Getting to know IN-CORE 2.0

Last change 07/17/19
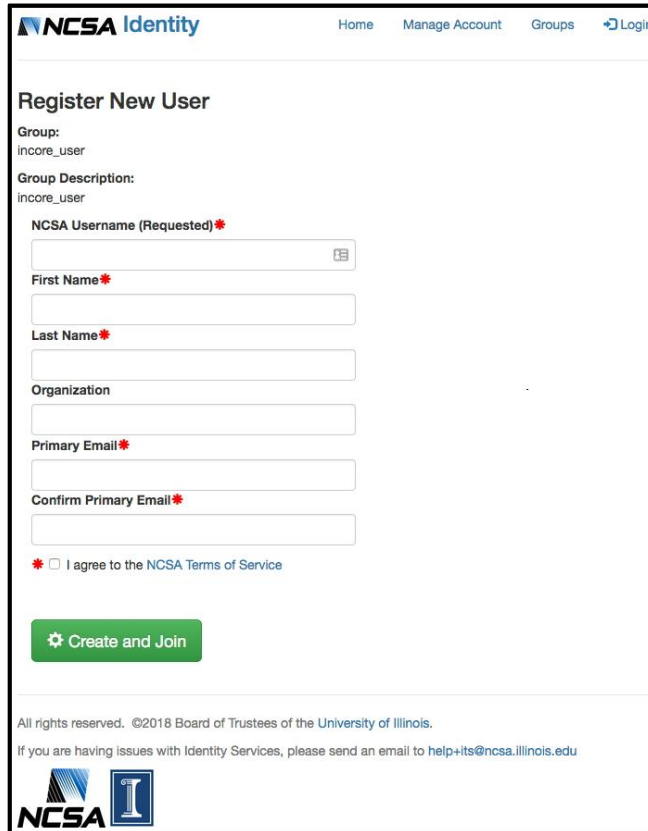
NCSA Team, incore-dev@lists.illinois.edu

Table of Contents

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

NOTE: Use your institutional email if possible.

- The username/password is used for accessing IN-CORE services. You can test your registration 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_<version>.tar.gz**) and Jupyter Notebook test file (**buildingdamage.ipynb**) at https://incore2.ncsa.illinois.edu/.

2 Installing pyIncore

2.1 Prerequisites

Please read through the instructions at least once completely before actually following them to avoid any installation problems!

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

Virtual environment
We recommend that users get familiar with Python virtual environment manager called Anaconda or Miniconda.
- This is a tools that help keep dependencies separate for different projects. If you decide, however, to use a virtual environment manager you must do it now, in this prerequisite step.
- An environment managers are available by downloading OS specific installers. Note that Anaconda/Miniconda distribution will include Python (Anaconda also includes a collection of over 1,500+ open source packages), so installing Python first isn't needed if you use Anaconda/Miniconda. With Anaconda you already have installed Jupyter notebook. The `conda` is the preferred interface for managing installations and virtual environments with the Anaconda/Miniconda Python distribution.

Python 3.5+ (https://www.python.org)
- It is common to have more than one Python version installed on your computer. Make sure you are running the correct, Anaconda version of Python (you can check by running `python –version` and/or `import sys;sys.executable` in Python console)

Jupyter notebook (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. Jupyter Notebook is already installed with Anaconda distribution; it has to be installed separately in your virtual environment on Miniconda distribution:

In the next section we provide installation instructions for environment manager using Miniconda (https://docs.conda.io/en/latest/miniconda.html). Similar instructions apply to full Anaconda (https://docs.anaconda.com/anaconda/install/) manager. Python 3.x will be installed with both distributions. The following instructions were tested for Mac, Windows and Linux 64-bit OS (The 32-bit has not been tested yet).

