<Company> System, Inc.

Intelligent Automation

Bot Development Code Review Checklists

Confidential

Versioning

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 14 Dec 2018 | 1.0 | Draft | Mayank Patel & Shubhen Sarangi, Datamatics |
|  |  |  |  |
|  |  |  |  |

# Summary

Code Review, or Peer Code Review, is the act of consciously and systematically convening with one's fellow programmers to check each other's code for mistakes, and has been repeatedly shown to accelerate and streamline the process of software development.

While reviewing the code, ask yourself the following basic questions:

1. Am I able to understand the code easily?
2. Is the code written following the coding standards/guidelines?
3. Is the same code duplicated more than twice?
4. Can I unit test / debug the code easily to find the root ca use?
5. Is this function or class too big? If yes, is the function or class having too many responsibilities?

# Common Checklist

* Does the code work? Does it perform its intended function, the logic is correct etc.
* Is all the code easily understood?
* Does it confirm to your agreed coding conventions? These will usually cover variable and function names, no. of script lines, indentations, formatting, and comments.
* Is there any redundant or duplicate code?
* Is the code as modular as possible?
* Is there any commented out code?
* Do loops have a set length and correct termination conditions?
* Do the names used in the script convey intent?

# Performance

* Are there any obvious optimizations that will improve performance?
* Can any of the code be replaced with library or built-in functions?
* Can any logging or debugging code be removed?

# Security

* Are all data inputs checked (for the correct type, length, format, and range) and encoded, masked?
* Where third-party utilities are used, are returning errors being caught?
* Are output values checked and encoded?
* Are invalid parameter values handled?

# Documentation

* Do comments exist and describe the intent of the code?
* Are all functions commented?
* Is any unusual behaviour or edge-case handling described?
* Is the use and function of third-party libraries documented?
* Are data structures and units of measurement explained?
* Is there any incomplete code? If so, should it be removed or flagged with a suitable marker like ‘TODO’?

# Testing

* Is the code testable? The code should be structured so that it doesn’t add too many or hide dependencies, is unable to initialize objects, test frameworks can use methods etc.
* Do unit/script tests actually test that the code is performing the intended functionality
* Could any test code be replaced with the use of an existing API?