






nasaaccess - Installation Guide


[nasaaccess](#)






Select Watershed Boundary



 Upload New Watershed

Select DEM



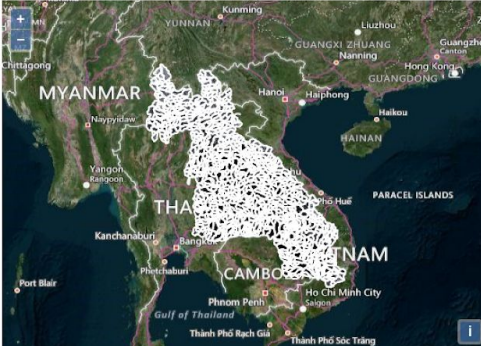
 Upload New DEM

Select Date Range

to

Select Functions

Function	Description
<input checked="" type="checkbox"/> GLDASpolycentroid	Create daily air temperature time-series files at the centroid of each polygon within the selected boundary. Generated from NASA GLDAS remote sensing products.
<input type="checkbox"/> GLDASwat	Create SWAT-compatible daily air temperature daily time-series files evenly distributed (on a grid) over the selected boundary. Generated from NASA GLDAS remote sensing products.
<input type="checkbox"/> GPMpolycentroid	Create daily rainfall time-series files at the centroid of each polygon within the selected boundary. Generated from NASA GPM remote sensing products.
<input type="checkbox"/> GPMswat	Create daily rainfall time-series files evenly distributed (on a grid) over the selected boundary. Generated from NASA GPM remote sensing products.



Run nasaaccess

Download Data

Overview

nasaaccess is a software tool built in R that streamlines the retrieval and processing of the global NASA earth observation data products (GPM and GLDAS) for use in models such as SWAT. The core functionality of nasaaccess can be summarized as:

- Access the NASA Goddard Space Flight Center (GSFC) servers to download earth observation data
- Clip needed grids to an input shapefile of a user study watershed
- Handle temporal and spatial inconsistencies
- Generate daily climate gridded data files and definition files compatible with SWAT and other models


nasaaccess was built as an R library containing the four separate data processing functions described in the table below. It is a very efficient system for accessing earth observation data. The nasaaccess web application was built to allow users to access the full functionality of the nasaaccess package without needing a working knowledge of R.

Function	Definition
GLDASwat	Generates SWAT compatible air temperature files
GPMswat	Generates SWAT compatible precipitation files
GLDASpolycentroid	Generates an air temperature station file at the centroid of each polygon within the input watershed boundary
GPMpolycentroid	Generates a precipitation station file at the centroid of each polygon within the input watershed boundary

How it works

The nasaaccess web application is simply a user interface for passing arguments into the nasaaccess functions.

Using a combination of dropdowns, datepickers, and checkboxes, the app allows the user to select a watershed boundary, DEM, daterange, and nasaaccess function(s) to pass to the server for running the selected nasaaccess function(s).


nasaaccess
?
×

UNDER DEVELOPMENT: GPM and GLDAS Poly Centroid functionality will be available soon

Select Watershed Boundary

Select Boundary Shapefile

Upload New Watershed

Select DEM

Select DEM


Upload New DEM

Select Date Range

Start Date
 to
 End Date

Select Functions

Function	Description
<input type="checkbox"/> GLDASpolycentroid	Create daily air temperature time-series files at the centroid of each polygon within the selected boundary. Generated from NASA GLDAS remote sensing products.
<input type="checkbox"/> GLDASwat	Create SWAT-compatible daily air temperature daily time-series files evenly distributed (on a grid) over the selected boundary. Generated from NASA GLDAS remote sensing products.
<input type="checkbox"/> GPMpolycentroid	Create daily rainfall time-series files at the centroid of each polygon within the selected boundary. Generated from NASA GPM remote sensing products.
<input type="checkbox"/> GPMswat	Create daily rainfall time-series files evenly distributed (on a grid) over the selected boundary. Generated from NASA GPM remote sensing products.