2.2 Windows 64-bit

1. Download the latest Miniconda3 installer for Windows from the Miniconda web page. (https://docs.conda.io/en/latest/miniconda.html)
2. Run the installer setup locally (select the "Just Me" choice) to avoid the need for administrator privileges.
3. Leave the default folder path (`C:\Users\<user>\..\miniconda3`).
4. Do not add Anaconda to the PATH. Do, however, register Anaconda as the default Python environment.
5. Open up an Anaconda prompt from the Windows Start menu. The `base` environment is being activated and the prompt changes to: `(base) C:\Users\<user>`
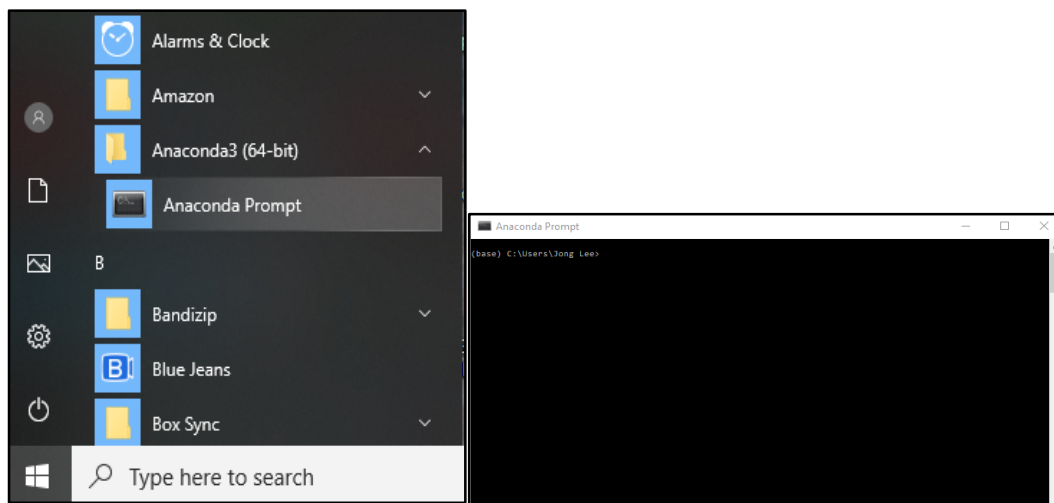
6. Create the python environment (`pyincore` for example) and activate it:
   ```
   conda create -n pyincore python=3
   conda activate pyincore
   ```

7. Add conda-forge (https://conda-forge.org/) package repository to your environment:
   ```
   conda config --add channels conda-forge
   ```

8. Install Jupyter Notebook. Jupyter Notebook is already installed with Anaconda distribution; it has to be installed separately in your virtual environment on Miniconda:
   ```
   conda install jupyter
   ```

2.3 Mac and Linux OS

1. Download the latest Miniconda3 installer from the Miniconda web page. (https://docs.conda.io/en/latest/miniconda.html)
2. Run the installer setup locally (select the "Install for me only" on Mac/Linux) to avoid the need for administrator privileges.

3. Leave the default folder path (`/Users/<username>/miniconda3` or `/home/<username>/miniconda3`).
4. Do not add Anaconda to the PATH. Do, however, register Anaconda as the default Python environment.
5. Open up a Terminal. The `base` environment is being activated and the prompt changes to: `(base)/Users/<username>` or `(base)/home/<username>/`
6. Create the python environment (`pyincore` for example) and activate it:

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

7. Add conda-forge (https://conda-forge.org/) package repository to your environment:

```
conda config –add channels conda-forge
```

8. Install Jupyter Notebook. Jupyter Notebook is already installed with Anaconda distribution; it has to be installed separately in your virtual environment on Miniconda:

```
conda install jupyter
```

Mac OS 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 line

```
backend: Agg
```

into the `~/.matplotlib/matplotlibrc` file. You must create the file (`matplotlibrc`) if it does not exist.

2.4 pyIncore Installation

pyIncore package

These steps will guide you on how to install pyIncore.

1. Add NCSA's pyIncore repository, conda channel to your environment (single line command without spaces):

```
conda config --append channels
https://channel_credentials@incore2.ncsa.illinois.edu/
conda/pyincore/
```

Contact us at the incore-dev@lists.illinois.edu for details of `channel_credentials`

2. To install pyIncore, navigate to the directory you want to use for running Jupyter Notebooks and run the following command:

```
conda install pyincore
```

3. (optional) Remove the NCSA's pyIncore repository, conda channel from your environment (single line command without spaces):
```
conda config --remove channels
https://channel_credentials@incore2.ncsa.illinois.edu/
conda/pyincore/
```

pyIncore credentials

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 prompt. The typical location of a HOME directory is `C:\Users\<username>` on Windows OS, `/Users/<username>` on Mac OS and `/home/<username>` on Linux based machines.

Note: The folders and files starting with "." (dot prefix) are hidden in Operating systems with Unix roots. There are few ways (https://nektony.com/how-to/show-hidden-files-on-mac and https://macpaw.com/how-to/show-hidden-files-on-mac) to view hidden files on Mac OS and Linux (https://askubuntu.com/questions/232649/how-to-show-or-hide-a-hidden-file).

