Name: Trask Manley Date: 6/5/2022

Final Project Requirements

Important Dates

Date	Event
5/16	Project Idea and Requirements Document/Discussion Deadline
6/4	Optional – Test run and preload your software configuration on the instructor computer and test with the projector.
6/4 or 6/6	Class Presentations
6/7	Due Date to Submit Source Code and Documentation

Objective

The main goal of the final project is to combine the skills and concepts that you have learned throughout the course and put together a real-world solution that leverages the best that .NET has to offer. Every week's lab was an exploration into a different topic of .NET, giving you the building blocks to create a feature rich application.

Note, this is not meant to be as big or detailed like a senior project. This is meant to be a small fun project to explore new things you have learned and showcase some neat things in a presentable demonstration application.

Some example project ideas might include:

- A photo world mapper that loads photos, reads the EXIF metadata for GPS information, calls the Google Maps API, and pins the photos to a map of the world.
- The user interface for a smart display with a WCF host for data. Display aggregated data from a WCF host in a summarized digest form. A popular project for smart mirrors in home automation projects.
- A tool for file hash and signature validation. Generate hashes for files and sign them with your private key to identify if a file has been tampered with.
- A puppy world mapper. Load pets up for adoption on Petfinder's web API and use Google Maps API to plot the location on a map. See which pets are nearest you for adoption on a neat world map.

Project Idea and Scope Discussion

Before you get started on coding your project, you will post on the Canvas discussion and discuss the project idea and define the scope of your project. The scope will consist of a list of features that your project will implement and identify the completion point of the project. This list will be used to identify how complete the project is at the point of submission.

I know, from a personal standpoint, that some projects can start off small and end up larger and larger during implementation. Sometimes, in order to implement a requirement, several more pop up and it possibly could turn into a never-ending list of things to implement before you can even get to the feature you wanted.

This is why discussion of project ideas is important! We can learn and make use of each others' experience and help each other out identifying hidden scope.

After you make your discussion post introducing your project idea and scope requirements, you will also make a comment on **two** other student's posts. Discuss the topic and scope requirements. I will leave the discussion generally open whatever you want to talk about, but try to provide some feedback. Whether it is about the topic or requirements, it is up to you.

Once you have made your post and have received feedback, compose a Microsoft Word document of your final topic and scope requirements and commit it to your Labs repository. Then start coding!

If you feel the scope has changed during the implementation of the project, let the instructor know as soon as possible so the list of requirements can be updated to meet with the project's new specifications.

Sometimes you find that during implementation (or even before) you find yourself having trouble with something, reach out to me or the class for help! Post a message to your post or ask during lecture!

If you feel like you need help implementing a specific feature, I will be more than happy to schedule either office hours or some time before or after class to help out.

Remember, this should be a fun activity! This is not a senior project!

Project Requirements

The following requirements are intentionally generalized to allow for flexibility in the project. All requirements will have been lectured in class prior to the end of the term and have generally been the focal point of the weekly labs.

- The project must be written in .NET 6.
  - You can choose any platform for your target, Windows, Mac, Linux, Mobile, etc…
  - You can use any libraries either downloaded from the internet or obtained via NuGet. However, you must provide attribution for the libraries you linked in.
    - Watch out for framework dependencies!
- The project must include a user interface*
  - If you are deploying to an IoT device like a Raspberry PI, the user interface will be the API consumer.
  - In all other cases, the user interface must:
    - Have their data acquired by databinding.
    - At least some level of rudimentary input validation preventing the user from inputting malformed information. E.g., incorrect file paths or numbers that don't parse from string inputs.
  - Any functions that are not instant has to be wrapped in an async/await function as to not block the UI thread. Use asynchronous methods if available and if not, wrap it yourself in an async method.
- The program must also include at least one topic from the following list of previous lab topics:
  - Plugin Architecture or Assembly Reflection
  - Cross platform development
  - Serialization (Binary, XML, JSON, etc…)
  - Web Service API Integration (Consuming or hosting)
  - WCF Communication

    - Parallel Processing
    - Data Encryption or Hashing
- The project must accomplish some non-trivial task. The goal of the project is to create a rich application that leverages the power of the .NET Framework. Simply downloading and saving serialized data from a web endpoint is not sufficient. Use the data in some way.
- All source code must have sufficient comments so that anyone who is reading the source code can understand what it is doing. Also, self-describing variable names are helpful.
- Adequate exception handling must be implemented. For example, if you access a web service and the network is down, you need to be able to catch the web exception and display a helpful message to the user indicating the network is down.

What To Submit

- The repository tag and commit hash for the source code for the project.
- A Microsoft Word document that describes your project details. It should contain:
    - A description of the problem statement
    - The objectives that you are trying to accomplish
    - Any limitations of your solution
    - A list of any nonstandard .NET libraries you have used. Include:
        - The name of the library package.
        - The URLs of where you obtained the libraries or NuGet repository address.
        - The specific version numbers of the libraries used.
    - Any required sample inputs or preset configurations
    - An example of the expected output or results.
    - How to build your project if it isn't as simple as F6.

Basically, a summary of what your project does and a list of what needs to be included to build and replicate your project to reproduce the same results.

What You Will Be Presenting

For the class presentation, you will prepare a short 5-10 minute presentation where you will include:

- The purpose of your project
- What your project does
- A physical demonstration of your project

For the demonstration, you will have the choice to use the instructor computer to run your application to display on the projector or you can connect your personal computer to the projector.

Note: If using your personal computer, make sure it has the ability to connect to a 15 pin VGA connector.

Note: If using the instructor computer, please come to class early to provide enough time to go through a dry run to solve any unforeseen problems so as to not hold up the rest of the presentations.

A test run of your presentation using the projector or instructor computer would be ideal to do on the lab day prior to the presentation day.

Grading Rubric and Information

Metric	Description
Requirements (20 points)	Implemented all the project requirements as specified in this document.
Scope (20 points)	Implemented all the functionality as described and agreed upon during the project idea and scope meeting.
Documentation (20 points)	Word document containing all the project details as described in this document.
.NET Code (20 points)	Source code compiles with no errors and is sufficiently commented. A minimum amount of error checking and exception handling preventing the code from crashing from erroneous user input.
Presentation (20 points)	The presentation introduced the project's purpose and described the functionality. The demonstration correctly displayed the intended output of the project.
Total: 100 points	