Capstone Final Report Outline

2025/04/03

# Overview

Your final report is the top of a giant pile of documentation, code, CAD, blood, sweat and tears. It concisely describes what you were given, what you did, what the results were, how to replicate what you did, and where you think the project should go from here.

Let me emphasize the part about “how to replicate what you did”. Part of this final report is to allow your industry sponsor and future project teams to reproduce what you’ve done. If they can’t easily reproduce what you’ve done based on this document, then you delivered a dead project that no one will be able to use. That’s not a good look.

# Audience

Your audience for this document is two-fold: your industry advisor, and the next team that’s going to take over this project. A good model for the next team is your team back in December: technically competent, but doesn’t know anything about this particular project or technology. You’re going to kickstart that team so they can grab what you did and extend the project with a minimal ramp-up time. In particular, you want to justify the design, include the dead ends that they shouldn’t pursue, and in general, try to answer all of their questions before they ask them.

# Writing Style

Please don't use the passive voice. Your document should be written directly and concisely. Some of the best final reports are not more than a dozen pages long. There's no reason to sound "fancy" or "scientific", which are code words for convoluted, passive, hard-to-read sentences that people think make you sound smart. Use direct, short, active sentences. Use "we" and "our". Again, write this to yourself back in December: be descriptive, be frank, be real. And don’t hold back on failures! Failures are some of the most important information the next team is going to need to know.

# Formatting

All the things you should already know to do for a formal document:

* Sane margins and a cohesive font choice.
* Title, author, date, version number.
* Number the pages. Best is a format like "i/j" pages so the reader knows where they are.
* A table of contents (automatically generated, of course) if there are more than about ~4 pages, or there are a lot of sections.

# Copy Early, Copy Often

*Most of this document should already be written in other documents!* Your job is to pull all of your existing documentation together into one cohesive document. Copy as much as you can from the PDS, from WPRs, from presentations you did, from your poster, from your sponsor, from the internet, etc. That said, do cite sources if you copy from someone else who isn’t you or the sponsor.

# Important!

The following outline and description is a template only. Parts of this template may not apply to your project. Feel free to delete sections, add others, whatever you need to do to concisely and cogently summarize your project.