1. Locate a file called **`.incorepw`** in the `.incore` folder in your HOME directory.
2. 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.

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.
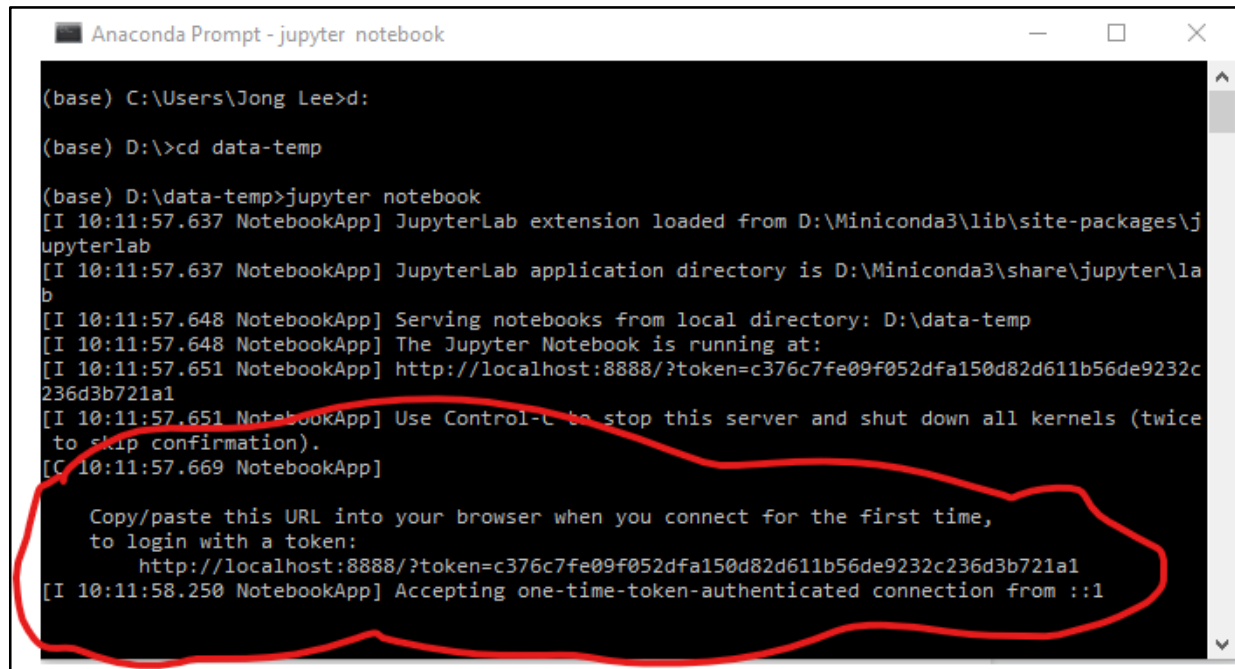
3.1 Running a Building Damage Analysis Locally

- 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:
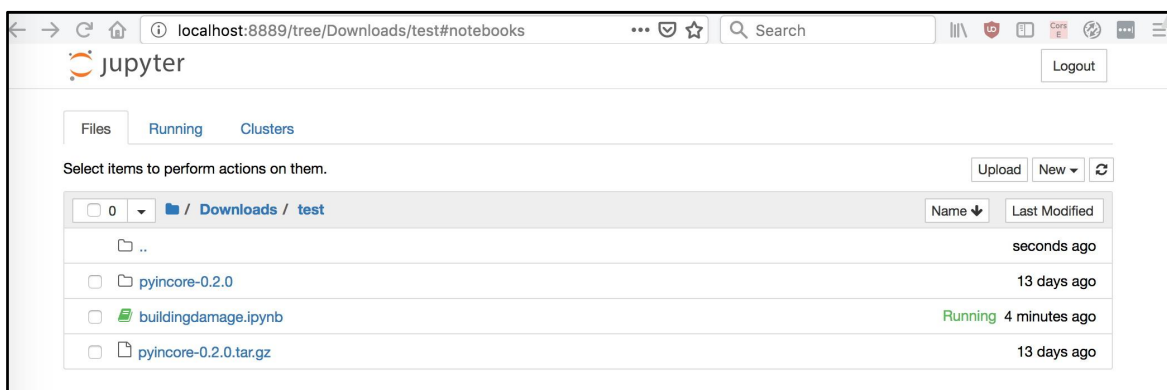

- Click on the `buildingdamage.ipynb` in the Jupyter Notebook browser.

