**Project Report**

**SolidWorks API Project Manager**

CCET 4610 Spring 2023

Developed by: Weston Shakespear

# I. Introduction

The main inspiration for this idea is to have the ability to manage the parts and sub-assemblies that are used in projects. I want to build a fully standalone application for Windows that will communicate with Solidworks.

This client program will allow for new projects to be created and old projects to be downloaded and/or managed. The main feature will be the versioning for components and the ability to re-use already designed or acquired components in new projects without messy file paths or organization.

For both company and personal data and file management having centralized and networked file servers is paramount for file security. By creating a way to host my projects in one location, I’ll be able to more quickly create projects as well as safely backup changes and modifications.

# II. Problem Statement

When creating an assembly for a new project I often find myself navigating folders and looking for parts I’ve created in the past. It’s difficult to find or remember which part is the most current since I will update the details or configurations for a part when using it in a new project. This is a problem I have been wanting to solve before I even started attending SUU. Coming from a programming background I am used to revisioning and collaboration tools like git and GitHub. While these solve the problem for programmers perfectly, these tools don’t fit the cad and design environment as well.

# III. Project Objectives

|  |  |  |  |
| --- | --- | --- | --- |
| **Rubric** | | | |
| **Objective** | **A** | **B** | **C** |
| **1** | Research and become comfortable with using the SolidWorks API by writing libraries to aid in the following tasks. | | |
| Starting, attaching to, and controlling the running SolidWorks instance. | A library has been created and has all the needed functions managing the SolidWorks instance. | While code has been written to communicate with SolidWorks, it has not been wrapped up into a library. | There is some code for managing SolidWorks, but it is not reusable and not efficiently written. |
| Listening to and interacting with the events in the current running instance. | A library has been created and has all the functions needed for interacting with and listening to events from SolidWorks. | While code has been written to interact with events, it has not been wrapped up in a library. | There is some code for listening to events in SolidWorks, but it is not reusable and not efficiently written. |
| Interacting with the components in assemblies, and individual features and sketches in parts. | A library has been created and has all the functions needed to read and modify the components, features, and sketches. | While code has been written to interact with components, features and parts it has not been wrapped up into a library. | There is some code for interfacing with components, features, and sketches but it is not reusable and not efficiently written. |
| **2** | Create a standalone desktop application that will communicate with SolidWorks and aid in the creation and collaboration of projects. | | |
| Both local and cloud parts and assemblies should be viewable and manageable. | There is an easy-to-use interface to upload, download and manage local and cloud-based documents and projects. | There is a way to upload, download and manage local and cloud-based documents and projects but the interface is clunky or not intuitive. | Creating or managing projets is difficult or impossible to do with the current state of the software. |
| New projects will be created via the interface and can have existing parts pre-inserted. | There is an easy-to-use interface to create a new project, assign properties, and auto-insert existing designs. | There is a way to use interface to create a new project, assign properties, and auto-insert existing designs but the interface is clunky or not intuitive. | There are issues creating new projects or inserting existing parts into newly created projects. |
| Parts in projects can be replaced and updated when new versions are available from the server. | There is an easy-to-use interface to replace parts with newer versions or with other parts. | There is a way to use interface to replace parts with newer versions or with other parts, but the interface is clunky or not intuitive. | There are issues with replacing or updating parts in assemblies. |
| **3** | Create a server based application to facilitate the storage and serving of projects and part files. | | |
| This should store all data for projects, file locations, and versions in an SQL database. | The server software can store and manage all relevant data about parts, assemblies, and projects. | The server software will log incoming data and will facilitate the program working, but the data is not persistent. | The server software is unable to function well enough for projects to be managed on the desktop application. |
| Projects and files should be easily downloaded and uploaded by the desktop application. | The server can communicate with the desktop application and supports downloading and updating projects. | The desktop application is able to download parts and projects but is unable to upload to the server. | The server software is unable to function well enough for projects to be managed on the desktop application. |

# IV. Evaluation of Objectives

The code and documentation for all of the objectives can be found in the IX. Project Results section.

## 1A. Starting, attaching to, and controlling the running SolidWorks instance.

The SWLib library that I built handles all aspects of this requirement. An instance of Solidworks can be started or connected to with only one line of code, and from there most of the basic calls can be accessed through the SWLib methods. Because I was able to achieve all of the functionality in this objective, I would give myself a 10/10.

## 1B. Listening to and interacting with the events in the current running instance.

Although I only completed the code for event handling and didn’t wrap all of it into a library, I think I would still give myself a 9.5/10 for this objective. Since I wasn’t very familiar with all the intricacies of the Solidworks API I didn’t realize how easy it was to attach the event handlers and how I wouldn’t really want those function calls obscured inside of a library.

I think if I had a better understanding of C#, it would be easier to write a library with all the needed event handlers wrapped up inside. The issue comes with wanting to have these handlers access a myriad of other parts of the code, and implementing this I might run into some issues with C#’s lack of pointers.

