North Carolina Water Supply Dashboard

May 2021

# Introduction

The Internet of Water (IoW) strives to make data more findable, accessible, and usable. The North Carolina Water Supply Dashboard demonstrates the value of improved data infrastructure by finding data from multiple sources, accessing those data through API’s, and using those data by integrating them into a single dashboard. Additionally, we created new pathways to integrate utility data, improving data access and sharing for water supply and demand in the Triangle region of North Carolina. This project was done in partnership with the North Carolina State Climate Office, the Carolinas Integrated Sciences and Assessments, and the North Carolina Department of Environmental Quality. Stakeholders who contributed to the project include the City of Durham, the Town of Cary, the City of Raleigh, the Fayetteville Public Works Commission, Orange Water And Sewer Authority, and the Triangle J Council of Governments. All files and code can be found HERE (GITHUB?). The online dashboard can be found here (NCDENR website… they need to get their own mapbox api problably)?

Below, we walk through the step by step process to update data and the website depending on where you are starting.

# Set by Step Process

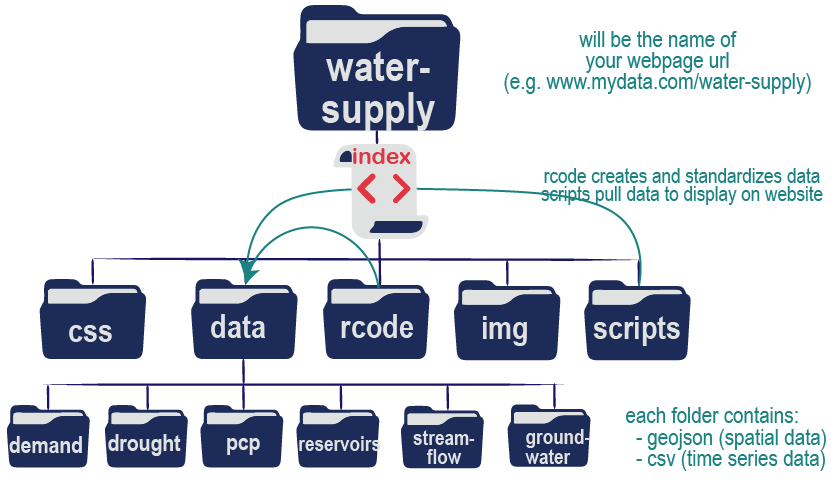
## Set up your folder structure

Clone the git hub folder: “we don’t have this version yet… I cloned r parts for you”

Rename the nc-water-supply folder to whatever you want the url endpoint to be. For example, the “nc-water-supply” folder is part of the url where the app is located: [https://**nc-water-supply**.internetofwater.app/](https://nc-water-supply.internetofwater.app/)

The statewide version of the app is: <https://nc-water-supply.internetofater.app/nc_state.html>

The scripts finding data and creating the dashboard are dependent on a specific file structure. The file structures should be as follows:



**Figure 1**: Folder structure needed for all scripts to run propertly. The **rcode folder** holds the R scripts that run to access and transform data found in the **data folder**. The **css** and **img folders** contain files that stylize the dashboard and provide static image. The **scripts folder** contains javascript files that read in data from the **data folder** to create the charts and maps displayed in the dashboard.

If you are starting a new state, you can delete all the data within the data folder and create the historic data through the access scripts. You will need to modify access scripts for groundwater and precipitation data sources relevant to your state.

If you are building onto the pre-existing dashboard (North Carolina), you can keep the historic data and add new data through the use scripts.

## Install R and a Text Editor (if needed)

### Install R.

To run the files in the rcode folder that will access and update data, you will need to install R.

You can install R for Linus, Mac, or Windows here: <https://cran.r-project.org/>

### Install RStudio (optional)

Rstudio is a really helpful platform for running R and being able to visualize code and plots together. You can install R Studio Desktop here: <https://www.rstudio.com/products/rstudio/download/> for windows or mac. You can run R through RStudio.

### Install a Text Editor.

To modify the files in the css folder (css), scripts folder (javascript), and the index file (html), you will need a text editor.

Any text editor will do, but Visual Studio Code (<https://code.visualstudio.com/>). This is a free software that has some nice features for debugging and testing changes on your computer before it is pushed to a server (check out the “Go Live” feature). The editor changes font color and spacing based on the type of file (.css, .js, and .html).

Another commonly used, free, open source text editor is notepad++: <https://notepad-plus-plus.org/downloads/>. Notepad++ also stylizes the text based on file type and can be used to make edits to these files. Additional packages can be downloaded for debugging, etc.

## Get your own api keys for generating maps and accessing data

### Mapbox API Key

To create the map layers you will need to obtain a mapbox api key. Mapox api keys are free and can be created once you have an account. For more information, go here: <https://docs.mapbox.com/help/getting-started/access-tokens/>

Copy and paste that token into your **water-supply/index.html** file. Search for (ctrl+F) “mapoxgl.accessToken” and paste your token behind the equal sign.

