sadowski et al.)
- Shipshape
- Tricium
- Parfait (Cifuentes et al.)

But USABILITY is not addressed…

- Caitlin Sadowski, Jeffrey van Gogh, Ciera Jaspan, Emma Söderberg, and Collin Winter. 2015. Tricorder: building a program analysis ecosystem. In *Proceedings of the 37th International Conference on Software Engineering - Volume 1* (ICSE '15), Vol. 1. IEEE Press, Piscataway, NJ, USA, 598-608.
- Cristina Cifuentes and Bernhard Scholz. 2008. Parfait: designing a scalable bug checker. In *Proceedings of the 2008 workshop on Static analysis* (SAW '08). ACM, New York, NY, USA, 4-11. DOI=http://dx.doi.org/10.1145/1394504.1394505

UX 1

[RQ 1] Does a separate list or single list help the user to identify the common bug?



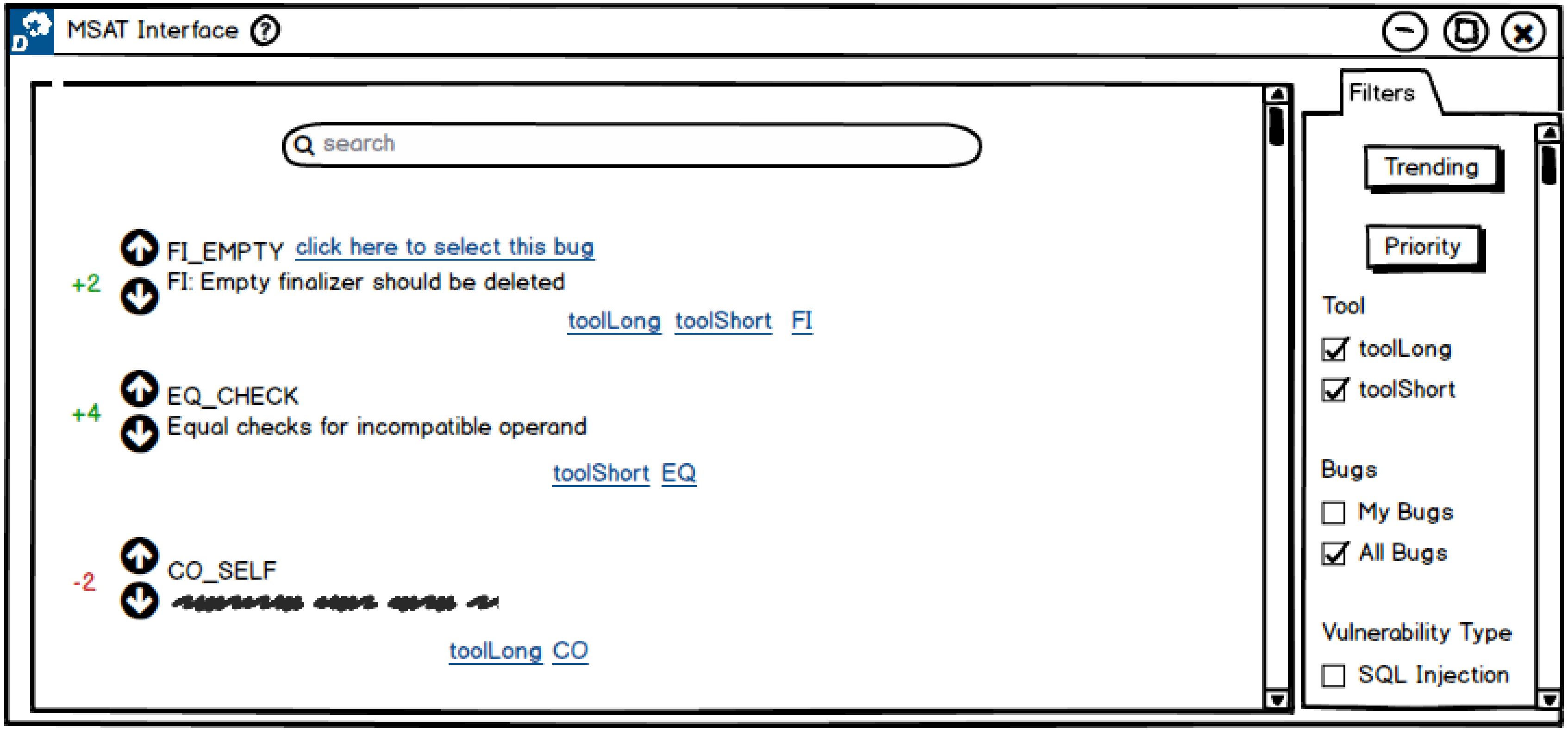
Name (Bug title)	Tool	Type	Fix Location	Assignee
FI_EMPTY	toolShort, toolLong	FI	12.4 EditList.java	Varma
EQ_CHECK	toolLong	EQ	6.3 LoopHelper.java	Max
CO_SELF	toolShort	CO	11.2 StringComparer.java	Un-assigned
XSS_REQUEST	toolLong	XSS	5.4 HttpSender.java	John
DMI_EMPTY	toolShort	DM	3.3 DatabaseHelper.java	Elina

FI: Empty finalizer should be deleted (FI_EMPTY)

Empty finalize() methods are useless, so they should be deleted.

Separate list outwins the single list with a slight majority of 3 out 5.

[RQ 1] Will having tags help in scalability of bugs?


MSAT Interface

search

+2 FI_EMPTY click here to select this bug
FI: Empty finalizer should be deleted
toolLong toolShort FI

+4 EQ_CHECK
Equal checks for incompatible operand
toolShort EQ

-2 CO_SELF
[redacted]
toolLong CO

Filters
Trending
Priority

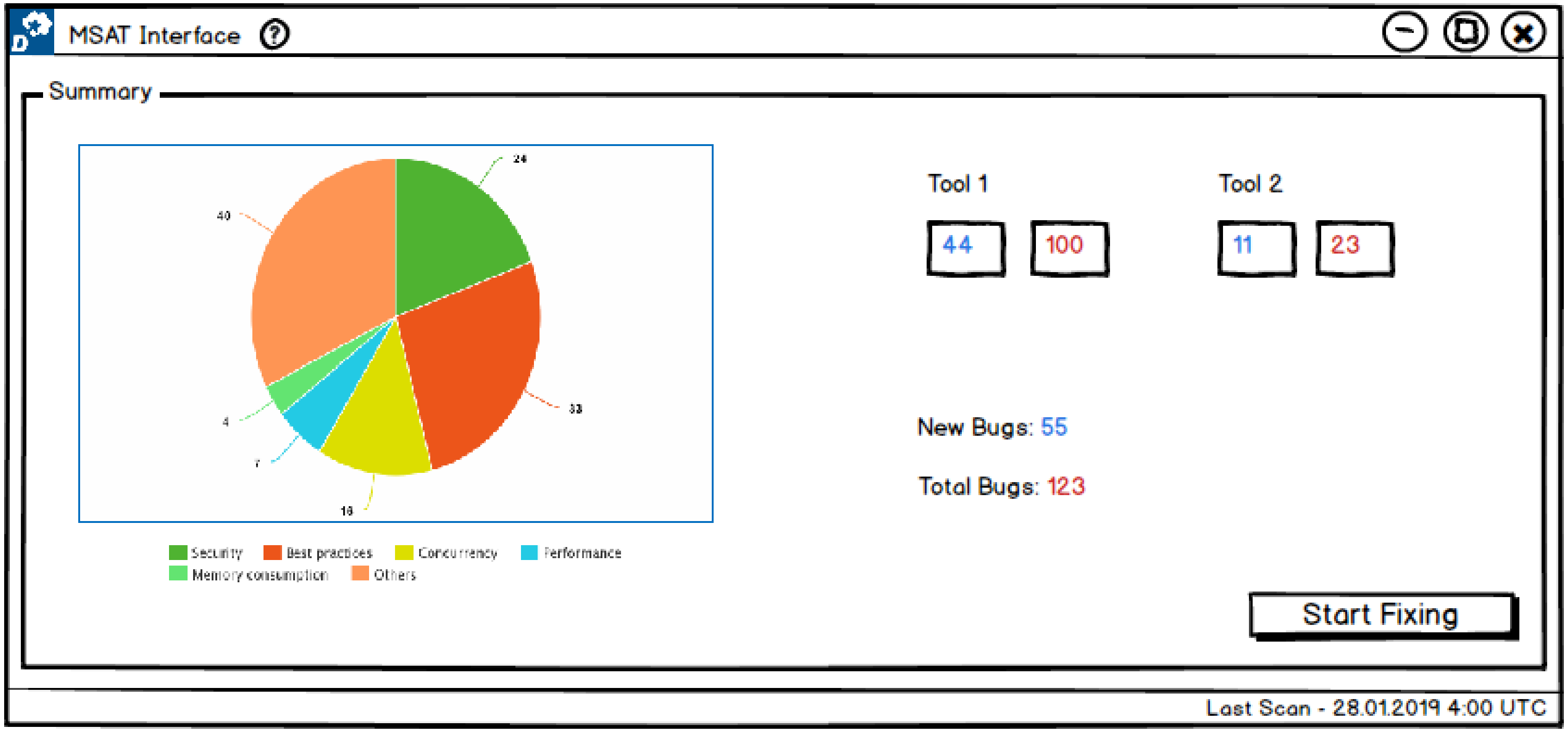
Tool
☑ toolLong
☑ toolShort

Bugs
☐ My Bugs
☑ All Bugs

Vulnerability Type
☐ SQL Injection

Yes! However, the interface is confusing. Users preferred single list in table format
solution idea as it suffices this scalability concern.

[RQ 1] Does the given statistics screen help the user in understating the analysis results overview?



Tool 1

44 100

Tool 2

11 23

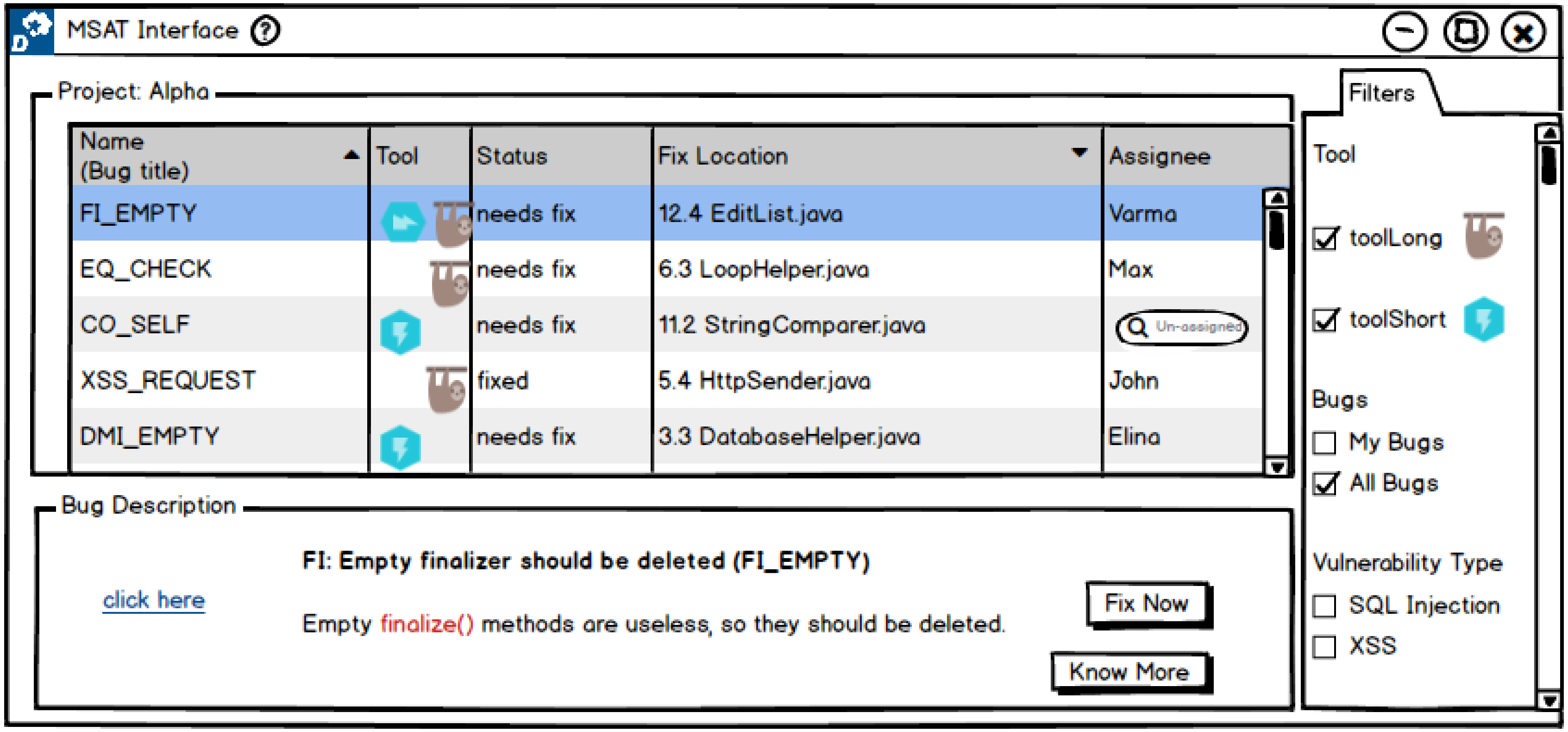
New Bugs: 55

Total Bugs: 123

Last Scan - 28.01.2019 4:00 UTC

Yes! It is conducive, especially when codebase is vast, and we use more analysis tools.