Run nasaaccess

Download Data

Normally, the nasaaccess functions take a considerable amount of time to run. For this reason, the app requests the user's email address when the user submits a job. After entering an email, the job is sent to the server for processing and the user is able to leave the app. When the process has completed, an email is automatically sent to the user's email with instructions on how to download their data and a unique access code for them to enter into the app.

Contact Information
 ×

Depending on the watershed size and the date range you selected, the nasaaccess process may take some time. Please provide your email here and we will contact you with a 6 digit access code for downloading your data when the process is complete.

Email Address:

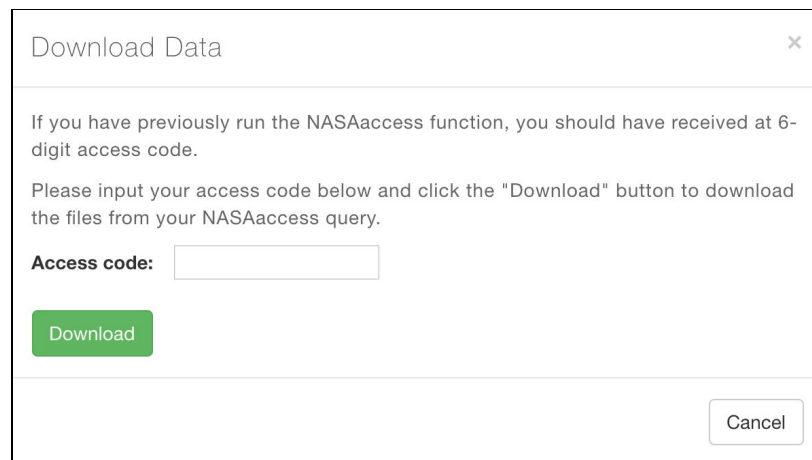
Your email will only remain in our database until your access code has been sent out. The access code you receive will then be valid for 3 days.

Submit

Cancel

Hello,
 Your nasaaccess data is ready for download at <http://tethys-servir-mekong.adpc.net/apps/nasaaccess>
 Your unique access code is: **00SGL2**

The “Download Data” button allows users to download the data that was created from their request.



The screenshot shows a dialog box titled "Download Data" with a close button (X) in the top right corner. The dialog contains the following text: "If you have previously run the NASAaccess function, you should have received a 6-digit access code." and "Please input your access code below and click the 'Download' button to download the files from your NASAaccess query." Below this text is a label "Access code:" followed by a text input field. At the bottom left of the dialog is a green button labeled "Download", and at the bottom right is a button labeled "Cancel".

Installation/Setup Instructions

1 Create an EarthData account

The nasaaccess functions download and process raw rainfall and air temperature data from NASA’s EarthData website. For these functions to work, the app needs to have an authorized account in the EarthData system. The following steps outline the process of setting that account up and linking the account to the app.

Create account

To create an account, follow the instructions found [here](#):

Link Earth Data account credentials to server

Now that you have an account, follow the instructions [here](#) ('curl for Mac/Linux') to ensure that the nasaaccess app knows which account to use to download data.