### North Carolina API Key for North Carolina State Climate data

The state climate office in North Carolina provides an API key to access data. A free api key can be created here: <https://api.climate.ncsu.edu/>

Copy and paste your token into your **water-supply/rcodes/global0\_set\_apis\_libraries.R** script by finding (ctrl+F) the ncsco.key and pasting your api key where it says “API KEY HERE”. The api key should be wrapped in quotes (e.g. “abcdefg1234567”).

### NOAA API Key

If you are located in another state and want to access precipitation data, NOAA provides an api key here: <https://www.ncdc.noaa.gov/cdo-web/token>

The data can be accessed in R using the api key (or token) and the rnoaa library.

I have a version of this code that works… we can provide it here or let a student develop it for a different state.

# Finding Data

Finding data was largely a manual process based on stakeholder and IoW team knowledge. In the future, [Geoconnex](https://info.geoconnex.us/) will make finding these initial datasets easier. Data were obtained from federal, state, and local governments (Table 1).

**Table 1:** Data and sources

|  |  |  |
| --- | --- | --- |
| Layer | Description | Source(s) |
| Utilities | -Service area boundaries for drinking water systems -Water Shortage Plans  -Utility Demand Data | -[ABOUT-US](https://aboutus.internetofwater.dev/layers/aboutus_data:geonode:nc_statewide_CWS)  -[Local Water Supply Plans](https://www.ncwater.org/WUDC/app/LWSP/search.php)\* -Utility provided |
| County | County boundaries | [NC OneMap](https://www.nconemap.gov/datasets/NCDOT::ncdot-county-boundaries) |
| River Basin | HUC6 river basin boundaries | [USGS Watershed Boundary Dataset](https://www.usgs.gov/core-science-systems/ngp/national-hydrography/access-national-hydrography-products) |
| Sub-Basins | HUC8 watershed boundaries | [USGS Watershed Boundary Dataset](https://www.usgs.gov/core-science-systems/ngp/national-hydrography/access-national-hydrography-products) |
| Water Source | North Carolina water supply watersheds  **DOES THIS EXIST FOR TEXAS?** | [NC DEQ](https://data-ncdenr.opendata.arcgis.com/datasets/nc-surface-water-supply-watersheds) |
| Rivers | Major rivers in North Carolina  **FIND TEXAS VERSION** | [NC DEQ](https://data-ncdenr.opendata.arcgis.com/datasets/majorhydro) |
| Stream Gages | USGS Stream Gauge Locations | [USGS National Water Information System](https://waterdata.usgs.gov/nc/nwis)\* |
| Reservoirs | US Army Corps of Engineers reservoir locations  **- PULL USBR DATA FOR TEXAS** | [USACE Water Management Data Dissemination](https://water.usace.army.mil/a2w/f?p=100:1:0:::::) |
| Groundwater | -USGS Groundwater Data  -**CHECK FOR TX Groundwater Data** | -[USGS National Water Information System](https://waterdata.usgs.gov/nc/nwis/gw)\*; -[Groundwater Database](https://www.ncwater.org/?page=343)\* |
| Weather Stations | -NOAA National Climate Data CenterU  -**CHECK FOR TX STATE CLIMATE OFFICE** | [NOAA Climate Data Online](https://www.ncdc.noaa.gov/cdo-web/)+  [Climate Office Unified Data System](https://api.climate.ncsu.edu/locations) |
| Drought | US Drought Monitor | [GIS Data Files](https://droughtmonitor.unl.edu/Data/GISData.aspx) |
| Precip 7-day Observ | NOAA Advanced Hydrologic Prediction Service | [7 Day Observed Precipitation](https://water.weather.gov/precip/)\* |
| Precip 7-day %Normal | NOAA Advanced Hydrologic Prediction Service | [7 Day Observed Precipitation](https://water.weather.gov/precip/)\* |
| Precip 6-10 Day Forecast | NOAA Climate Prediction Center | [6 to 10 Day Outlooks](https://www.cpc.ncep.noaa.gov/products/predictions/610day/) |
| Temp 6-10 Day Forecast | NOAA Climate Prediction Center | [6 to 10 Day Outlooks](https://www.cpc.ncep.noaa.gov/products/predictions/610day/) |
| QPF 7-day Forecast | The National Weather Service Weather Prediction Center’s Quantitative Precipitation Forecasts | [7 Day Total QPF](https://www.wpc.ncep.noaa.gov/qpf/day1-7.shtml) |

# Running R Scripts

## Manual Steps 🡪WE ARE SKIPPING ANYTHING WITH UTILITY DEMAND

The script currently filters the utility shapefile geodatabase for those selected by the state. The file that does this is the basic\_info.csv. If you want to filter the utility layer to include a subset of data, then create the **basic\_info.csv** file. The file **must** contain the following columns:

* pwsid in the format of capital state abbreviation + 7 digits. (e.g. NC0332010)
* utility\_name in the format desired to display on the dashboard (this may come from demand data)
* data = a “yes”/”no” column that tells the dashboard whether to make the utility clickable on the map.

In North Carolina, there was a desire to link the utilty to their water supply watershed. We manually created the **link\_pwsid\_watershed.csv** file. This file **must** contain:

* pwsid in the format of capital state abbreviation + 7 digits. (e.g. NC0332010)
* utility\_name in the format desired to display on the dashboard (this may come from demand data)
* ws\_watershed is the name of the water supply watershed in the shapefile provided by the state. There is no unique identifier for the water supply watersheds and so the name must match exactly.