[RQ 2] Will the animation (rotation) of icons for tools suffice the feedback required by the user?



Name (Bug title)	Tool	Status	Fix Location	Assignee
FI_EMPTY		needs fix	12.4 EditList.java	Varma
EQ_CHECK		needs fix	6.3 LoopHelper.java	Max
CO_SELF		needs fix	11.2 StringComparer.java	Un-assigned
XSS_REQUEST		fixed	5.4 HttpSender.java	John
DMI_EMPTY		needs fix	3.3 DatabaseHelper.java	Elina

Bug Description

click here

FI: Empty finalizer should be deleted (FI_EMPTY)

Empty finalize() methods are useless, so they should be deleted.

Fix Now

Know More

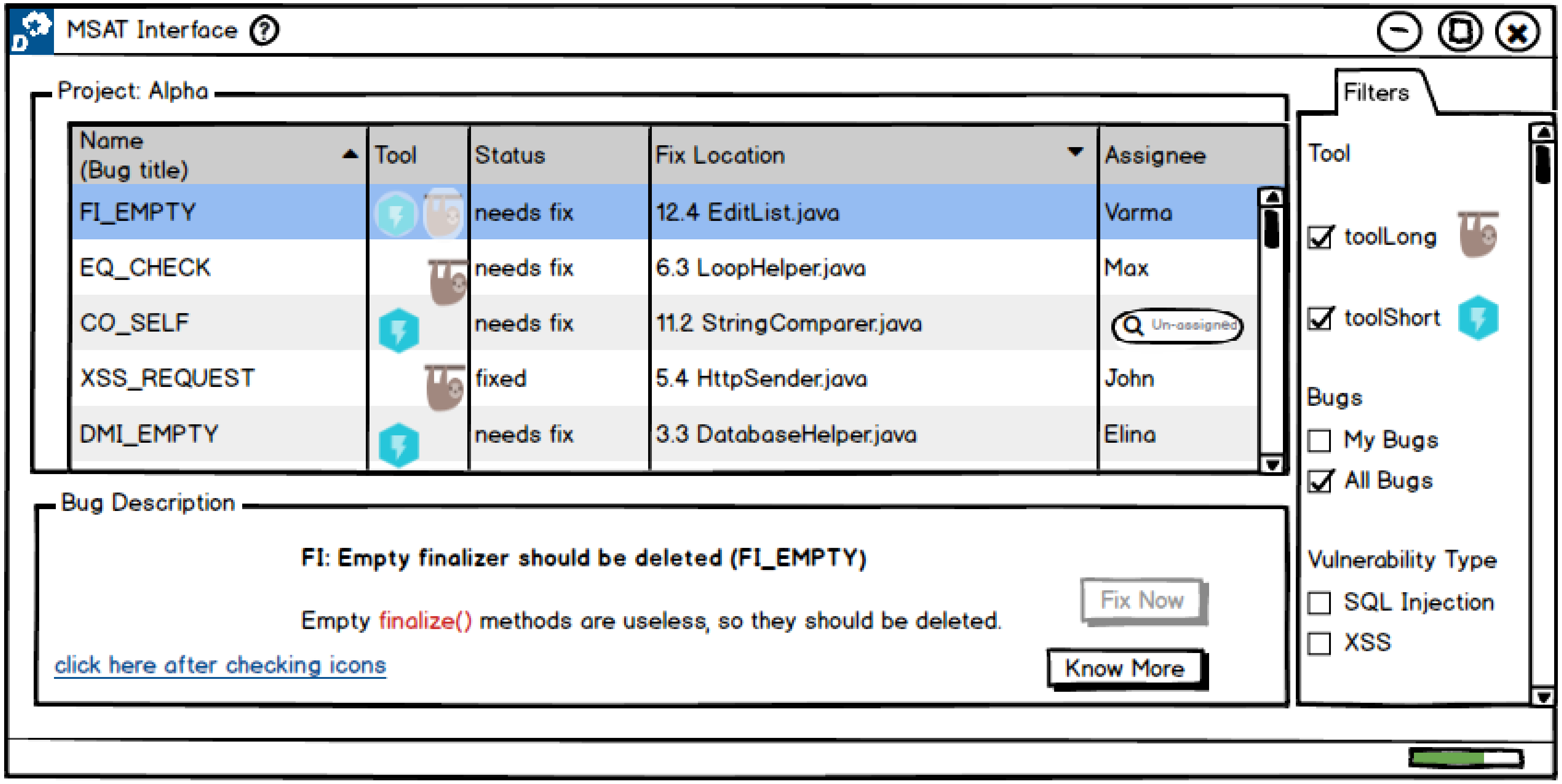
Tool
- [x] toolLong
- [x] toolShort

Bugs
- [ ] My Bugs
- [x] All Bugs

Vulnerability Type
- [ ] SQL Injection
- [ ] XSS

No! Users interested to see how far the analysis done and also to have more detailed information on it.

[RQ 2] Will stating the progress of analysis for each tool be better than animation provided as feedback to the user?



MSAT Interface

Project: Alpha

Name (Bug title)	Tool	Status	Fix Location	Assignee
FI_EMPTY		needs fix	12.4 EditList.java	Varma
EQ_CHECK		needs fix	6.3 LoopHelper.java	Max
CO_SELF		needs fix	11.2 StringComparer.java	Un-assigned
XSS_REQUEST		fixed	5.4 HttpSender.java	John
DMI_EMPTY		needs fix	3.3 DatabaseHelper.java	Elina

Bug Description

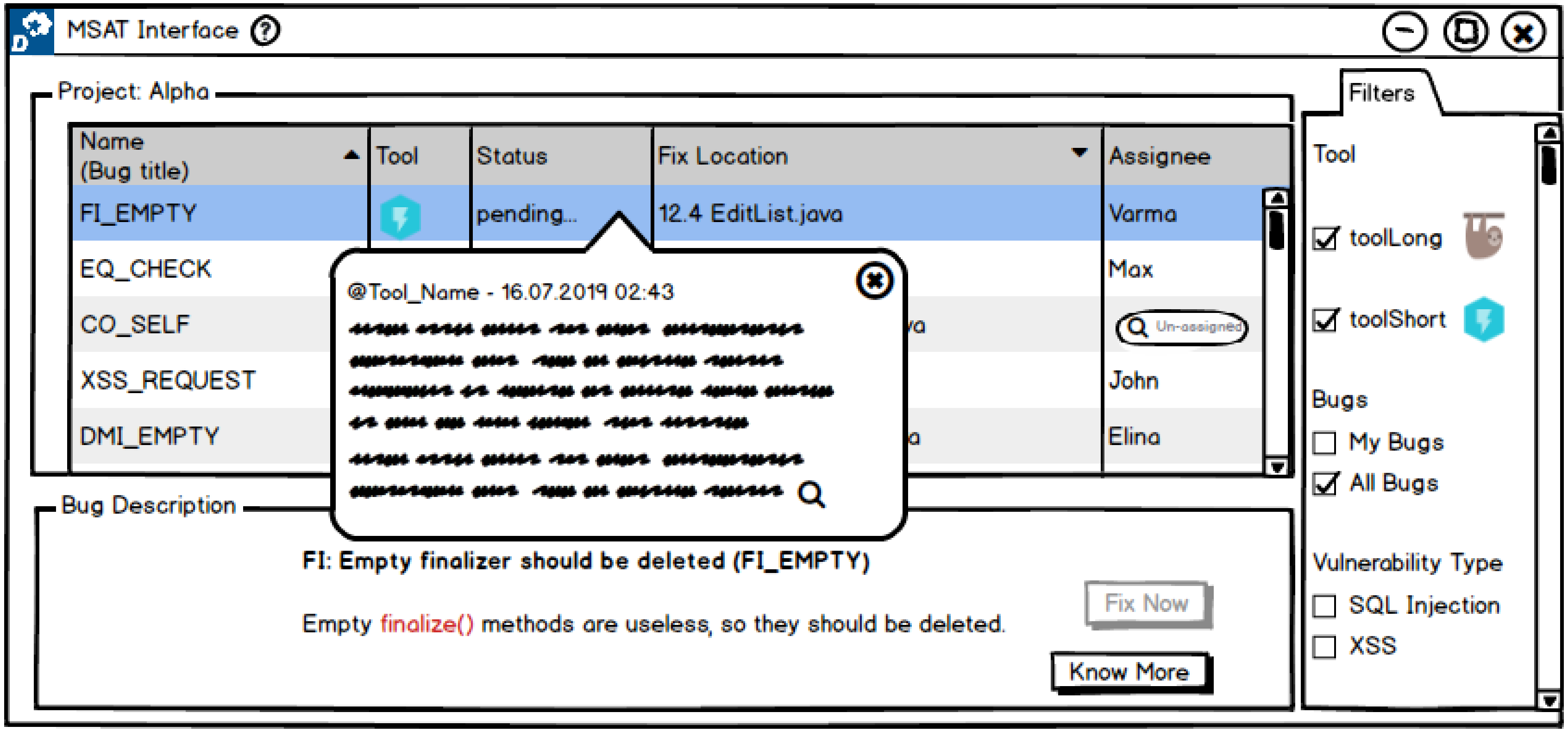
FI: Empty finalizer should be deleted (FI_EMPTY)

Empty finalize() methods are useless, so they should be deleted.

click here after checking icons

Yes! in terms of knowing the progress in completion of analysis.

[RQ 2] Does having more textual information with a popup feedback is required by the user?



Name (Bug title)	Tool	Status	Fix Location	Assignee
FI_EMPTY		pending...	12.4 EditList.java	Varma
EQ_CHECK				Max
CO_SELF			va	Un-assigned
XSS_REQUEST				John
DMI_EMPTY			a	Elina

@Tool_Name - 16.07.2019 02:43

FI: Empty finalizer should be deleted (FI_EMPTY)

Empty finalize() methods are useless, so they should be deleted.

Yes! More information as possible is more beneficial from a user perspective being a developer or tester.

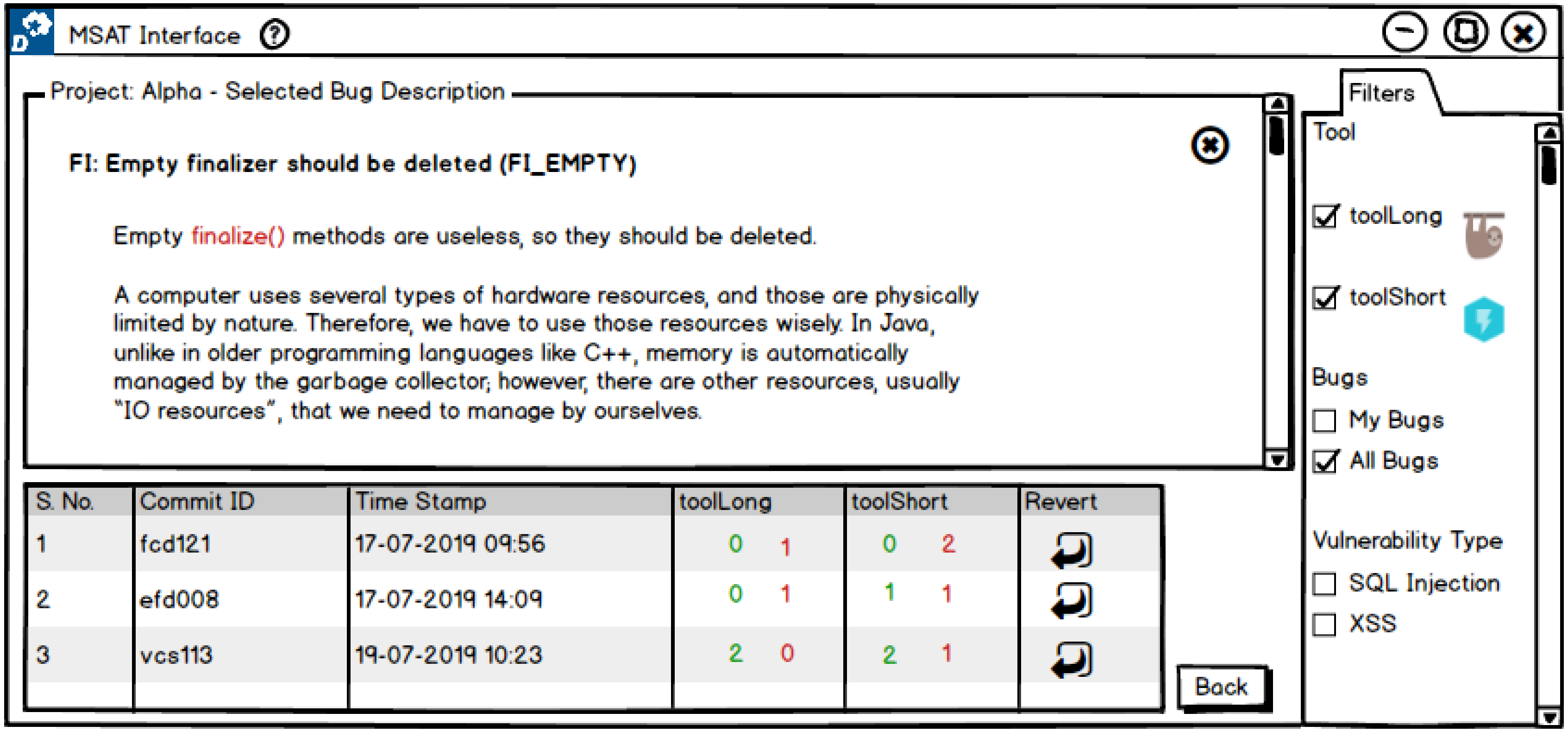
[RQ 2] Do users require multiple feedbacks, i.e., any combination of animated icons, progress bar or pending status popup?

