
INC Flywheel Documentation

Release 0.1

Lena & Amy

May 16, 2023

GETTING STARTED

1	Contents
---	----------

3

INC FLYwheel has its documentation hosted on Read the Docs.
Check out the [Overview](#) section for introductory information.

Note: This project is under active development.

CONTENTS

1.1 Overview

Flywheel.io is an imaging platform used to receive, curate, manage, and analyze data from scientific facilities. Intermountain Neuroimaging Consortium (INC) is excited to host an “on premise” deployment of Flywheel.io for clinical, industry, and academic researchers within the region. The “on premise” instance means all data storage as well as computation is completed using University of Colorado services. INC at CU Boulder has leveraged the power of Flywheel.io and the considerable resources available through University of Colorado Research Computing to provide a seamless experience for all researchers from data collection at our MRI facilities to data analysis, visualization, and publication.

For those familiar with other “PACS” systems, Flywheel also provides a service to receive and view data directly from the Intermountain Neuroimaging Consortium’s 3T Prisma Fit MR scanner.

Most importantly, Flywheel is a FAIR compliant platform. **F-A-I-R** are a set of data principles adopted by both the National Science Foundation (NSF) and National Institute of Health (NIH, beginning 2023) to prompt transparency and reproducibility. **F**indability, **A**ccessibility, **I**nteroperability, and **R**euse cornerstone these principles. Flywheel supports the FAIR data practice mission by ensuring rich metadata for all data stored within the platform. Flywheel also enforces the use of provenance, meaning the storage of information about “who, what, when, how” in data creation and management. By conforming to these best data practices, INC hopes to elevate the importance of transparent and reproducible research in the neuroimaging community.

Interested in learning more? Check out our introductory video [here](#).

Also, check out Introduction to Flywheel Webinar [here](#).

1.2 At the Scanner

Before starting a new or existing study in Flywheel, please set up a meeting with the Intermountain Neuroimaging Consortium Staff ([website](#)) who can help discuss your specific needs. While Flywheel.io provides almost endless flexibility in implementation of the platform, there are a few **critical** actions that must take place to ensure every image collected at INC lands in Flywheel in the right place.

1.2.1 Naming your Flywheel Session

In order to ensure acquisitions are assigned to the correct Project, Subject, and Session this information **MUST** be entered at the scanner console correctly (in the field labelled **Accession Number**), using the following convention:

Flywheel Accession Number Naming Convention:
<project-label> / <subject-label> / <session-label>

While no constraints are placed on the format within each label, we highly recommend using [BIDS](#) compliant naming schemes for subject and session labels.

A minimal amount of additional information may be entered at the scanner for any study participant or scan session. This information includes:

Scanner Field	Usage	Notes
Referring Physician	Principal Investigator	(Required)
Accession Number	Flywheel Naming Convention	(Required)
Patient ID/Name	URSI Identifier	(Co-Enrollment in COINS Only)
Age	Age	(Co-Enrollment in COINS Only)
Gender	Gender	(Co-Enrollment in COINS Only)

INC currently records this information using the Scanner Requisition Form which should be submitted before each scan session. Still have questions? Check out our [FAQs](#) page.

Note: The naming convention outlined above is specific only to the INC instance of Flywheel. Other neuroimaging centers have individual conventions for tracking project, subject, and session IDs from during image acquisition.

1.2.2 Examples of Accession Numbers

Study A has enrolled John Snow into their study, this is a longitudinal study, where the participant will return for 3 separate neuroimaging sessions. Below is an example of what is entered for the Accession Number field on the scanner console.

StudyA/101/S01

The participant is assigned a subject-id of 101 and session-id of S01.

