POUZDANOST I KONTROLA KVALITETA SOFTVERA

**Inspekcija koda – Lista za provjeru**

**TIM**

|  |  |  |
| --- | --- | --- |
| **Broj greške** | **Ozbiljnost greške (veći broj = veća greška)** | **Opis** |
| General | | |
| 1. | 4 | Does the code work? Does it perform its intended function, the logic is correct etc. |
| 2. | 3 | Is there any redundant or duplicate code? |
| 3. | 2 | Can any global variables be replaced? |
| 4. | 1 | Is there any commented out code? |
| 5. | 2 | Can any logging or debugging code be removed? |
| 6. | 3 | Is the code as modular as possible? |
| 7. | 2 | Can any of the code be replaced with library functions? |
| 8. | 2 | Make sure that there shouldn't be any project warnings. |
| 9. | 3 | All unused usings need to be removed. |
| 10. | 4 | Is there any platform specific code? |
| Security | | |
| 11. | 3 | Are all data inputs checked (for the correct type, length, format, and range) and encoded? |
| 12. | 2 | Where third-party utilities are used, are returning errors being caught? |
| 13. | 2 | Are output values checked and encoded? |
| 14. | 2 | Are invalid parameter values handled? |
| Documentation | | |
| 15. | 2 | Do comments exist and describe the intent of the code? |
| 16. | 2 | Are all functions commented? |
| 17. | 3 | Is any unusual behavior or edge-case handling described? |
| 18. | 2 | Is the use and function of third-party libraries documented? |
| 19. | 2 | Are data structures and units of measurement explained? |
| 20. | 4 | Is there any incomplete code? If so, should it be removed or flagged with a suitable marker like ‘TODO’? |
| Testing | | |
| 22. | 2 | Is the code testable? i.e. don’t add too many or hide dependencies, unable to initialize objects, test frameworks can use methods etc. |
| 23. | 3 | Do unit tests actually test that the code is performing the intended functionality? |
| 24. | 4 | Are arrays checked for ‘out-of-bound’ errors? |
| 25. | 2 | Could any test code be replaced with the use of an existing API? |
| Data use and control | | |
| 26. | 2 | Minimized use of global variables |
| 27. | 1 | Variables – declarations with the smallest scope possible |
| 28. | 2 | Variables – declared with a specific type (the smallest type appropriate for the data) |
| 29. | 3 | Variables – clear names to identify the use |
| 30. | 2 | Variables - Data comparisons of the same type |
| 31. | 2 | Magic numbers avoided using constants and macros |
| Execution control | | |
| 32. | 3 | Do functions return proper values ? |
| 33. | 4 | Recursive functions – are there boundary safeguards? |
| 34. | 2 | Modularization use – no deep nesting of control statements. |
| 35. | 3 | Comparisons – proper bracketed evaluations (ensure right order of priority) |
| 36. | 3 | Do loops have a set length and correct termination conditions? |
| 37. | 3 | Are loops optimized? |