- All the 3 feedback features

- Animated Icons – What bugs are being analysed?
- Progress Bar – How far the bugs got analysed?
- Pending Status – more information on analysis

Yes! Especially progress bar with pending status popup.

[RQ 3] Whether the given UI, i.e., previous commits in the process of fixing a bug-finding with numbers determining the adding or removing of other bugs be able to address the scenario from the user perspective?



Project: Alpha - Selected Bug Description

FI: Empty finalizer should be deleted (FI_EMPTY)

Empty finalize() methods are useless, so they should be deleted.

A computer uses several types of hardware resources, and those are physically limited by nature. Therefore, we have to use those resources wisely. In Java, unlike in older programming languages like C++, memory is automatically managed by the garbage collector; however, there are other resources, usually "IO resources", that we need to manage by ourselves.

S. No.	Commit ID	Time Stamp	toolLong	toolShort	Revert
1	fcd121	17-07-2019 09:56	0 1	0 2	↩
2	efd008	17-07-2019 14:09	0 1	1 1	↩
3	vcs113	19-07-2019 10:23	2 0	2 1	↩

Tool
- [x] toolLong
- [x] toolShort

Bugs
- [ ] My Bugs
- [x] All Bugs

Vulnerability Type
- [ ] SQL Injection
- [ ] XSS

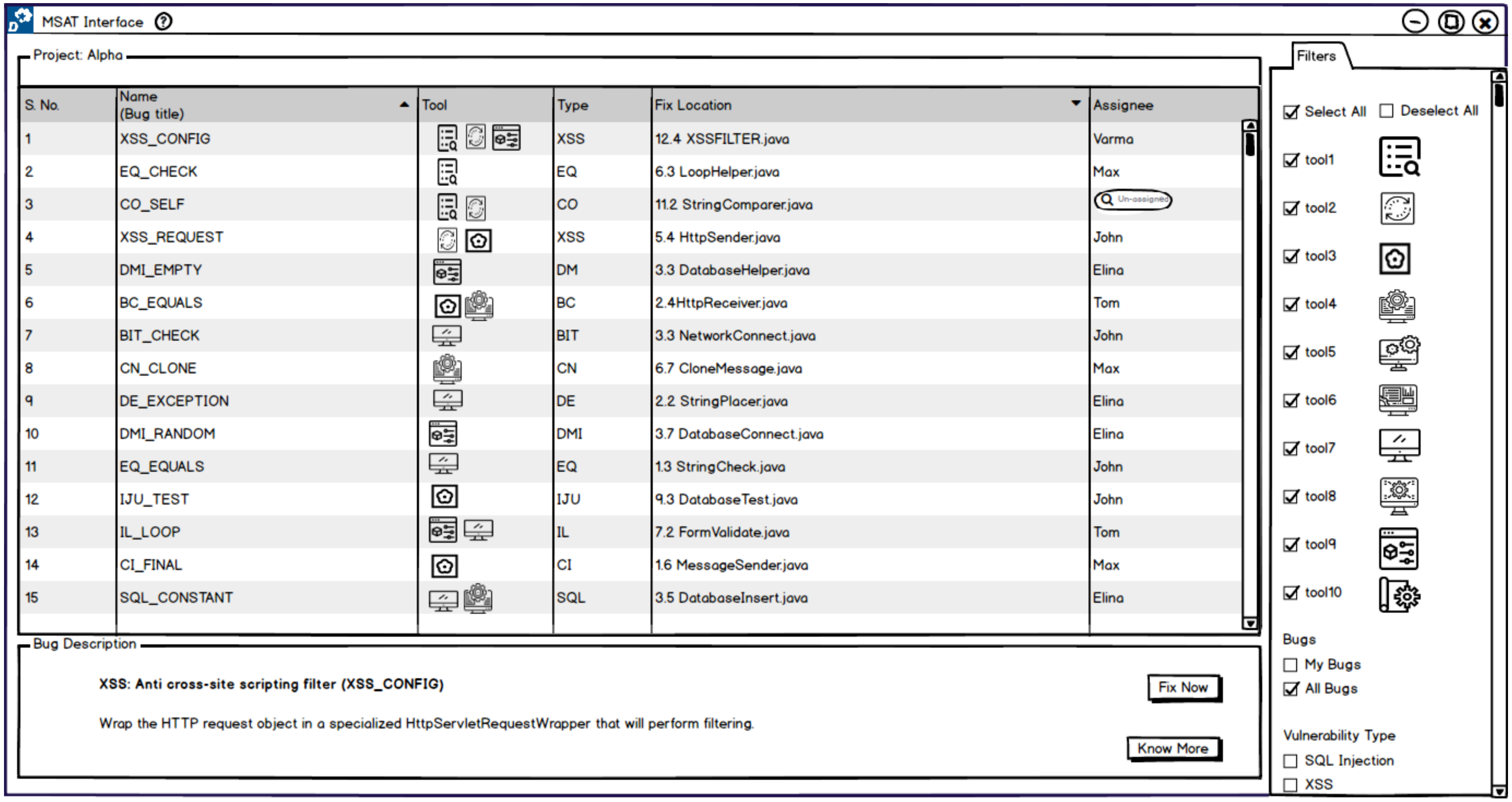
Yes! The proposed design is helpful for the given scenario.

[Post]
Does onboard phase is required to understand the UI better?

- No! enough text on screen would suffice.

UX 2

[RQ 1] From analysis view perspective, does a separate list or single list help the user to identify the common bug?



Project: Alpha

S. No.	Name (Bug title)	Tool	Type	Fix Location	Assignee
1	XSS_CONFIG		XSS	12.4 XSSFILTER.java	Varma
2	EQ_CHECK		EQ	6.3 LoopHelper.java	Max
3	CO_SELF		CO	11.2 StringComparer.java	Un-assigned
4	XSS_REQUEST		XSS	5.4 HttpSender.java	John
5	DMI_EMPTY		DM	3.3 DatabaseHelper.java	Elina
6	BC_EQUALS		BC	2.4HttpReceiver.java	Tom
7	BIT_CHECK		BIT	3.3 NetworkConnect.java	John
8	CN_CLONE		CN	6.7 CloneMessage.java	Max
9	DE_EXCEPTION		DE	2.2 StringPlacer.java	Elina
10	DMI_RANDOM		DMI	3.7 DatabaseConnect.java	Elina
11	EQ_EQUALS		EQ	1.3 StringCheck.java	John
12	IJU_TEST		IJU	9.3 DatabaseTest.java	John
13	IL_LOOP		IL	7.2 FormValidate.java	Tom
14	CI_FINAL		CI	1.6 MessageSender.java	Max
15	SQL_CONSTANT		SQL	3.5 DatabaseInsert.java	Elina

Bug Description

XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized HttpServletRequestWrapper that will perform filtering.

Fix Now

Know More

☑ Select All ☐ Deselect All

☑ tool1
☑ tool2
☑ tool3
☑ tool4
☑ tool5
☑ tool6
☑ tool7
☑ tool8
☑ tool9
☑ tool10

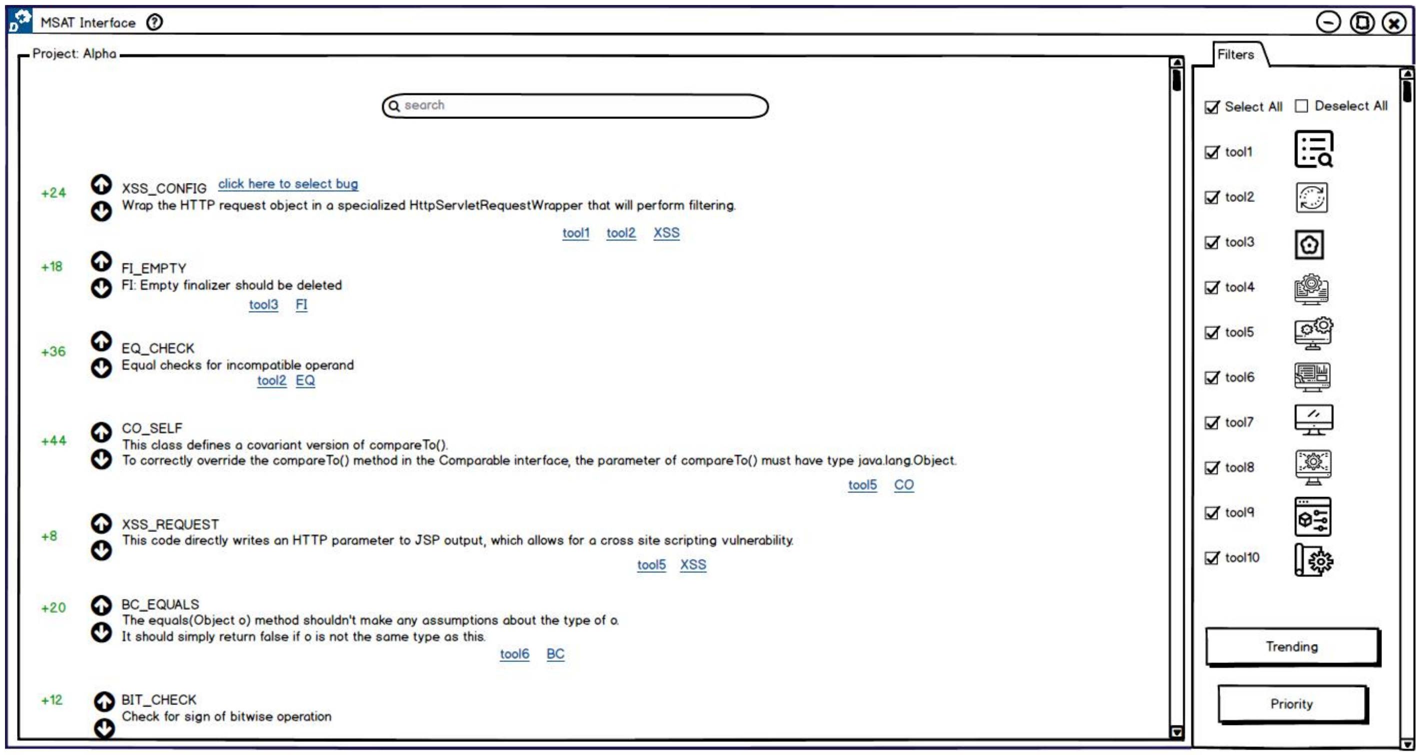
Bugs
☐ My Bugs
☑ All Bugs

Vulnerability Type
☐ SQL Injection
☐ XSS

Users preferred single list as it is more user-friendly and effortless to perceive.

HEINZ NIXDORF INSTITUT
UNIVERSITÄT PADERBORN

[RQ 1] From analysis view perspective, will tags help in scalability of bug results in comparison to separate list or single list?