## Order of scripts

The scripts are labeled in the order they need to run. If the scripts can be run in any order, they are assigned the same number.

There are three groups of code:

1. The first script sets up the R environment, libraries, and variables needed to run all the other scripts.
2. The second group of scripts are called “access” and are designed to access and build the historic database. These scripts take longer to run and only need to be used when building the initial database.
3. The third group of scripts are called “use” scripts. These scripts use the data. They also access new data since the last time the script was run. These scripts can be run on a daily, weekly, or monthly basis to keep update the data in the dashboard.

If you need to build the historic database, then you will run the scripts in this order:

🡪global0\_set\_apis\_libraries.R

🡪access1\_static\_map\_layers.R

🡪access2\_historic\_demand\_data.R. SKIP

🡪access2\_historic\_streamflow\_data.R

🡪access2\_historic\_precip\_data.R 🡪 pull from TX state climate office or build on code I started to pull NOAA data (noaa\_webserivces\_initial\_pull.R) 🡪 will need to figure out how to take that and integrate it to match.

🡪access2\_historic\_reservoir\_data.R. Add USBR and integrate with Corps

\*Find out how far back USBR data goes in API… if only to 2015 or so then we won’t worry about historic corps data… if goes back in time then I will help you

🡪access2\_historic\_groundwater\_data.R 🡪 will need to figure out for Texas or from national groundwater monitoring network (I could not get apis to work but they might have improved since then).

Then you will need to run the use scripts to calculate statistics and generate files for the dashboard.

🡪use1\_demand\_data.R. SKIP

🡪use1\_streamflow\_data.R

🡪use1\_precip\_data.R

🡪use1\_reservoir\_data.R

🡪use1\_groundwater\_data.R

If you only need to update the files then you run the following scripts in this order:

🡪global0\_set\_apis\_libraries.R

🡪use1\_streamflow\_data.R

🡪use1\_demand\_data.R

🡪use1\_precip\_data.R

🡪use1\_reservoir\_data.R

🡪use1\_groundwater\_data.R

## Running the code: loading libraries and variables used in several scripts

1. In R studio, open **global0\_set\_apis\_libraries.R**

* Add your API keys to the sript
* Change the state abbreviation and fips for your state.
* If you change the folder directory names, updated swd\_html to save the data in the appropriate data folder.
  + The Julian csv file is located in the data folder of the github repository. Make sure to include it in your folder.

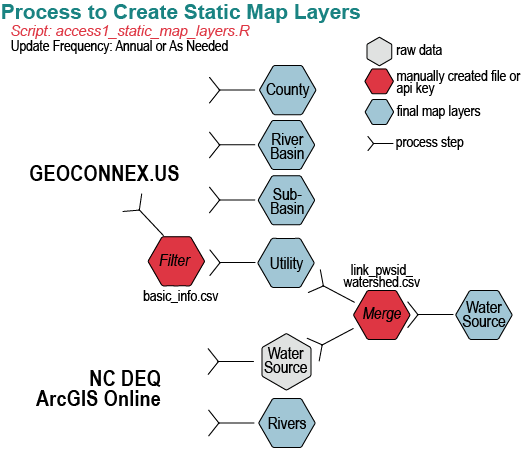
NOTE THAT THE LIST OF PWSID’S WAS MANUALLY CREATED. THIS MAY NOW BE CREATED IN THE DEMAND SCRIPT… IN WHICH CASE I WILL NEED TO CHANGE THE ORDER OF SCRIPTS.

## Running the code: creating static map layers and historic data

1. Open **access1\_static\_map\_layers.R**
   1. This files depends on “basic\_info.csv” and “link\_pwsid\_watershed.csv” files.
   2. You may need to update the basic\_info.csv to include more utility systems
   3. Most of the data come from [Geoconnex](https://geoconnex.internetofwater.dev/) and will have persistent links. You may need to update other urls if they change over time.

At the end of this script, you will create the following files:

|  |  |  |
| --- | --- | --- |
|  | **File Name** | **Description** |
| **Files needed** | ../data/basic\_info.csv | Used to filter to desired utilities in the state and whether demand data are available |
| ../data/link\_pwsid\_watershed.csv | Used to provide attributes linking utilities to the water source layer |
| **Files created** | ../data/county.geojson | Used to create the county map layer |
| ../data/huc6.geojson | Used to create the river basin map layer |
| ../data/huc8.geojson | Used to create the sub-basin map overlay |
| ../data/rivers.geojson | Used to create the rivers map overlay |
| ../data/  water\_supply\_watersheds.geojson | Used to create the water soure map overlay |
| ../data/nc\_utilities.geojson | Used to create the utilities map overlay |



**Figure X**. The process of acessing data from different sources and the map layers created.

1. Open **access2\_historic\_demand\_data.R**

**TBD**

1. Open **access2\_historic\_streamflow\_data.R**

This script uses USGS’s R package to find currently active stream gages with more than 5 years of data (this value can be adjusted for your needs). The script loops through each site to access daily streamflow data. The variables used to create the url that will pull daily mean discharge data include: pcode (discharge data), scode(mean data), and serv (daily). We limited data from 1990 onwards, but the start year could be changed based on preference in the global0\_set\_apis\_libraries.R script. The script saves out the location of stream sites and daily streamflow data available from 1990 onwards. (Table X).