2 Clone app

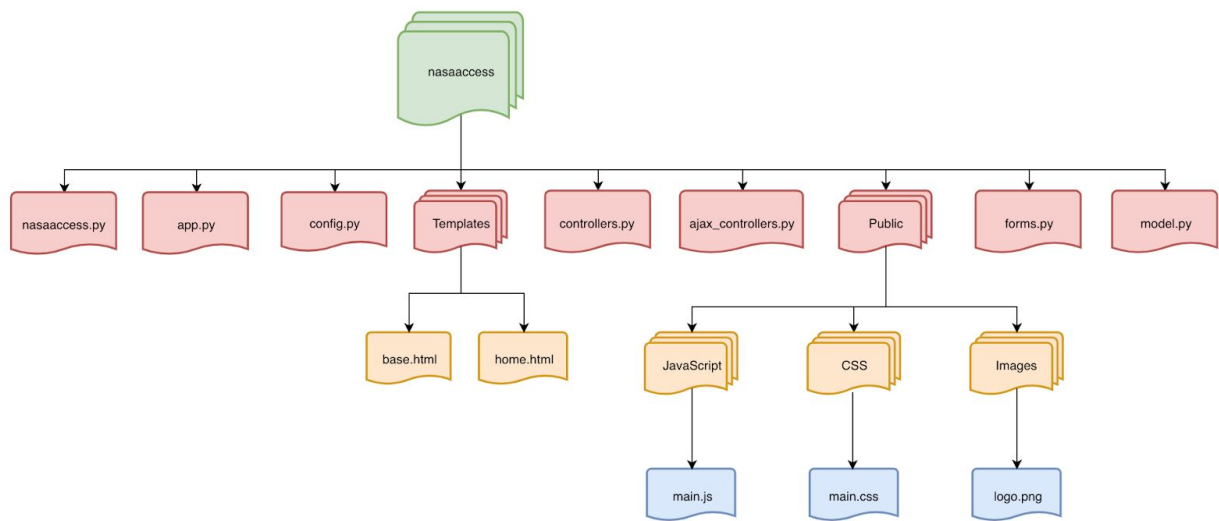
Within the 'tethys' python environment, clone the nasaaccess git repository to the folder you have decided to use for storing the app packages. If the apps will be installed on a production server, this folder should be located at `/home_directory/tethys/apps/`. Use the following command to clone the app.

```
git clone https://github.com/imohamme/tethys_nasaaccess.git
```

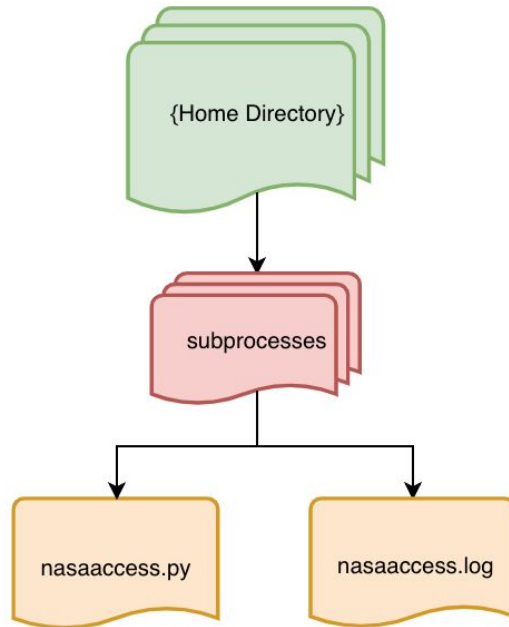
If you plan to make edits to the app and would like to track your own development within the app, forking the repository to your own github account is recommended.

The app package

After cloning the app onto the server/computer, the following file structure should be created in the tethysapp-nasaaccess folder.



Important note: the nasaaccess.py file (first file on the red level) is a python3 file containing all of the nasaaccess functions (see next section for more details). This script runs “detached” from the app, and so it is recommended to copy it to a separate folder (in the image it was place in a folder called “subprocesses” in the home directory). This will make it easier for the app to reference that file later.



Wherever you end up copying the nasaaccess.py file to, create a new empty text file called nasaaccess.log. The app will use this file to create a running log of how the app is being used and will assist in debugging if issues arise.

Python package prerequisites

Within the 'tethys' environment, install the following packages using the command below:

```
conda install -c conda-forge package_name
```

shapely=1.5.17
rasterio=0.36.0
netcdf4=1.2.8
pandas=0.23.4
geopandas=0.4.0
georaster=1.25
xarray=0.10.9
ncurses=6.1
requests=2.18.4

It is possible that other versions of these packages will work, but the versions specified above have been tested and have proven to work.

4 Set up requisite file structure on the server

The nasaaccess functions take five arguments.