MSAT Interface

Project: Alpha

search

+24 XSS_CONFIG click here to select bug
Wrap the HTTP request object in a specialized HttpServletRequestWrapper that will perform filtering.
tool1 tool2 XSS

+18 FI_EMPTY
FI: Empty finalizer should be deleted
tool3 FI

+36 EQ_CHECK
Equal checks for incompatible operand
tool2 EQ

+44 CO_SELF
This class defines a covariant version of compareTo().
To correctly override the compareTo() method in the Comparable interface, the parameter of compareTo() must have type java.lang.Object.
tool5 CO

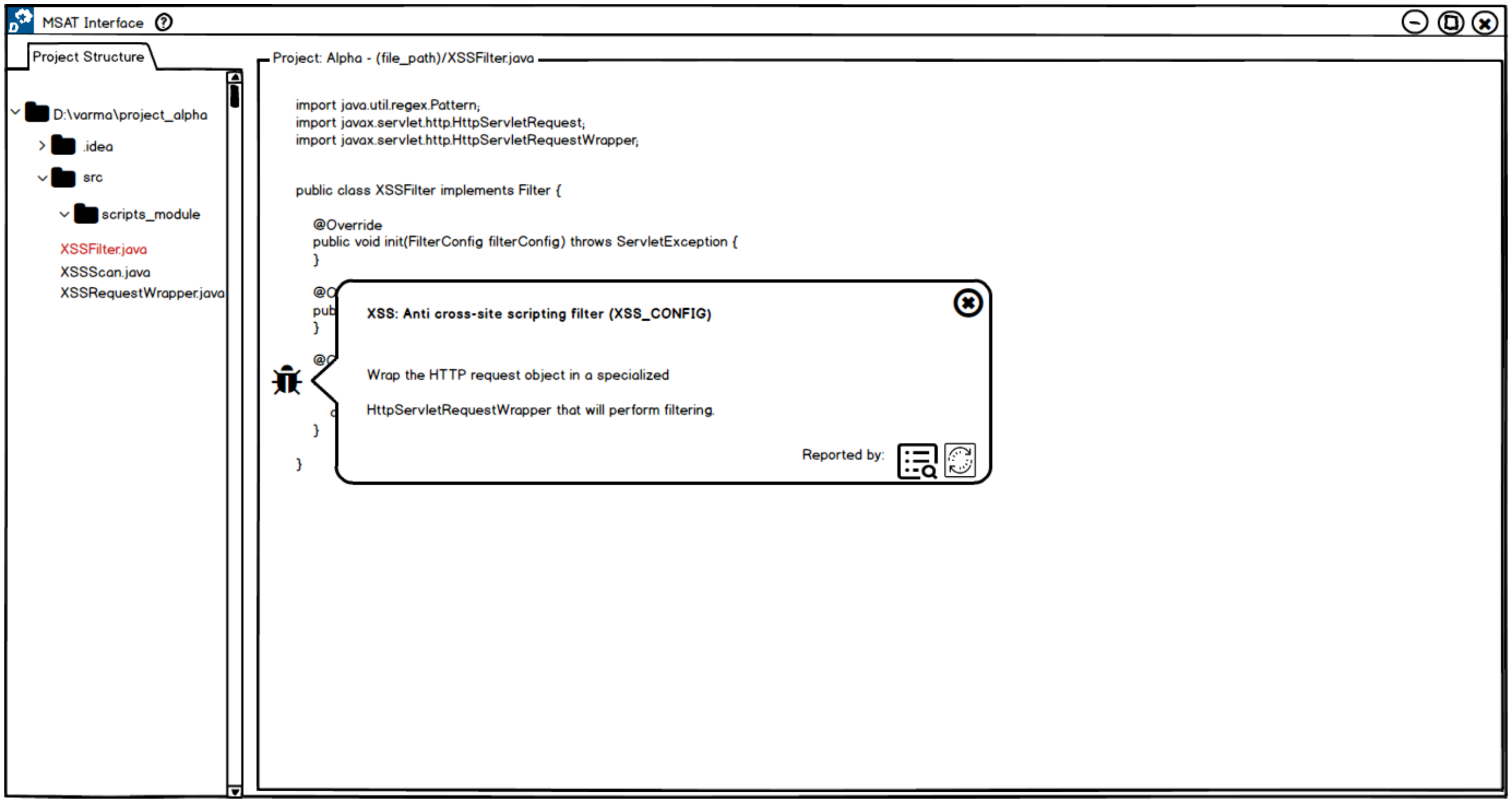
+8 XSS_REQUEST
This code directly writes an HTTP parameter to JSP output, which allows for a cross site scripting vulnerability.
tool5 XSS

+20 BC_EQUALS
The equals(Object o) method shouldn't make any assumptions about the type of o.
It should simply return false if o is not the same type as this.
tool6 BC

+12 BIT_CHECK
Check for sign of bitwise operation

No! Although the proposed design is novel and more open, couple of users explicitly stated it to be confusing. Most users agreed with single list solution idea for this scenario.

[RQ 1] From code view perspective, will single icon suffice the showing of different tools icons?



```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;

public class XSSFilter implements Filter {

  @Override
  public void init(FilterConfig filterConfig) throws ServletException {
  }

  @O
  pub
  }

  @
  }

}
```

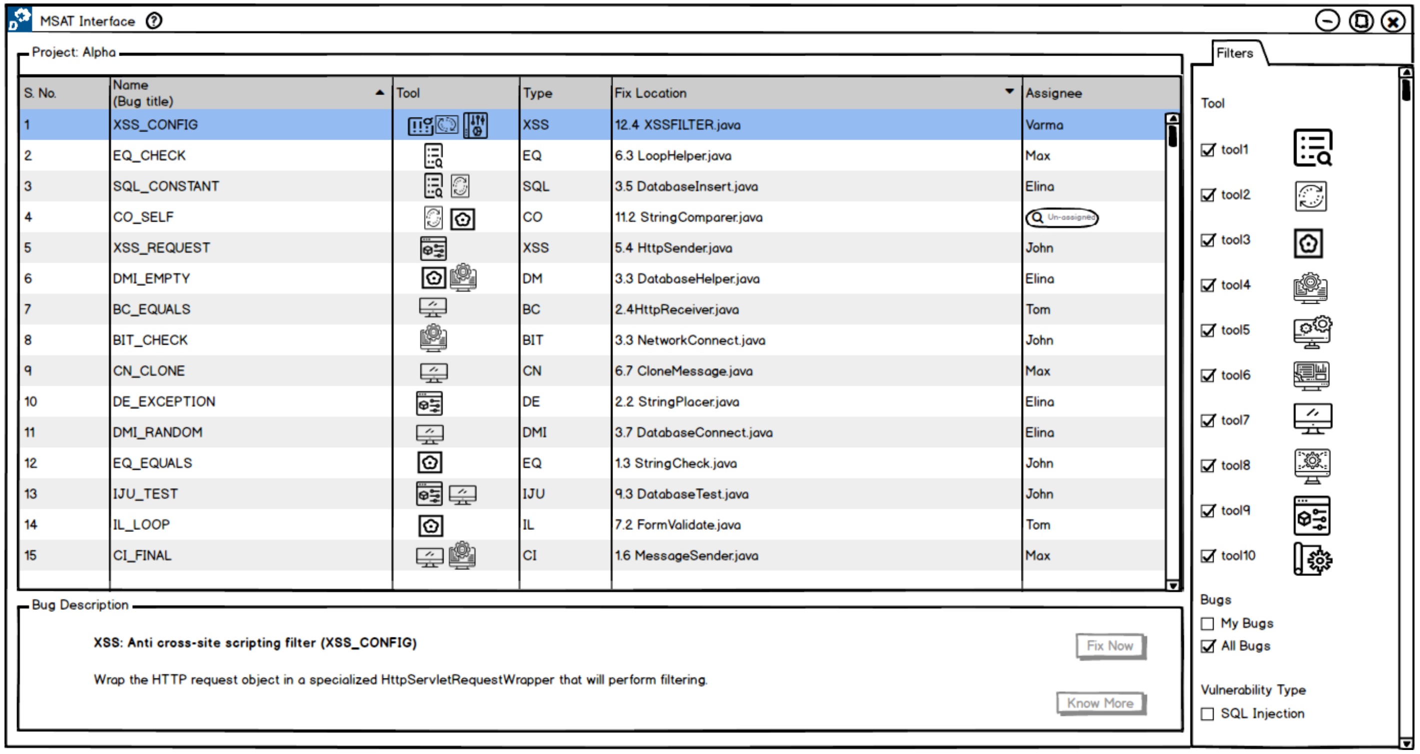
XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized

HttpServletRequestWrapper that will perform filtering.

Yes! Single icon solution idea outwins multiple icons solution idea with a majority of 5 out of 7 users.

[RQ 2] When submitting the bug for analysis, what feedback does user feel convenient among animation, progress bar or popup?



MSAT Interface

Project: Alpha

S. No.	Name (Bug title)	Tool	Type	Fix Location	Assignee
1	XSS_CONFIG		XSS	12.4 XSSFILTER.java	Varma
2	EQ_CHECK		EQ	6.3 LoopHelper.java	Max
3	SQL_CONSTANT		SQL	3.5 DatabaseInsert.java	Elina
4	CO_SELF		CO	11.2 StringComparer.java	Un-assigned
5	XSS_REQUEST		XSS	5.4 HttpSender.java	John
6	DMI_EMPTY		DM	3.3 DatabaseHelper.java	Elina
7	BC_EQUALS		BC	2.4HttpReceiver.java	Tom
8	BIT_CHECK		BIT	3.3 NetworkConnect.java	John
9	CN_CLONE		CN	6.7 CloneMessage.java	Max
10	DE_EXCEPTION		DE	2.2 StringPlacer.java	Elina
11	DMI_RANDOM		DMI	3.7 DatabaseConnect.java	Elina
12	EQ_EQUALS		EQ	1.3 StringCheck.java	John
13	IJU_TEST		IJU	9.3 DatabaseTest.java	John
14	IL_LOOP		IL	7.2 FormValidate.java	Tom
15	CI_FINAL		CI	1.6 MessageSender.java	Max

Bug Description

XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized HttpServletRequestWrapper that will perform filtering.

Fix Now

Know More

Filters

Tool

- ☑ tool1
- ☑ tool2
- ☑ tool3
- ☑ tool4
- ☑ tool5
- ☑ tool6
- ☑ tool7
- ☑ tool8
- ☑ tool9
- ☑ tool10

Bugs

- ☐ My Bugs
- ☑ All Bugs

Vulnerability Type

- ☐ SQL Injection

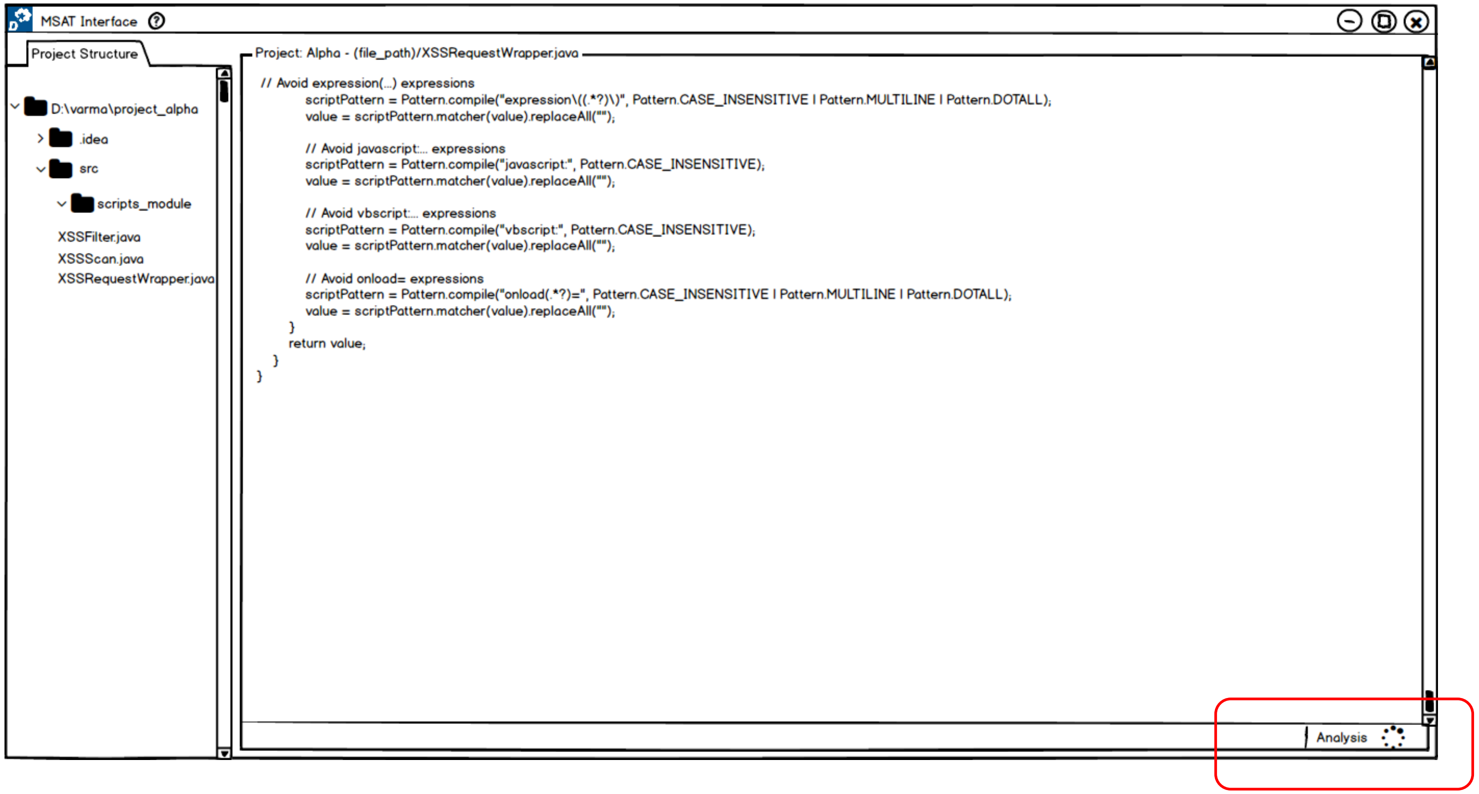
Each has its prominence. However, users felt that pending status is more useful among them.

