**Lab Report 1**

Title: Introduction to APIs and Data Interoperability

Notice: Dr. Bryan Runck

Author: Evening Hade

Date: 30 September 2024

**Project Repository:**<https://github.com/eveningsaria/gis5571>

**Google Drive Link:** not applicable

**Time Spent:** 20 hours

**Abstract**

This lab exercise begins with comparing three different application programming interfaces (APIs) within the GIS world. First, these APIs were explored and deconstructed to compare their interfacing differences Next, Python/ArcPy is used to pull data from these sites with requests.get(), project them onto a map making reprojections if necessary, create a spatial join, print the head of an attribute table, and save the feature layers as a geodatabase. Throughout the process, it became clear that the different APIs were not only visually distinct, but characteristically so, altering the way one must interact with them.

**Problem Statement**

To compare three different APIs from three different websites: Minnesota Geospatial Commons’ CKAN API, Esri ArcGIS Online’s REST API, and the North Dakota Agricultural Weather Network (NDAWN), they must be visually examined, dissected, and interacted with by downloading data from them and manipulating it using a GIS interface.

*Table 1. Description of data used in this project.*

| **#** | **Requirement** | **Defined As** | **(Spatial) Data** | **Attribute Data** | **Dataset** |
| --- | --- | --- | --- | --- | --- |
| 1 | County Boundaries in Minnesota | Raw input dataset from MNDOT | Counties of Minnesota represented as a shapefile | Polygonal county geometry | [County Boundaries in Minnesota](https://gisdata.mn.gov/dataset/bdry-counties) |
| 2 | COUNTY\_NAM | Names of counties in MN | Projected using UTM Zone 15N NAD83 | Polygonal county geometry | [County Boundaries in Minnesota](https://gisdata.mn.gov/dataset/bdry-counties) |
| 3 | Counties Free Reduced Lunch 2015 to 16 | Raw input dataset from the Minnesota Department of Health via Nicole Helgeson | Nonspatial .csv | County geometry | [Counties Free Reduced Lunch 2015 to 16](https://umn.maps.arcgis.com/home/item.html?id=c9fa2c75ec6e494da701348cc513658f) |
| 4 | Region | Names of counties in MN | Nonspatial | County geometry | [Counties Free Reduced Lunch 2015 to 16](https://umn.maps.arcgis.com/home/item.html?id=c9fa2c75ec6e494da701348cc513658f) |
| 5 | Average Temperature | Raw input dataset from NDAWN | Lat/Long coordinates | Weather station point geometry | [NDAWN Weather Station Data](https://ndawn.ndsu.nodak.edu/table.csv?station=78&station=174&station=118&station=87&station=124&station=226&station=219&station=184&station=2&station=220&station=223&station=93&station=183&station=156&station=70&station=173&station=185&station=187&station=119&station=4&station=82&station=225&station=120&station=71&station=103&station=116&station=114&station=3&station=115&station=121&station=61&station=181&station=60&station=122&station=5&station=91&station=182&station=117&station=6&station=222&station=92&station=123&station=95&station=148&variable=mdmxt&year=2024&ttype=monthly&quick_pick=&begin_date=2023-09&count=12) |

**Input Data**

To answer this question, three datasets are required: one from each API. These three datasets were all manipulated using Jupyter Notebooks integration with ArcGIS Pro. The dataset, “County Boundaries in Minnesota”, displays the official county boundaries of the State of Minnesota. The Minnesota Department of Transportation created this dataset. It was accessed through the Minnesota Geospatial Commons. Of the three datasets, this first one has the most accessible presentation. The dataset, “Counties Free Reduced Lunch 2015 to 16” (originally written as one word with no spacing), records which counties in Minnesota participated in the Free/Reduced Lunch program for schoolchildren as well as how many meals were served, how many people were served, and the estimated state of need in the county. Nicole Helgeson created this dataset through the University of Minnesota, though the data itself is from the Minnesota Department of Health. It was accessed through Esri ArcGIS Online. The dataset, “Average Temperature” (originally unnamed), displays average daily temperatures measured over a month-long period by NDAWN-affiliated weather stations in Minnesota. This data belongs to NDAWN and was accessed via their website. Of the three datasets, this last one certainly had the most data, but the most confusing presentation.