## 1C. Interacting with the components in assemblies, and individual features and sketches in parts.

I was able to write the code to read the components in an assembly and the features and references in a part. While this is a good start to finishing this objective there is still a lot of code that would need to be written to modify the components or features. Because of all of that I would give myself a 7.5/10 on this objective.

## 2A. Both local and cloud parts and assemblies should be viewable and manageable.

The local and cloud hosted parts are all viewable in the application and the user can easily download files that don’t exist locally. The preview for the files makes it easy to find the correct one and didn’t actually take that long to implement. Since I was able to finish this objective all the wasy it will be a 10/10 for this one.

## 2B. New projects will be created via the interface and can have existing parts pre-inserted.

I was able to get a project creation dialog created, with all of the features I needed to get the rest of the objectives done. I wasn’t able to get the insertion of pre-existing parts done, this came more from time constraints as I kind of blew past my estimated time limit and kept going throughout the last few weeks. Because of this I would give myself a 9/10.

## 2C. Parts in projects can be replaced and updated when new versions are available from the server.

When there are newer parts on the server compared to the local parts there are no issues downloading these updated parts and having them updated in the assembly. The only issue with this process is if any of the files are open or if there are issues with the dependencies. There are next to no examples of a lot of the dependency-related functions and most of the plm functions are buried inside inaccessible apis for the type of licensing the school has.

Because some of the issues were outside of my control and I was able to get the main part of the objective done I would give myself a 10/10.

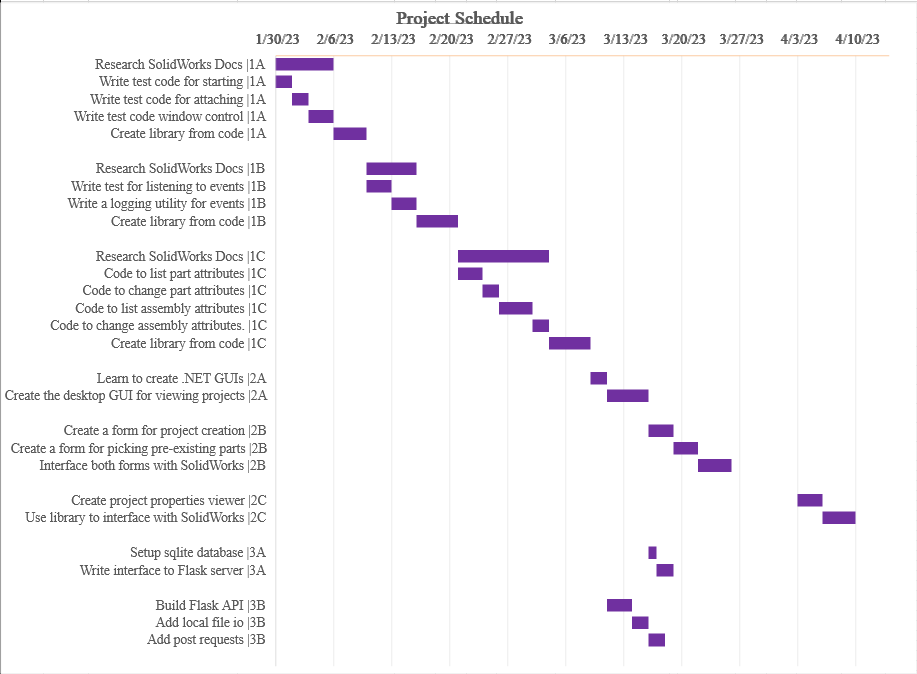
## 3A. This should store all data for projects, file locations, and versions in an SQL database.

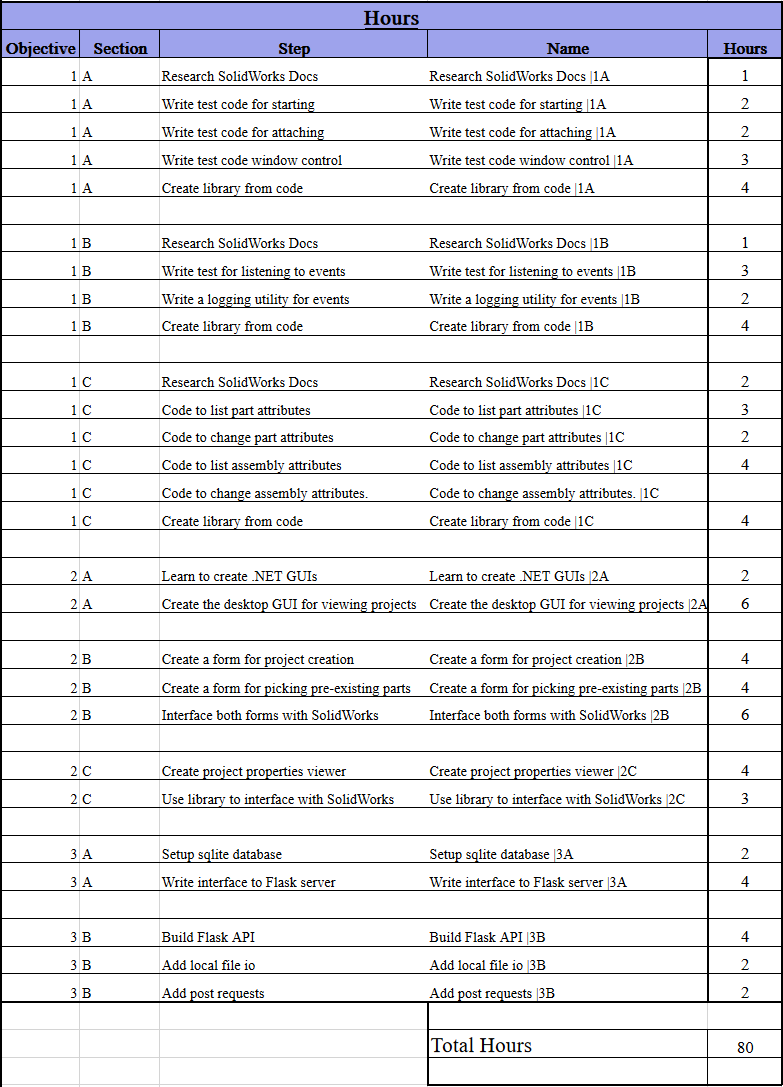
The server code is able to quickly decode the sent files, keeps all the versions and is able to recall the newest or any version, and actively tracks all this information in a SQL database. Because all these requirements were fulfilled, I would give myself 10/10 on this objective.