[RQ 2] Does a single type of feedback suffice or requires combination?

All 3 solution ideas as each could be depict different understandings.

Users felt the requirement of the combination of all feedbacks as each serve in its scope.

[RQ 2] From code view perspective, i.e., once user fixed a bug and submitted for analysis and then off the analysis results screen, then is popup notifications with analysis progress information better to busy status (spinner)?

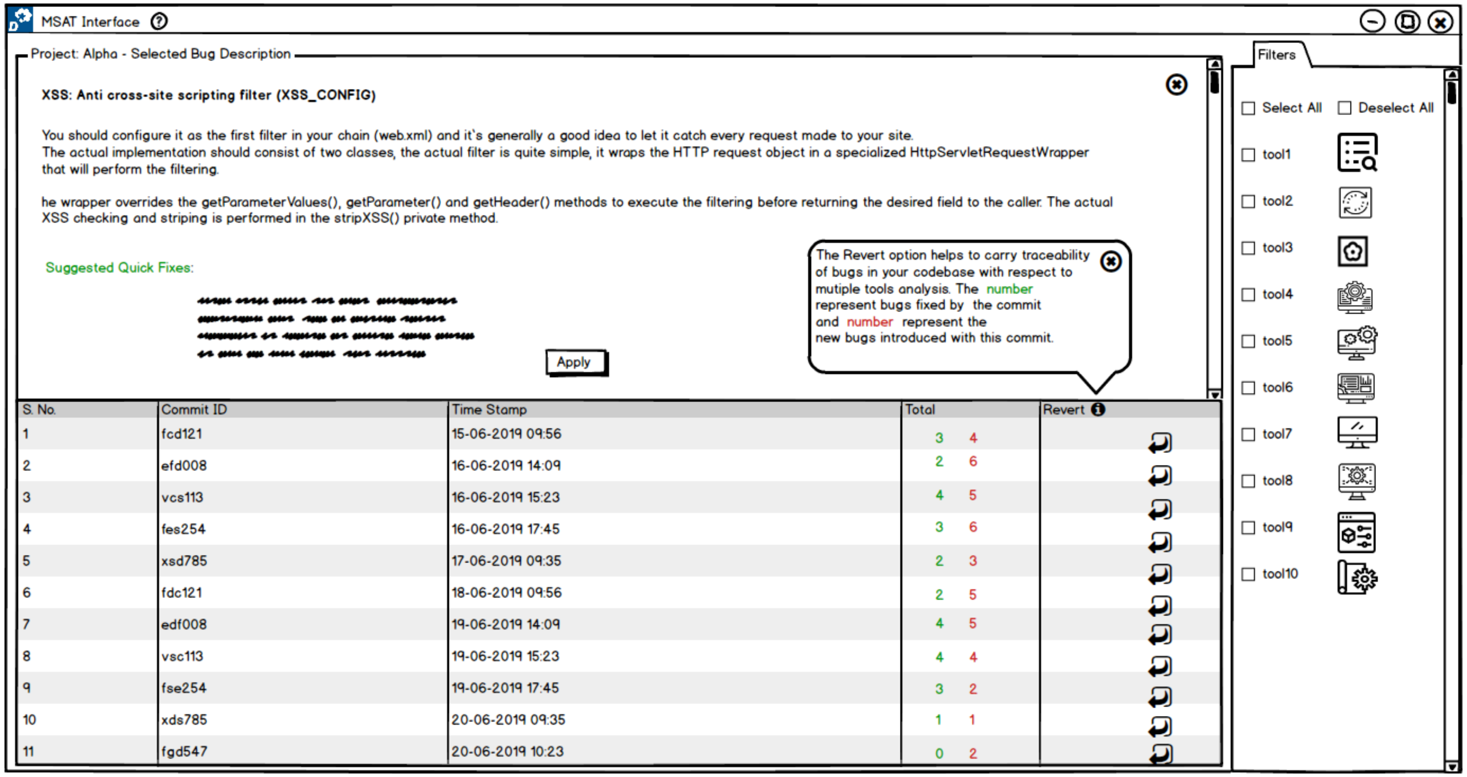


Project: Alpha - (file_path)/XSSRequestWrapper.java

```
// Avoid expression(...) expressions
        scriptPattern = Pattern.compile("expression\((.*?)\)", Pattern.CASE_INSENSITIVE | Pattern.MULTILINE | Pattern.DOTALL);
        value = scriptPattern.matcher(value).replaceAll("");

        // Avoid javascript:... expressions
        scriptPattern = Pattern.compile("javascript:", Pattern.CASE_INSENSITIVE);
        value = scriptPattern.matcher(value).replaceAll("");

        // Avoid vbscript:... expressions
        scriptPattern = Pattern.compile("vbscript:", Pattern.CASE_INSENSITIVE);
        value = scriptPattern.matcher(value).replaceAll("");

        // Avoid onload= expressions
        scriptPattern = Pattern.compile("onload(.*?)=", Pattern.CASE_INSENSITIVE | Pattern.MULTILINE | Pattern.DOTALL);
        value = scriptPattern.matcher(value).replaceAll("");
      }
      return value;
    }
}
```

Popup notifications solution idea outwins status spinner with majority of 4 out of 7. Although remaining said it is annoying, but if needs implemented they would prefer to have when bug fix fails.

[RQ 3] In tracing, will the user need to know the changes made to fix a bug affecting the analysis of other tools?



Project: Alpha - Selected Bug Description

XSS: Anti cross-site scripting filter (XSS_CONFIG)

You should configure it as the first filter in your chain (web.xml) and it's generally a good idea to let it catch every request made to your site.
The actual implementation should consist of two classes, the actual filter is quite simple, it wraps the HTTP request object in a specialized HttpServletRequestWrapper that will perform the filtering.

he wrapper overrides the getParameterValues(), getParameter() and getHeader() methods to execute the filtering before returning the desired field to the caller. The actual XSS checking and striping is performed in the stripXSS() private method.

Suggested Quick Fixes:

[illegible]

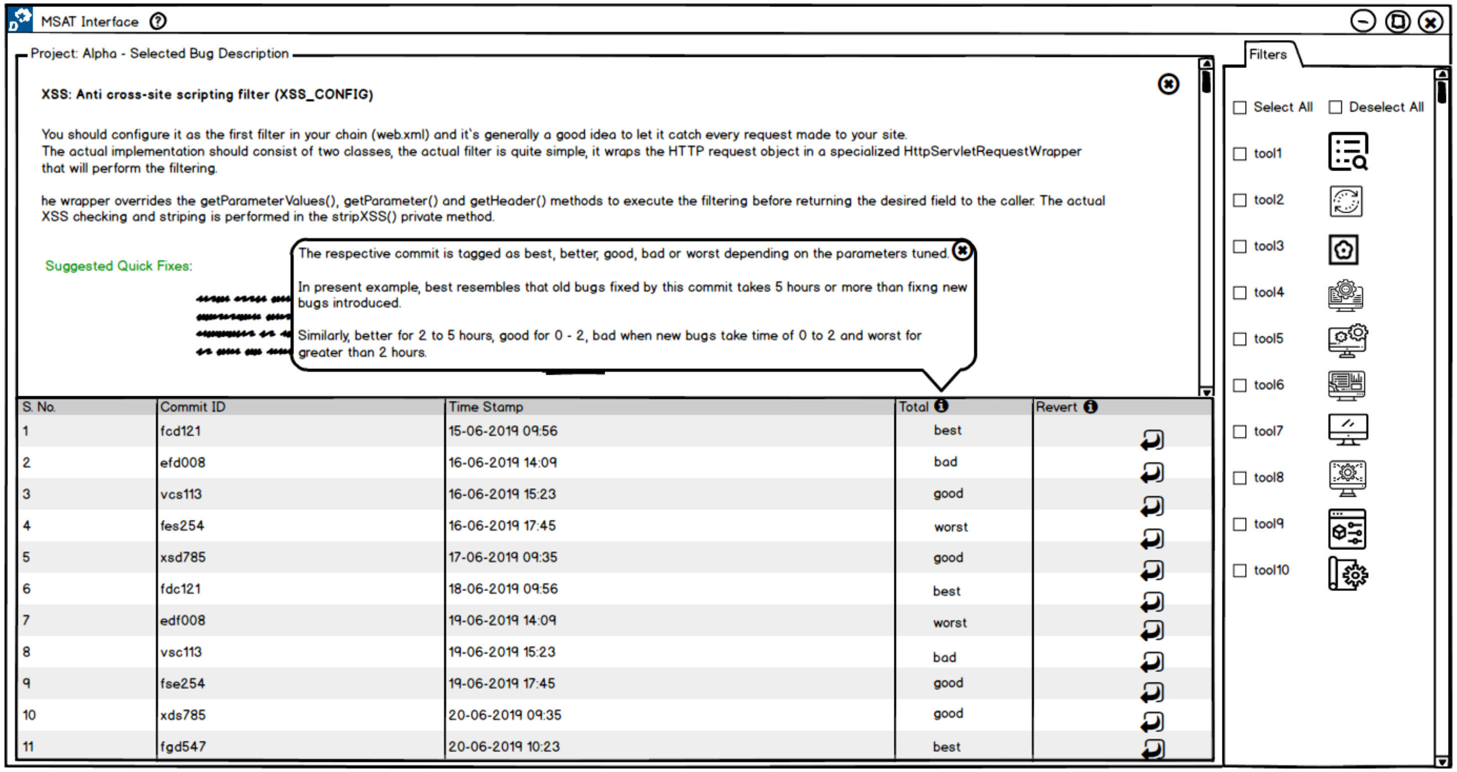
The Revert option helps to carry traceability of bugs in your codebase with respect to mutiple tools analysis. The number represent bugs fixed by the commit and number represent the new bugs introduced with this commit.

S. No.	Commit ID	Time Stamp	Total		Revert
1	fcd121	15-06-2019 09:56	3	4	
2	efd008	16-06-2019 14:09	2	6	
3	vcs113	16-06-2019 15:23	4	5	
4	fes254	16-06-2019 17:45	3	6	
5	xsd785	17-06-2019 09:35	2	3	
6	fdc121	18-06-2019 09:56	2	5	
7	edf008	19-06-2019 14:09	4	5	
8	vsc113	19-06-2019 15:23	4	4	
9	fse254	19-06-2019 17:45	3	2	
10	xds785	20-06-2019 09:35	1	1	
11	fgd547	20-06-2019 10:23	0	2	

☐ Select All	☐ Deselect All
☐ tool1	
☐ tool2	
☐ tool3	
☐ tool4	
☐ tool5	
☐ tool6	
☐ tool7	
☐ tool8	
☐ tool9	
☐ tool10	

Yes! With number representation, it is good, but those do not represent difficulty.

[RQ 3] Does adjective mapping ease the user to trace the changes made in code in terms of bugs existence?



Project: Alpha - Selected Bug Description

XSS: Anti cross-site scripting filter (XSS_CONFIG)

You should configure it as the first filter in your chain (web.xml) and it's generally a good idea to let it catch every request made to your site.
The actual implementation should consist of two classes, the actual filter is quite simple, it wraps the HTTP request object in a specialized HttpServletRequestWrapper that will perform the filtering.

he wrapper overrides the getParameterValues(), getParameter() and getHeader() methods to execute the filtering before returning the desired field to the caller. The actual XSS checking and striping is performed in the stripXSS() private method.

Suggested Quick Fixes:

The respective commit is tagged as best, better, good, bad or worst depending on the parameters tuned.