All variables to adjust location and period of record are found in the global0\_set\_apis\_libraries.R script. No adjustments should be needed in this script.

|  |  |  |
| --- | --- | --- |
|  | **File Name** | **Description** |
| **Files created** | ../data/streamflow/  stream\_gauge\_sites.geojson | Used to create the location of stream gauges on the map |
| ../data/streamflow/all\_stream\_data.csv | Daily streamflow data from 1990 to present |

1. Open **access2\_historic\_reservoir\_data.R**

This script uses Army Corps API to pull historic data from Army Corp reservoirs providing water supply in North Carolina. The current Army Corps API only provides data from 2015 onwards; however, the Nicholas Institutes’ Water Program had previously done research with the Corps and has daily reservoir data from the start of reservoir construction to present time and the location of sites (Table X). This script combines the historic data with new data available since 2015.

The user will need to download and combine data for reservoirs of interest from here: <https://nicholasinstitute.duke.edu/reservoir-data/>. Select a reservoir in the map and the reservoir name will appear to download the data in the drop-down menu below. The user will need to know the three digit code for their district, which can be found in the reservoir data.

The user can change the “last\_number” and the “last\_unit” to determine how much data they need to download from the Corps API. The new data are stitched together with the old data. The old data assumes the same operational targets are present as in 2012. Those assumptions will need to be checked.

|  |  |  |
| --- | --- | --- |
|  | **File Name** | **Description** |
| **Files Needed** | ../data/reservoirs/matchNID\_LocID.csv | Matches the National Inventoryof Dams ID (used by the historic reservoir dataset) with the Location ID used by the Corps to incorporate data older than 2015 |
| ../data/reservoirs/usace\_dam.csv | Historic reservoir daily data |
| **Files created** | ../data/reservoirs/usace\_sites.geojson | Used to create the location of reservoirs on the map |
| ../data/reservoirs/usace\_dams.csv | Combines daily reservoir data prior to 2015 with data from 2015 to present. |

1. Open **access2\_historic\_precip\_data.R**

This script accesses the North Carolina State Climate Office precipitation data from two networks (COOP and ECONET) that provide daily precipitation data through an API. The API limits the number of data calls each month. It required 3 months of effort to pull together the historic state data base for daily precipitation data from 1990 to present. This script creates a file with the location of the weather station and a file with the daily precipitation data.

The user should provide their api key in the global0\_set\_apis\_libraries.R and the start date to gather data for the period of interest. No other modifications should be required for this script.

**Table X.** The files (data) needed for access2\_historic\_precip\_data.R and the files (data) created by this script.

|  |  |  |
| --- | --- | --- |
|  | **File Name** | **Description** |
| **Files Needed** | ../data/pcp/ncsu\_api\_locations.csv | Used to provide the location ids for desired precipitation gages. This was manually created. |
| **Files created** | ../data/pcp/pcp\_locations.csv | Used to create the location of stream gauges on the map |
| ../data/pcp/pcp\_data.csv | The historic precipitation data available for sites from 1990 to the day this script was run. |

1. Open **access2\_historic\_groundwater\_data.R**

This script begins by manually creating a list of groundwater sites from the National Groundwater Monitoring Network: <https://cida.usgs.gov/ngwmn/>. Unfortunately, we were not able to use the API to download data. In North Carolina, data were provided by the USGS and NC. We pulled the groundwater data from the USGS using their API and scraped the data from the NC DEQ database. The URL links for the NC DEQ database appear to change over time; therefore, the script finding those urls must be run in both the access and use files. This script is fragile and may require more upkeep.

**Table X.** The files (data) needed for use1\_streamflow\_data.R and the files (data) created by this script.

|  |  |  |
| --- | --- | --- |
|  | **File Name** | **Description** |
| **Files Needed** | ../data/gw/ALL\_SITE\_INFO.csv | Provides site information and ids from the National Groundwater Monitoring Network that are used to pull data from the North Carolina DEQ database. |
| ../data/julian-daymonth.csv | Manually created and used to link julian days with clean datea formats on charts. |
| **Files created** | ../data/gw/gw\_sites.csv | CSV file of groundwater sites with location and metadata. |
| ../data/gw/all\_gw\_levels.csv | Daily groundwater levels over time |

