

Breakout Session

Getting to know IN-CORE 2.0

1:00 - 2:30 pm on 5/3/2019 (Friday)
NCSA Team, incore-dev@lists.illinois.edu

Things you will learn during this session:

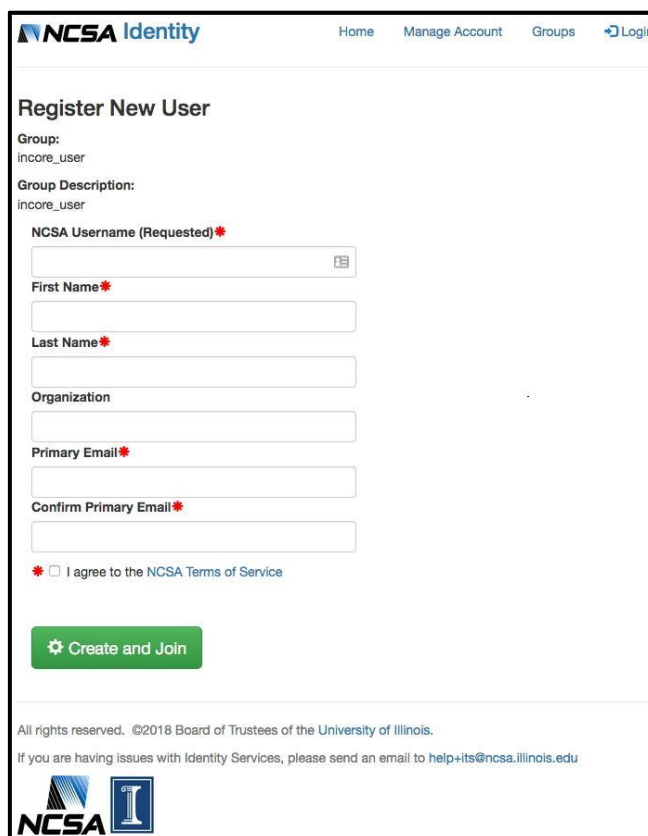
- How to acquire IN-CORE account
- How to install pyIncore
- How to use IN-CORE lab
- How to contact and work with NCSA

Table of Contents

1	Acquiring IN-CORE account.....	2
2	Installing pyIncore	3
2.1	Prerequisites.....	3
2.2	Installing pyIncore Package	5
3	Testing pyIncore Installation.....	7
3.1	Running a Building Damage Analysis Locally	7
4	Using IN-CORE Lab.....	9
4.1	Running Jupyter Notebook in IN-CORE Lab	9
5	How to Contact and Work with NCSA	11
6	Information for pyIncore Developers	11
7	Additional Information	12
7.1	Technical Documentation.....	12
7.2	IN-CORE Web tools.....	13
7.3	Additional Information about IN-CORE Lab.....	15

1 Acquiring IN-CORE account

- A user must have a valid IN-CORE account recognized by the IN-CORE service. Please **register** at <https://identity.ncsa.illinois.edu/register/UUMK36FU2M>
- Your credentials (Username and Password) will be required in later steps.



The screenshot shows the 'Register New User' page of the NCSA Identity service. The page has a header with the NCSA Identity logo and navigation links for Home, Manage Account, Groups, and Login. The main content area contains the following fields and options:

- Group:** incore_user
- Group Description:** incore_user
- NCSA Username (Requested):** A text input field with a red asterisk indicating it is required.
- First Name:** A text input field with a red asterisk indicating it is required.
- Last Name:** A text input field with a red asterisk indicating it is required.
- Organization:** A text input field.
- Primary Email:** A text input field with a red asterisk indicating it is required.
- Confirm Primary Email:** A text input field with a red asterisk indicating it is required.
- Agreement:** A checkbox labeled 'I agree to the NCSA Terms of Service' with a red asterisk.
- Create and Join:** A green button with a gear icon and the text 'Create and Join'.

At the bottom of the form, there is a footer with the following text:

All rights reserved. ©2018 Board of Trustees of the University of Illinois.
If you are having issues with Identity Services, please send an email to help+its@ncsa.illinois.edu

The footer also includes the NCSA logo and the University of Illinois logo.

NOTE: Use your institutional email if possible.

- The username/password is used for accessing IN-CORE services. You can test your credentials by accessing the **IN-CORE page** at <https://incore2.ncsa.illinois.edu/>.
- This is also used for accessing the **documentation** and downloading pyIncore package (**pyincore_0.3.0.tar.gz**) and Jupyter Notebook test file (**buildingdamage.ipynb**) at <https://incore2.ncsa.illinois.edu/>.

2 Installing pyIncore

2.1 Prerequisites

IN-CORE account

- A user must have an IN-CORE account. If you don't have an account, see IN-CORE account section above.

Python 3.5+ (<https://www.python.org>)

- If you are on Windows, go to Windows 64 bit section under GDAL section.
- It is common to have more than one Python version installed on your computer. Make sure you are running the correct version of Python (you can check by running `python -version`) with corresponding path added to the PATH system variable. The following links will help you navigate through various installations guides
 - <https://realpython.com/installing-python/>
 - <https://docs.python-guide.org/#the-hitchhiker-s-guide-to-python>
 - OS specific downloads: <https://www.python.org/downloads/>
- We recommend that users get familiar with virtual environments (<https://www.pythonforbeginners.com/basics/how-to-use-python-virtualenv/>) or environment manager (<https://www.anaconda.com/distribution/>);
 - These are tools that help keep dependencies separate for different projects. If you decide, however, to use a virtual environment or manager you must do it now, in this prerequisite step.
 - Note that GDAL installation is global on Windows and Linux, even if you use virtual environments (see next step).

