7-2 Final Project Submission

Daniel Clark  
CS 499  
8/10/22

*GitHub e-Portfolio link:*

*Self-Assessment:*

Hello, my name is Daniel Clark, and I am in the final year of the Computer Science program at Southern New Hampshire University. In the following sections, I will be reviewing artifacts and enhancements made for an application that I developed called CAL (Customer Agreement List). CAL functions like a shared document, but on a very large scale (10 million records). The front end and user interface is a simple Excel spreadsheet. There are three main sheets, all of which connect to a cloud database and corresponding tables. The user has the ability to view and update records by customer and can see live updates as they are made by other users in the system.

This capstone has been incredibly beneficial as I conclude the CS program and begin planning for new job roles. I have never made a formal portfolio and the practice/feedback provided by my instructor has been clear and helpful. Looking forward, I plan to specialize in full-stack development. I hope this was made somewhat clear when looking at my project – although I used Excel as the UI, I quite enjoy the design and thought that goes into front end development, and the complexity that goes into backend development. I very much like they idea of working projects end-to-end.

As I reflect on my time in this Computer Science program, I can’t help but to be excited for my future. This has been a very challenging program for me, someone with no technical background/experience prior to enrollment. I have been able to learn and apply principles from each course both in personal practice and at work. As a result, I have been able to progress in my current occupation while building the foundation for my future career.

*Software Design and Engineering:*

This artifact serves as an event handler, and it sits behind an Excel workbook. It monitors the workbook for specific changes (events) and performs specific actions depending on the conditions (how/why/when) of the event trigger. This module was initially created two years ago. My enhancement sends a specific alert when users try to add data outside of the allowed range.

Text

Description automatically generated

This artifact has been included in my ePortfolio because it delivers clear and concise visual communications that are coherent, technically sound, and appropriately adapted to the active users of this application. As such, I have met the course objectives that I planned to meet with this enhancement in Module one.

A challenge that I faced in the process of enhancing this artifact was simply recalling how the project functions. I built this two years ago, and there were many modules that let me stumped – luckily the project is well commented. After re-familiarizing myself with the project, this update was very straight forward. It is a very simple thing to implement, I just didn’t think of it at the time of development.

*Algorithms and Data Structure:*  
 This artifact identifies user updates and performs CRUD to the server tables. My enhancement is simple – create a hidden sheet that retains the original data (read only), and when saved, compare the hidden sheet to the updated sheet using SQL to find mismatching (updated) line items. Once the updates are saved, refresh the hidden sheet and repeat. This is core functionality behind the project – it is designed to essentially act as a shared file on a massive scale. This module was initially created two years ago. My enhancement includes faster algorithms to identify changes (update/insert/delete).

This routine identifies new records by returning a primary key for items that exist on the user-maintained sheet, but not the hidden sheet.

Text

Description automatically generated

This routine identifies updated records by returning a primary key that exist on both the user-maintained sheet and the hidden sheet, with values that do not match in every field.

Text

Description automatically generated

This routine identifies deleted records by returning a primary key for values that exist on the hidden sheet, but not on the user-maintained sheet.

Text

Description automatically generated

This routine contains insert/update/delete procedures using the values returned in the previous routines.

Text

Description automatically generated

This routine refreshes the hidden sheet once changes are synced to the server.

Text

Description automatically generated

I included this artifact in my ePortfolio because the previous algorithms were time consuming and inefficient. This enhancement serves to improve functionality and performance of the product while following computer science practices and standards, and managing the trade-offs involved in design choices. I have met the course objectives that I planned to meet with this enhancement in Module one.

One challenge that I faced in the process of enhancing this artifact was accidentally impacting downstream processes. There are many modules in this project, and there are many custom functions that are utilized by those modules throughout. In making this enhancement, I was faced with the decision of creating a new function or updating functions that would in turn require I update any module or sub that referenced that it. Of course, I don’t want to have redundant code. I ended up having to handle these scenarios on a case-by-case level, depending on the complexity of the update.

*Databases:*

This artifact stores all records for the project. There are three tables in this project, each containing specific customer data. The programs table stores supplier/customer contract details, customer\_profile contains customer contact and system configuration details, and the deviation\_loads table contains customer eligibility for group purchasing organizations. These tables were created two years ago. My enhancement includes creating indexes for the searchable columns to increase table query speeds.

This is the SQL used to create the indexes – Customer is the only field that users search by, so it is the only field that needs in indexing.

Graphical user interface, text, application

Description automatically generated

To emphasize the impact of indexing, I ran two queries on identical tables, one before indexing and another after. As you can see, even though this is an incredibly simple query and it does not run *slow* in either test, searching with an indexed field is far faster.

Without Indexing:

Graphical user interface, text, application, email

Description automatically generated

With Indexing:

Graphical user interface, text, application

Description automatically generated

I included this artifact in my ePortfolio because querying the existing tables (without any indexes) can be time consuming and do impact the overall performance of the project. This enhancement serves to improve the functionality and performance of the product while maintaining core functionality and security. I have met the course objectives that I planned to meet with this enhancement in Module one. I was able to implement the indexes without issue, and the increased in query speed is noticeable for the user.

One challenge that I faced in the process of enhancing this artifact was learning how to execute indexes without knowing how the change may impact the project. This is my first-time adding indexes to any project, and the first step before just following a stackoverflow tutorial, was cloning my database tables and making a backup. I was worried that adding indexes would somehow impact table functionality in a way that I wasn’t aware of, so I had to make very sure that I could undo any changes with ease.