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

```
In [10]:  from pyincore import IncoreClient
          from pyincore.analyses.buildingdamage import BuildingDamage
```

Type *Markdown* and LaTeX: α^2

```
In [11]:  client = IncoreClient()
```

```
In [12]:  # EQ Building Dataset - Memphis Hospitals
          bldg_dataset_id = "5a284f0bc7d30d13bc081a28"
```

```
In [13]:  # New madrid earthquake using Atkinson Boore 1995
          hazard_type = "earthquake"
          hazard_id = "5b902cb273c3371e1236b36b"

          # Building damage ratios
          dmg_ratio_id = "5a284f2ec7d30d13bc08209a"
```

```
In [14]:  bldg_dataset_id = "5a284f0bc7d30d13bc081a28"
```

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)

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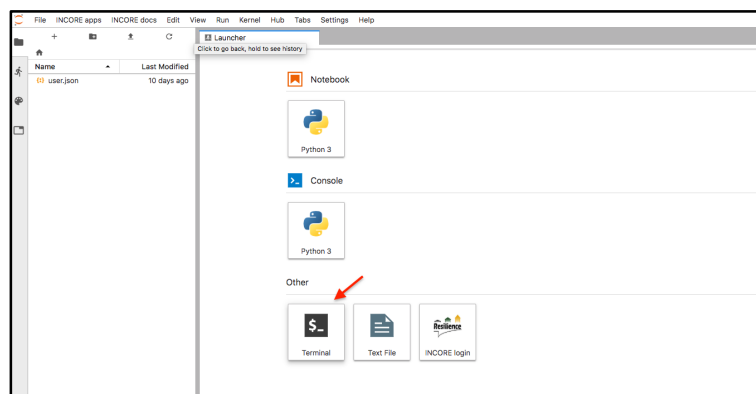
    b. In the terminal, make sure you are in your HOME directory. Type:
       ```
       pwd
       ```

to see the current path and
```
cd ~
```
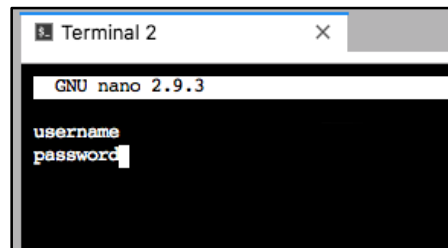to get into your home directory (`/home/<username>`).

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 **nano** text editor:
```
cd .incore
nano .incorepw
```
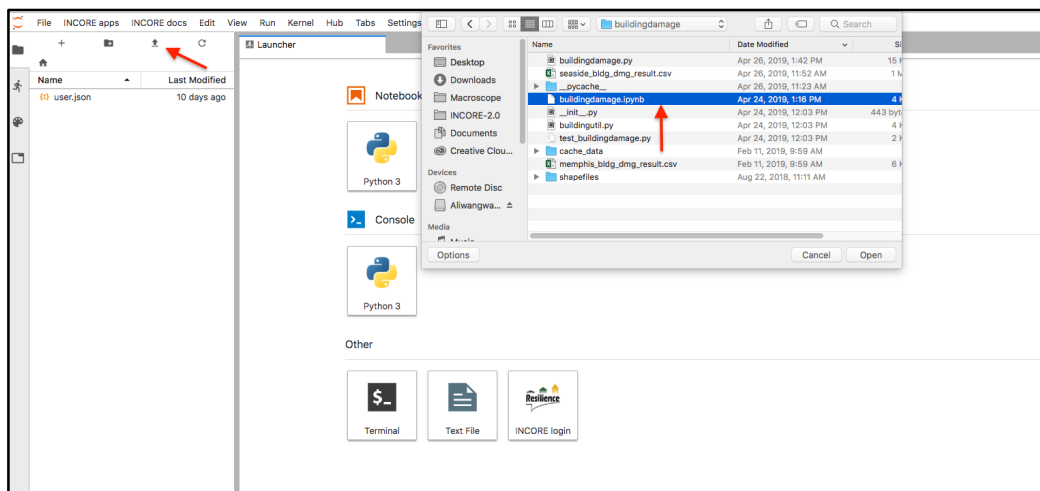

e. Save the file with `Ctrl+O` and `Enter` commands
f. Close the text editor and return to your shell with `Ctrl+X` command

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.ipybn** .