*Table 2. Datasets used in this project.*

| **#** | **Title** | **Purpose in Analysis** | **Link to Source** |
| --- | --- | --- | --- |
| 1 | County Boundaries in Minnesota | Raw input dataset of Minnesota’s counties from MNDOT | [County Boundaries in Minnesota](https://gisdata.mn.gov/dataset/bdry-counties) |
| 2 | Counties Free Reduced Lunch 2015 to 16 | Raw input dataset from the Minnesota Department of Health via Nicole Helgeson | [Counties Free Reduced Lunch 2015 to 16](https://umn.maps.arcgis.com/home/item.html?id=c9fa2c75ec6e494da701348cc513658f) |
| 3 | Average Temperature | Raw input dataset from NDAWN | [NDAWN Weather Station Data](https://ndawn.ndsu.nodak.edu/table.csv?station=78&station=174&station=118&station=87&station=124&station=226&station=219&station=184&station=2&station=220&station=223&station=93&station=183&station=156&station=70&station=173&station=185&station=187&station=119&station=4&station=82&station=225&station=120&station=71&station=103&station=116&station=114&station=3&station=115&station=121&station=61&station=181&station=60&station=122&station=5&station=91&station=182&station=117&station=6&station=222&station=92&station=123&station=95&station=148&variable=mdmxt&year=2024&ttype=monthly&quick_pick=&begin_date=2023-09&count=12) |

**Methods**

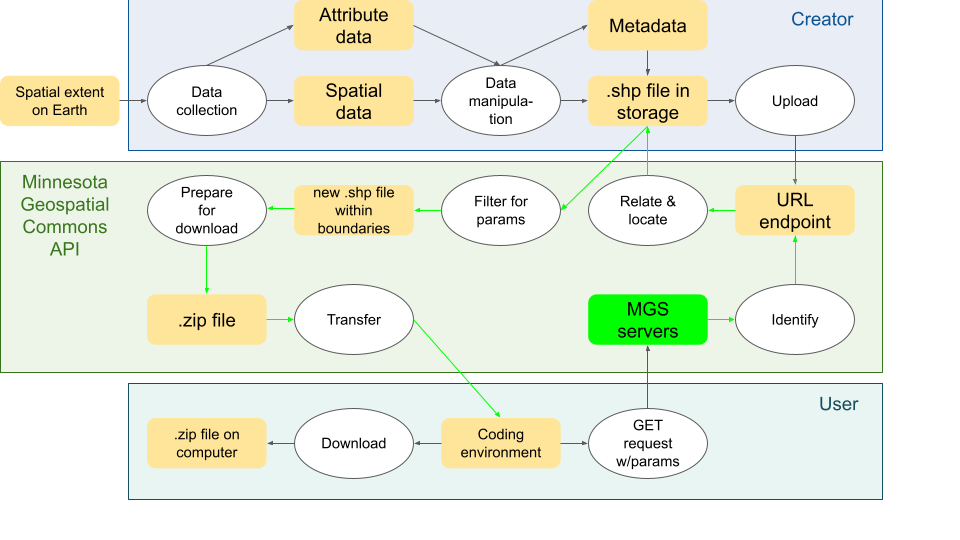
This project was divided into two parts: examining the APIs, and pulling data from the APIs to manipulate. Part One involved researching what the APIs were and determining what their goals might be. Inspecting the API code was the next step, resulting in connections being drawn between the code, the website’s appearance, and the API’s goals. In particular, the abstractions between the raw code and the GUI were examined, and the abstractions between the API and the actual data were pondered.

Part Two of this project investigated how interactions with these APIs differed between the three even in the same environment, ArcGIS Pro’s Jupyter Notebooks integration. First, data was pulled from all three APIs using requests.get()into a virtual OS. This involved three discrete methods, one for each API. It would be interesting to see if a more general function could be written by a more experienced programmer despite the APIs’ differences.

Next, spatial data was projected to WGS84 by iterating a function over a list of downloaded data. The following step involved spatially joining two datasets, one with county boundaries and one with free/reduced lunch information for those counties, via the name of the county. In doing so, the free/reduced lunch data was added to the attribute table for the county boundary data and displayed on the map’s pop-up. These steps aided in determining the interoperability of the data from the two APIs.