# Outline

* **Front Matter**
  + Title Page
    - Include project title, sponsor organization and individuals, team number and full names of team members, faculty advisor
  + Contact Information
    - Not required, but if you feel so inclined, give them your non-PSU email address so they can contact you with questions. Sometimes, a short 5 minute conversation will save months of time for the next team.
  + Table of Contents
* **Executive Summary**
  + A one page or less summary of this project, understandable by a person without an engineering degree. Overview of the sponsor, what the problem is to solve or what to build, brief project deliverables, your approach, and your final results including successes and failures.
* **Background**
  + Commercial, technical, scientific, and other domain specific background information you need. You can probably copy this from your sponsor and from your PDS. Include useful background, like links to introductory videos, similar systems, block diagrams, pictures, etc. This background should be comprehensive enough such that the next team has their research and ramp up all laid out for them.
* **Requirements**
  + What your sponsor asked you to deliver. Brief concept of operations for your project. List of stakeholders. Requirements. Specifications. User stories. Again, take this from your PDS.
  + If your requirements were a moving target, record that moving target here. Start with the initial requirements, discuss why and how things changed, and be sure to list the final requirements.
* **Objectives and Deliverables**
  + Now take those requirements and give us a *concise* description of the specific objective of this project and what your sponsor wanted delivered. At the end of this project, what did you and your sponsor *want* to have?
  + Did this change too? Tell the audience how it changed.
* **Approach**
  + You may or may not want to have this section. It should describe how you approached the problem, how you broke it up, the actual phases of the project.
  + This might include a brief "history of this project" - the surprises, twists and turns, and big changes in what your team did.
* **Existing Design**
  + You may or may not want to have this section. Don’t include it if this is a clean sheet design.
  + If it’s a revision of an existing product / system, then you should probably include what was done before you started your version of it. Emphasize what worked, and what didn’t, on that design.
  + In particular, the reader must be able to figure out what existed before your capstone, and what you changed/added.
* **Design**
  + OK, now we’re to the heart of what you actually did!
    - This should be your final design. You can talk about what changed, but you should present what you ended up with.
  + Have sections for each major subsystem including hardware subsystems, firmware, software, data mode, servers, etc.
  + Include block diagrams, UML views, models, use-cases, schematics, layouts, flow charts, server configuration, and detailed discussions on each subsystem.
  + Describe your design with enough details that someone reading this section will (1) understand why you designed the system the way you did and (2) will dramatically help them understand the rationale of your schematic, firmware, and software implementations.
* **Test Plan**
  + Describe how you tested the system to prove that it does what you expected it to do. This might just reference an actual test plan in an appendix, or if the test plan is short enough, you could include it here.
* **Results**
  + Concise but detailed results of your project. Deliverables achieved and not achieved. Success you had, and failures you had. Tests and measurements performed. How well did it meet stated requirements? Be quantitative when you can be, and present data appropriately (tables, graphs, etc.).
  + Some of this might be repeated from the Design section.
* **Post Mortem**
  + What worked?
  + What was great?
  + What didn’t?
  + What sucked?
  + What would you do differently?
  + What do you wish you would have known back in December? (Don’t be snarky - let the audience know what you wished you would have known)
  + Review the ECE 411 post-mortem slides in the [project management lecture](https://docs.google.com/presentation/d/19cgQAIzVydDGRDL_uwVzIHm2QU9o44Gwfhlh4pQGuOI/edit#slide=id.p33). In particular, don’t mope, or try to point the blame, about what didn’t work. Explain why it didn’t work, and move on.
  + **What would you do if you had more time?**
  + **What do you think the next team should do?**
* **Project Resources**
  + Links to collaboration sites and repositories with a description of where to get all major parts of the projects (schematics, firmware, software, configurations, etc).
  + List of tools you used (with versions!!)
  + Other resources (chat? File storage? Test equipment you used? Anything to point the way for future teams).
  + Bills of materials (BOMs) might be handy here, both for your project and for anything around your project (tooling, for example).
  + Include your budget here if you have one.
* **Conclusion**
  + OK, wrap it all up. A short section summarizing your entire capstone.
* **Appendices**
  + **CRITICAL: Tooling**
    - YOU ABSOLUTELY MUST include all hardware and software tools used, their version numbers, and your project’s specific configuration.
    - This includes step-by-step guide, if necessary, on how to install and configure the required tools if it’s not COMPLETELY obvious.
    - For very esoteric tools, consider delivering their installer/source code as a ZIP file when delivering your project files to your industry sponsor.
  + **CRITICAL: Developer’s Manual**
    - From the project developer’s point of view, you ABSOLUTELY MUST include detailed instructions on how to:
      * assemble,
      * program, and
      * test subsystems for proper operation (testing could optionally be in a test plan),
    - *The lack of any of these instructions makes the project not reproducible, which is essentially project failure. Do not skip these or suffer the wrath of Andrew and future capstone teams.*
  + **CRITICAL: User’s Manual**
    - You ABSOLUTELY MUST include detailed instructions on how to operate your system, from the system user’s point of view,
    - *Unless trivial, the lack of user instructions makes the project not reproducible, which is essentially project failure. Do not skip these or suffer the wrath of Andrew and future capstone teams.*
  + **CRITICAL: Test Plan**
    - This is where your full test plan, with results, should go, if the full plan didn’t fit in the main section of the final reports.
  + And then finally, normal appendices (whatever else is too big to fit in the main flow of the final report). For example:
    - You could, if you thought it was important, include WPRs / project notes.
    - Schematics and layouts are printed out in clear and pretty ways.
    - Firmware listings. Don’t bother if they’re too long; that’s what links are for.
    - Software listings. Don’t bother if they’re too long, that’s what links are for.