SAF FAQ: What is a recommended best practice for Code Review?

Answer: Teams should follow these code review recommendations:

Implement Project Code Standards for DevOps:

1. Employ code quality standards (for measuring code quality)
    a. Linting - used for checking for code clarity and neatness in code
    b. Code Coverage to 90-95% - unit/functional testing to exercise 90-95% of all functions
    c. Complexity Reduction - to increase maintainability and modularity, helps developers fix multiple instances of security problems with fewer corrections to code.
    d. Static Code review (security review)
        i. Seek a tool that attempts to check as many applicable Common Weaknesses Enumeration (CWE) types CWE, from SANS Top 25.
        ii. Note however that few tools reach much higher than a 30% true positive rate.[1]
        iii. Running tools from different sources can help identify true positives.
    e. Peer code review by team members (informal, analogous to independent release-based review)
2. Integrate all tools into the CI/CD pipeline – Automate to save time! Spend time evaluating the results rather than running the tool manually.
3. Code quality tools are selected as appropriate for the entire code base used for the project (e.g., node.js (JavaScript), ruby, etc.)[2]

Employ a Good Workflow:

1. Use branch/merge pull request model that rebases against the main line:
   → Upon every merge of a pull request into the main line, run code quality tools.
2. In the source code repository: All code commits must be tagged/related to a planned project issue ticket
3. In the project issue tracker: All planned project issue tickets for coding/recoding a function must be tagged with applicable security controls, with the help of the on-staff security engineer:
    a. Tag (identify) the security controls the function supports for the application - e.g., the application's authentication service directly supports (performs) AC-3 (Access Control Enforcement)
    b. Tag security controls that support the security of the function itself (e.g., the application's authentication service is protected by file permissions, encryption, employed with infrastructure components)

1, 2, and 3 together provide a way to map pull requests to controls, to support security-based change control tracking.

---

[1] https://rawgit.com/OWASP/Benchmark/master/scorecard/OWASP_Benchmark_Home.html

[2] Sonarqube plugin for javascript exists - https://docs.sonarqube.org/display/PLUG/SonarJS