Data Recognition Corporation (DRC) - SB11 Development Teams code review process

1.) SB11 Code review is part of the on-going DRC Agile development process. This is partially accomplished via Daily standups. Developers communicate what they are working on and discuss potential design or coding issues. These items are then tabled for the standup and a white-board session takes place as soon as possible following the standup meeting. Design/code changes can be made before the development task is completed, preventing significant refactoring efforts.
2.) Code quality tools run on scheduled intervals and are integrated with the development build process. Output of these tools is then used to correct potential issues. Developers are required to justify exceptions, which may result in a reconfiguration of the tool to not produce these false negatives, or the peer review simply allows the exception condition. Quality checks are performed for:
    a. Code structure/syntax
    b. Potential common coding mistakes/bugs.
    c. Test Coverage
    d. Cyclomatic complexity
3.) Build/Deployment processes are automated and run with each code push to the master repository. Unit and Integration tests are run with each build. Test coverage tools reveal code that is not being tested via automated means. In certain cases, integration tests can be difficult to run in build environment, for these exceptions, the lack of automated Integration testing is justified by the developer.
4.) Agile development tasks are managed in a centralized tool with a workflow process. When all stories that complete an Epic are "ready for test", the code is peer reviewed. Once the code is validated, the tasks can be moved to "completed" status.

Code Reviewer Guidelines:

1.) Primarily a code review should look at the functional aspects of the code. Does it meet the requirements and are there potential issues? Checks for appropriate validation controls.
2.) Look for redundant code, potentially considering re-use across other projects. Can the code be moved to a common library?
3.) Performance considerations, are there any obvious performance bottlenecks?
4.) Security issues. Any clear security holes? For SB11, any projects delivered prior to the development of a global security strategy will not focus on the security aspects.
5.) Error handling. Are all potential errors being addressed by the code? Is the code handling errors in a consistent manner as established by the development team.