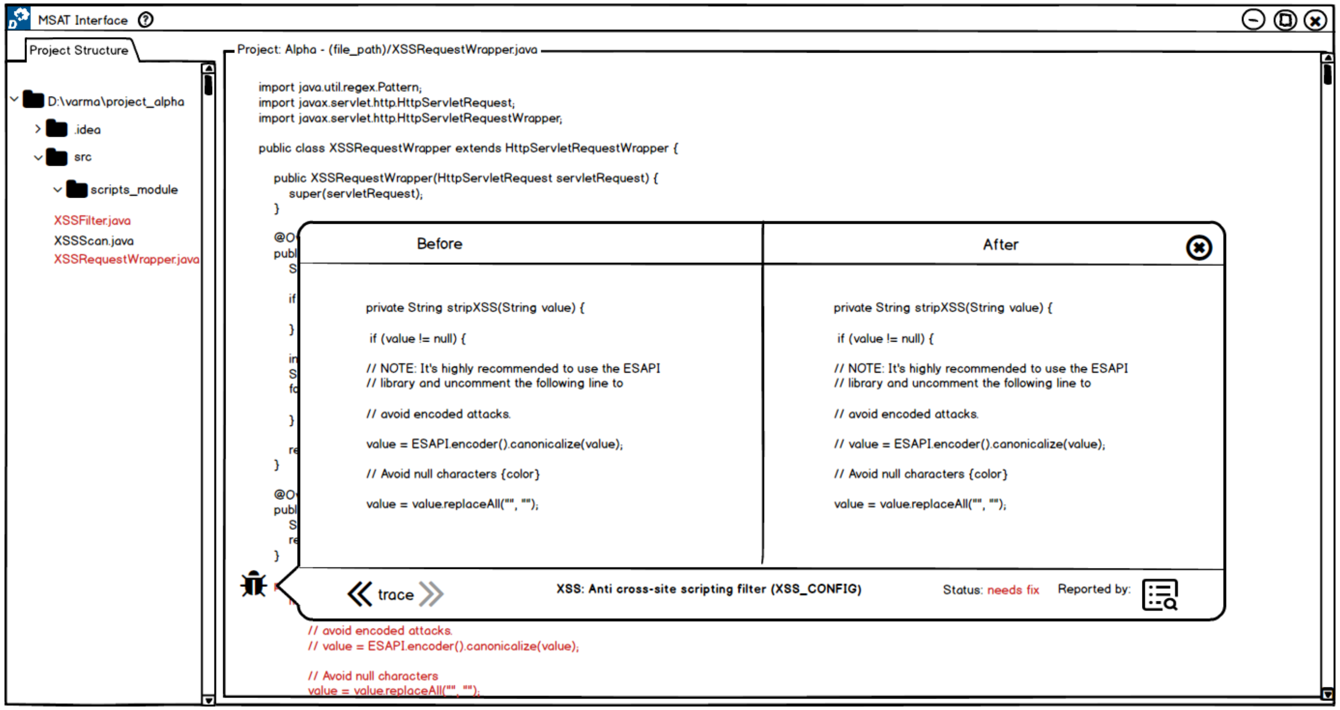
In present example, best resembles that old bugs fixed by this commit takes 5 hours or more than fixng new bugs introduced.

Similarly, better for 2 to 5 hours, good for 0 - 2, bad when new bugs take time of 0 to 2 and worst for greater than 2 hours.

S. No.	Commit ID	Time Stamp	Total	Revert
1	fcd121	15-06-2019 09:56	best	↩
2	efd008	16-06-2019 14:09	bad	↩
3	vcs113	16-06-2019 15:23	good	↩
4	fes254	16-06-2019 17:45	worst	↩
5	xsd785	17-06-2019 09:35	good	↩
6	fdc121	18-06-2019 09:56	best	↩
7	edf008	19-06-2019 14:09	worst	↩
8	vsc113	19-06-2019 15:23	bad	↩
9	fse254	19-06-2019 17:45	good	↩
10	xds785	20-06-2019 09:35	good	↩
11	fgd547	20-06-2019 10:23	best	↩

Yes! With number representation, it is good, but those do not represent difficulty.

[RQ 3] From code view perspective, will the bug tool icons with before/after code help understand the user in easing to fix it?



```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;

public class XSSRequestWrapper extends HttpServletRequestWrapper {

  public XSSRequestWrapper(HttpServletRequest servletRequest) {
    super(servletRequest);
  }
```

Before:

```
private String stripXSS(String value) {

 if (value != null) {

// NOTE: It's highly recommended to use the ESAPI
// library and uncomment the following line to

// avoid encoded attacks.

value = ESAPI.encoder().canonicalize(value);

// Avoid null characters {color}

value = value.replaceAll("", "");
```

After:

```
private String stripXSS(String value) {

 if (value != null) {

// NOTE: It's highly recommended to use the ESAPI
// library and uncomment the following line to

// avoid encoded attacks.

// value = ESAPI.encoder().canonicalize(value);

// Avoid null characters {color}

value = value.replaceAll("", "");
```

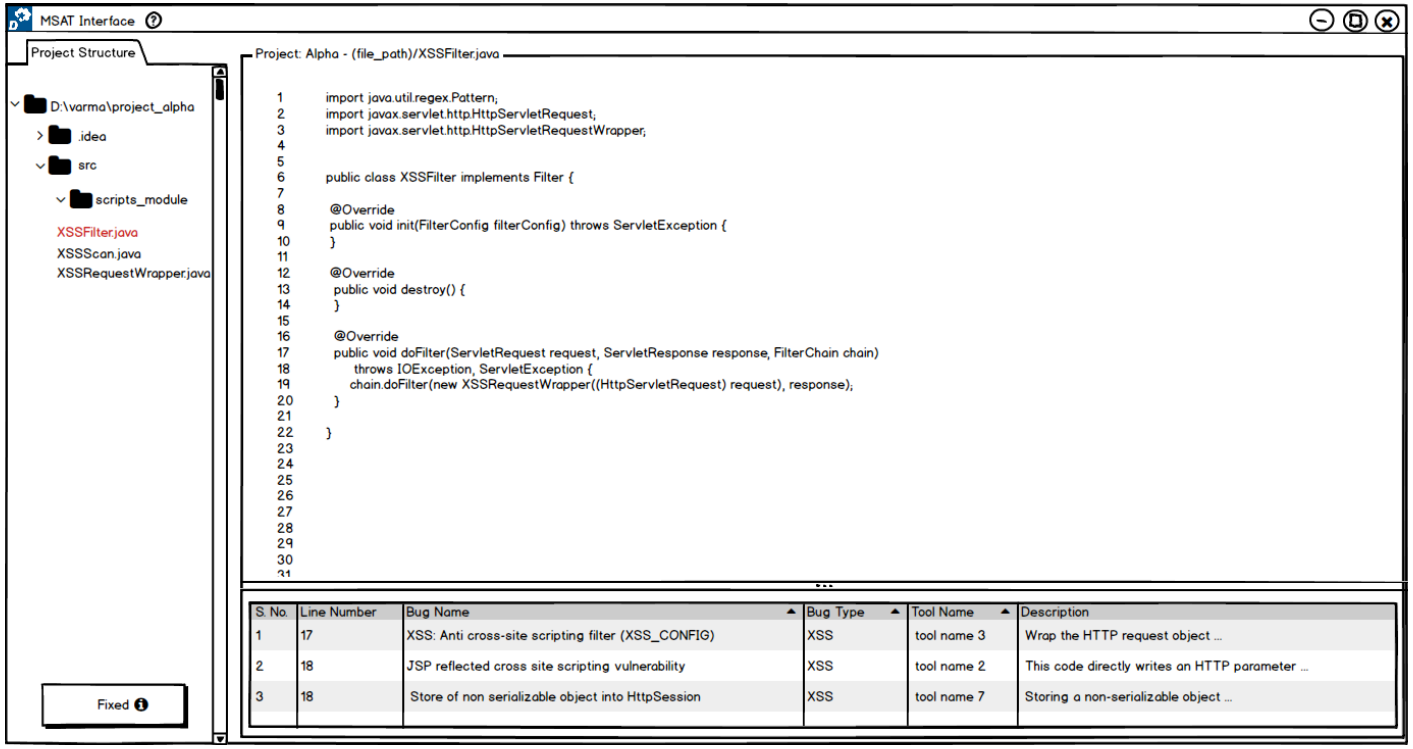
XSS: Anti cross-site scripting filter (XSS_CONFIG)

```
// avoid encoded attacks.
// value = ESAPI.encoder().canonicalize(value);

// Avoid null characters
value = value.replaceAll("", "");
```

Yes! Users felt helpful in tracing, although they did not understand the design in first glance as it is novel.

UX 3

[RQ 1]
Do users prefer bug icons or list view for bugs in same file?

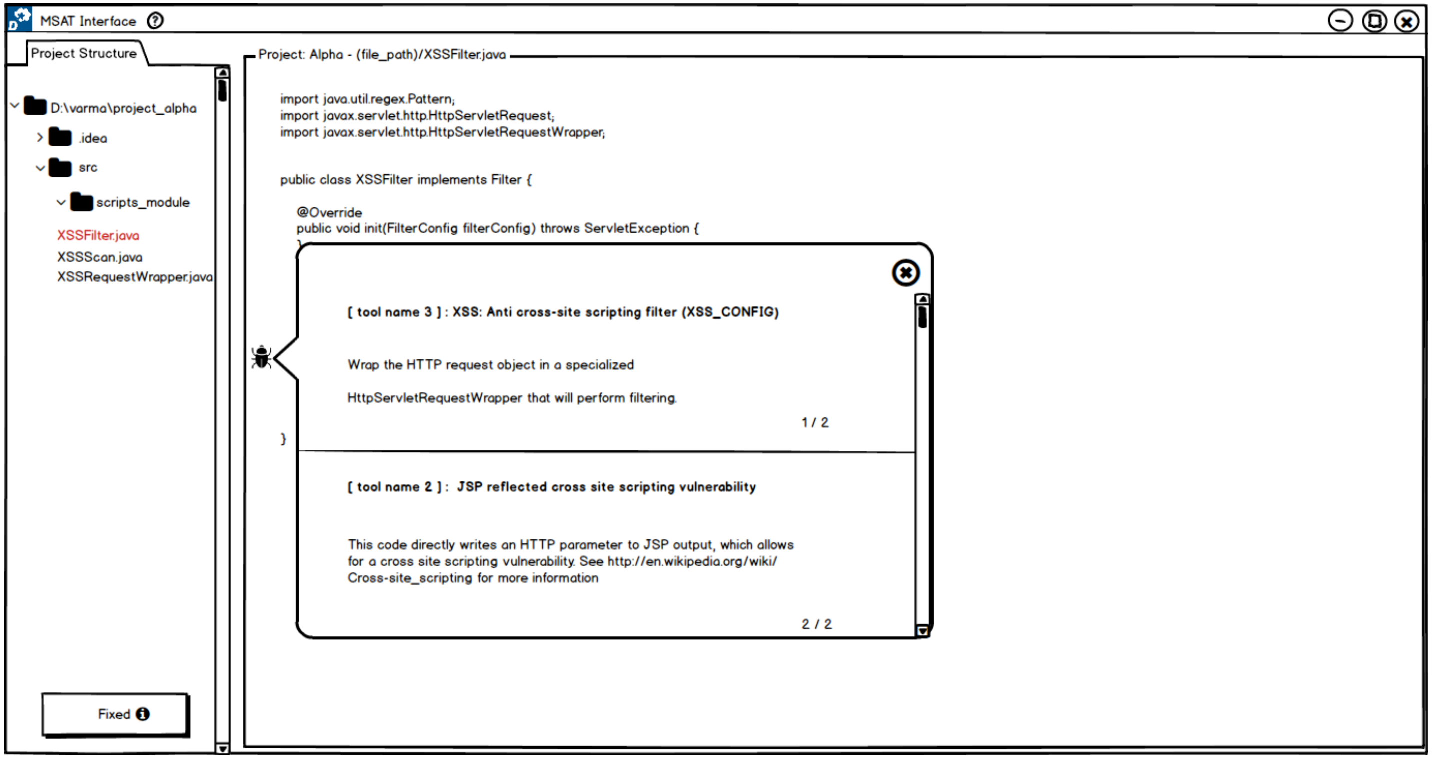


```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;


public class XSSFilter implements Filter {

 @Override
 public void init(FilterConfig filterConfig) throws ServletException {
 }

 @Override
  public void destroy() {
  }

  @Override
  public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
       throws IOException, ServletException {
      chain.doFilter(new XSSRequestWrapper((HttpServletRequest) request), response);
  }

}
```

S. No.	Line Number	Bug Name	Bug Type	Tool Name	Description
1	17	XSS: Anti cross-site scripting filter (XSS_CONFIG)	XSS	tool name 3	Wrap the HTTP request object ...
2	18	JSP reflected cross site scripting vulnerability	XSS	tool name 2	This code directly writes an HTTP parameter ...
3	18	Store of non serializable object into HttpSession	XSS	tool name 7	Storing a non-serializable object ...