## Adding new data and transforming data into format used in the dashboard

1. Open **use1\_demand\_data.R**

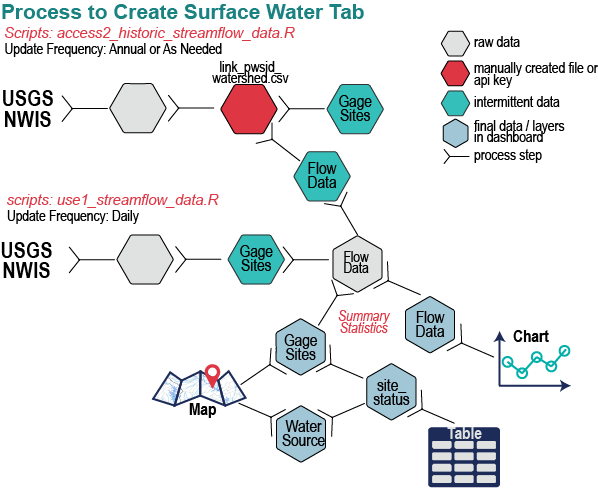
WILL INCLUDE UPDATING CONSERVATION STATUS TOO AND ALL INITIAL UTILITY INFO… WILL REPLACE MANUAL SECTION UP TOP I SUPPOSE

1. Open **use1\_streamflow\_data.R**

The script pulls daily or instantaneous data for the selected sites, calculates the 7-day rolling average, and calculates the streamflow statistics for each day of the year. . The files created are used to generate map layers, charts, and tables in the Surface Water Tab of the dashboard (Table X; Figure X).

**Table X.** The files (data) needed for use1\_streamflow\_data.R and the files (data) created by this script.

|  |  |  |
| --- | --- | --- |
|  | **File Name** | **Description** |
| **Files Needed** | ../data/streamflow/  stream\_gauge\_sites.geojson | Created in **access2\_historic\_streamflow\_data.R** and used to determine which gages to obtain new data. |
| ../data/  water\_supply\_watersheds.geojson | Created in **access1\_static\_map\_layers.R** and used to identify streamflow conditions based on a utilities water supply watershed. |
| ../data/streamflow/all\_stream\_data.csv | Contains all historic daily streamflow data pulled from **access2\_historic\_streamflow\_data.R** and any new data that has been pulled by running this script. |
| **Files created** | ../data/streamflow/all\_stream\_data.csv | Builds on the historic streamflow database. |
| ../data/streamflow/stream\_stats.csv | Summary statistics of the 7day rolling average streamflow for each day of the year. It includes the 7-day daily average flow for the last two years as well for charts in the Surface Water Tab. |
| ../data/streamflow/site\_status.csv | Provides the status of stream gages by watershed and is used to create the table in the Surface Water Tab. |
| ../data/streamflow/  stream\_gage\_sites.geojson | Geojson file of stream sites with the current streamflow condition included as an attribute. |



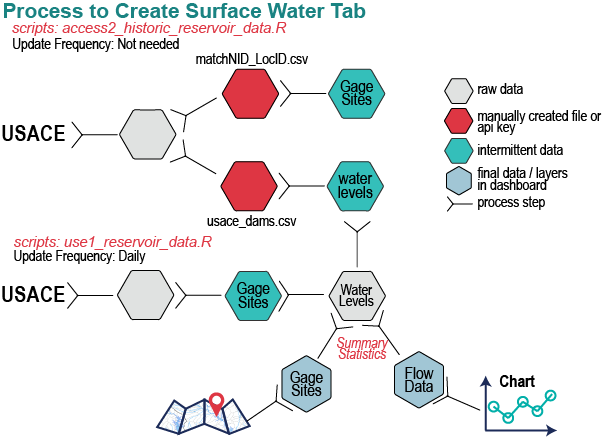
**Figure X:** Process of creating streamflow files used in the map and Surface Water Tab of the North Carolina Water Supply Dashboard. Notice that the table only works if the utility and water supply watersheds have been linked together.

**use1\_reservoir\_data.R**

This script uses the US Army Corps of Engineers API to update reservoir data, continously updating the daily reservoir database. The script calculates the percent of the reservoir that is full relative to operational target for any given day of the year (e.g. rule curve or guide curve). The reservoir is considered full at 100% of the operational target, below full when less than 100%, and above full at more than 100%. The files created are used to generate map layers, charts, and tables in the Surface Water Tab of the dashboard (Table X; Figure X).

**Table X.** The files (data) needed for use1\_streamflow\_data.R and the files (data) created by this script.

|  |  |  |
| --- | --- | --- |
|  | **File Name** | **Description** |
| **Files Needed** | ../data/reservoirs/  usace\_sites.geojson | Created in **access2\_historic\_reservoir\_data.R** and used to show location on maps with current conditions. |
| ../data/reservoirs/usace\_dams.csv | Created in **access2\_historic\_reservoir\_data.R** and provides historic reservoir level data from the period of record to the date of last being udpated. |
| ../data/julian-daymonth.csv | Manually created and used to link julian days with clean datea formats on charts. |
| **Files created** | ../data/reservoirs/usace\_dams.csv | Contains all historic daily reservoir data pulled from **access2\_historic\_reservoir\_data.R** and any new data that has been pulled by running this script. |
| ../data/reservoirs/  reservoir\_stats.csv | Summary statistics of the percent of storage (100% is full storage) for each day of the year. It includes the percent volume for the last two years as well for charts in the Surface Water Tab. |
| ../data/reservoirs/  usace\_sites.geojson | Geojson file of reservoir site with the current conditions included as an attribute. |