Argument	Description	User Specified?
Output path	File path to the location where the function outputs will be written. The app will automatically generate this path when a user initiates the nasaaccess functions	No
Shapefile path	File path to the location containing the watershed boundary shapefile. The app will generate this path using the watershed name selected in the user interface	Yes, in code
DEM path	File path to the location containing the DEM TIFF file. The app will generate this path using the DEM name selected in the user interface	Yes, in code
Start date	First day that the user wants to obtain data for	Yes, in app interface
End date	Last day that the user wants to obtain data for	Yes, in app interface

For the nasaaccess app to work, it needs to know where to find these files. The following figures depict how the file structure should be set up on the server.

Data requirements and file structure

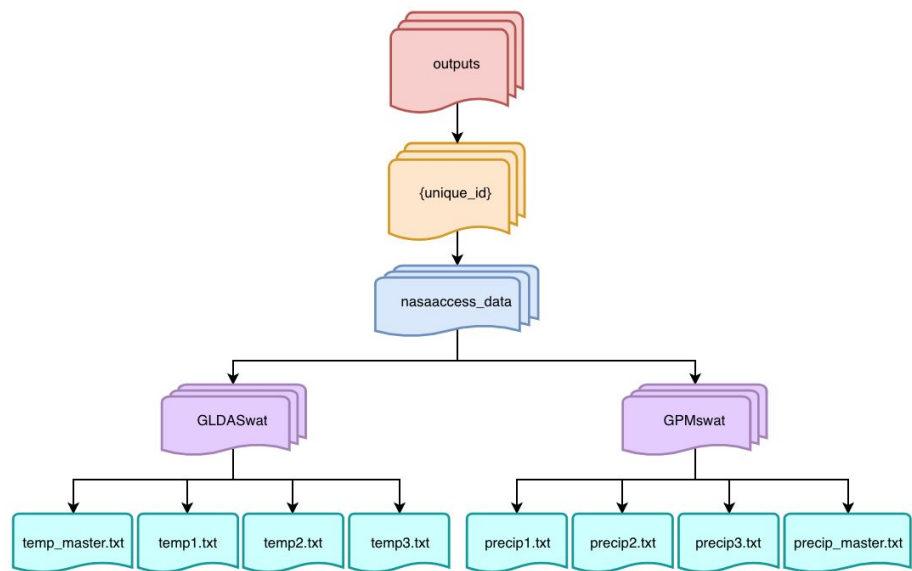
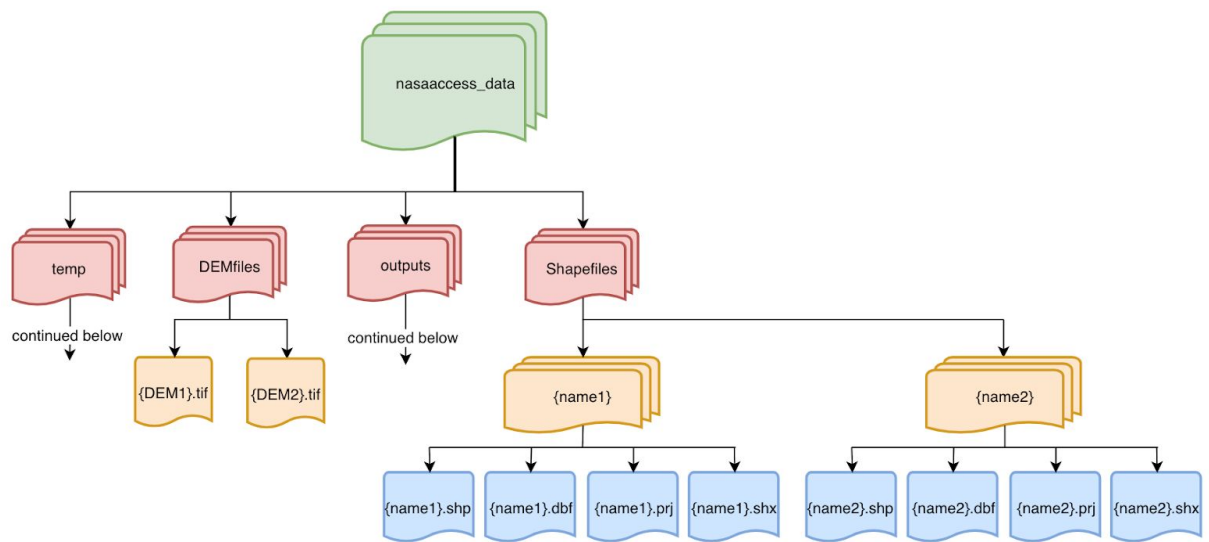
The nasaaccess web app uses two different folders to store and write data to: (1) the nasaaccess_data folder in the server and the user_workspace folder in the app package.