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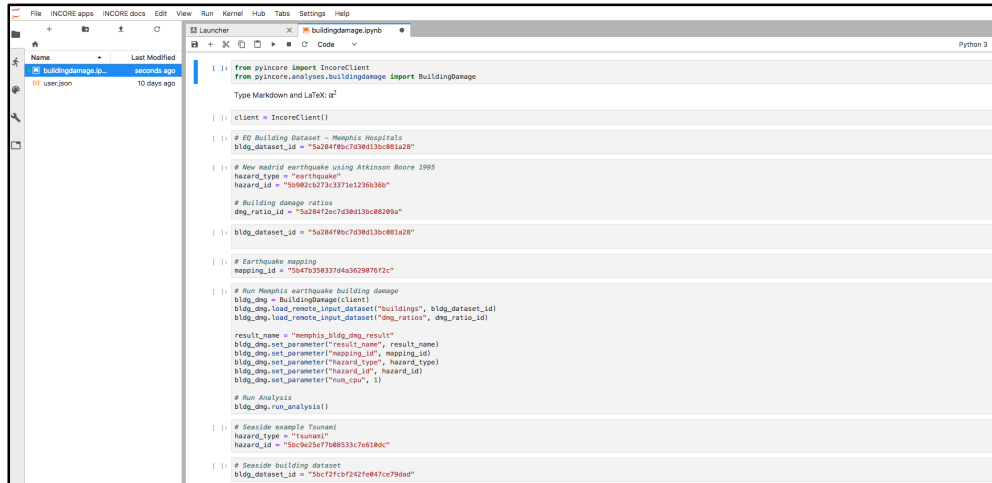
5. Run it. Instructions on how to run building damage analysis, please refer to previous section *Running a Building Damage Analysis*.

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
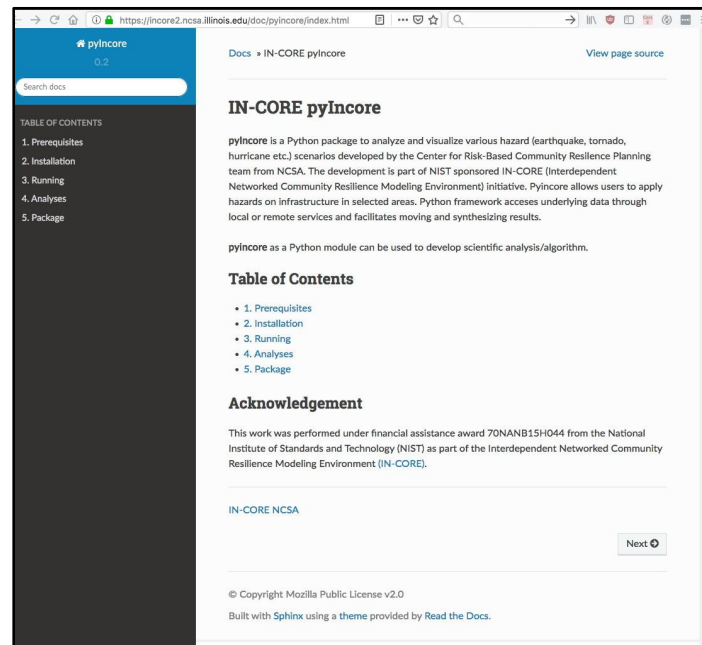
7 Additional Information