Users preferred list view as it is more comfortable and friendly UI. In case of huge codebase, bug icons solution idea would take more time in scrolling to identify bugs.

[RQ 1]
Do users prefer to see bugs one by one or at once in the context of multiple bugs at the same time?



```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;

public class XSSFilter implements Filter {

  @Override
  public void init(FilterConfig filterConfig) throws ServletException {
  }
}
```

[tool name 3] : XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized

HttpServletRequestWrapper that will perform filtering.

1 / 2

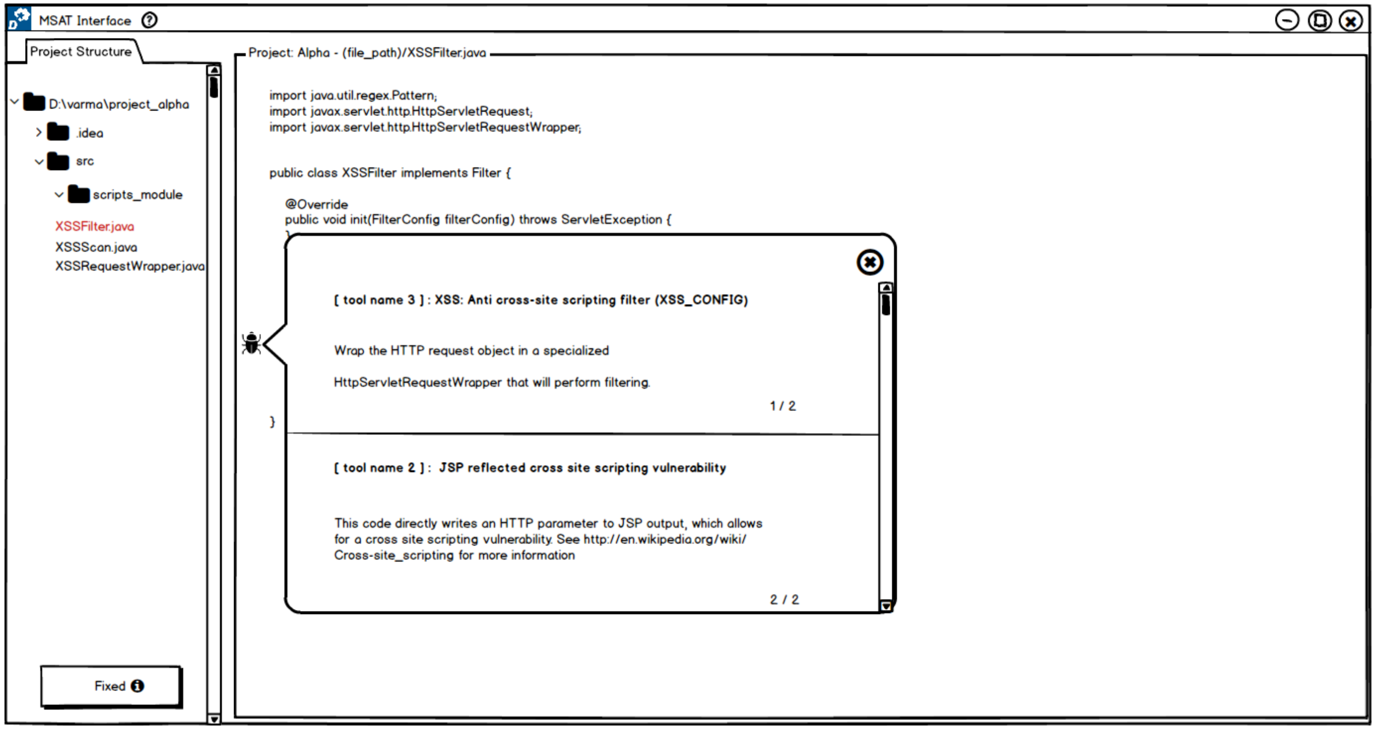
[tool name 2] : JSP reflected cross site scripting vulnerability

This code directly writes an HTTP parameter to JSP output, which allows for a cross site scripting vulnerability. See http://en.wikipedia.org/wiki/Cross-site_scripting for more information

2 / 2

Users preferred horizontal view as it helps in comparing results when using multiple tools. In general, users would like to go one by one and understand the results.
73

[RQ 1]
Does vertical view help in getting an overview of the presence of multiple bugs over horizontal views?



```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;

public class XSSFilter implements Filter {

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
    }
}
```

[tool name 3] : XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized

HttpServletRequestWrapper that will perform filtering.

1 / 2

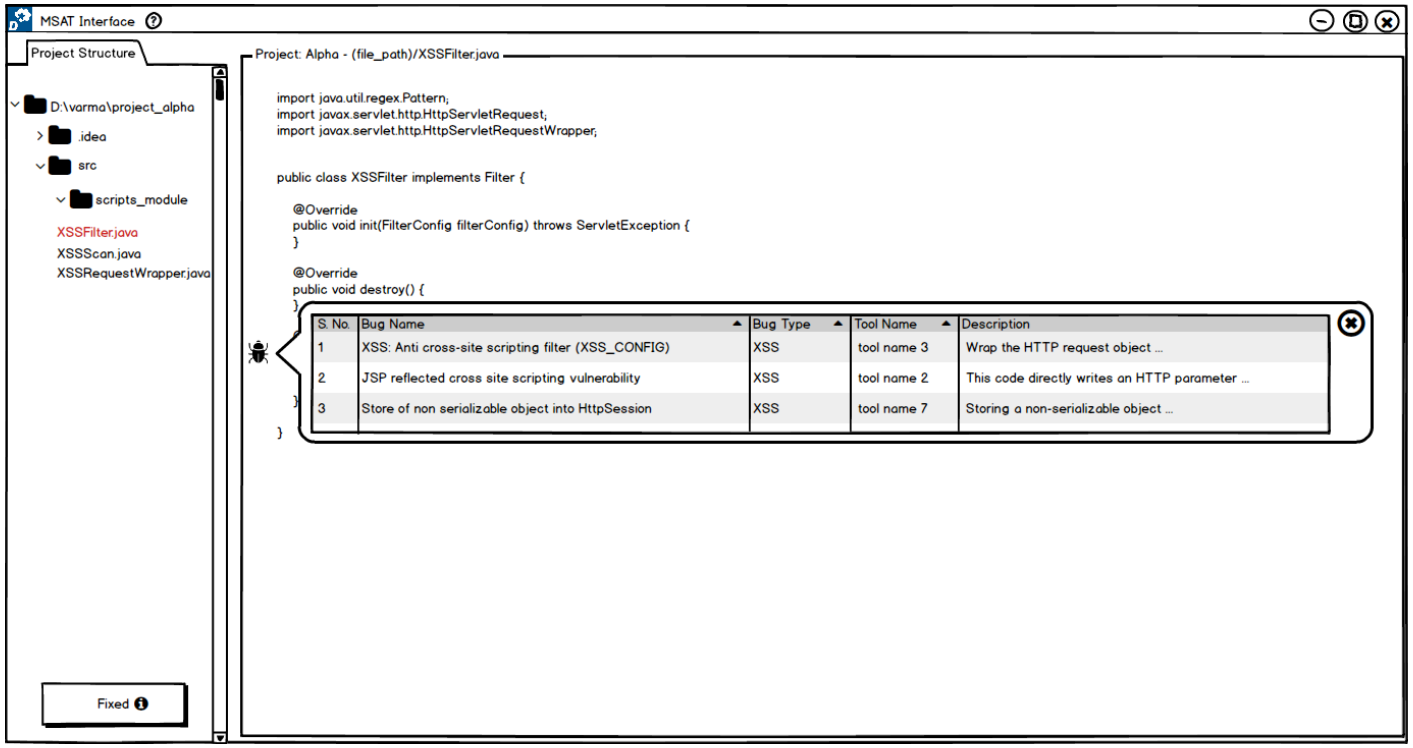
[tool name 2] : JSP reflected cross site scripting vulnerability

This code directly writes an HTTP parameter to JSP output, which allows for a cross site scripting vulnerability. See http://en.wikipedia.org/wiki/Cross-site_scripting for more information

2 / 2

The users mostly prefer horizontal view solution idea as they got used to such proposed UI concerning scrolling. In case of vertical view solution idea, users felt it is best suited for more landscape screens and touch screens.

[RQ 1]
Do users prefer for table view over text description shown for multiple bugs at a line of code?



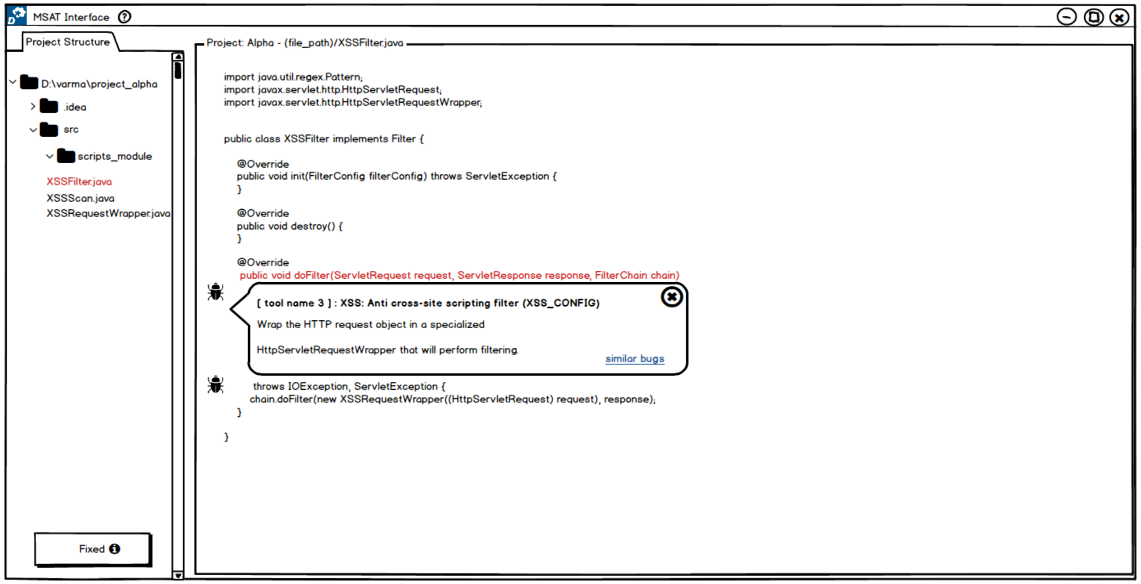
Project: Alpha - (file_path)/XSSFilter.java

```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;


public class XSSFilter implements Filter {

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
    }

    @Override
    public void destroy() {
    }

    }
}
```

S. No.	Bug Name	Bug Type	Tool Name	Description
1	XSS: Anti cross-site scripting filter (XSS_CONFIG)	XSS	tool name 3	Wrap the HTTP request object ...
2	JSP reflected cross site scripting vulnerability	XSS	tool name 2	This code directly writes an HTTP parameter ...
3	Store of non serializable object into HttpSession	XSS	tool name 7	Storing a non-serializable object ...

Users preferred table view over text descriptions as it helps to sort the results and so support comparison.

[RQ 1]
In context of same bug identified but with different line numbers, would have ‘similar bugs’ in bug description with on click pops up similar bug description boxes at the identified line or a list at the bottom help user in locating actual line where bug exist?



```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;


public class XSSFilter implements Filter {

   @Override
   public void init(FilterConfig filterConfig) throws ServletException {
   }

   @Override
   public void destroy() {
   }

   @Override
   public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)

       throws IOException, ServletException {
     chain.doFilter(new XSSRequestWrapper((HttpServletRequest) request), response);
   }

}
```

[tool name 3] : XSS: Anti cross-site scripting filter (XSS_CONFIG)

Wrap the HTTP request object in a specialized

HttpServletRequestWrapper that will perform filtering.

similar bugs

- list

Users preferred list solution idea as with additional popups; it would be confusing and time-consuming.

[RQ 2] Evaluation Set Up: 5 Feedbacks – MSAT-UI Vs Native UIs

- Three different native UI tools for a single JavaScript project.

