# Final Project for CPTN230

**Name:** Final Project.

**References:**

Any and All.

### Project Overview:

1. Create a project that does something “interesting.”
2. The project must be complex enough in features and techniques to demonstrate you mastery of the C++ topics covered in this course.

Sample Project Ideas.

* Create an inventory application of objects. You should be able to add, remove, alter and view the objects in the inventory.
* Create an appointment calendar application. You should be able to add, remove, alter and view the appointments in your appointment calendar.
* Create a sports team statistics application. You should be able to add, remove, alter and view team member statistics.
* Propose your own project.
* Create an application that demonstrates the use of at least 75% of the C++ topics covered in class. This is the default project for those that do not choose their own project. It’s boring but will be accepted.

You must work in teams. All project choices must be approved by the instructor.

1. The project must use a simplified Software Development Life Cycle (SDLC) approach for design and implementation using provided Microsoft Word Document Outlines.

In industry, application developers are implicitly expected to know how to use the basics of Microsoft Word, therefore Microsoft Word must be used for all SDLC documentation.

1. You must prepare a maximum 15 minute presentation with Microsoft Power Point slides. And a 2 minute presentation won’t cut it.

In industry, application developers are implicitly expected to know how to create simple Microsoft Power Point Presentations, therefore Microsoft Power Point must be used for the presentation.

1. Submit all deliverables to Blackboard via Completed/View Assignments Link.

### Project Deliverables:

1. Copy of all your C++ Program Source Code. This includes all objects, the applications and test driver.
2. A copy of all SDLC documents. These include:

* Project Plan
* Requirements Document
* Design Document
* Test Plan
* User Documentation

1. A copy of your Power Point Slides.

Do not zip these files. They must be individual file submissions via BB. Standardize all file names by following the conventions used in the assignments.

### Project Grading:

* Working Project 60%
* SDLC Documentation 25%
* Presentation 15%

### Strenuously Recommended Approach:

This section is a recommendation of how to complete the assignment. It is how I would do the assignment if it were assigned to me. You should use it as a step by step procedure for completing the assignment. It is developed from many years of operating system and application software development and lessons learned. It may seem like more work up front but it will save you time. Guaranteed!

1. Understand the Problem

You must know what you are building and the process by which you will build it. That means read this entire document before proceeding. You need to know what documents are needed, what tools you will need, as well as any other resources. This is the time to select your project and team. Again, all projects must be approved by the instructor prior to continuing. In fact, unless otherwise stated, you should not move on to a step until all previous steps have been completed.

1. Gather your resources

Decide what computer you will use for project development. Create a separate directory for the project. You may want to consider separate sub directories for the code, documentation and presentation.

Insure that you have the required software development tools on that system. These include, but are not limited to, compilers, runtime environments and Microsoft Word and Power Point. The computers in Room 171 do have the required software.

Download the 5 Word Document Outlines for the 5 required SDLC documents.

1. Create and Start the Project Plan

Notice this step is “Create and Start” the Project Plan. A Project Plan is a living document that gets updated as the project progresses. You will periodically come back to this step as the project becomes more defined and as implementation and testing progress. The Project Plan Outline lists topics that are often included in a Project Plan. For this project, to save time, only fill in those sections discussed in class.

1. Create the Requirements Document

Note, this does not read “Create and Start” as it did for the Project Plan. This document should be completed in one pass. You cannot know how to design, implement and test a product if you don’t know what it does! As the project progresses, you may need to make some minor refinements to this document but no significant changes. The Requirements Document Outline lists topics that are often included in a Requirements Document. For this project, to save time, only fill in those sections discussed in class.

1. Create the Design Document

This step is the same as the previous one except it applies to the Design Document. Make sure you provide the algorithm for every feature listed in the Requirements Document.

1. Create the Test Plan

Again, this document should be created in one pass. The “black box” testing is created from the Requirements Document and will later serve as the basis for a lot of the User Documentation and Presentation. The “white box” testing is implemented by the creation of a Test Driver program for one of your objects. For this class you will need to document “black” and “white” box testing.

1. Create the Application and Test Driver

This is not a trivial step but is made much easier by completing the previous steps. Trust me!

Create at least one support class and a test driver to test it. Verify the support class works. If you create more than one support class you only need to provide a test driver for one of them.

For the real application, implement one of the features listed in the Requirements Document and verify (test) that it works. When it does, then implement the next feature from the Requirements Document and verify it works as well as the previous feature. Continue this process until all features have been implemented and verified. Don’t miss any features and do not add any features. Only implement the features listed in the Requirements Document. A good way to do this is to copy and paste the algorithms from the Design Document into your source code and make them the comments in the source code. This insures the code follows the design and that all features are implemented without missing any or adding any extras.

1. Test the application

Although listed as a separate step, this step is often done in parallel with the previous step. For this project, I suggest you follow this procedure.

1. Create the User Documentation

As before, the User Documentation Outline lists topics that are often found in User Document. For this project, to save time, only fill in those sections discussed in class. The “how to use” section is often inspired by the “black” box testing listed in the Test Plan.

1. Create the Presentation

This is done in Microsoft Power Point and must address the following topics:

* Introduction. Describe the features of the Application.
* Demonstrate the Application. The presentation demo is usually taken from the “how to use” section of the Users Documentation.
* Lessons learned. What went wrong with the project and why? What went well and why?

The presentation should be about 15 minutes per team. But absolutely, no longer!

Rehearse (out loud) the presentation in the forum (classroom) with the system in the classroom that you will actually use for the real presentation.

1. Synchronize all documents, code and the presentation for consistency.

This is a final sanity check that all the documents, code and the presentation are in sync with each other. This should not require any major changes to anything but is to ensure the details mesh. Minor refinement of the documents, presentation and code comments is all that should be required.

1. Submit the deliverables

Submit all the deliverables via the “View/Complete Assignment” link. Again, do not compress or zip files together. Only one set of submissions per team is required.