GDAL (<https://www.gdal.org>) - Geospatial Data Abstraction Library

pyIncore uses the GDAL library, which has to be installed separately. The following are instructions for each operating system

- **Linux**

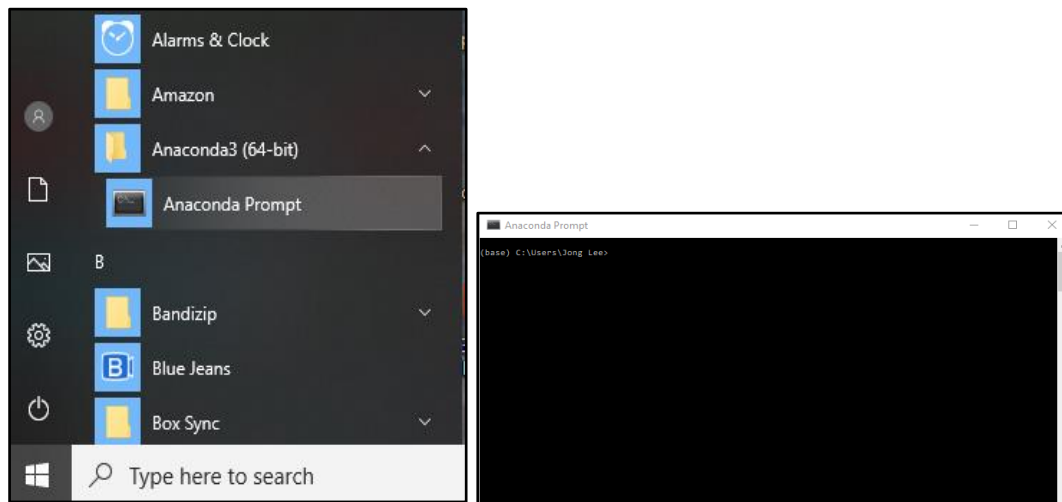
Additional information can be found at the wiki page How to install GDAL (<https://github.com/domlysz/BlenderGIS/wiki/How-to-install-GDAL>).

- Install **gdal-bin**
`sudo apt-get install gdal-bin`
- Install **libspatialindex-dev**
`apt-get install libspatialindex-dev`
- Also, update your Pip Python package manager
`pip3 install --upgrade pip`

- **Windows 64-bit** (The 32-bit has not been tested yet.)

We provide installation instructions for Anaconda environment manager using Miniconda (<https://docs.conda.io/en/latest/miniconda.html>). Python 3.x and GDAL library will be installed with Miniconda. The following instructions were tested for Windows 64-bit:

- Download the latest Miniconda3 installer for Windows from the Miniconda web page. (<https://docs.conda.io/en/latest/miniconda.html>)
- Run the installer setup locally (select the “Just Me” choice) to avoid the need for administrator privileges.
- Leave the default folder path (C:\Users\<user>\..\miniconda3).
- Do not add Anaconda to the PATH. Do, however, register Anaconda as the default Python environment.
- Open up an Anaconda prompt from the Windows Start menu:



- Create the python environment (required for the pyincore) and activate it:

```
conda create -n pyincore python=3
```

```
conda activate pyincore
```
- Install dependency packages in the following order:

```
conda install rasterio
```

```
conda install fiona
```

```
conda install rtree
```

- **MacOS**

Use Homebrew (<http://mxcl.github.com/homebrew/>), a MacOS package manager. If you don't have Homebrew, please install it. Additional information about installing GDAL can be found at https://medium.com/@vascofernandes_13322/how-to-install-gdal-on-macos-6a76fb5e24a4

- Install **gdal**:
`brew install gdal`
- Install **spatialindex** library
`brew install spatialindex`
- Also, update your pip Python package manager
`pip3 install --upgrade pip`

Jupyter (<https://jupyter.org/>)

We recommend using Jupyter Notebook for ease of running **pyIncore** projects. It is an open-source application that allows you to create projects (documents) that contain live Python code, visualizations and documentation.

- Installing Jupyter (<https://jupyter.org/install.html>) can be done again with `pip` (on Miniconda; in your virtual environment) or `pip3` as indicated below:
`pip3 install jupyter`

2.2 Installing pyIncore Package

These steps will guide you on how to install pyIncore.

1. Download pyIncore (pyincore_0.3.0.tar.gz) at <https://incore2.ncsa.illinois.edu/> to a directory on your computer.

2. To install pyIncore, navigate to the directory you used on step 1 and:

- From the Terminal (Mac/Linux) run:

```
pip3 install --user pyincore_0.3.0.tar.gz
```

Note: If you are using a virtual environment, you will need to activate it and then run the installation command.

- From the Anaconda prompt (Windows):
 - i. Activate the pyincore environment created on the *Prerequisite* section

```
conda activate pyincore
```

- ii. Run the following command:

```
pip install --user pyincore_0.3.0.tar.gz
```

3. The installation installs pyIncore and creates an `.incore` folder in your HOME directory to store cached files. A message “pyIncore credentials file has been created at <HOME

directory>/.incore/.incorepw” appears in the terminal/prompt. The typical location of a HOME directory is:

- C:\Users\- /Users/<username> on MacOS
- /home/<username> on Linux based machines

Linux specific notes

- If you see the `OSError: Could not find libspatialindex_c library file` error, make sure that you installed **libspatialindex-dev** (see GDAL section above)

MacOS specific notes

- We use the matplotlib library to create graphs. There is a Mac specific installation issue addressed at StackOverflow <https://stackoverflow.com/questions/4130355/python-matplotlib-framework-under-macosx> and <https://stackoverflow.com/questions/21784641/installation-issue-with-matplotlib-python>

In a nutshell, insert the line `backend : Agg` into the `~/matplotlib/matplotlibrc` file.

