## Team details

|  |  |
| --- | --- |
| Batch & Team name | MS2 & Team No: 5 |
| Team members | Ambika K. C., Sreelal E. S., Srivatsan R. |
| Date | 24/09/2019 |
| Status | In-progress |

Notes:

* In this form, ‘your code’ is the code you own now – i.e., Phase-2 code
* If you link the evidence (e.g., to a file in GitHub), you don’t need to update the doc every time

## Compiler warnings and Linting

|  |  |  |
| --- | --- | --- |
|  | Assertion | Exceptions |
|  | Metric: The warning + linting count is zero | Server Api is not refactored. |
|  | Sustain: The ‘treat warnings as errors’ setting is enabled |  |

#### What linting tool is used?

SonarLint

#### How have the warnings fared, since you took over in Phase-2?

Normal

### Evidence

Attach/link: build & linting logs/screenshots

<https://github.com/srivats07/StaticCodeAnalyzer/blob/master/sonarlintClient.PNG>

<https://github.com/srivats07/StaticCodeAnalyzer/blob/master/waeConsole.PNG>

## Static Analysis / Coding standards

#### What is the max cyclomatic complexity of a function in your code?

3 per function. For server api it’s more

#### Where did you reduce the complexity of the code in Phase2?

Client applications

#### What are the instances where you improved the naming for enhanced clarity / consistency?

Libraries and classes are named according to the functions they perform

### Evidence

Attach/link: static-analysis / complexity report

<https://github.com/srivats07/StaticCodeAnalyzer/blob/master/CodeMetric.PNG>

<https://github.com/srivats07/StaticCodeAnalyzer/blob/master/CodeMetricsOfClientSide.xlsx>

## Duplication

|  |  |  |
| --- | --- | --- |
|  | Assertion | Exceptions |
|  | Metric: The duplication count is zero |  |
|  | Sustain: We have a mechanism to eliminate duplication |  |

#### How much duplication did you reduce in Phase2? List the instances.

Kept it as minimum as possible

### Evidence

Attach/link: Simian output

<https://github.com/srivats07/StaticCodeAnalyzer/blob/master/SimianConsole.PNG>

<https://github.com/srivats07/StaticCodeAnalyzer/blob/master/SimianForGUIClient.PNG>

## Unit testing

|  |  |  |
| --- | --- | --- |
|  | Assertion | Exceptions |
|  | Metric: We met our code-coverage target  (if you didn’t fix a target, take 75% as reference) | Code coverage greater than 75% |
|  | Sustain: We have a mechanism to enhance coverage and not let it slip. |  |

#### For each not-covered-code: What is the consequence of not-covering?

Explain in terms of development-time, for someone who receives your code

### Evidence

Attach/link: coverage report / screenshot

<https://github.com/srivats07/StaticCodeAnalyzer/blob/master/CodeCoverageConsole.PNG>

## Dynamic Analysis

|  |  |  |
| --- | --- | --- |
|  | Assertion | Exceptions |
|  | Metric: We have timing- and usability-related tests | Not performed |
|  | Sustain: We have a mechanism to avoid slippage |  |

### Evidence

Attach/link: timing- and usability- tests and checks

## Functional / end-to-end tests

#### How many end-to-end tests do you execute, to verify all the functionality?

Manual tests were performed to check functionalities

#### How many are automated?

### Evidence

Attach/link: test-scripts and/or test-spec

## Retrospection

If you had to do Phase-1 again (with the same time & scope), what would you do different?

Keep it minimal and clear as possible. Maintainability plays an important task in code extension.