Importantly, Flywheel is different from COINS in that you do not need to pre-register your participants with his/her ID apriori (before the scan). In fact, INC recommends that the participants are not pre-registered in Flywheel, but rather that the studies check Flywheel after the scan completes to make sure there was no typo at the scanner. **No personally identifiable information can exist on the Flywheel platform (see MOU)**. In short, this means study teams must retain the key to their *Coded* data in a secure location outside Flywheel such as REDCap or on paper. Please consult the University of Colorado Institutional Review Board (IRB) regarding appropriate steps that must be taken to secure *Coded* and Personally Identifiable Information for human subject research.

Study B has enrolled Snow White into their study. This participant has already participated in another study conducted in the same laboratory. This study is cross-sectional with a single cohort, and therefore the study does not want to include a session flag.

Incorrect

StudyB/102/ or StudyB/102

We have arbitrarily assigned the participant a subject-id 102, no link is required to the prior study **BUT** we are missing a required session label!

Flywheel naming convention is rigid and requires Project, Subject, **and** Session label to ingest and route the data to the correct location.

Correct

StudyB/102/S1

While this example study has only one session, we must enter all three labels: project, subject, and session every time!

Note: *What happens if this naming goes wrong?* If a flywheel session was incorrectly named, all acquisitions associated with that session will be stored in an “Unsorted” project. This project is unique to each Principal Investigator (Flywheel “Group”). Study teams should take great care to ensure any missing or incorrectly named scans are caught quickly! Once a study has identified an incorrectly labelled scan, they should contact INC personnel immediately who will correct the error.

1.2.3 What is Pre-Registration? and Why it Matters?

For current INC users, you are likely accustomed to storing participant information and subject-ids in COINS. COINS uses pre-registration to check that images generated on the scanner “match” coded information already entered into the COINS database. With this protocol, INC staff could immediately identify and flag scanning sessions with incorrectly entered participant codes. Flywheel provides a more streamlined approach that does not necessitate pre-registration. The bottom line here: in Flywheel, INC staff can no longer play any role in the confirmation that participant information was entered correctly at the scanner.

Warning: Users should check all scans entering Flywheel **immediately** after the scan session is complete. Closely inspect that all participant information is correct and matches the information stored in your participant key outside Flywheel!

1.2.4 I Started my Study in COINS, What Happens Now?

All studies who wish to continue pre-registering and importing your data into COINS have the option to do so. All studies opting to continue using COINS will also have all new acquisitions stored in Flywheel. All scanner fields necessary for COINS convention are compatible with Flywheel convention. The one notable exception: while COINS has no restrictions on the value entered into “Accession Number”, this field **MUST** conform to the Flywheel naming convention to comply with both COINS and Flywheel requirements.

1.2.5 What Information Can I Include in Flywheel?

INC at University of Colorado supports an “on premise” deployment of Flywheel.io. As all data and compute is conducted within UCB systems, we must conform to all data and privacy policies set forth by University of Colorado Research Computing (CURC) and Flywheel.io. As such, data must be de-identified before entering Flywheel. **NO** protected health information (PHI) and **NO** personally identifiable information (PII) may be stored in Flywheel. Examples of protected information includes:

- first or last name
- email address
- phone number
- mailing address

- study enrollment or collection date (when paired with other identifying information)
- detailed health history

Not sure if your data is correctly de-identified? Please contact your IRB representative before placing any data in Flywheel!

1.3 Navigating The User Interface

Flywheel.io's user interface is a flexible powerful platform where users can do almost anything from creating and viewing data, to running analyses, and inviting collaborators to participate. The following provides a *brief* sample of the actions that can be taken within the Flywheel user interface. Please attend INC Courses on Using Flywheel to learn more!