4. Locate a file called **.incorepw** in your HOME directory and write your LDAP credentials in it; the first line contains your username and the second password.
 - This information is used for communicating with IN-CORE web service.
 - The file is located in the `.incore` folder created during installation in your HOME directory.
 - The typical path is C:\Users\

3 Testing pyIncore Installation

- For these instructions we assume that users develop their python script by using pyIncore in their own **project folder** (create folder if you don't have one)
- Download the Jupyter Notebook file for Building damage analysis (<https://incore2.ncsa.illinois.edu/doc/examples/buildingdamage.ipynb>) to your **project folder**. We will verify your installation of pyIncore by running this file.

3.1 Running a Building Damage Analysis Locally

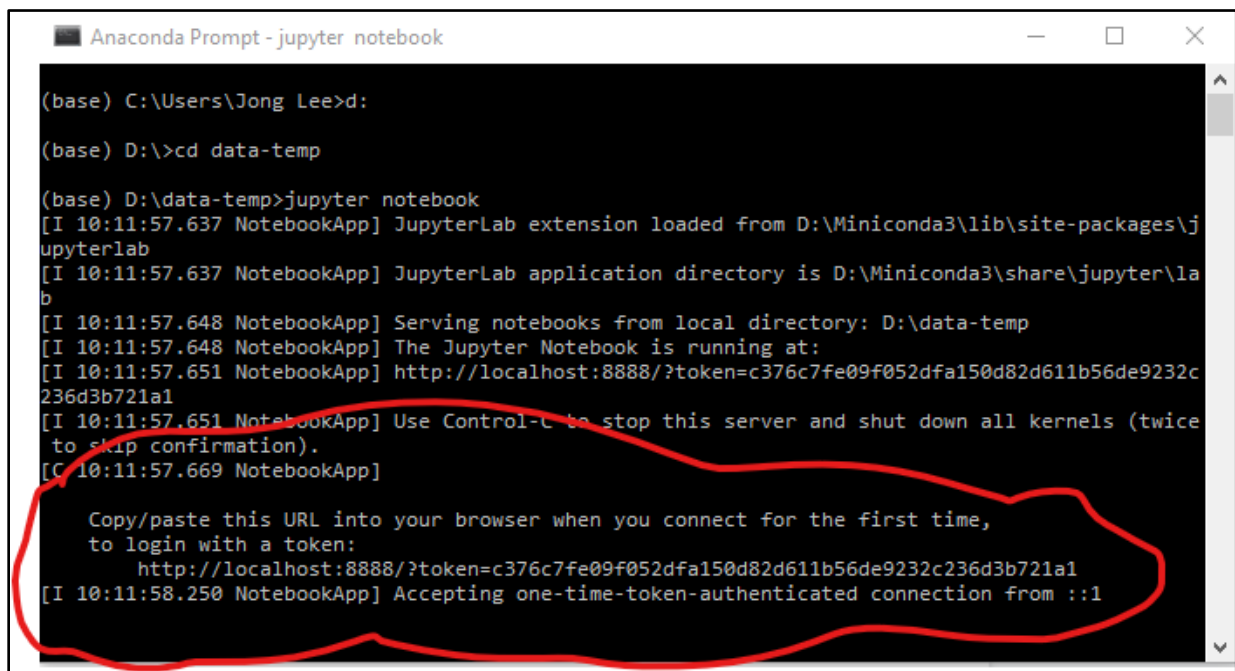
- Start a local **Jupyter Notebook** by running the following command in the terminal or command prompt from your **project folder** (change directories to the particular project folder at the command prompt):

```
jupyter notebook
```

or if **Jupyter Notebook** is not recognized in Anaconda

```
python -m notebook
```

