Github Repos to demonstrate

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Source URL | Additional URL | Simple Description |
| Configuration Manager - Keepass database Storage | <https://www.github.com/skyhoshi/ConfigManager-Keepass> | <https://keepass.info> | Configuration process to store settings in a Keepass Database. |
| Face Recognition - In Dot Net | <https://github.com/skyhoshi/FaceRecog-LiveCoding> | <https://www.education-ecosystem.com/skyhoshi/lNbDO-how-to-integrate-facial-recognition-into-any-dotnet-app> | A Demonstration Windows Forms application to develop a Facial Recognition Registration application.  We talk through possible projects and how to use Azure Congnitive services to include a Facial recognition process into a dotnet application. This Project remains unfinished but achieves the goal of laying out the foundation for cognitive services integration into a dotnet application. |
| Webhook Consumption and Response website | <https://github.com/skyhoshi/WebhookCap> |  | Application I reciently created for a company that managed their own database.  Purpose was to catch a webhook (XML) and parse it, push it to the database and offer ability to validate users and turn the app on and off.  If you want to run, update the appsettings.json and run the update commands listed in the given contexts, you'll need to run all three. It was intended to allow the end user to move/store the data in different locations all by adjusting the connection strings. |
| Installation Displacement Powershell Module | <https://github.com/skyhoshi/VHDXToFolders> | <https://gist.github.com/skyhoshi/dab7d6d3812428dbf5bb652c6c00b965> | Concept is to displace application files from a higher performing hard drive to a lesser one. In other words, you can install an application using a VHDx File as the storage location in place of the folder using a Symbolic Link (Junction) and move the VHDx Around as needed to better manage storage and performance rather than uninstalling and reinstalling applications. |
| WinGet |  |  | I was attempting to add a feature [its currently incomplete] demonstrates ability to expand and conform to lesser known or weaker strength languages and architectures. |
| Skyhoshi.App.TimeMarker |  |  | An Application I developed to allow someone to use shortcut keys (HotKeys) to record timestamps |

Project: Configuration Manager – Keepass database Storage  
Source: <https://www.github.com/skyhoshi/ConfigManager-Keepass>  
Additional Url’s: <https://keepass.info>  
Project Description:

This Library represents early work on secrets management. It uses a keepass database as a secure store, (visit <https://keepass.info> for more information) the concept is simple, you create a Keepass Database, store User/Application/System Secrets as entities within the database which can include nearly limitless amounts of additional data. (Limited by 3 things, One: File size (Load Time), Two: Storage Available, and Three: Search Method(s) Conditions) The application uses the search function built into the library to find the entities by name.

My Customization allows you to call the configuration object like you’re used to. This returns the string object that is stored in the keepass database. If it’s not found in the keepass database it will look to local configuration files and storage methods included.

```  
Configuration.AppSetting[SettingsName]

```

The next version of the project has plans to implement the Microsoft.Extensions.Configuration library and allow loading and consumption of enumerated lists of settings files and locations, the keepass database will be included in that list.

Project: Face Recognition - In Dot Net  
Source: <https://github.com/skyhoshi/FaceRecog-LiveCoding>  
Additional Urls: <https://www.education-ecosystem.com/skyhoshi/lNbDO-how-to-integrate-facial-recognition-into-any-dotnet-app>  
Project Description:

This Application was a demonstration of the Azure Cognitive services integration into a User Registration application developed for the Video series.

The concept wrapped around having a Home Automated Door Deadbolt Lock and Eyehole Camera. When you presented yourself to the door, the automated door Deadbolt Lock application (not apart of this project) would take a picture, using Accord framework (<http://accord-framework.net/>) to locally identify if the picture had a face within it. When a face was found, the image would then be sent to an API (again Not apart of this project) that would then be sent off to Azure for recognition and categorization, if the face belonged to a registered group (Registered within Azure Cognitive Services Face-API) the API using that information would use that information to check the local registration to validate the authorization the user had to the entry way. If they were authorized the system would then send off a signal to unlock the door. Unauthorized users would be cataloged, and an email would be sent to be reviewed.

Project: Webhook Consumption and Response website  
Source: <https://github.com/skyhoshi/WebhookCap>

Additional Urls:   
Project Description:

This has been one of my latest development projects as of October 2020. This simple application has a UI Interface that has several features.

Project: Installation Displacement Powershell Module  
Source: <https://github.com/skyhoshi/VHDXToFolders>  
Project Outline Url: <https://gist.github.com/skyhoshi/dab7d6d3812428dbf5bb652c6c00b965>  
Project Description:

Development started in DiskPart Settings Text Files and now PowerShell 7 is used to manage all the things that Diskpart and a settings file would do. PowerShell 7 is required.

The TLDWTR (Too Long and don’t want to read) Version:

Concept is to displace application files from a higher performing hard drive to a lesser one. In other words, you can install an application using a VHDx File as the storage location in place of the folder using a Symbolic Link (Junction) and move the VHDx Around as needed to better manage storage and performance rather than uninstalling and reinstalling applications.

History of the Project:

This project was one of necessity. After 20+ years of collecting, developing and loosing/misplacing projects and source files, When Windows 7 ( + Windows Server 2008) presented the ability to create Virtual Machines. Virtual Hard drives became a standard in storing a lot of “Shared” data. I had virtual hard drives that I would attach and detach from virtual machines to do maintenance and installations.

A few years ago with the introduction of Windows 10, Hyper-V and Hyper-V PowerShell Module I was able to script out the creation of Virtual Hard Drives and took the concept of sharing one step further by attaching these drives to folders (Folder Mounting).

Theories and Practice:

With this I could in theory, install games onto to my SSD drive but when I wasn’t playing the game but needed additional space on the then limited space of my SSD I could dismount and displace the VHD to a Cold storage Spindle based drive (lesser performing drive) or even push the drive up to network storage for archiving.

I took the theory one step further when I started working for GiftCertificates.com and used the virtual hard drive storage process to section of projects and configurations in order to better my development performance and time spent between different projects all while managing the space, and what was presented to me at any given time in the work environment.

Passing on the concept:

I’ve taken my time to convert the large group of scripts and methods down to a concise and simple PowerShell Module that will allow you to take a given folder on your system with a virtual hard drive that is mounted to the folder but is stored separately.

Currently the project is unfinished, but the base scripts needed to mount existing virtual hard drives at startup are present. The generation methods for creating the mounting config files and scripts are in the source but the automation of converting a given folder to a virtual hard drive backed storage folder is rather scattered and currently not included in the source due to liability reasons. (an issue that I ran into is corruption and deletion of files during moves, until I can ensure that this does not happen / cannot happen within my scripts I’ve left these out. I, myself am required to manually move important folders due to the scripts tenuous nature.)

Project: WinGet  
Source: <https://github.com/skyhoshi/winget-cli>  
Additional Urls:   
Project Description:

I was attempting to add a feature [its currently incomplete] demonstrates ability to expand and conform to lesser known or weaker strength languages and architectures.