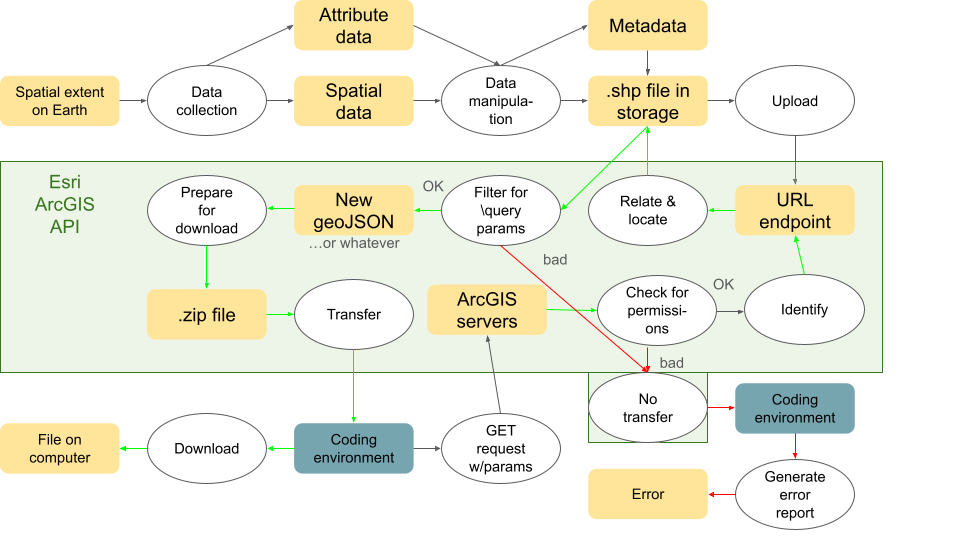
For each of the following figures, I will describe what role the API plays in this process.

*Figure 1. Data flow diagram for Minnesota Geospatial Commons.*



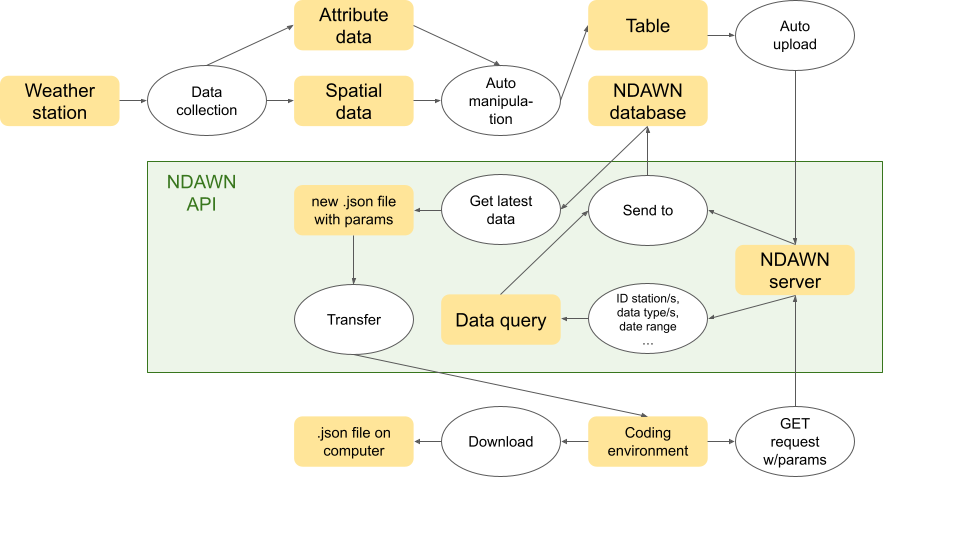
MGS’s API receives the GET request, identifies the file that the URL endpoint is referring to, checks to see if the user has specified a spatial extent for the data using its built-in boundary box feature, makes a new shapefile within that spatial extent (or merely copies the original if the boundary box includes all the file’s data), prepares the data for download by creating a .zip file, then transfers that data back to the coding environment on the user’s computer. It is also possible to download non-spatial data from MGS and in this case, a copy of the JSON or CSV or whatever is sent to the coding environment; it is not zipped. Regardless of the file format, that file the site created for the user is only available to the user, which is to say it is not displayed on the website anywhere. If another user wanted to get that data via MGS’s API, they would have to do their own GET request, and the API would run through all the steps again. Something important to note is that you can only download certain file types for certain datasets. The file type has to exist somewhere in storage in order for you to download it: you cannot convert the file using MGS’s API. (At least not as a novice programmer.)

