[Introduction]

- ☐ consent form
- ☐ start recording

Pre-test Questionnaire:

Q. How often do you do software development (i.e., coding)?

- Is it daily, weekly etc?

Q. Have you used static analysis tools?

- tool to find bugs in your code!

Q. What tools have you used?

- Is it IDE integrated tool or any other dedicated tools like FindBugs, PMD etc.

Q. Which is your favourite one?

Q. Why it is your favourite?

- Any correlation to its better user interface feature?

Assume you are working on a project and want to find bugs in your code. There are ten tools linked to your codebase to have better coverage of vulnerabilities. Now let us walkthrough 3 main questions with respect to its user interface.

[Research Question 1]

The first one;

Q. How to display the results of the same codebase from different analysis tools?

(present the prototype one after other in random order, do cognitive walkthrough – think aloud)

(for first three prototypes – results perspective)

Scenario: Assume you as a Software Developer working on a project called “Alpha”. On next working day, you are about to see analysis results from multiple tools and your primary intention is to make your code base bug free.

Task: Understand the results and Identify the common bug reported by tools.

Prototype [single list]:

Success Criteria:
User find the **Bug name – XSS_CONFIG** which is common by **clicking** on **'Select all'** tools filter.

Additional:
Demo the user interface by showing multiple screen for each tool and significance of icon.

Prototype 2 [separate list]:

Success Criteria:
User scrolls to see the results **until tool 6** and find the **Bug name – XSS_CONFIG** which is common.

[why not user scroll until tool 10 – reason: there are no bugs reported by tool 7, 8, 9 and 10. It signifies real time scenario.]

Additional:
Demo the user interface by showing the bugs related to different tools separately and the bugs related to tool 1 and tool 3

Prototype 3 [tags]:

Success Criteria:
User scrolls down **3 screens** to see the results and find the **Bug name – XSS_CONFIG** which is common.

Additional: Demo the user interface by showing alternative approaches to see bugs reported by tool1.

(after presenting all related prototypes)

Follow up:

Q. Among the 3 solution ideas presented, which one do you feel convenient with for given task?

- easy to use?

Q. Why?

Q. Rate: [O be low, 10 be high] – all designs in comparison

Q. Which UI do you think scales better with increase in bugs?

(code view perspective)

Scenario: Assume you are a developer working on project Alpha precisely on software package called 'scripts_module'. On your next working day morning opened your code editor to start your work.

Task: What is the bug being reported by your analysis tools? And how many tools reported it?

Prototype 4 [multiple icons]:

Success Criteria:
On two clicks with different tool icons know that the bug is 'Anti cross site scripting filter'.
The answer for number of tools is 2.

Prototype 5 [single icon]:

Success Criteria:
On single click on bug icon know that the bug is 'Anti cross site scripting filter'.
The answer for number of tools is 2.

(after presenting both prototypes)

Follow up:

Q. Among the 2 solution ideas presented, which one do you feel convenient with?

- easy to use?

Q. Why?

Q. Rate: [O be low, 10 be high] – in comparison

[Research Question 2]

Let's switch to second main question;

Q. What feedback works to know that the bug fixing is on-going?

(present the prototype one after other in random order, do cognitive walkthrough)

(now on results perspective)

Prototype 1,2,3: [animated icon, progress bar, popup]

Scenario:

Assume that you worked on a bug and changed some code related to it.
Next, submitted bug fix code for analysis.

Task:

- Observe what happens after clicking on 'Fixed' button.

[no success criteria as designer has to demo the feedback effect]

Follow up:

Q. Among the 3 solution ideas presented, which one do you feel convenient with?

- easy to use?

Q. Why?

Q. Rate: [O be low, 10 be high] – in comparison

(now on code view perspective)

Prototype 4,5: [alert, status]

Scenario:

After submitting code for analysis, instead of being on the bug results window, you moved on to next task in code editor.

Task:

- Observe the visuals with alert box and status bar with spinner as demo.

(after presenting related prototypes)

Follow up:

Q. Among the 2 solution ideas presented, which one do you feel convenient with?

- easy to use?

Q. Why?

Q. Rate: [O be low, 10 be high] – in comparison

Q. Would you prefer in having multiple feedbacks?

[Research Question 3]

Let's switch to final main question;

Q. How to carry traceability of bug fixing?

(Explain what traceability in this scenario is.)

(Here in this scenario traceability mean to see how each attempt / commit to fix a bug effects the metrics of the analysis tools. This ensures some safety to prevent new bugs.)

Prototype 1 & 2:

(results perspective)

Scenario:

You have been working on some part of code to fix a bug in last few working days. Now you would like to see how the changes (commits) you made are affecting the analysis results of other tools.

(present the prototype one after other in random order)

Task:

Watch the results of all tools and decide on a best commit among last 3 to revert as to start from that point.

Success Criteria:

Prototype 1: Selecting the commit id – fse254

Prototype 2: Selecting the commit id – fgd547

[Understand the scenario to take decision to revert to what level if needed.]

(after presenting prototype)

Follow up:

Q. Among 2 solution ideas, which one do you feel convenient with your workflow?

Q. Rate: [O be low, 10 be high] - in comparison

Prototype 3 [Before/After]:

Scenario:

As usual you are working on a code editor and notice a bug reported. You remember you have worked on this part of code before to fix some other bug.

Task:

Find out why you changed the code before and plan to come up with better solutions that satisfies the tools reporting same part of code. In this case, what was the code line you changed earlier to fix the bug reported earlier?

Success Criteria:

It is found out that code line " **value = value.replaceAll("","");** " is uncommented.

Follow up:

Q. Do you feel convenient by this UI with your workflow? Yes/No

Q. Why?

[Conclusion]