15-110 F26 - Final Project Guide

The final project is an opportunity for you to take everything you've learned in 15-110 and apply it to a context that matches your own interests and needs. You will come up with an idea for a project, design a plan for how to create it, implement that plan using code, and reflect on how your implementation connects to the topics we've discussed throughout the semester.



You are allowed to use generative AI on this assignment. However, you should still make sure you understand everything that is produced and can explain it in your own words.

You should work on the final project through Weeks 10-14 of the course. Part 1 and Part 2 are due as parts of Hw6 and Hw7. Part 3 and 4 are submitted as a separate assignment due on the last day of classes via **Final Project - Implementation** and **Final Project - Reflection** on Gradescope. You will complete Part 5 in an oral assessment during the finals period.

Part 1: Ideate - due Monday 11/16
Part 2: Plan - due Monday 11/23
Part 3: Implement - due Friday 12/04
Part 4: Reflect - due Friday 12/04
Part 5: Discuss - due Monday 12/14

Part 1: Ideate - due Monday 11/16

First, come up with an idea for a Python project you'd like to design and implement. This can be something connected to the work you do in your major that would make a useful portfolio piece, or something to support one of your hobbies, or a useful tool for everyday life, or just something for fun!

You're encouraged to get creative with your project idea. The only requirements are:

- The idea must be something that will be implemented using Python code.
- The project must be complex enough to require 4+ functions of reasonable length that call each other / work together to support a central part of the project.
- There must be a clear input provided to the project (parameters, data, or user input) and a clear output produced (text, visual graphics, or files).
- The project cannot be something we have already done in class or something you built before taking this class.

If you're having trouble coming up with an idea, consider the following starting points:

- Find a data set online, come up with a few questions about it, perform data analysis to address those questions, and automatically generate a short report.
- Find a training data set online, use machine learning to build a model and test its accuracy, and create a small text interface where users can run new inputs through the model.
- Come up with a scenario you want to simulate, identify the important components and rules for that scenario, build an interactive simulation of the scenario, and use probabilistic methods to answer questions about the scenario.
- Think of an interesting variation to a game you like to play. Use Python to implement that modified game from scratch.
- Find a Python library (internal or external) that you think will be useful for you in the future. Browse the documentation and find a few functions that can be put together to make an interesting demo project for the library.

Once you've settled on your idea, write a short paragraph describing a) your project idea and b) what the project should look like if it's successful. When grading, we'll give you feedback on your idea that you can take into account before moving on to Part 2. You should submit your paragraph as part of Hw6.

Part 2: Plan - due Monday 11/23

Once you've finalized your project idea, make a step-by-step plan for how you will implement it. This should be similar to the high-level steps provided in the Mini-Projects and the project breakdowns we worked on in the Problem Decomposition lecture. You should plan to start with a single feature/step, build up to a minimal viable product, then add additional/optional features as desired.

Write up your sequence of steps with a short description of what will be done in each step and submit it as part of Hw7. We'll give you feedback on your plan and any adjustments you might want to make before you move to Part 3.

Part 3: Implement - due Friday 12/04

Once you've finalized your plan, you may start work implementing your project. You are welcome to either use generative AI to produce code or write the code yourself, whichever you prefer. While implementing the project, try to follow your plan as much as possible, though it's okay to make changes if it becomes clear that parts of the plan will not work. Make sure you follow the development principles we've learned this semester: work iteratively, read all code to make sure you understand it, and test to ensure code works correctly.

While you work on the implementation, be sure to track your **process.** When using generative AI, save all conversations with agents and export them into a PDF once done. For all work you do yourself (including testing), keep a log with notes about what you've done and any challenges/surprises you run into.

We will grade your implementation based on your process, not the final outcome. If you encounter a bug that you cannot fix despite reasonable debugging attempts or a feature you cannot implement despite numerous attempts, that's okay and we won't penalize you. Instead, we will grade your implementation based on the following:

- Evidence that you created the project through iterative development
- Evidence that you made a reasonable attempt to follow the step-by-step plan
- Evidence that you used testing to ensure correctness (both testing of individual functions and broader system testing of the whole project)
- If there are bugs / features that do not work, evidence that you used debugging to attempt to fix the problem(s)

When your implementation is done, put together a folder that contains the following files:

- **readme.txt -** a document that tells a user unfamiliar with your project how they can run it. Should include a list of any external libraries that need to be installed.
- **process -** a folder containing all your process documents. Should include:
    - PDFs of all your generative AI conversations related to the project (note that most tools let you export these automatically)
    - Logs of the notes you took to document your own process while working (including testing and debugging)
- **code** - a folder containing the codebase of the project. Should include:
    - All code files
    - Any supporting files needed to run the project (data, images, etc)

Compress these files into a single .zip file and submit them to the **Final Project - Implementation** assignment on Gradescope.

Part 4: Reflect - due Friday 12/04

While working on the implementation of the project, review the **Final Project - Reflection** document available on the Assignments page. Complete the problems in this reflection document to connect the work you're doing in the project to the topics we've discussed throughout the semester.

When you're finished, submit the PDF file to the **Final Project - Reflection** assignment on Gradescope.

Part 5: Discuss - due Monday 12/14

After you've submitted your implementation and reflection, sign up for a meeting slot on the Oral Assessment 3 page. In this meeting a course TA or instructor will talk to you about your project and ask you questions about the implementation, your process, and your reflection answers. The usual oral assessment rules apply - if you fail the oral assessment, your final project will automatically be marked as a 0 until you meet with a course instructor to discuss what went wrong in the assessment.

Note that you do not need to understand every single line of code in your project for this meeting (there may be a lot of code!). However, you should understand what each function is doing at a high level, and you should be prepared to answer questions about your process and reflection.