A message “The Jupyter Notebook is running” appears in the terminal/prompt and you should see the notebook dashboard open in your browser. Note that you might be asked to copy/paste a URL into your browser when you connect for the first time as shown below:



```
Anaconda Prompt - jupyter notebook

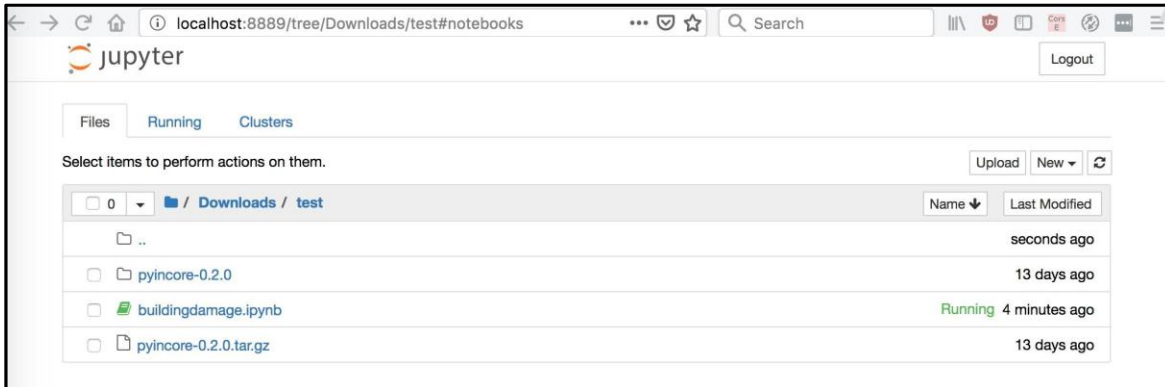
(base) C:\Users\Jong Lee>d:

(base) D:\>cd data-temp

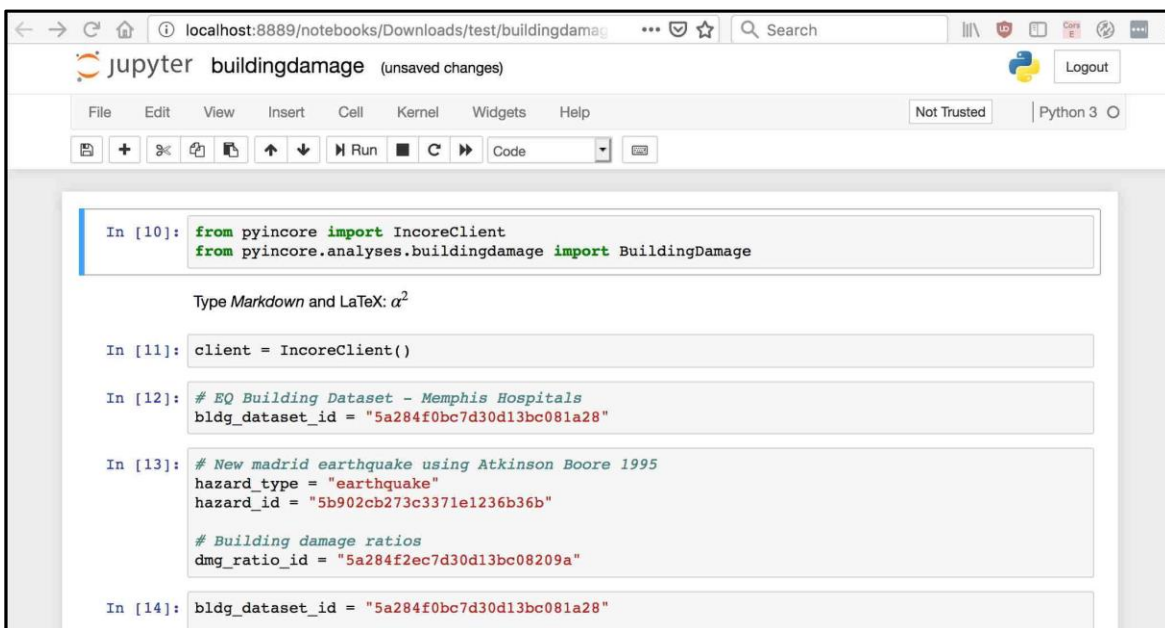
(base) D:\data-temp>jupyter notebook
[I 10:11:57.637 NotebookApp] JupyterLab extension loaded from D:\Miniconda3\lib\site-packages\jupyterlab
[I 10:11:57.637 NotebookApp] JupyterLab application directory is D:\Miniconda3\share\jupyter\lab
[I 10:11:57.648 NotebookApp] Serving notebooks from local directory: D:\data-temp
[I 10:11:57.648 NotebookApp] The Jupyter Notebook is running at:
[I 10:11:57.651 NotebookApp] http://localhost:8888/?token=c376c7fe09f052dfa150d82d611b56de9232c236d3b721a1
[I 10:11:57.651 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 10:11:57.669 NotebookApp]

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
    http://localhost:8888/?token=c376c7fe09f052dfa150d82d611b56de9232c236d3b721a1
[I 10:11:58.250 NotebookApp] Accepting one-time-token-authenticated connection from ::1
```

- Click on the `buildingdamage.ipynb` in the Jupyter Notebook browser.



Your web page should now show multiple cells of code like this:



Right now you are not actually running a notebook yet. Running a cell means that you will execute the cell's contents. To execute cells in order you can just select the first cell and click the **Run** button at the top.


Note that **Building damage** is a long running analysis and there is little indication that it's running except by either looking at the Jupyter Notebook file and seeing the `[*]` for the notebook cell where that block of code is being executed or by looking at the Task Manager in the Notebook dashboard to see there is a python process running. Alternatively, you can look at the Jupyter Notebook dashboard to see if the `csv` file with results has been created yet.

For details of running and manipulating `ipynb` files refer to Jupyter documentation (<https://jupyter.readthedocs.io/en/latest/running.html#running>).

4 Using IN-CORE Lab

IN-CORE Lab is a customized Jupyter Lab for running and editing Notebooks accessible at <https://incore-lab.ncsa.illinois.edu>.

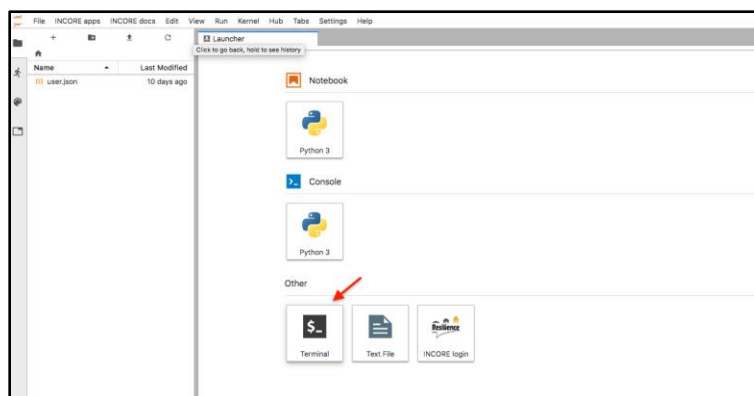
- Login to IN-CORE Lab with you IN-CORE account info (you created at the beginning of this session)

A sign-in form with an orange header bar containing the text "Sign in". Below the header, there are two input fields: "Username:" and "Password:". Each field has a small "x" icon on the right side. At the bottom of the form is an orange button labeled "Sign In".

4.1 Running Jupyter Notebook in IN-CORE Lab

In *Testing pyIncore Installation* section we described how to run Building damage Notebook locally. This section focuses on step-by-step instructions of running Notebooks on the IN-CORE Lab.