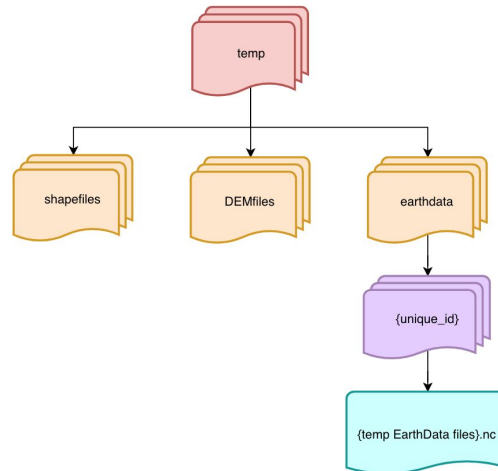
nasaaccess_data folder

The nasaaccess_data folder is where the app will look to find the input shapefile, DEM file, and output folder to write the outputs to.

Within nasaaccess_data, there are four subfolders:

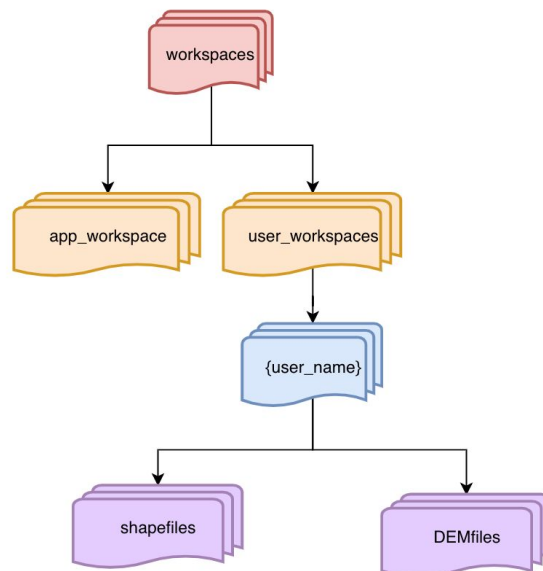
- 1) DEMfiles
 - a) Contains all the tif files that the app admin wants to make available for all users of the app
 - b) These files need to be in a geographic coordinate system for the functions to work
- 2) Shapefiles
 - a) Contains all the shapefiles that the app admin wants to make available for all users of the app
 - b) Each watershed/boundary is given it's own subfolder in the Shapefiles folder that contains the 4 separate shapefile component files (shp, shx, dbf, and prj).
 - i) the name of this folder is what will show up in the user interface dropdown and should be the same name used for each of the 4 files
 - c) These files need to be in a geographic coordinate system for the functions to work
- 3) Outputs
 - a) This is the folder that the app will write the function outputs to.
 - b) Each new user request from the app, creates a new uniquely name subfolder within the outputs folder and the outputs are written there
- 4) Temp
 - a) This is where all of the intermediate files (i.e. raw data netCDF files downloaded from EarthData) are stored.
 - b) These files will be deleted when the nasaaccess functions have completed so this folder should remain empty unless a process is running





User Workspace folder

The nasaaccess app also allows users to upload their own watershed boundary and DEM files. The app uses the “user_workspaces” folder (shown below) within the app package to store a specific user’s data for them to use in the future. When the app is opened, the watershed and DEM dropdown menus are automatically updated to include all of the files available within the nasaaccess_data folder on the server and the current user’s user_workspace.



