CST8002 Programming Language Research Project

# Practical Project Part 4 – Project Release – See Brightspace for due dates

* Refer to the Weekly Schedule document posted in Brightspace under Course Information for additional requirements common to all assignments as well as details on the required use of the data set specified for the course.
* Make reference to the documents in the dataset content area in Brightspace for the dataset to use, note that all columns need to be used unless your professor has indicated otherwise, the dataset column names must be used in your source code as variable names as well to verify you are using the dataset provided.
* The first record in the data set may contain the column names, if so, you may skip over this record when reading in and parsing the data set data.
* You may use your previous work in this course as a reference / starting point but I expect modifications, i.e. passing in older work again with none to very small changes will not earn marks.

## Tasks

### Version Control

* Using your GitHub account:
  + You should have already created private remote repository for your project development over the semester, and have given the repository a name like “CST8002\_PracticalProject\_Section\_YourName”
    - The section would be your lab section, e.g. 350, 360 and so on.
    - YourName would be your name as it appears in ACSIS or in Brightspace.
  + If you have not done so already, invite your professor to your repository as a collaborator, so they can review your commit history as well as clone your repository for marking purposes. See the contact information area in Brightspace for your professor’s name and email address.
* As you work on your project you are expected to follow an iterative development process, making commits that have brief, but detailed comments explaining what program feature or functionality was achieved. These will correspond to tasks / requirements below.
* For Practical Project 4 create a new branch called Project4, based off your branch from project3. See [1]
  + E.g., git branch project4
  + E.g., git checkout project4
* Tag your last commit for practical project 4 as V4.0. [2]

### Program Functionality

* Modify your project to add additional functionality by utilizing the data set data to offer a new novel project feature to the user, **Choose one of**

1. Visualization (Charts) one of:
   * Horizontal Bar Chart
   * Vertical Bar Chart
   * Pie Chart
   * You can work on a different visualization, with written permission from the professor. Contact your professor first.
2. Search or Filter records based on multiple columns at the same time.
3. Sort records based on multiple columns at the same time.

The user must be able to interact with the program at run-time to customize the output, for example what parts of the data to chart or perform a search or sort on.

Console programs can fall back on generating ASCII Art versions of charts.

Web Service projects can return a formatted result that can be either viewed as ASCII art in a testing program or used by a client to render results in the case of charts.

You can figure out the logic and write the code on your own or reference a web resource and adopt it to your own use or use a framework either part of the language API or a free third-party alternative (no paid packages). I.e. Your professor is not concerned with how you make it work, only that you make it work, that you can explain how it works, and that you documented learning resources and / or code library licensing using IEEE reference style with both citations and references in both your source code and your report.

Create a commit on each step in refactoring your code to add the novel feature selected, example steps would be to modify the presentation or view program logic, modify the business layer or controller logic, modify the persistence layer logic, and documenting the code changes.

**Warning**

* **You must integrate the novel feature you select above into your existing project as it was developed through Practical Project Parts 1, 2 and 3 retaining the previous Projects’ functionality. A new stand-alone program that focuses on just this new feature will receive a mark of zero. If you did not fully complete the earlier practical projects seek guidance from your professor.**
* **There is a requirement for the novel feature to be separated out into smaller parts and incorporated appropriately into the layers or parts of MVC in your application. Do not write a single method that handles user interaction, data processing, and display. Refer to Hybrid 05 for details.**

## Your single MS Word document for your Practical Project 4 report should have this general format

* Cover page with your full name within it, the content of your report should start on page 2.
* Heading with name “Evidence of Learning”
  + Code figures to display the implementation of the Practical Project Part 4 novel feature you selected.
  + Then Either:
    - Use small code examples for each topic taken from your project with brief descriptive text, or
    - Indicate what line numbers in a larger code sample (including the file name) illustrate each concept within brief descriptive text (line numbers must be present in the code examples as well to match).
    - **You must indicate clearly to your professor that you can identify what parts of your code illustrate and match each programming concept.**
    - Use small code examples for the topic, or
    - Indicate what line numbers in a larger code sample illustrate the concept.
    - **You must indicate clearly to your professor that you can identify what parts of your code illustrate and match it to the programming concept.**
* Heading with name “Program Changes”
  + You need to outline where you modified your existing project code to add the novel feature’s logic, as well as the user interface.
  + You are not required to provide any UML diagrams for practical project 4.
  + You are not required to write a unit test for your novel feature.
* Heading with name “Program Demonstration via Screen Shots”
  + Include screen shots of your running program to showcase the novel feature.
  + Provide brief descriptions using one to two sentences for each screen shot.
* Heading with name “Source Code Commenting Example”
  + Copy and paste at least **source code file(s)** from your project to demonstrate you can write programming comments using documentation-comments. Use a font size of 10 point, with a monospaced font of your choosing.
  + This should be the file(s) that has (have) your novel feature logic and user interface changes implemented.
* Heading with name “References”
  + Use IEEE reference style for learning resources related to your selected program feature.
* Do not copy and paste code from the web or AI agents into your demonstration program, it must be your own work. In other words, even properly cited and referenced code copied from a website will not earn marks, as you must provide your own work. Your professor, while grading your submission will make this determination.

## Demonstration in Lab Period

* The week following the due date, you will be expected to demonstrate your program to your lab professor in the lab period.
* You will be asked to run the program to show that it works, then will be asked to answer two questions. These may be either theory and / or a request to modify your code.
* Sample Theory Questions
  + Why did you choose to author all the code and logic for your novel feature?
  + Why did you choose an API or Library for your novel feature?
  + Similar questions on theory knowledge are possible at the discretion of the lab professor.
* Sample Code Modification Questions
  + You may be asked to modify your program logic so that the appearance of the generated visualization, or behavior of the search or sort is altered.
  + Similar questions on modifying the code are possible at the discretion of the lab professor.