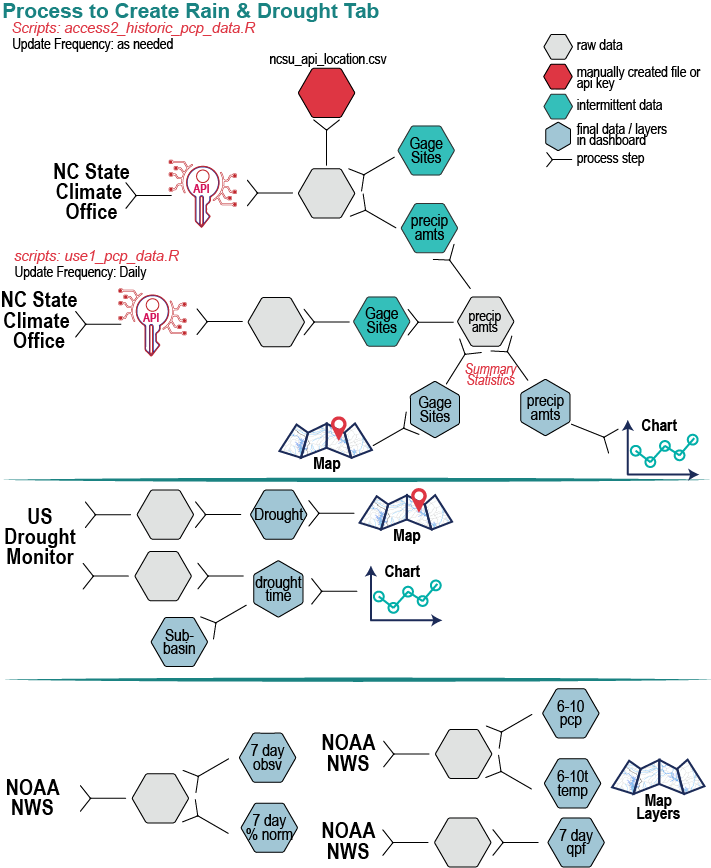
**Figure X:** Process of creating reservoir files used in the map and Surface Water Tab of the North Carolina Water Supply Dashboard.

**use1\_precip\_data.R**

This script pulls spatial weekly map from the drought monitor and updates the table. The drought monitor map updates weekly while the remaining precipitation data may update daily. The forecasts data are provided as kmz layers from different sources and we convert them into geojsons. Lastly, the precipitation data consists of 7 day observed precipitaiton and 7 day percent of normal precipitation spatial data, as well as daily precipitation data from the NC State Climate office. The daily precipitation data are provided as the total precipitation for each month and the cumulative precipitation for each year by site. The files created are used to generate map layers, charts, and tables in the Rain & Drought Tab of the dashboard (Table X; Figure X).

**Table X.** The files (data) needed for use1\_precip\_data.R and the files (data) created by this script for drought, forecast, and daily precipitation data.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **File Name** | **Description** |
| **Drought** | **Files Needed** | ../data/huc8.geojson | Created in access1\_static\_map\_layers.R and is used to identify sub-basins in North Carolina. |
| ../data/drought/  percentAreaHUC.csv | Created earlier using this script, but going back to the start of data (2000). Pulling earlier data makes the script run faster when updating. |
| **Files Created** | ../data/drought/  current\_drought\_geojson | Provides the map layer showing drought for the current week (full US coverage). |
| ../data/drought/  percentAreaHUC.csv | Provides the percent area of the sub-basin in different levels of drought and used to create the table in the Rain & Drought Tab. |
| **Forecasts** | **Files Created** | ../data/pcp  /pcp610forecast.geojson | Provides the 6 to 10 day precipitation forecast in the US in terms of wetter or drier than average map overlay. |
| ../data/pcp/  temp610forecast.geojson | Provides the 6 to 10 day temperature forecast in the US in terms of hotter or colder than average. |
| ../data/pcp/  qpf1-7dayforecast.geojson | Provides the 7-day total precipitation forecast. |
| **Precipitation** | **Files Needed** | ..data/pcp/  ncsu\_triangle\_locations.csv | Created in access2\_historic\_precip\_data.R and provides the location of weather stations and unique IDs for getting new data. |
| ..data/pcp/  ncsu\_triangle\_data.csv | Created in access2\_historic\_precip\_data.R and provides the historic precipitation data from weather stations. |
| **Files Created** | ../data/pcp/  pcp\_7day\_obsv.geojson | Seven day observed precipitation map overlay |
| ../data/pcp/  pcp\_7day\_percent\_normal  .geojson | Seven day percent of normal map overlay |
| ..data/pcp/ncsu\_  triangle\_locations.geojson | Provides the locaiton of weather stations and current conditions (based on cumulative precipitation to date). |
| ../data/pcp/  ncsu\_triangle\_data.csv | Continues to add to the time series of daily precipitation data. |
| ../data/pcp/  pcp\_months\_total.csv | Total precipitation for each month and year. Used to create the top chart in the Rain & Drought Tab. |
| ../data/pcp/  pcp\_cum\_total.csv | Cumulative precipitation for each day of the year. Used to create the bottom chart in the Rain & Drought Tab. |



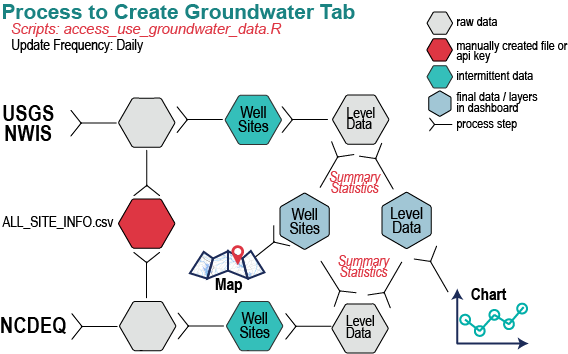
**Figure X:** Process of creating drought, forecast, and precipitation data used in the map and Rain & Drought Tab of the North Carolina Water Supply Dashboard.