1. Create a credential file with IN-CORE username/password (same information you used to login to IN-CORE Lab) in order to use IN-CORE services. This is similar to the authentication step described in *Testing pyIncore Installation* section except the authentication file `.incorepw` is being created on the IN-CORE Lab server running Linux OS:
 - a. Open the terminal on IN-CORE lab Launcher page:



- b. In the terminal, make sure you are in your HOME directory. Type:
`pwd`

to see the current path and

```
cd ~
```

to get into your home directory (/home/<username>).

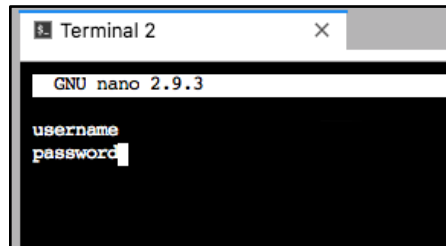
- c. Create a hidden (therefore dot prefix) folder:

```
mkdir .incore
```

- d. Create a hidden credential file in the folder you just created and type IN-CORE username and password using for example **nano** text editor:

```
cd .incore
```

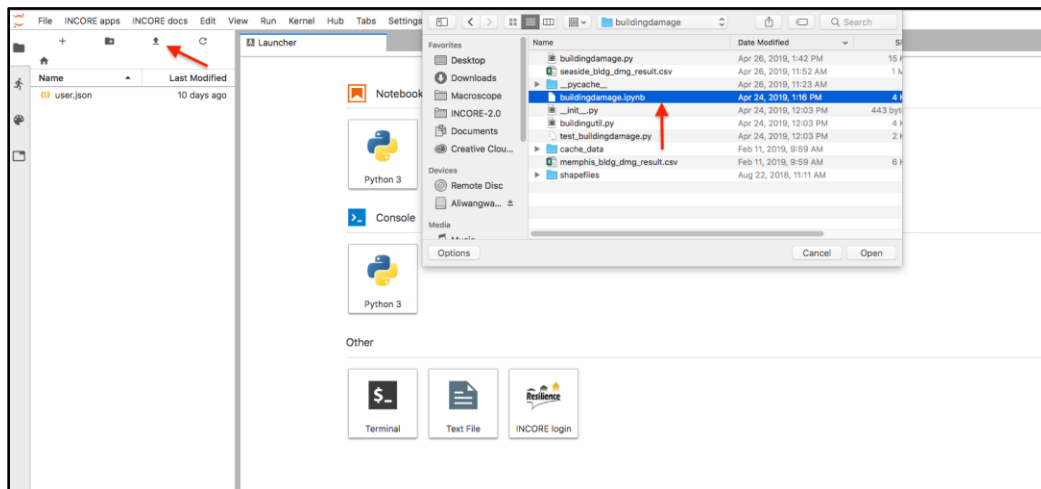
```
nano .incorepw
```



- e. Save the file with **Ctrl+O** and **Enter** commands

- f. Close the text editor and return to your shell with **Ctrl+X** command

2. Upload the Building Damage Notebook from your local machine to IN-CORE lab by clicking the **Upload** icon in the left panel and select **buildingdamage.ipynb**.



3. The building damage Notebook shows up in the left panel after a successful upload.
4. Double click to open it in the main area.

```

1 from pyincore import IncoreClient
2 from pyincore.analyses.buildingdamage import BuildingDamage
3
4 # Type Markdown and LaTeX: q2
5
6 client = IncoreClient()
7
8 # ID Building Dataset - Memphis Hospitals
9 bldg_dataset_id = "5a284f8bc7d38c13b485a28"
10
11 # New madrid earthquake using Atkinson House 2995
12 hazard_type = "Earthquake"
13 hazard_id = "59982c673c3371e12386366"
14
15 # Building damage ratios
16 dmg_ratio_id = "5a284f8bc7d38c13b485a28"
17
18 bldg_dataset_id = "5a284f8bc7d38c13b485a28"
19
20 # Earthquake mapping
21 mapping_id = "55470583376a4362987672c"
22
23 # Run Memphis earthquake building damage
24 bldg_dmg = BuildingDamage(client)
25 bldg_dmg.load_remote_input_dataset("buildings", bldg_dataset_id)
26 bldg_dmg.load_remote_input_dataset("dmg_ratios", dmg_ratio_id)
27
28 result_name = "memphis_bldg_dmg_result"
29 bldg_dmg.set_parameter("result_name", result_name)
30 bldg_dmg.set_parameter("mapping_id", mapping_id)
31 bldg_dmg.set_parameter("hazard_type", hazard_type)
32 bldg_dmg.set_parameter("hazard_id", hazard_id)
33 bldg_dmg.set_parameter("num_cpu", 1)
34
35 # Run Analysis
36 bldg_dmg.run_analysis()
37
38 # Seaside example Tsunami
39 hazard_type = "Tsunami"
40 hazard_id = "5bc9e25e77d88533c7e188c"
41
42 # Seaside building dataset
43 bldg_dataset_id = "5bcf27c9f242f8d47ce798ad"

```

- Run it. For instructions on how to run building damage analysis, please refer to previous section *Running a Building Damage Analysis*.

5 How to Contact and Work with NCSA

- Contact an individual programmer developer by email and copy incore-dev@lists.illinois.edu if you work closely with NCSA on a code conversion and/or improvement of your hazard analysis.
- Contact the incore-dev@lists.illinois.edu email list if you do not work directly with NCSA.
- Response time during the week will be in approximately 24 hours or less. Weekend emails will be responded to on the next business day, typically Monday.