## Submission Requirements

* Upload your MS Word document by the due date to Brightspace.
* Compress your local project repository into a zip file and upload this to Brightspace as well.
* Provide a plain-text link to your project GitHub repository in the comment section of your Brightspace submission. You should have already invited your professor as a collaborator when you started working on this assessment.
* Demonstrate your practical project to your lab professor in the lab period following the submission due date.
* **While the demonstration carries no direct marks, missing the demonstration will result in a zero for this assignment. Additionally, if you cannot answer simple questions to demonstrate concepts in this assignment to your lab professor you will lose up to 100% of the score on this assignment. Please refer to the Course Outline under Department Information.**

## Grading (Total 18 Points)

**Note: There is no entry in this rubric for a cover page with your full name in it, however a mark deduction of 3 points will be applied if you do not have a cover page in your MS Word document.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Criteria | Poor/Missing (0) | Below Expectations (1) | Meets Expectations (2) | Exceeds Expectations (3) |
| Evidence of Learning | Poor/Missing or program does not use the dataset as required, or program is not using correct data set. IEEE citation and reference are missing.  Project is not based on practical project 3 and / or does not have the features developed through practical projects 2 and 3. | Novel feature does not work, or project no longer builds and runs. Reference is incomplete, and / or not in IEEE Reference style. Code figures and / or descriptive text and / or figure-text are missing for several items. | Novel feature is present, and project builds and runs, but is not fully functional, i.e. it does not work to produce correct results or does not meet expectations as specified. Reference is complete, but loosely patterned on IEEE reference style. Code figures and / or descriptive text and / or figure-text are missing for some items. | Novel feature is complete, project is functional, and working correctly and meets or exceeds specified expectations. Reference is complete and formatted using IEEE reference style. Code figures and / or descriptive text and / or figure-text present for all items. |
| Program Changes | Learning topic was not incorporated into the software by refactoring existing code but is a stand-alone proof of concept or is an attached showcase item. Student does not outline where the novel feature logic is located, or program is not using correct data set. Project is not based on practical project 3 and / or does not have the features developed through practical projects 2 and 3. | Vague explanation of where the novel feature logic occurs in the source code. | Student discusses where the novel feature logic occurs within the software. | Student also discusses how layered architecture, or alternative architecture, has helped with refactoring and incorporation of novel feature logic. Provides an overview of what architecture modules needed to be modified. |
| Screen Shots, Running Program | Poor/Missing  E.g. missing full name from all images, image file not within the MS Word document. Screen shot does not demonstrate use of the dataset. Screen shot does not show or indicate new novel feature, or program is not using correct data set. Project is not based on practical project 3 and / or does not have the features developed through practical projects 2 and 3. | Screen shots are within MS Word document. No explanation of the image except for headings or subheadings. Has partial name or nickname or but full name missing from some screen shots. Some project functionality missing or program crashes. The requested novel feature is incomplete. | Screen shots are within MS Word document. Student provides brief generalized description of most images using one to two sentences. Has full name in nearly all screen shots. Most of the requested novel feature is working. | Screen shots are within MS Word document. Student provides brief yet detailed description using one to sentences for each image. Has full name in all screen shots. New novel project feature is documented and working fully. |
| Source Code, programmer comments. | Poor/Missing  E.g. missing full name as programmer comment at the top of the file as author of the file or program is not using correct data set. Project is not based on practical project 3 and / or does not have the features developed through practical projects 2 and 3. | Student uses minimal comments in source code, e.g. the student does not comment (m)any class members. | Student comments some class and class members, however does not use documentation comments. | Student uses documentation comments in an accepted coding style specific to their language of study. If the language does not support documentation comments student provides evidence of this from reputable source, yet still comments code following best practices. |
| Version Control | Poor / Missing or there are no incremental commits in the commit history or program is not using correct data set. Project is not based on practical project 3 and / or does not have the features developed through practical projects 2 and 3. | Student project development shows the use of a few commits; however, commits do not have a detailed description or no description. Not all the requested commits are present. | Student project has regular commits providing evidence of incremental builds, not all commits briefly overview what was changed. Missing at most one of the requested commits. | Student project has regular commits providing evidence of incremental builds, each commit details what was changed. Requested commits are present. | |
| Source Code and Source Code files | Poor / Missing or program is not using correct data set. The professor was not given access to the remote GitHub repository or there is no repository. Project is not based on practical project 3 and / or does not have the features developed through practical projects 2 and 3. | Student submitted one of: MS Word document, remote repository link, or zip file with project source code. | Student submitted two of: MS Word document, remote repository link, or zip file with project source code. | Student submitted all requested parts, and professor was added as collaborator to remote repository. | |

# Recommended Resources / Sources Cited

[1] git (n.d.) 2.6 Git Basics – Tagging. git-scm.com. [online] Available at: <https://git-scm.com/book/en/v2/Git-Basics-Tagging> [Accessed on Nov 10, 2024]

[2] git (n.d.) 3.1 Git Branching – Branches in a Nutshell. [online] Available at: <https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell> [Accessed on April 24, 2025]

Download GitHub Desktop. (n.d.). desktop.github.com. [online] Available at: <https://desktop.github.com/download/> [Accessed on Nov 10, 2024]

## Additional Notes

Video Game Software projects are not acceptable in this course.

Your source code within the MS Word document should match the code in your source code files in your repository, this includes the programmer comments. If there are large or many differences, you will lose marks.

If you do not provide your GitHub link and include your professor as a collaborator so they can review your commit history and source code files you will receive a score of zero for this assessment.

Not using the correct data set for your course section will result in a zero for this assessment, note that a Canadian Cheeses dataset if present was for demonstration purposes only and is not the correct data set for your project.