**use1\_groundwater\_data.R**

This script begins by pulling in the site locations and historic groundwater level data pulled in the original script. The groundwater data are scraped and there is no way to only update recent files for DEQ data. Additionally, the unique url for NC DEQ files seems to change over time. This requires scraping the current URL and then pulling the data. The files created are used to generate map layers, charts, and tables in the Groundwater Tab of the dashboard (Table X; Figure X).

**Table X.** The files (data) needed for use1\_streamflow\_data.R and the files (data) created by this script.

|  |  |  |
| --- | --- | --- |
|  | **File Name** | **Description** |
| **Files Needed** | ../data/gw/gw\_sites.csv | The groundwater sites found in NC. |
| ../data/gw/all\_gw\_levels.csv | The historic groundwater levels accessed in **access2\_historic\_groundwater\_data.R**. |
| **Files created** | ../data/gw/gw\_sites.geojson | Geojson file of groundwater sites with the current conditions included as an attribute. |
| ../data/gw\_stats.csv | Summary statistics of groundwter levels for each day of the year. Many wells have sporadic data, making summary statistics difficult on a daily basis. Used for charts in the Groundwater Tab. |
| ../data/gw/gw\_levels\_time.csv | Daily groundwater data over time and plotted based on historic levels for that day in a year. |
| ../data/gw/gw\_annual\_level.csv | Plots the median groundwater depth below surface for all monitoring observations of a site each year. |



**Figure X:** Process of creating groundwater data used in the map and Groundwater Tab of the North Carolina Water Supply Dashboard.