1.3.1 Logging Into Flywheel

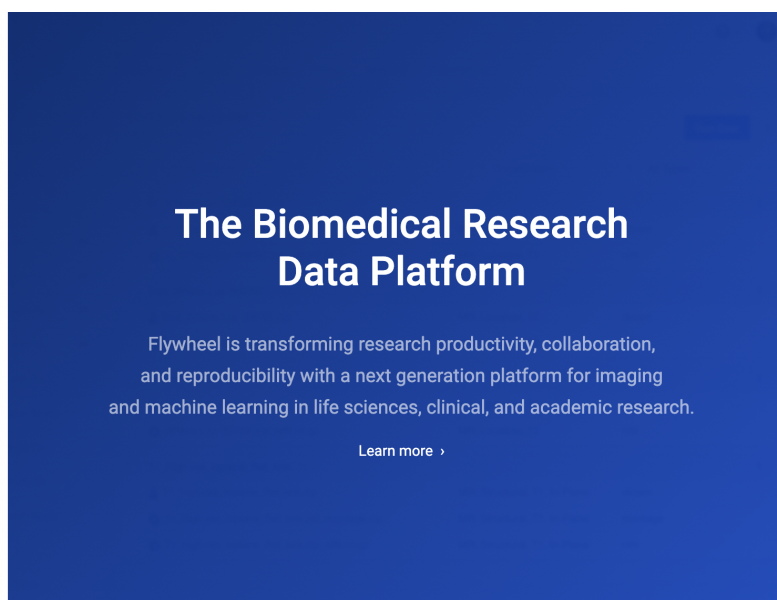
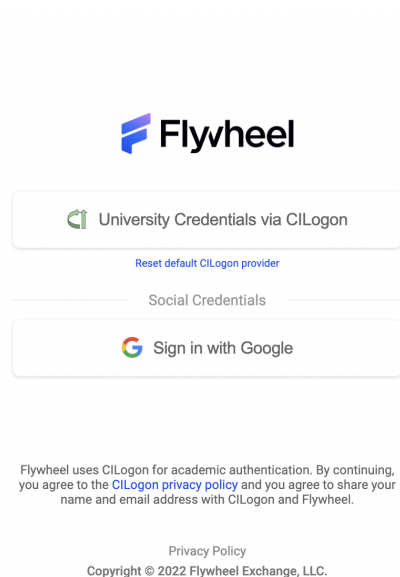
Flywheel uses CILogon service to manage access to their platform. CILogon is used by most academic institutions around the world to manage institutional accounts and therefore makes it very easy to add users, and log in with externally managed University credentials!

University of Colorado Users

To logon as a UCB user, you need only a University of Colorado identikey. If you're not sure if you have one, contact INC staff.

Warning: Experiencing issues logging in? Try **changing browsers**. For more info visit our [FAQs](#) page.

1. From your web browser, go to Flywheel flywheel.rc.colorado.edu
2. Select University Credentials via CILogon




3. If this is your first time logging on, you will be redirected to the CILogon portal to select your organization.

- Find University of Colorado Boulder
- Check “Remember this Selection”
- Click “Log on”



Select an Identity Provider

ORCID 

☐ Remember this selection

By selecting "Log On" you agree to the terms of service.

- University of Col
- University of Colima**
- University of Colombo School of Computing
- University of Colombo, Sri Lanka
- University of Colorado at Boulder
- University of Colorado Denver | Anschutz Medical Campus

For questions about this site, please see the [FAQs](#) or send email to help@cilogon.org.
 Know your responsibilities for using the CILogon Service.
 See [acknowledgements](#) of support for this site.

4. You will be directed to the CU Boulder logon page. Enter your identikey and password
5. You are now on the CU Boulder Flywheel Instance!

External Users

Do you have an account with a University, ORCID, or another organization that uses the CILogon system?
 If you are not sure, you can check [here](#).

I already have a CILogon Connected Account

Contact your collaborator at UCB with the appropriate credentials. Follow the instructions above to log in.

I do not have a CILogon Connected Account

Request a University of Colorado Boulder Affiliate Account through your UCB collaborator. These accounts will provide you access to the UCB systems for a period of one year, and usually can be generated within 5 business days.

1.3.2 The Flywheel Hierarchy

Before we get into how to navigate around the Flywheel interface, the following three sections are important building blocks to understand how Flywheel is designed. Understanding the Flywheel hierarchy, the back-end storage, and the container principle will help you navigate Flywheel and address your questions more readily.