5 Uploading shapefiles and tiff files to Geoserver

For files that will be available to all users

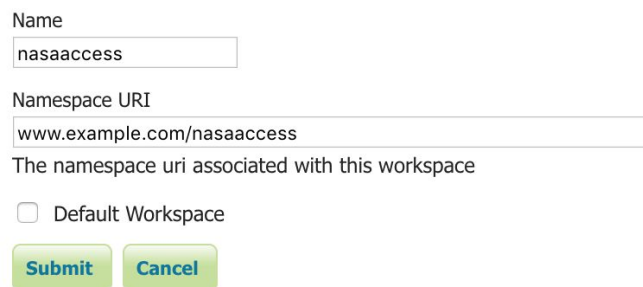
The nasaaccess app has a map built into its interface that allows users to view the shapefile and DEM that they selected. The map visualization does not affect the functionality of the app but allows the user to verify that their shapefile and DEM cover the same area (the shapefile needs to fit completely within the DEM).

To upload a new shapefile to the Geoserver, first zip the 4 shapefile component files (shp, shx, dbf, and prj) into a zip folder with the name of the shapefile (Should be the watershed name).

If it doesn't exist already, create a new workspace in the Geoserver called "nasaaccess".

New Workspace

Configure a new workspace



The screenshot shows the 'New Workspace' configuration form in Geoserver. It has two input fields: 'Name' with the value 'nasaaccess' and 'Namespace URI' with the value 'www.example.com/nasaaccess'. Below the URI field is a label 'The namespace uri associated with this workspace'. There is an unchecked checkbox labeled 'Default Workspace'. At the bottom are two buttons: 'Submit' and 'Cancel'.

Then in server/computer's terminal, navigate to the folder containing the zip file and run the following curl command after replacing the names in {curly brackets} with the information specific to the file being uploaded.

```
curl -v -u {user}:{pass} -XPUT -H "Content-type: application/zip" --data-binary @{Zip_File_Name}.zip  
{geoserver_url}:{port}/geoserver/rest/workspaces/{Workspace_name}/datastores/{Zip_File_Name}/file.shp
```

Uploading a new DEM TIFF file is very similar to uploading a shapefile. There is no need to zip the TIFF file.

If the 'nasaaccess' workspace already exists on the Geoserver, navigate to the folder containing the TIFF file and run the following command after replacing the names in {curly brackets} with the information specific to the file being uploaded.

```
curl -v -u {user}:{pass} -XPUT -H "Content-type: image/tiff" --data-binary @{TIFF_File_Name}.tif  
{geoserver_url}:{port}/geoserver/rest/workspaces/{Workspace_name}/datastores/{TIFF_File_Name}/file.geotiff
```

For files uploaded by specific users

The app features two separate file upload features that allows users to upload their own shapefiles and DEM tiff files. Included in the file upload functions are geoserver upload capabilities. In other words, the app will take care of uploading user-uploaded shapefiles and DEMs.

6 Reference file paths and Geoserver URL in config.py and main.js

The nasaaccess_data folder and the nasaaccess.py script can be placed anywhere on the server as long as their paths are referenced in the config.py file in the app package as shown below. The config.py is used by the app to locate the various scripts and files that it needs.

```
data_path = os.path.join('/Users/imohamme/Documents/Web_NASAaccess/nasaaccess_data/')  
nasaaccess_py3 = os.path.join('/Users/imohamme/anaconda3/envs/tethys/bin/python3')  
nasaaccess_script = os.path.join('/Users/imohamme/Documents/Web_NASAaccess/subprocesses/nasaaccess.py')  
nasaaccess_log = os.path.join('/Users/imohamme/Documents/Web_NASAaccess/subprocesses/nasaaccess.log')  
geoserver = {'rest_url': 'http://localhost:8080/geoserver/rest/',  
             'wms_url': 'http://localhost:8080/geoserver/wms/',  
             'user': 'admin',  
             'password': 'geoserver',  
             'workspace': 'nasaaccess',  
             'URI': 'nasaaccess'}
```

