M.E. Systems Engineering Final Report

Notes for the SYSEN 5900 Quantitative Finance Project: Fall 2025

Overview

As Systems Engineering students in the second semester of the Software Systems for Quantitative Finance, your grade for the course this semester will also be your grade for your Final Project. That grade will be determined by:

1. Your contribution to a working software system that solves a real-problem in quantitative finance; and
2. A final report meeting the requirements outlined in the "Master of Engineering Systems Engineering Project Guidelines and Final Report Requirements."

We've used the weekly meetings to keep all students on track for the working software system part of the requirement. The purpose of this document is to provide some guidance on how to produce a successful final report.

Final Report Requirements In Context

This section will offer a few suggestions and reminders that may be helpful in meeting all of the requirements based on the unique nature of this project course.

1. ***An analysis of the context of the presented problem*** – A good treatment of this topic would help the reader understand who the user is, what they do, and why this tool is valuable. This section should introduce the reader to enough of the quantitative finance subject matter to understand why we built this tool. What is a portfolio manager's job? Why do we use measures like Sharpe Ratio and Information Ratio to measure the success of an investment strategy? What are factor portfolios and why are they potentially useful for a portfolio manager? In addition, this would be a good place to introduce some of the more practical details of the portfolio manager's job including market data, and the specific details of this business that make computer security so important.

2. **Well defined requirements that are agreed-upon to define a satisfactory solution to the problem** – Standard system engineering practice dictates requirements stated as verifiable "shall" statements. These overlap in purpose with the User Stories that we crafted throughout this project. We used demos as our main verification tool: If we could "act out" the user story in the demo, then we would say the feature was complete.

   Consider using this section to share some of the key user-stories, and showing how these

map to “shall” statements, and demos that define the verification of each completed feature. This may also be a good opportunity to introduce your general “Definition of Done” requirements as well.

3. **Determination of some possible functional definitions of a solution** – For the key user stories, we also considered different functional approaches to user-interaction. For example, with the Cayuga Fund project, we contemplated delivering a web-app, a desktop program, an Excel plugin, and a shared collab notebook. We had similar iterations around how our system would consume input data, provide results to the user, etc. Use-Case diagrams and Interaction Diagrams may be helpful in illustrating functional definitions.

4. **Determination of some possible structural ways of realizing the functions that are being proposed as alternatives** – Considering each of the different functional definitions led us to think about different structural solutions. For example, had we decided to deliver this as a web-app we would have needed to find some hardware to deploy that application and make it available to our users. Choosing an excel-plugin, or a desktop app, would have caused us to consider different structural decisions. In this section, consider ramifying structural solutions for a few different functional approaches we may have taken. Consider how hard each may have been to maintain, to secure, to provide user-friendly systems to our users, to provide a cost-effective solution, etc. Class Diagrams and Deployment diagrams may be helpful in illustrating structural design.

5. **Analysis and optimization** – Why did we choose the system we chose? What were the trade-offs? What did we gain? What did we give up? Use this section to describe the design and architecture of the final solution delivered. Use diagrams. Describe the evolution of your design over the different iterations – e.g. for the Cayuga Fund project, your system evolved from a single cell in a notebook to a full object-oriented system with unit-tests.

6. **Planning and carrying out implementation, testing, and development** – Consider using this section to describe how you used agile techniques to demonstrably deliver value in an iterative and incremental fashion. You may also want to discuss how you used the MVP mindset to get early feedback. You can also use this section to cover the more technical details of your process such as how you approached system testing, how you used tools like git and continuous integration to facilitate collaboration, etc.

FAQ

Is the Final Report a group or individual assignment?

Either is fine.

We may have worked on different projects over the two semesters. Do I need to cover both of them?

You can focus on just one project if that's easier. However there may be valuable lessons or experience you took from the other semester's, which you are encouraged to include.

Can I share unfinished drafts with the professor to get early feedback?

Yes. In fact I encourage you to do so. If there are any issues or misunderstandings, this will help us catch them in time for you to make revisions by the final deadline.

How You'll Get An A

1. Structure your report around the 6 points specified in "Masters of Engineering Systems Engineering Project Guidelines and Final Report Requirements". Make sure each point is well answered from the perspective of a Systems Engineer.

2. Answer sections 3 and 4 in a way that demonstrates a clear understanding of the difference between functional decisions (i.e. **What** are the things the system does?) and structural decisions (i.e. **How** does it do these things?)

3. **Use Diagrams!** Use UML to convey software systems concepts. In addition, use some of the other tools and techniques you've learned in your Systems Engineering courses (e.g. SYSEN 5100.)