CST8002 Programming Language Research Project

# Hybrid 05 - Working with Layered Architecture / MVC Pattern

**See Brightspace for Due Date**

**Refer to the Weekly Schedule document posted in Brightspace under Course Information for additional requirements common to all assessments.**

## Overview

* For a review of Layered Architecture, and/or the Model View Controller (MVC) Design Pattern, please refer to Hybrid 03 of this course.
* Practical Project 4 will ask you to add a new feature to your project, one of Visualization, or searching / sorting / filtering across multiple columns at the same time on the dataset data.
* Where to place the logic for the new feature within the layers is an important choice.

## Design Considerations with Layered Architecture / MVC Design Pattern

* Both Layered Architecture and MVC Design Pattern support the concept of Separation of Concerns in software design [1].
* Review the article [1] using this link (required reading):  
  <https://nalexn.github.io/separation-of-concerns/>
* When you implement the new feature, you may need to make changes to code files in more than one layer to implement the feature correctly.
* Separate the feature out into parts: user interaction(s), manipulating data or model, reading or writing to data store. Then modify the presentation layer (or view), business layer (or controller + model), or persistence layer (or controller).
* For example, if implementing a visualization like a chart, the chart should be placed on screen via program logic within the presentation layer (not the business or persistence layer). The presentation layer may need summarized or aggregate data for the chart but generating this data should be done in the business layer or even in the persistence layer if a database query is to be used. The lower layers would supply the data upwards to the presentation layer which generates the chart.
* For multiple column searching or sorting, the presentation layer would interact with the user to request what columns they want to manipulate, then a request for the sorting or searching would be made to the business layer which may use a library to manipulate the data in memory, or the business layer may make a request for the sorting or searching to the persistence layer which uses queries against the database to generate the results. The data would then be passed upwards through the business layer to the presentation layer which displays the results on screen.

## Examples of what **not to do**

* Create one method that does all the parts…
* Persistence or Business Layer creates and shows a chart on the screen directly
* Persistence or Business Layer interacts with the user requesting the columns they want to sort on.
* Presentation Layer opens its own connection to the database to run the SQL queries.
* Business Layer interacts with the user and / or shows charts on screen.
* Model or Controller interact with the user directly without using the View.
* In other words, placing programming logic into a layer or part of MVC that does not have that role.

## What is the Bigger Picture?

* The Layered Architecture and the Model View Controller Design Pattern help us to plan and design our software for deployment as physically separate modules, on separate computers in a client-server environment. (Tiered Architecture, see [2][3][4] for overviews).
* Placing the program logic into the wrong layer will make it nearly impossible to easily physically separate the layers into Tiers. For example, if the persistence layer is running on a remote server there is no user to interact with the program in the terminal, and there would be no user to view a chart (or close it) should it appear on the server’s screen.
* Many popular web MVC frameworks are distributed implementations, the Controller and Model reside on the web server, and the View is the remote web browser. One of the challenges of generating and showing charts to the user in the web browser is getting the chart to them either as an image which is generated and temporarily saved on the servers hard drive with the View (browser) calling back to the server to fetch the image file, or a client-side JavaScript framework is used to generate the chart based on data requested from the server.

## Grading (10 Points Total)

* Hybrid Quiz 05 – 10 questions, 1 point per question.

# Resources / Sources Cited

[1] Alexey Naumov. (Jan 16, 2020). Separation of Concerns in Software Design. nalexn.github.io. [online] Available at: <https://nalexn.github.io/separation-of-concerns/> [Accessed on Dec 19, 2024]

[2] IBM. (n.d.). What is three-tier architecture? [www.ibm.com](http://www.ibm.com). [online] Available at <https://www.ibm.com/topics/three-tier-architecture> [Accessed on Dec 19, 2024]

[3] Wikipedia. (Oct 10, 2024). Multitier Architecture. en.wikipedia.org. [online] Available at <https://en.wikipedia.org/wiki/Multitier_architecture> [Accessed on Dec 19, 2024]

[4] Appsierra. (Apr 01, 2024). Software Architecture: N Tier, 3 Tier, 1 Tier, 2 Tier Architecture. www.appsierra.com. Available at <https://www.appsierra.com/blog/tiers-in-software-architecture> [Accessed on Dec 19, 2024]