7.1 Technical Documentation

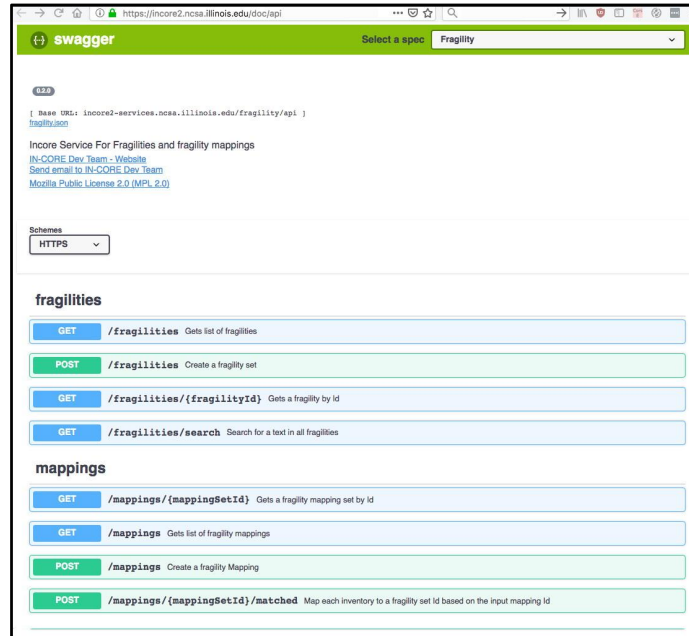From the IN-CORE landing page at https://incore2.ncsa.illinois.edu/ a user can access:
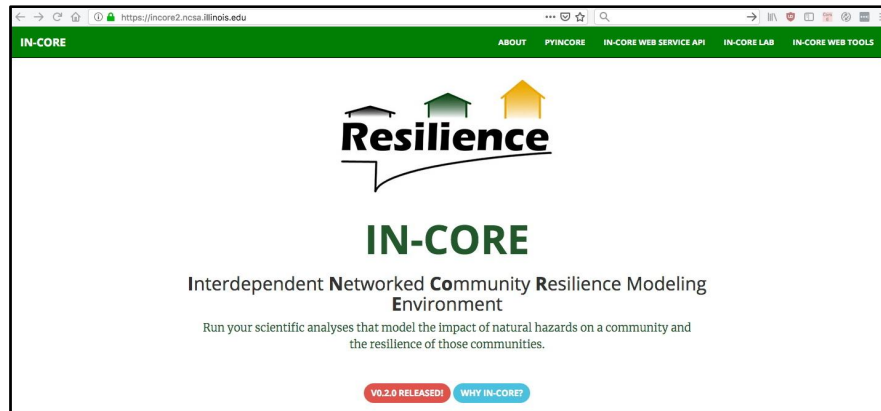
In this site, you can download pyIncore package and access to other services and documentations.

- 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.
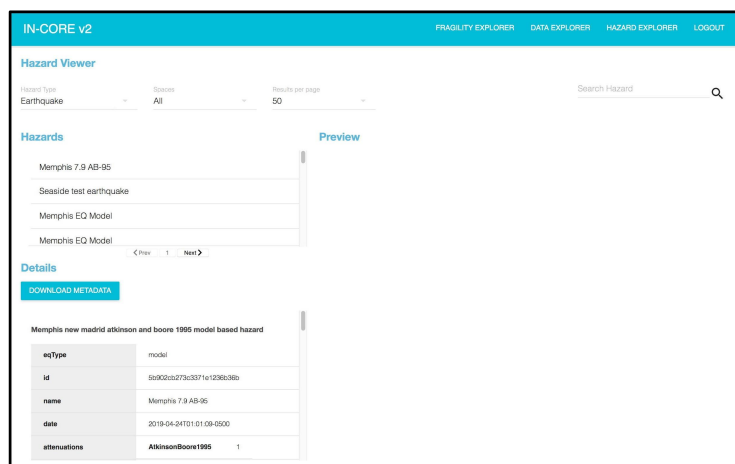
7.2 IN-CORE Web tools

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:
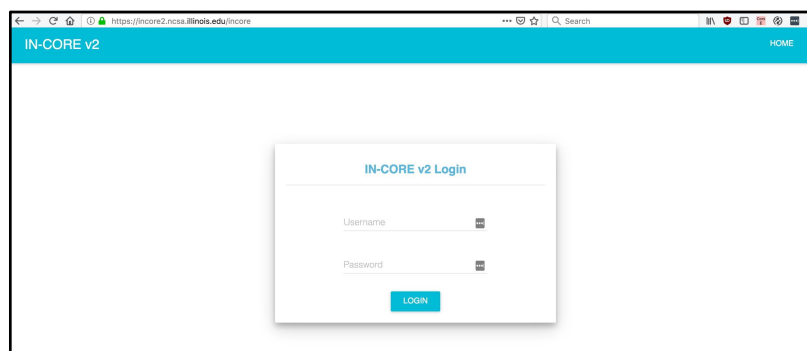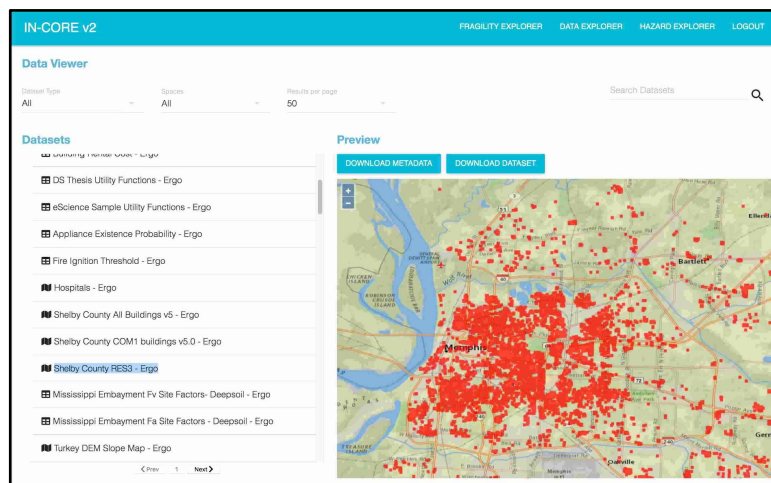