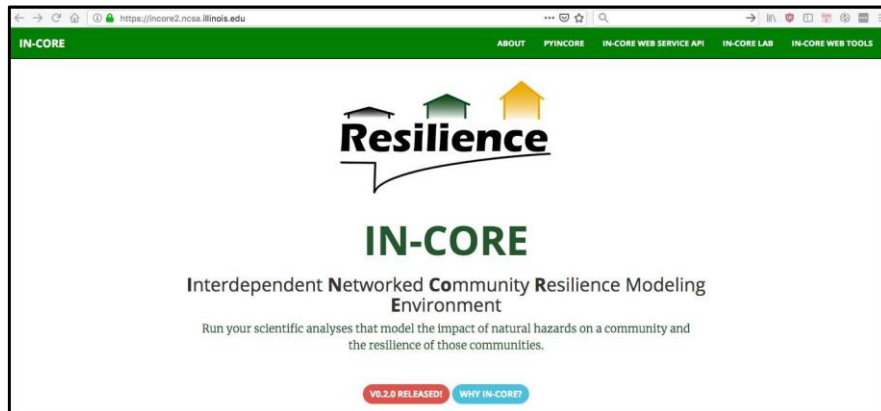
6 Information for pyIncore Developers

- IN-CORE programming guideline:
<https://opensource.ncsa.illinois.edu/confluence/display/INCORE1/IN-CORE+Programming+Guideline>
- Python programming tips:
<https://opensource.ncsa.illinois.edu/confluence/display/INCORE1/Programming+Tips+for+Python>

7 Additional Information

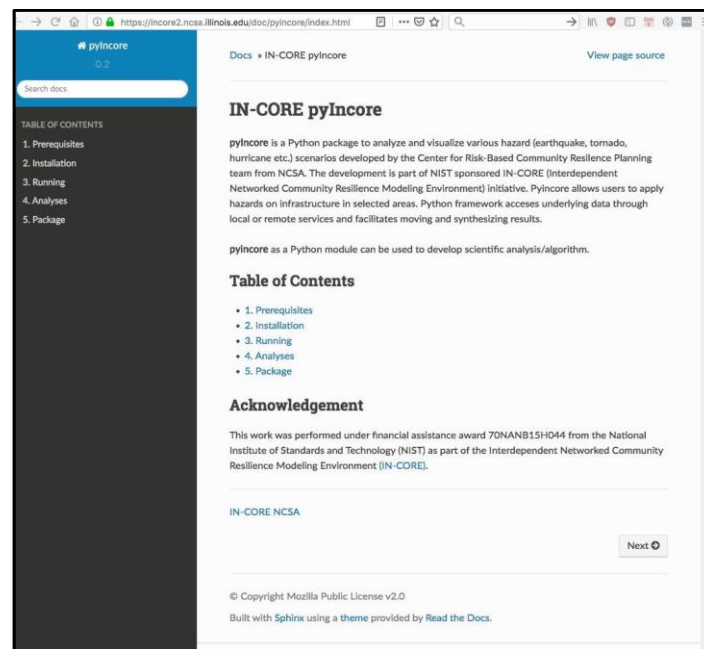
7.1 Technical Documentation

From the IN-CORE landing page at <https://incore2.ncsa.illinois.edu/> a user can access:

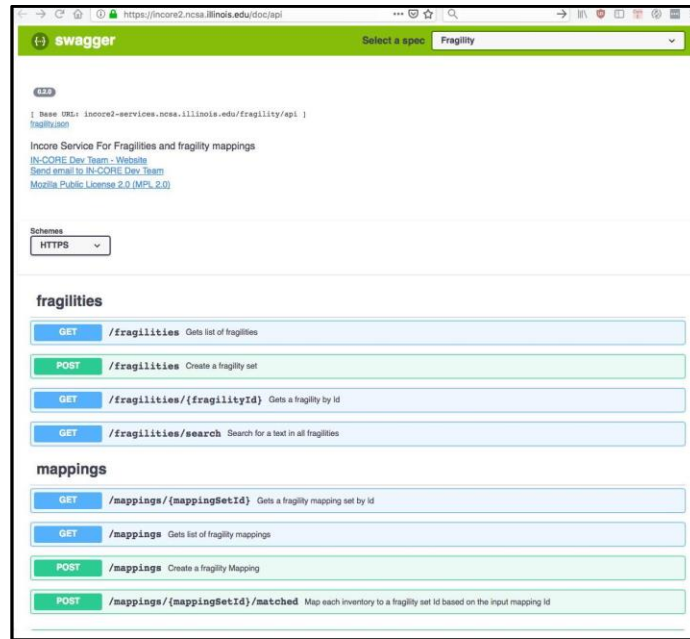


In this site, you can download pyIncore package and access to other services and documentations.

- Documentation of **pyIncore** is at <https://incore2.ncsa.illinois.edu/doc/pyincore/>



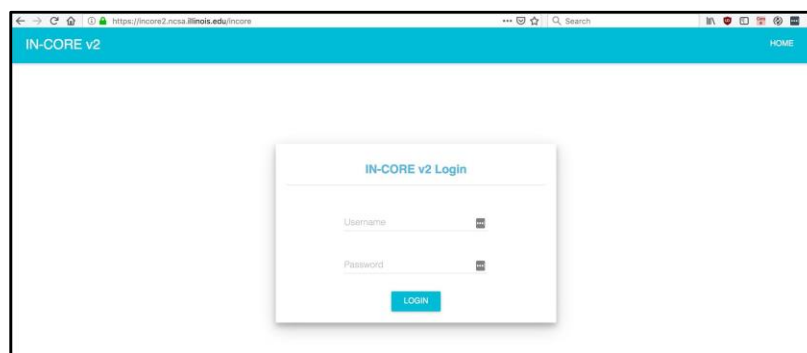
- Documentation of **IN-CORE Web Service** is at <https://incore2.ncsa.illinois.edu/doc/api/>



