

System Verification and Validation Plan Checklist

Spencer Smith

November 16, 2025

- Follows writing checklist (full checklist provided in a separate document)
 - L^AT_EX points
 - Structure
 - Spelling, grammar, attention to detail
 - Avoid low information content phrases
 - Writing style
- Follows the template, all parts present
 - Table of contents
 - Pages are numbered
 - Revision history included for major revisions
 - Sections from template are all present
 - Symbolic constants are used rather than “magic” numbers. Symbolic constants are used to improve maintainability and to increase understandability
 - Specific values are provided for all symbolic constants
- Grammar, spelling, presentation
 - No spelling mistakes (use a spell checker!)

- No grammar mistakes (review, ask someone else to review (at least a few sections))
- Paragraphs are structured well (clear topic sentence, cohesive)
- Paragraphs are concise (not wordy)
- No Low Information Content (LIC) phrases ([List of LIC phrases](#))
- All hyperlinks work
- Every figure has a caption
- Every table has a heading
- Symbolic names are used for quantities, rather than literal values
- LaTeX
 - Template comments do not show in the pdf version, either by removing them, or by turning them off.
 - References and labels are used so that maintenance is feasible
- Overall qualities of documentation
 - Test cases include SPECIFIC input
 - Test cases include EXPLICIT output
 - Description over specification, when appropriate
 - Plans for what to do with description data (performance, usability, etc). This may involve saying what plots will be generated.
 - Plans to quantify error for scalar values using relative error
 - Plans to quantify error for vector and matrix values using a norm of an error vector (matrix)
 - Plans are feasible (can be accomplished with resources available)
 - Plans are ambitious enough for an A+ effort
 - Survey questions for usability survey are in an Appendix (if appropriate)
 - Specific plans for task based inspection, if appropriate (not just saying inspection will be done, but details on how)

- Provided adequate detail on non-dynamic testing. Statements like “We will perform a code walkthrough with our stakeholders” are accompanied by details, such as a checklist of items to go through during a walkthrough.
- Very careful use of random testing
- Specific programming language is listed
- Specific linter tool is listed (if appropriate)
- Specific coding standard is given
- Specific unit testing framework is given
- Investigation of code coverage measuring tools
- Specific plans for Continuous Integration (CI), or an explanation that CI is not being done and why not
- Specific performance measuring tools listed (like Valgrind), if appropriate
- If you are referencing an outside standard like the Web Content Accessibility Guidelines ([WCAG](#)), refer back to it when talking about it. Don’t just say “perform WCAG checks to validate accessibility” – say what tests you are planning on performing. If they are provided by WCAG, reference the specific tests you’d like to use.
- Traceability between test cases and requirements is summarized (likely in a table). The traceability matrix shows a test case for each requirement, or a non-dynamic technique is used for that requirement.
-

- Avenue rubric

- More than 5 peer review issues created for another team
- Should have enough redundancy in testing. Ideally there should be more than one approach for verification for each requirement.
- Extras should be clearly identified and should be feasible. The TA should have enough information to be able to provide feedback.
- A case should be made for why the extras will improve the project, and thus prove that they are not an afterthought.