# Code Review Checklist

## Documentation

* All source code contains @author for all authors.
* Codes are properly commented to improve its maintainability.
* Each class should have only one responsibility. Same rule applies to the methods also.
* Followed open/closed principle. (According to this principle, an object-oriented object must be open for extension but must be closed for modification.)
* Followed CamelCase Notation- Naming Convention. Gave meaningful & easy to understand names, to increase its readability.
* Methods have approximately 15 lines at maximum. Larger methods cut into smaller pieces. Also length of the class did not cross 200 lines, which makes it easier to understand what the class is supposed to do.
* Describe behavior for known input corner-cases.
* Complex algorithms should be explained with references. For example, document the reference that identifies the equation, formula, or pattern. In all cases, examine the algorithm and determine if it can be simplified.
* Incomplete code is marked with //TODO or //FIXME markers.
* All public and private APIs are examined for updates.

## Error Handling

* Invalid parameter values are handled properly early in methods (Fast Fail).
* NullPointerException conditions from method invocations are checked.
* Consider using a general error handler to handle known error conditions.
* An Error handler must clean up state and resources no matter where an error occurs.
* Catch block should not be empty, it should at least log the exception. Ex: catch (Exception ex) {}.

## Performance

* Objects are duplicated only when necessary. If you must duplicate objects, consider implementing Clone and decide if deep cloning is necessary.
* Avoid large objects in memory, or using String to hold large documents which should be handled with better tools.
* Do not leave debugging code.
* Avoid System.out.println(); statements in code, or wrap them in a Boolean condition statement like if(DEBUG) {...}