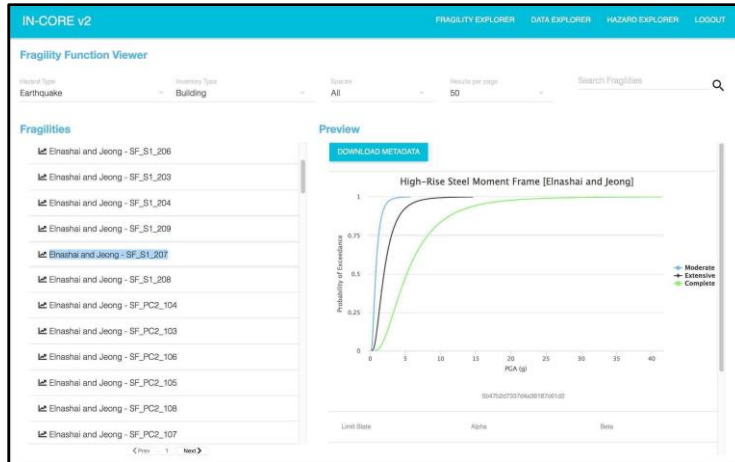
- IN-CORE Lab is at <https://incore-lab.ncsa.illinois.edu>
- IN-CORE Web Tools are for interacting with the service layer. They enable users to browse and search the **Datasets**, **Hazards** and **Fragilities**, view the metadata and visualizations, and download the datasets.

7.2 IN-CORE Web tools

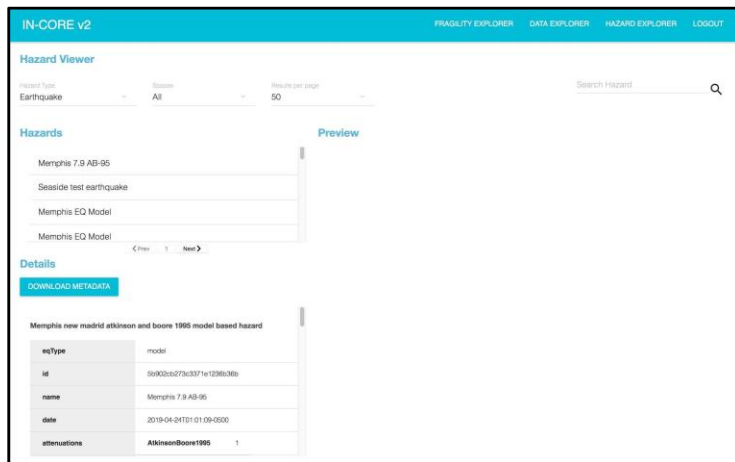
In-Core Web tools are a dashboard with viewers for various services. Currently these are **Fragility**, **Data** and **Hazard** services. A user must login with IN-CORE username and password in order to access the viewers:



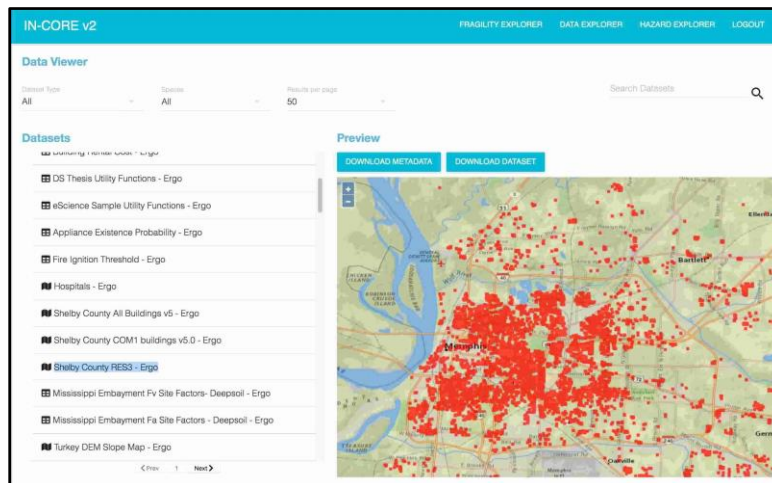
Fragility service. This is an example of a viewer showing a selection list (left) of Fragility curves. Hazards and types of structures are selected in the pull down menus. The data can be downloaded in json format.



Hazard service viewer.



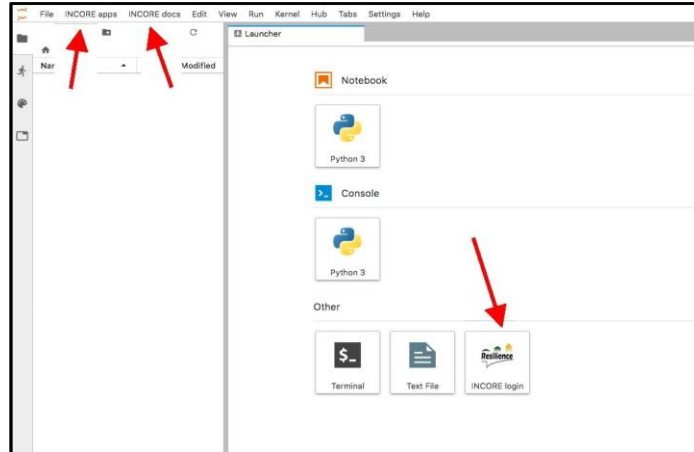
Data service viewer.