A hierarchy is simply the system we rank or organize data according to a parent-child relationship. You might think of this as a folder on your computer that contains other folders and files. In this case the ‘parent’ folder has other ‘child’ sub-folders and files.

Flywheel uses a hierarchical data model to store data. In this way, data is automatically stored in an ordered way by principal investigator, study, subject, session, and acquisition.

1.3.3 Object Based Storage Principles

We are not going to get into the weeds here... What is important is that Flywheel uses object based storage to store all raw and derived neuroimaging data. Object based storage is a type of data storage. Object based storage is generally more efficient and attaches a more information about how the data was created, modified, or used within the data structure itself. What does this mean for you? Neuroimaging storage on Flywheel takes up less disk space (its cheaper!) and contains a lot more information to search or retrieve data later. Interested in [learning more](#)?

1.3.4 What are Containers in Flywheel?

Containers are the data storage building blocks within Flywheel. Why does this matter? If you are thinking about retrieving data, running analyses, or even reviewing data already stored, you need to think about how to retrieve this data from a container. In layman’s terms, a container could be thought of as a “folder” on your computer which can contain other “folders” or containers, as well as files or metadata. If you are unfamiliar with the concept of metadata, think of it as information about that folder, such as when it was created or modified, its name, etc.

In Flywheel containers are used to store “groups”, “projects”, “subjects”, “sessions”, “acquisitions”, and “analyses”. We get into the meaning of each of these containers below, but you can think of these containers as folders in Flywheel that bundle metadata and data together.

Below is the basic Flywheel hierarchy:

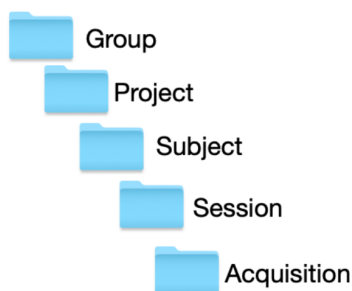


Image duplicated from docs.flywheel.io

1.3.5 Accessing My Groups

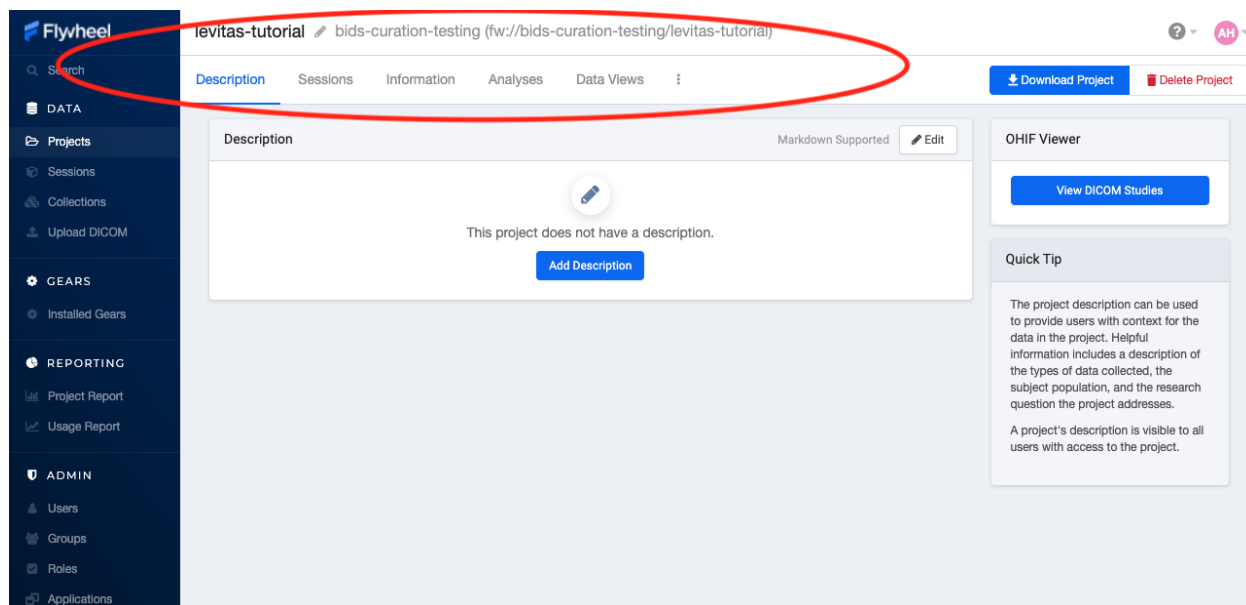
At INC, we use “Groups” to assign a principal investigator or laboratory. Here “Groups” can store multiple different projects or “studies”, have specific users and user permissions, and have administrative roles to add / edit / delete data and metadata for every container within. In Flywheel, you can identify Groups by the “tag” associated with any of your projects. As an Admin, you can also make changes to user permissions and projects within your “Group”. For more information on this topic, please refer to our documentation on “[User Permissions](#)”.