Fragility service. This is an example of a viewer showing a selection list (left) of Fragility curves. Hazards and types of structures are selected in the pull down menus. The data can be downloaded in json format.
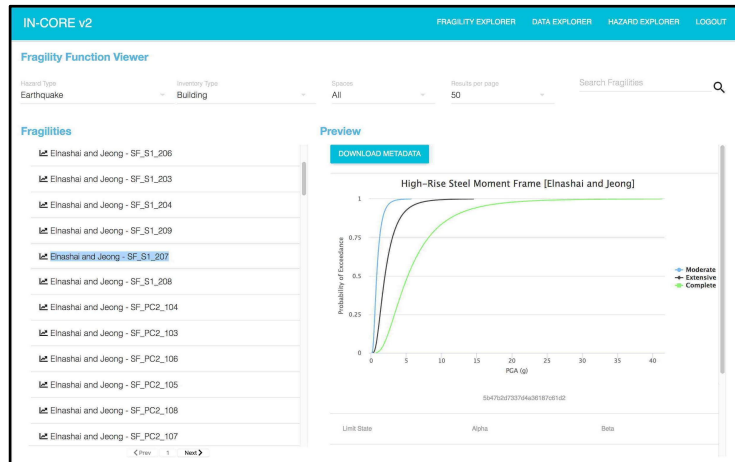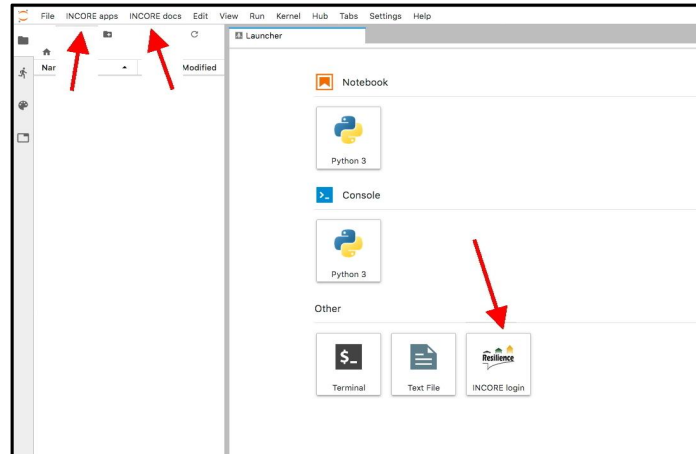
Hazard service viewer.
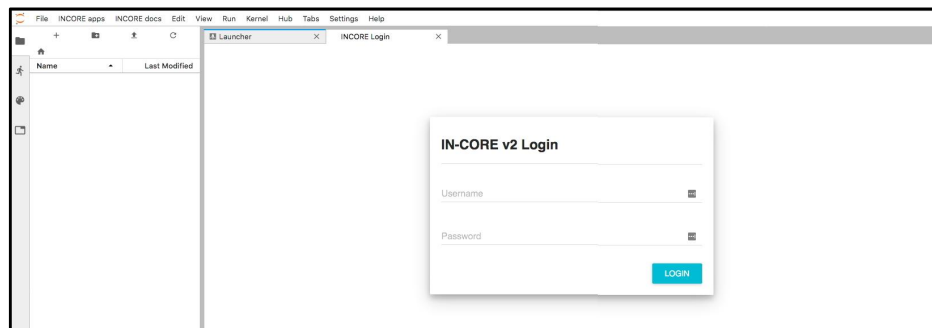

Data service viewer.