  - CLI – ESLint
  - IDE – SonarLint
  - WEB - SonarQube

[RQ 2]

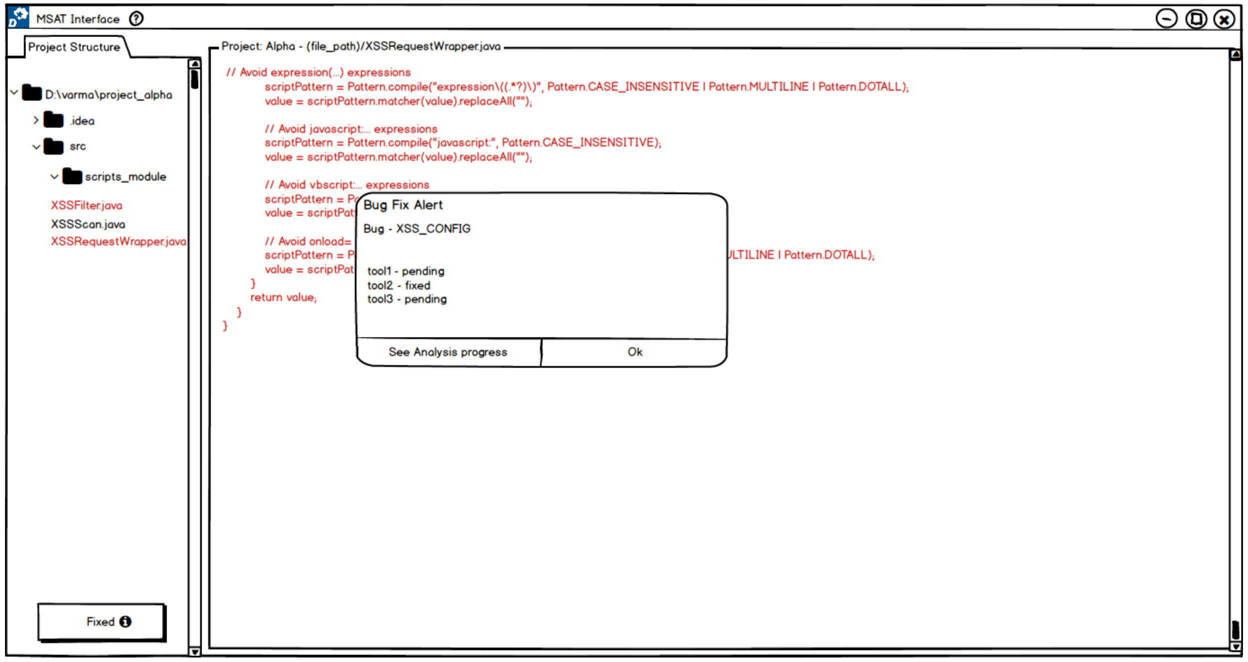
How usable are each feedback functionality compared to the scenario of using unified UI to native UIs?

- Animated Icons
- Progress Bar
- Pending Status Popup
- Alerts
- Status

- Almost all users agreed the ideas being novel and hardly present with native UIs.

Each proposed feedback play an essential role in providing the information to the user. Some are absent in existing tools.

[RQ 2]
Does alert notification help in fixing more bugs in contrast to its absence in current tools UI?



```
// Avoid expression(...) expressions
        scriptPattern = Pattern.compile("expression\\((.*?)\\)", Pattern.CASE_INSENSITIVE I Pattern.MULTILINE I Pattern.DOTALL);
        value = scriptPattern.matcher(value).replaceAll("");

        // Avoid javascript:... expressions
        scriptPattern = Pattern.compile("javascript:", Pattern.CASE_INSENSITIVE);
        value = scriptPattern.matcher(value).replaceAll("");

        // Avoid vbscript:... expressions
        scriptPattern = Pa
        value = scriptPat

        // Avoid onload=
        scriptPattern = P                                          LTILINE I Pattern.DOTALL);
        value = scriptPat
      }
      return value;
    }
}
```

```
Bug Fix Alert
Bug - XSS_CONFIG

tool1 - pending
tool2 - fixed
tool3 - pending

See Analysis progress    Ok
```

Users felt it as useful to have. As in case of success, it helps the developer have positive fulfilment in fixing the bug and in case of failure, the developer could re-try the bug fix again easily.

[RQ 2]
Does MSAT UI with five different mechanisms helps in fixing more bugs in comparison to using multiple tools with native user interfaces?

- Alert - when bug fix failed, helps to work on the bug again (state of work flow)
- Status – time for analysing

Yes! Users will be attentive with the provided feedbacks.

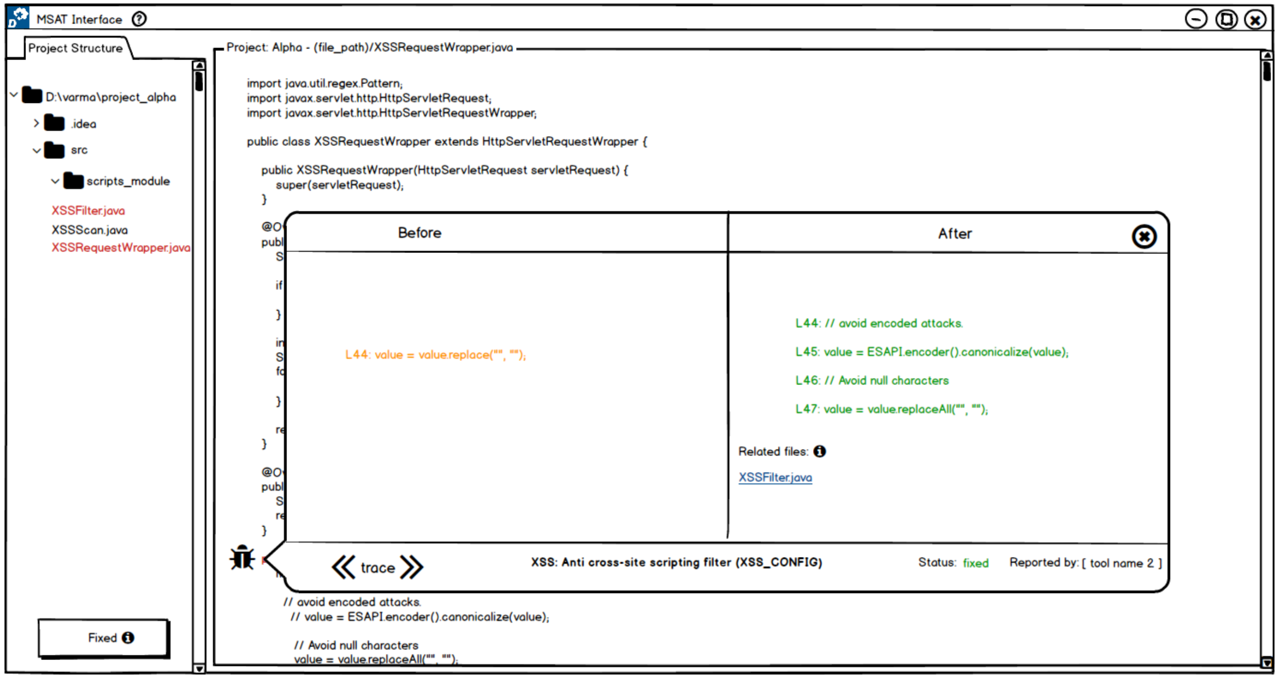
[RQ 2]

Does MSAT UI with five different mechanisms helps in fixing the bugs in a faster way in comparison to using multiple tools with native user interfaces?

- Visualisations provided by these 5 feedback helps!
- Example: Progress Bar – helps in waiting than making system hang

Yes! with the help of direct visualisations provided by the proposed feedbacks.
Notably, ‘alert’ is helpful to try again to fix the bug immediately in case of bug fix fail.

[RQ 3]
Do users prefer having multiple windows to single window in tracing previous bug fixes in a method?



Project: Alpha - (file_path)/XSSRequestWrapper.java

```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;

public class XSSRequestWrapper extends HttpServletRequestWrapper {

  public XSSRequestWrapper(HttpServletRequest servletRequest) {
    super(servletRequest);
  }

  @O
  publ
    S

    if

    }

    in
    S
    fo

    }

    re
  }

  @O
  publ
    S
    re
  }

    // avoid encoded attacks.
     // value = ESAPI.encoder().canonicalize(value);

     // Avoid null characters
     value = value.replaceAll("", "");
```

Before:

```
L44: value = value.replace("", "");
```

After:

```
L44: // avoid encoded attacks.
L45: value = ESAPI.encoder().canonicalize(value);
L46: // Avoid null characters
L47: value = value.replaceAll("", "");
```

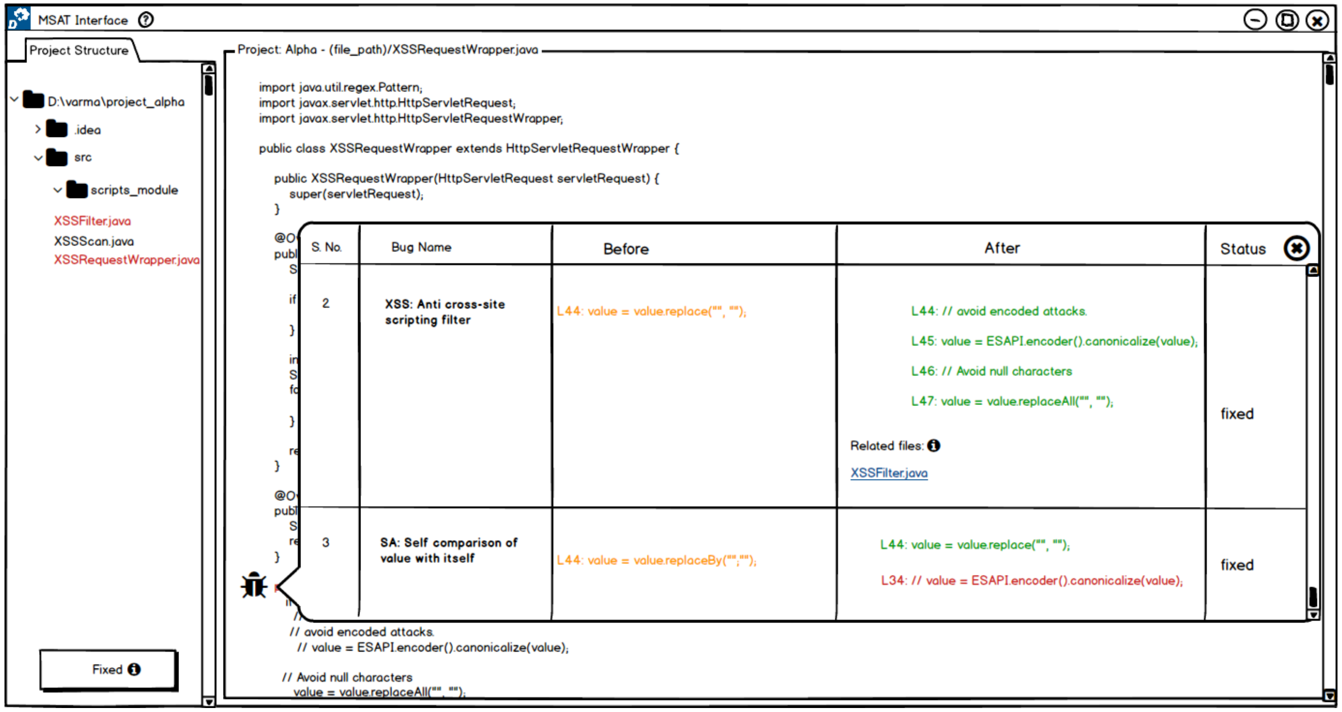
XSS: Anti cross-site scripting filter (XSS_CONFIG) Status: fixed Reported by: [tool name 2]

Users preferred single window as it is easy to perceive results.

[RQ 3]
Do users be able to keep up in state of workflow as tools scale?

- Yes! proposed solution ideas promised to keep up the scalability.
- However, users preferred ‘table view’ as easy with their consistency model.

[RQ 3]
While tracing previous bug fixes in a method, do users prefer a table view to a before/after windows?



Project: Alpha - (file_path)/XSSRequestWrapper.java

```
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;

public class XSSRequestWrapper extends HttpServletRequestWrapper {

  public XSSRequestWrapper(HttpServletRequest servletRequest) {
    super(servletRequest);
  }

  @O
  publ
    S

    if

    }

    in
    S
    fo

    }

    re
  }

  @O
  publ
    S
    re
  }

    // avoid encoded attacks.
     // value = ESAPI.encoder().canonicalize(value);

   // Avoid null characters
     value = value.replaceAll("", "");
```

S. No.	Bug Name	Before	After	Status
2	**XSS: Anti cross-site scripting filter**	L44: value = value.replace("", "");	L44: // avoid encoded attacks. L45: value = ESAPI.encoder().canonicalize(value); L46: // Avoid null characters L47: value = value.replaceAll("", ""); Related files: XSSFilter.java	fixed

3	**SA: Self comparison of value with itself**	L44: value = value.replaceBy("","");	L44: value = value.replace("", "");
			L34: // value = ESAPI.encoder().canonicalize(value);
				fixed

Users found both are useful, but in case of scalability, 4 out of 5 users preferred table view.

HEINZ NIXDORF INSTITUT
UNIVERSITÄT PADERBORN

Q. Do users prefer having tool names in general?

- Yes!

- Compare tool performance
- Helps to have much information as possible