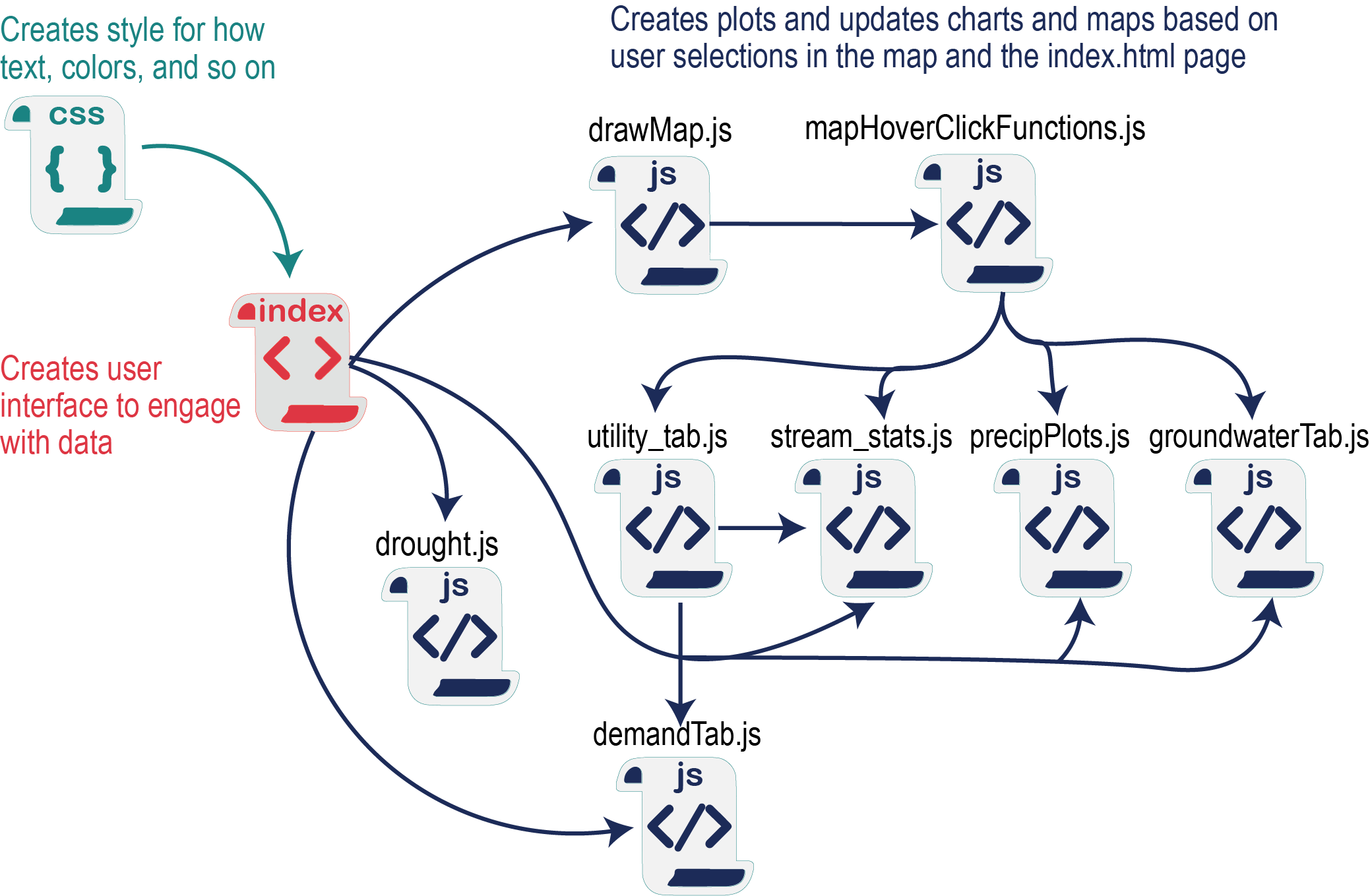
# Visualizing the Data

The dashboard was built using html, css, and javascript. It pulls in the files created by the R scripts. The Rscripts that are “using” data can be automated to run on a daily or weekly schedule to update the data in the dashboards. The scripts may break if the data providers change their API’s or are missing data for a given date. End users can build a variety of visualizations on the platform of their choice using the same underlying data.

The following scripts create the dashboard.

The user must create a mapbox account and get an api key (free for so many uses of the website). They must place their key in the line with: mapboxgl.accessToken – “putTokenHereInQuotations”

The only update that needs to be made annually in in the index.html file is to add additional checkbox years to both the checkPCPYear form and the checkDemandYear forms. Remove the “checked” status from the previous year and add it to the new year.



**Figure X:** Diagram showing how the different scripts interact with each other. Details are provided below.

# CSS File

..//css//main\_content.css

This file contains classes for divs and stylizes how many of the elements render

# HTML Index File

The head of the file calls scripts used to:

* create the map (mapbox)
* create plots (plot.ly and d3)
* create the dashboard elements and format (bootstrap)
* create the fonts (gstatic and googeapis)
* call functions (jquery)

The body sets the placement of divs and global variables used in javascript.

Creates code to draw original map and establish the geocoder.

Creates the code that links together the map layers and the tabs.

# Javascript Files

There are a series of javascript files responsible for creating the plots and maps.

1. drawMap.js
   1. Contains the function **drawMap**.
      1. This draws the map using the preloaded utility spatial layer.
   2. This layer loads in and creates all the layers that can be added to the map
   3. Contains the code that turns layers on and off based on user selection
      1. $(‘button’).on(‘click’, function(){…}
   4. Contains the code to create map legends and turn those legends on and off
2. mapHoverClickFunctions.js
   1. Contains the functions for hovering and clicking on the utility and block group layers.
      1. Includes info boxes
      2. Includes rules for clicking on the top most layer only
      3. Contains the rules for filtering water supply watersheds based on the utility selected
      4. Contains rules for highlighting selected utilities, stream gages, precipitation layers, and so on in the map. Contains the scripts to create the chart headers.
   2. Returns variables based on selection of stream gages (streamID), groundwater wells (gwID), and weather stations (pcpID)
3. utility\_tab.js
   1. Contains the function toTitleCase(str) which will capitalize the first letter of a string
   2. Contains the function myUtilityInfo(myUtility) to highlight the selected utility and zoom to that map layer
      1. Returns the myUtility variable
   3. Creates the text and table seen in the Utility Tab
4. demandTab.js
   1. contains the function: createTraceDemand() to draw traces based on user selection of the check box
      1. returns checkedDemand
   2. contains the function: createDemandInfo(myUtilityID, checkedDemand)
      1. creates the text and plot in the Demand Tab
5. stream\_stats.js
   1. contains the function createBlankSummary() to create the table in the streamstats prior to a utility being selected
   2. contains the function createCurrentSummary(myUtility) to create the table that counts the number of stream gages in each water supply watershed of the selected utility
   3. contains the function toggleStats(target) to let users toggle between two different views of the streamflow chart
      1. returns streamPlotType variable
   4. contains the function createDailyStatistics(streamID, streamPlotType) to draw the streamflow and supporting chart text
6. droughtPlot.js
   1. contains the function setHUCThis(target) to determine which HUC time series to plot
      1. return myHUC
   2. contains the function plotDroughtTime(myHUC) to plot the selected huc data and create the table showing what percent of the watershed is in different drought conditions
7. precipPlots.js
   1. contains function createTracePCP(target) to draw the traces selected in the checkbox
      1. returns checked variable
   2. contains the function plotPrecipitation(pcpID, checked) to draw the monthly and cumulative precipitation plots based on the site selected in the map and the checkbox selection.
8. groundwaterTab.js
   1. contains the function toggleGWstats to determine which plot type to draw
      1. returns gwPlotType
   2. contains the createGWTab(gwID, recentDate, gwPlotType) function to create the groundwater plots

To do: Add image showing how Github, Docker, R, and Javascript/HTML/CSS all interact to create dashboards