Final Project Instructions

John Brown University Engineering

CS3743 Advanced Computer Programming, Fall 2024

Document revision date: 10/31/2024

Summary:

In this project you will have a chance to build a real-world software engineering project, related to an area that you are personally interested in. Please discuss with the instructor before starting, in order to validate the complexity of your idea.

Timeline:

19-Nov - Final Project Outline due (please consult with instructor BEFORE this date)

10-Dec - Progress report due (1 page, listing which parts of project are working or not working yet, and what specific problems are being encountered)

17-Dec - Final Project Presentations and Final Report due (both are due at the time of the presentations)

Program/code requirements:

* Preferably written in C# / .NET (or get special instructor permission for other language)
* Must have some "real-world" connection or applicability or usefulness
* Must have a GUI of some type - desktop, mobile app, web app, etc
* Must involve some processing of some type of "data" (doesn’t need to be numerical data though)
* Must include at least one custom class, other than the driver class that holds the “main()” function and the Test (TDD) classes
* You must use source control (Bitbucket, Github, etc).
* (See also Report Requirements below)

Report Requirements:

Your Project Report should include the following four sections, in addition to your full computer code.

1. Introduction – explain why this project is useful / important in the real world, and what aspects of computer programming it demonstrates.
2. Methods – explain the overall functioning of your code. Explain any special or tricky parts of your code. If you borrowed any code from the Internet, cite your sources.
   * You must include a set of UML class diagrams for all the custom classes you write. You may also include other UML diagrams as appropriate… state diagrams, activity diagrams (flowcharts) timing diagrams, sequence diagrams, etc.
3. Results – explain your testing plan. Which combinations of inputs did you test? (You will also need to demonstrate the working program to the instructor). If any parts of your code were still not working, explain this.
   * You must use unit tests (e.g. Visual Studio unit tests) as part of the testing of your final code.
4. Conclusions – What did you learn from this project? What were the hardest parts of the code and integration? What would you do differently next time? What suggestions do you have for improving the project for next year?

Final Project Presentation:

Your Final Project Presentation should be 5-6 minutes. It should use 4 – 15 Powerpoint slides, which should be emailed to me before that morning or brought on a flash drive. Since time is limited, you may or may not have time to run a quick demo of your project during your presentation. You will not have time to log in to your account, so you can bring it on a flash drive or email it – but test this beforehand to make sure it works. Either way, it would be great to have (as a backup?) some slides with photos of your project in action, or a short video clip. We will need to start exactly on time and carefully stick to the schedule to make sure that the final teams do not run out of presentation time. Therefore, please practice your presentation beforehand. In order to make a good presentation, plan to stop working on your project one or two days beforehand, allowing you to put time into a nice Powerpoint presentation.

I will looking for the following **four things** in your presentations:

* A very brief, clear, simple, overall description of your project.
* A brief overview of the system design of your project. One strategy that is often helpful is to start at a high level, then “zoom in” “down” to more detailed levels of the project, one at a time. Also remember that pictures and diagrams are often better than text for technical presentations (make the pictures/diagrams as large as possible on the screen). When you need to use text, make sure it is bulleted and that the font size is large enough.
* A discussion of what features you were able to get working, and what features you were not able to get working. Explain (or demonstrate) how you tested your program. Explain (to the best of your knowledge) why the non-working parts are not working.
* Finally describe what things you have learned overall during your project… for example, things about C#/.NET, about engineering design, about the software design process, about Visual Studio, about project scheduling and team coding, etc. What would you do differently if you were doing this project over again?

Suggestions of project ideas (this list is **confidential** to this class – please do not share outside this class)

* combo project if also taking EE3323 or EE3123 or another CS class (e.g. DE2-based oscilloscope? Or MSP430-based?) or EE4413 or for EN4323 (controls)
* Machine-learning / Artificial-Intelligence / Deep-Learning type of project, with some kind of ML C# library like ML.NET <https://dotnet.microsoft.com/apps/machinelearning-ai/ml-dotnet>
* video stabilization program: read-in .avi, estimate motion between frames, correct single-frame motion using a long-term-moving-average motion estimate to produce a smoothed video
* steganography program?, using a local GUI (jpg-in, jpg-out) AND either web programming or mobile app
* - a project related to a summer internship that would benefit that company (but beware of confidentiality issues)
* a significant expansion of one of our smaller course projects… especially a demonstration of graph algorithms or dynamic programming or other algorithmic concepts
* computing "related" documents, as in online ad bidding (for Google ads) or Amazon "related products"
* "statistically-improbable phrases" (use a set of nonfiction books from Gutenberg, leave-one-out and find the SIP's?)
* sort mp3 songs by "average volume" (in one thread) while other threads watch a 'cancel' button, display progress, etc... then when finished, can choose a 'quiet' song (from lower quartile) or 'loud' song, etc... advanced: sort by beat speed
* 'where's waldo' correlation-matching in image (possibly too simple by itself)
* K-Nearest-Neighbor pattern-recognition? Eigenface face-recognition?
* Computer game, using Unity and C# ?
* DNA-similiarity matching/ranking & searching? (if you do this project well, it could easily land you a bioinformatics job after graduating)
* JBU-campus "emotional-state-estimator/predictor" - a program that trawls all available social-media associated with current JBU students (Facebook/Twitter/etc) and computes a metric based on classifying all the posts, about the emotional state of the campus. This could be used for chapel speakers to tailor their words, for cafeteria workers to figure out how much food to cook, for counseling workers, etc. It could also compute a 'JBU-geist'/'trending' indicator which lists the 5 topics of greatest current interest, and incorporate various sources of data (predicted weather, JBU sports scores, the number of coughs/sneezes detected by a smart-microphone system in chapel and the cafeteria, etc) to predict what the emotional state of the campus tomorrow will be.
* invent a faster way (using a custom app) to type or speak text into a smartphone for SMS messaging ("texting")
* a program that would help handicapped or injured or disabled people in some way…?
* an educational program for children?
* image deblurring program, using a blur estimate to deconvolve and sharpen an input image – this would work great with a GUI !
* read in depth map from a Kinect, implement a game or other GUI-control based on gestures or actions
* Audio-processing program, such as adding audio effects like echos/reverbs/stereo?
* Help Dr Gilmour with some aspect of his de-mining robot project