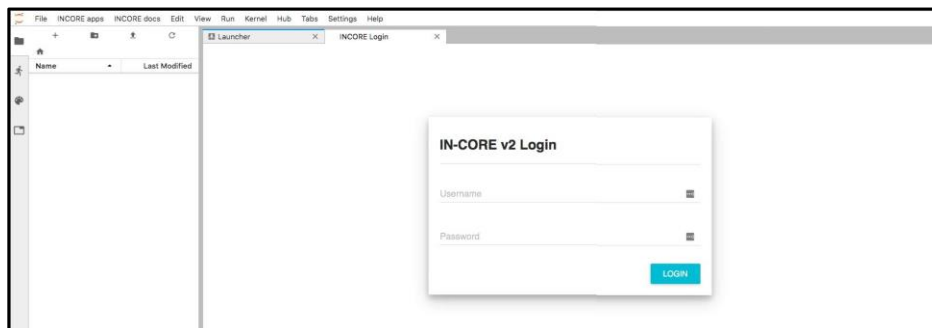
7.3 Additional Information about IN-CORE Lab

This section shows how to access IN-CORE Web Tools and documentations on IN-CORE Lab.

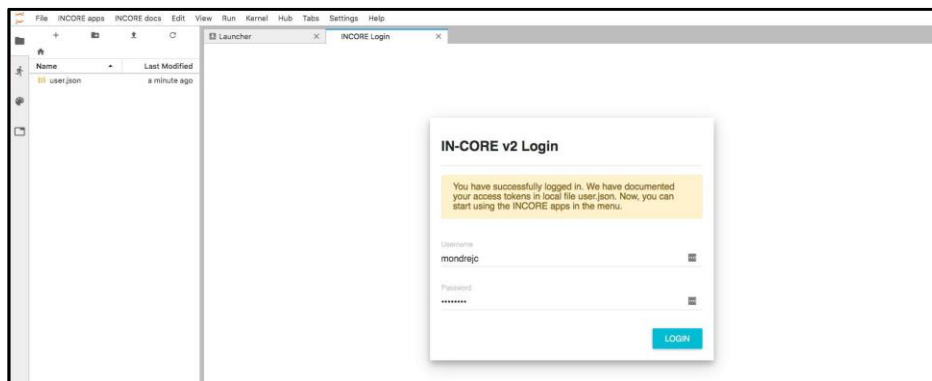
- Click on “INCORE Login” button as shown below.



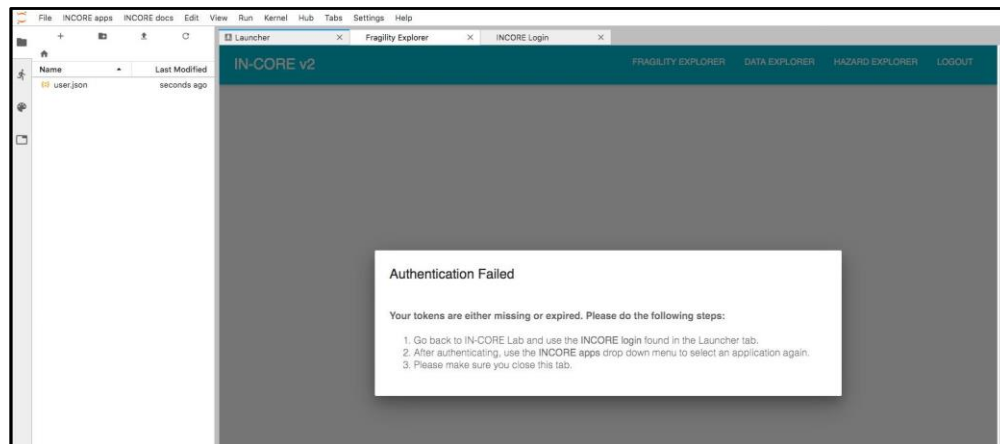
Same username and password for this part.



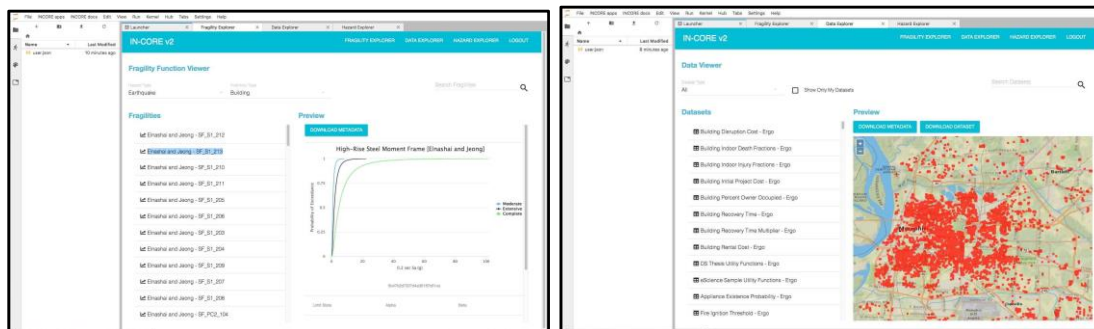
This login process generates a file named **user.json**. It appears in the File list manager on the left side. The file contains an authentication token required for development of new analyses using IN-CORE’s Application programming interface (API).



Fragility, Data and Hazard Explorers under INCORE apps menu become enabled after pressing LOGIN button AND reloading the current page in the browser. **NOTE:** A user must reload the whole Jupyter dashboard page (above) using the Reload button of the browser, not the Refresh File List (part of Jupyter's file navigation) otherwise a following Warning appears:



Viewer as part of INCORE Lab as shown below.



- The second IN-CORE menu (INCORE docs) allows user to see pyIncore documentation and API endpoints definitions for accessing Fragility, Data and Hazard server(s). Another IN-CORE login window opens up at the top of the browser's main window.

For ease of access - documentation is easily accessible from IN-CORE Lab.