## 3B. Projects and files should be easily downloaded and uploaded by the desktop application.

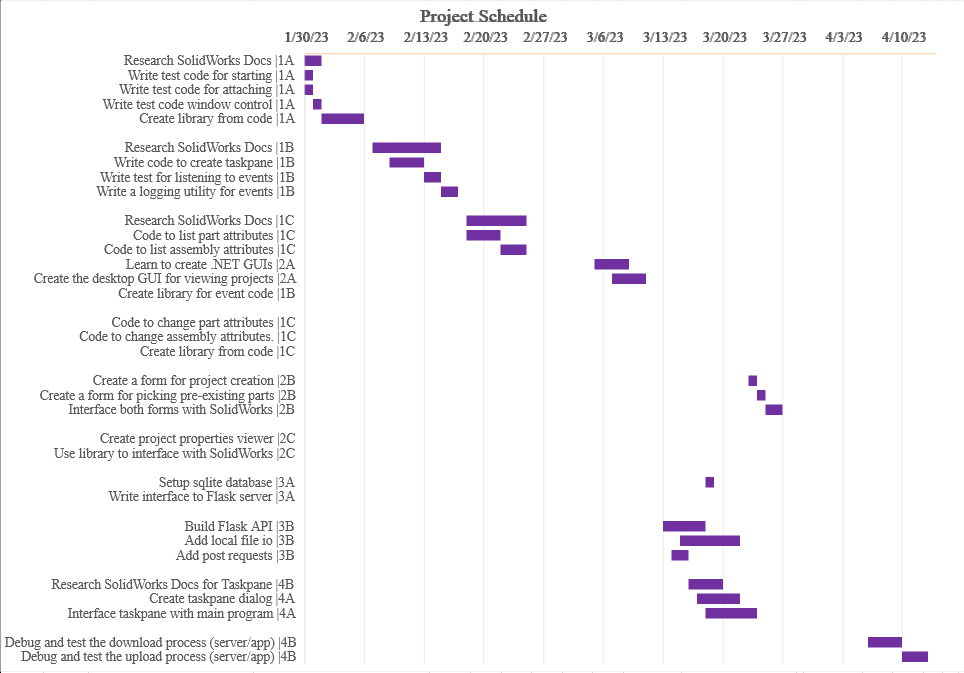
The functions for both downloading and for uploading files both have error handling and logging built into them. There could be more verbose messages sent to the end user, and I do plan on implementing this at some point in the future. I would give myself 10/10 on this objective as well.

# V. Proposed Schedule





# VII. Actual Schedule





# IX. Project Results

Since I have built a tool for my project, I think that supplying a manual and programmers guide to my application and server will be the most concise way to show the results. In the additional “Final\_Project\_Documentation” directory there are these documents in pdf form showing how to set up and use both programs.

# X. Reflection of Success

I was able to learn a ton about how Solidworks operates “under the hood” and already have a bunch of ideas for different tools I could build to improve my workflow. Buidling the tools that I need to build and create more applications that interact with Windows programs will probably be the biggest takeaway from all of this. As I had mentioned in my proposal and some of my status reports I don’t have very much experience developing things for Windows but feel a lot more comfortable after learning the ropes throughout this semester.

I was able to build a tool that is in a fully usable state as it is and only needs a little more functionality in order for me to use to day to day for my different projects. Because of all of these things I think this project was a pretty big success.

# XI. References

## SOLIDWORKS API Reference

https://help.solidworks.com/2023/english/SolidWorks/sldworks/c\_solidworks\_api.htm?verRedirect=1