*Figure 2. Data flow diagram for Esri ArcGIS Online.*

**

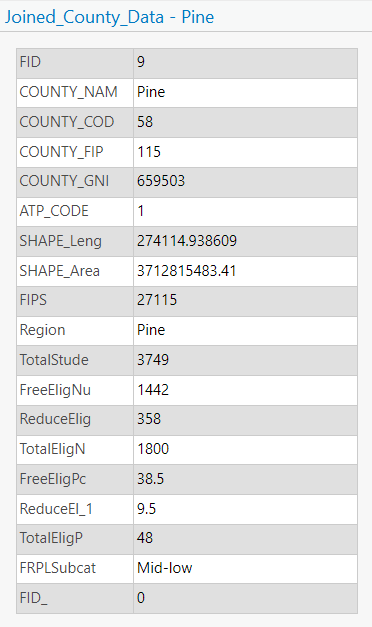
Esri ArcGIS Online’s API is by far the most advanced of the three. There are seemingly endless possibilities for the programmer who interacts with this API. Something important to note is that the ArcGIS servers will check for permissions based on an API key whenever you request data. If for any reason you do not have access to that data, you will not be able to download it or do much of anything else. If you do not have a personal or organizational ArcGIS account, the API will notice that this ID is missing from your request and it will not send you the data then either. One of the many things Esri ArcGIS Online’s API can do is convert data to a different file type; however, for this parameter and others, if it is not possible to complete it based on the data owner’s specifications or based on the data itself, then you will receive an error.