- 1) data_path: path to the top level of the nasaaccess_data directory.
 - a) If this folder is structured as shown above, the app will know how to access all the data it contains
- 2) geoserver: Specify URLs, username, password, and workspace that the app will use to display and upload spatial data
- 3) nasaaccess_py3: Path to the python3 executable in the nasaaccess python environment.
- 4) nasaaccess_script: Path to the nasaaccess.py script
- 5) nasaaccess_log: Path to the nasaaccess.log file

The geoserver WMS URL and workspace also need to be referenced in the main.js file as shown below.

```
var current_layer,
    element,
    layers,
    map,
    public_interface,           // Object returned by the module
    variable_data,
    wms_workspace,
    geoserver_url = 'http://localhost:8080/geoserver/wms',
    gs_workspace = 'nasaaccess',
    wms_url,
    wms_layer,
    wms_source,
    basin_layer,
    dem_layer,
    featureOverlaySubbasin,
    subbasin_overlay_layers,
    geojson_list;
```

7 Install the app

Once the config.py and main.js files have been updated with the new file paths and URLs, the app is ready to be installed.

Local (Development) Installation

Within the tethys environment, run the setup.py script as follows:

```
python setup.py develop
```

Production Installation

```
tuo
```

```
python setup.py install
```

```
tethys manage collectall
```

```
tso
```

```
tsr
```

8 Set file permissions for the app

Read and write permissions and ownership requirements

In a production installation of the nasaaccess application, there are various file permission and file ownership changes that need to be made to the server. When the app calls the nasaaccess.py script, it tries to run the script as the 'www-data' user instead of the the root user. Unless you change the ownership of that script to 'www-data' the server will not allow the script to run. In addition, because the app will be reading and writing new files to the server, full read, write, and execute permissions (rwx) need to be given to the folders containing data relevant to the app.

Below is a list of the directories that need to have ownership and permissions changed

Folder	Owner (user:group)	rwx
/path/to/nasaaccess_data/folder/*	www-data:www-data	777
/path/to/tethys/apps/*	www-data:www-data	775
/path/to/tethys/src/*	www-data:www-data	775
/path/to/tethys/static/*	www-data:www-data	775
/path/to/tethys/anaconda3/envs/*	www-data:www-data	775
/path/to/tethys/anaconda3/envs/tethys/lib/python3.7/site-packages/tethys_app/*	www-data:www-data	777
/path/to/nasaaccess_app/workspaces/*	www-data:www-data	777

* Include all files within that folder

note: 777 corresponds to full read, write, execute permissions. 775 corresponds to limiting write permissions for non-owner users

To change the ownership of a folder and all the subfolders and files it contains:

- 1) Navigate to that folder in the terminal
- 2) Run the following command:

```
sudo chown -R www-data:www-data .
```

To change the read-write-execute permissions for a folder and all the subfolders and files it contains:

- 1) Navigate to that folder in the terminal
- 2) Run the following command:

```
sudo chmod -R 0777 .
```

File upload size requirements

The file upload (both shapefile and DEM) features in the app rely on a number of settings that are available to change within the `global_settings.py` file in tethys portal package.

The `global_settings.py` file is located in the following file path:

```
~/tethys/miniconda/envs/tethys/lib/python2.7/site-packages/django/conf/global_settings.py
```

Open the file to edit and change the following settings if they don't already match what is shown here:

```
MEDIA_ROOT = ''
```

```
DATA_UPLOAD_MAX_MEMORY_SIZE = 100000000 # i.e.  
100MB
```