7.3 Additional Information about IN-CORE Lab

This section shows how to access IN-CORE Web Tools and documentations on IN-CORE Lab.
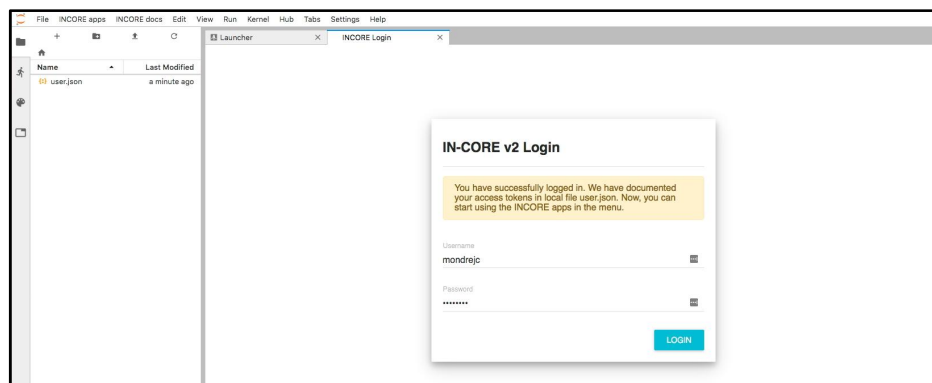
- Click on "INCORE Login" button as shown below.

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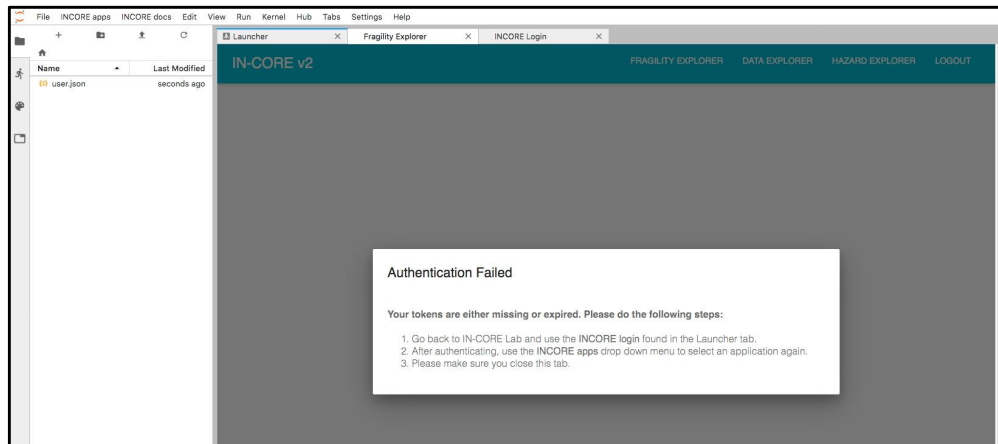
Viewer as part of INCORE Lab as shown below.
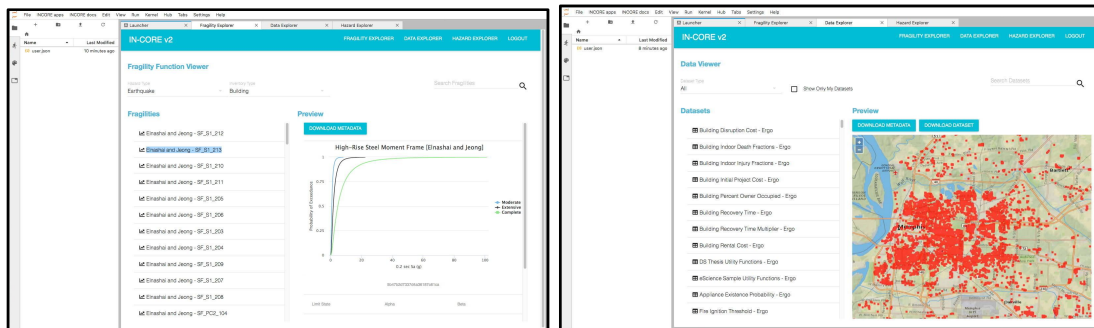
- The second IN-CORE menu (INCORE docs) allows user to see pyIncore documentation and API endpoints definitions for accessing Fragility, Data and Hazard server(s). Another IN-CORE login window opens up at the top of the browser's main window.

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