*Figure 3. Data flow diagram for NDAWN.*

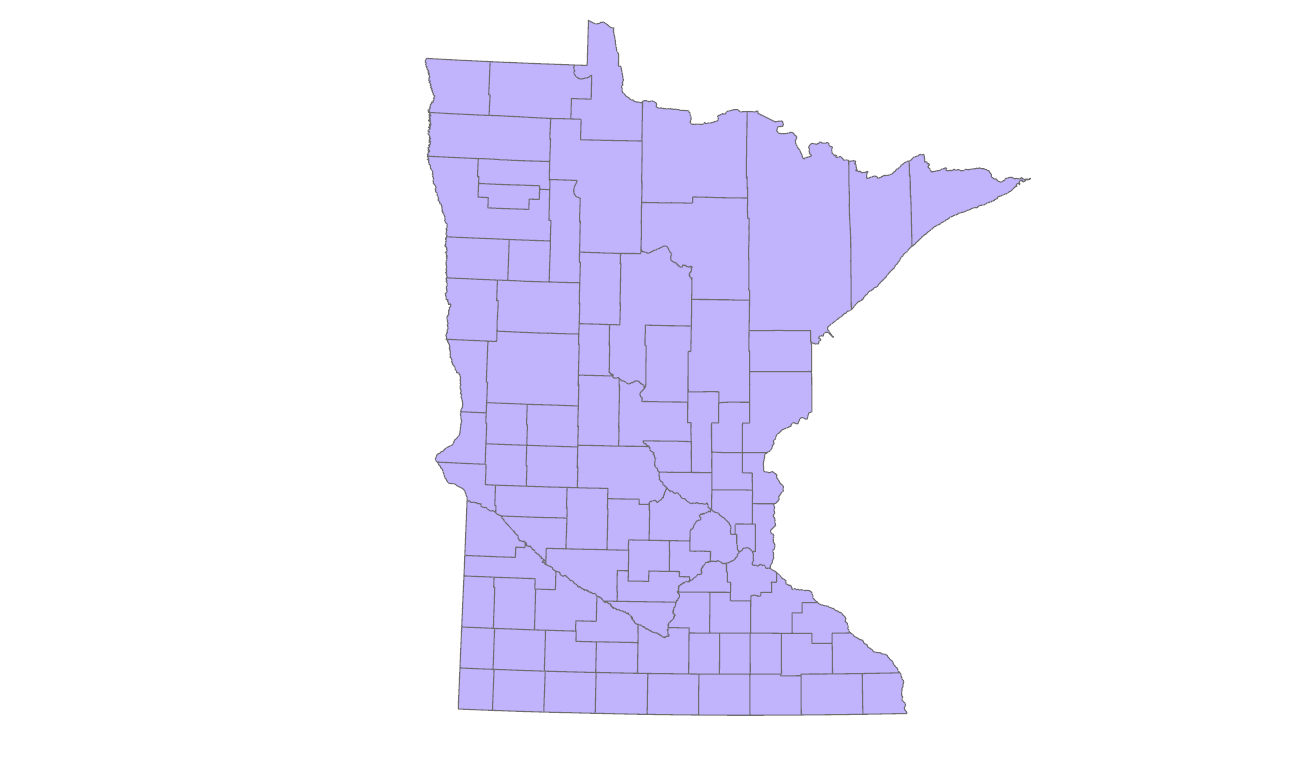
****

What really stands out about NDAWN’s API is that it is constantly interacting with NDAWN’s database. NDAWN has who knows how many scripts running as they do streamline-style data updates every five minutes, updates that are immediately communicated to NDAWN’s server. All of the data you acces via NDAWN’s API has been formatted and uploaded automatically, which makes it very different from the MGS and Esri data, neither of which are live service sites and both of which have datasets made and uploaded by certain creators. While this is very convenient for those who want the latest weather data, the lack of abstraction in the API means that finding the data you want is the hardest part of getting that data on your computer. Anyone can access NDAWN’s data, but one must know specific ID numbers used for every weather station and those ID numbers must be repeated one after another if you want data from more than one site. The resulting URL can be painfully long, and using parameters in Python hardly helps at all.

**Results**

The first API was Minnesota Geospatial Commons’ *Figure 4. Example pop-up screen for the*REST-ful API created with a CKAN framework. This *joined county data showing both MNDOT* website provided a [page](https://gisdata.mn.gov/content/?q=help/api) for developers with information *data and free/reduced lunch program data.* about their API. This signals an openness on the part of *These two APIs seemed tailor made for* Minnesota Geospatial Commons (henceforth abbreviated *spatial data, making joining this data easy.*  
as MGS) to individualized customization of their API to suit the needs of their audience. Yet the focus is not only on professional-level programmers. This API is very much tailored to spatial data while also maintaining an accessible interface, meaning that those with little programming experience can still interact with it easily. This is a result of a significant level of abstraction from the data, such that interfacing with the website seems much less technically complex than it actually is. For example, MGS offers users the option to create a ‘bounding box’ for the data they want to retrieve and the API will automatically filter a dataset. Overall, the API is easy to use, but acts as a starting point for more complex processes instead of facilitating them; this potential issue for developers is offset by the open source nature of the API, with CKAN providing [multiple resources](https://docs.ckan.org/en/ckan-2.1.5/api.html) for how to engage the API via programs on local computers.

The second API was Esri ArcGIS Online’s REST API. Esri has [documented](https://developers.arcgis.com/rest/) their API to provide users with an in-depth overview of how they might interface with it. While Esri is similar to MGS and CKAN in this way, a major difference is that its REST API is constructed with complex processes in mind. Using access tokens, developers can manipulate and analyze data with ease. This means that Esri’s API is less accessible to the general public, though. The system of access tokens, likely created because Esri’s REST API is a proprietary service as opposed to

an open source one, means that it is much more difficult

*Figure 5. Map created using ArcGIS Pro.* for the average user to interact with this API as opposed to the one used by MGS; however, there are significant advantages for professional developers and GIScientists. With a background of coding knowledge and some practice within Esri’s API, users can benefit from Esri’s robust interface and its extensive data management & analysis processes.

The third API was the North Dakota Agricultural Weather Network’s. There was no information available on the website regarding API development. With this being the third API on this list, it is perhaps at a disadvantage from the high expectations set by MGS’s API and the even loftier ones set by Esri. The largest limitation of NDAWN’s API is how it formats its endpoints. Python allows the user to set parameters, which can simplify the process, but really, the user has no way of knowing what their desired URL will be unless they’ve already downloaded the dataset they want to access. Thusly, when interfacing with NDAWN, interactions are clunky and inefficient, even though they are extremely straightforward with little abstraction between the data you are calling from the database and what appears on your local computer when you download it from the website. NDAWN is also unique in that it contains much more limited data than the other two sites; though this does not mean the data is necessarily of higher quality on average, you can be sure of what you are getting when you access a dataset from NDAWN. Also, some of the data is uploaded every five minutes from weather stations in North Dakota, Minnesota, and Montana. It is clear that NDAWN’s API is more focused on retrieving data than analyzing it programmatically. In conclusion, NDAWN’s API is the least developer friendly, but does what it seems to be designed to do, which is to retrieve data from weather stations on the back end, and allow users to download that data.

These results were reached via an examination of the websites and interacting with the data in ArcGIS Pro.

**Results Verification**

All of these APIs can be freely accessed by anyone in the community. For more information about Minnesota Geospatial Commons’ API and Esri ArcGIS Online’s API, refer to the links provided in the “Results” or “References” sections of this document; this information will also come up in a general web search. Importing arcpy requests zipfile io os and pandas into an ArcGIS Pro notebook will allow a user to replicate what I have done and experience what I experienced when interacting with the APIs. This is my first investigation into API functionality, and I would be interested in learning how a professional programmer’s perception of these APIs would differ from my own.

**Discussion and Conclusion**

In reality, the three APIs discussed throughout this document all necessarily have the same goal: to allow users to interact with the data made available through the respective websites. The ways in which the APIs are structured simply speak to different, specific audiences within the broad swath of visitors to these websites. While the NDAWN API seems to be made to be interacted with and not examined, which places a general, non tech-savy audience above the programming audience. The lack of abstraction within the NDAWN API means that it is both rather direct in its functions and burdensome in its presentation. On the opposite hand, the Esri REST API is targeted toward an audience of programming enthusiasts, consistent with Esri’s focus on creating cutting edge GIS technology in a world that is more and more cloud-focused. Somewhere in the middle lies MGS’s CKAN-based API, which easily facilitates basic functions in a manner accessible to the general public, yet is flexible enough to be useful to programmers as well.

*\*

**References**

*API Developer Resources*. Minnesota Geospatial Commons. (n.d.). https://gisdata.mn.gov/content/?q=help%2Fapi

Esri. (n.d.). *Build powerful apps with arcgis services: Esri developer rest apis documentation*. Esri Developer. https://developers.arcgis.com/rest/

Helgeson, N. (2018). *CountiesFreeReducedLunch2015to16*. ArcGIS Online. https://umn.maps.arcgis.com/home/item.html?id=c9fa2c75ec6e494da701348cc513658f

Minnesota Department of Transportation. (2024). *County boundaries in Minnesota*. Minnesota Geospatial Commons. https://gisdata.mn.gov/dataset/bdry-counties

North Dakota State University. (2024). *Monthly Weather Data*. NDAWN. https://ndawn.ndsu.nodak.edu/weather-data-monthly.html

*The CKAN API*. The CKAN API - CKAN Documentation 2.1.5 documentation. (n.d.). https://docs.ckan.org/en/ckan-2.1.5/api.html

**Self-score**

| **Category** | **Description** | **Points Possible** | **Score** |
| --- | --- | --- | --- |
| **Structural Elements** | All elements of a lab report are included **(2 points each)**:  Title, Notice: Dr. Bryan Runck, Author, Project Repository, Date, Abstract, Problem Statement, Input Data w/ tables, Methods w/ Data, Flow Diagrams, Results, Results Verification, Discussion and Conclusion, References in common format, Self-score | 28 | **28** |
| **Clarity of Content** | Each element above is executed at a professional level so that someone can understand the goal, data, methods, results, and their validity and implications in a 5 minute reading at a cursory-level, and in a 30 minute meeting at a deep level **(12 points)**. There is a clear connection from data to results to discussion and conclusion **(12 points)**. | 24 | **20** |
| **Reproducibility** | Results are completely reproducible by someone with basic GIS training. There is no ambiguity in data flow or rationale for data operations. Every step is documented and justified. | 28 | **28** |
| **Verification** | Results are correct in that they have been verified in comparison to some standard. The standard is clearly stated **(10 points)**, the method of comparison is clearly stated **(5 points)**, and the result of verification is clearly stated **(5 points)**. | 20 | **20** |
|  |  | 100 | **96** |