1.3.6 Accessing My Projects

At INC, “Projects” are used to differentiate studies conducted within a Principal Investigator’s laboratory (ie, studies within a Flywheel “Group”). Users may be added to multiple projects, and once granted permission will be able to view the each project in Flywheel. All accessible projects may be viewed from the left hand ribbon on the projects page:

Project	Group	Subjects	Sessions	Role	Users
levitas-tutorial	bids-curation-testing	3	2	admin	
RRAY	rkaiser	57	57	admin	AH LS

In the second column of the project list you will find the parent Group for each project. “Projects” have several attributes including a description, project files, subjects, sessions, custom data views and more! Check out our upcoming tutorials to learn more about how to customize your project to meet your needs.



1.3.7 Accessing My Subjects or Sessions

If this is a new project, you may not see any subjects or sessions linked to your project. If you have already started scanning, or have uploaded historical/retrospective data from your project you should see each scan session in “sessions.”

Note: *Still can't see your data?* Remember that pesky Accession Number? Well, if the first part of that string (ie the STUDY in STUDY/SUBJECT/SESSION) wasn't entered correctly at the scanner, your data doesn't know where to land on Flywheel. Not to worry, your data will be sitting in a project called Unsorted. If you can't see this project, contact INC staff or your lab admin of Flywheel who can add you to the Unsorted Project. From there, you can move that subject to the correct Project.

“Subjects” are used to bundle sessions collected on the same participant across multiple days or “sessions”. We identify subjects using a single Subject ID. This ID should be unique to the participant in the current study. If this ID needs to be “coded” with a reference to any personal identifiable information (PII), that PII **MUST** be stored outside Flywheel in a database such as COINS or REDCap. If you have questions about storing participant information, please contact INC!

From a project within Flywheel, the easiest way to access subjects and sessions is from the “Sessions” panel shown here:

The screenshot shows the Flywheel web interface. On the left is a dark sidebar with navigation links for DATA, Projects, Sessions, Collections, Upload DICOM, GEARS, Installed Gears, REPORTING, Project Report, Usage Report, ADMIN, Users, Groups, Roles, and Applications. The main content area has a top navigation bar with tabs: Description, Sessions (highlighted with a red arrow), Information, Analyses, and Data Views. Below this is a 'Sessions' panel with a table of sessions. The table has columns for 'Timestamp', 'Subject', and 'Session'. Two sessions are listed: one from 2021-07-08 12:06 for Subject 085, and another from 2020-01-22 14:29 for Subject 10462@thwjames.... To the right of the sessions table is an 'Acquisitions' panel for Session S2, showing a list of files with their descriptions, creation dates, and classifications.

Within the sessions panel, you may notice the sessions are sorted by date of collection, and show a summary of the Subject ID and Session ID for that set of acquisitions. To view the same data in “Subject view” you need to select the Subjects’ icon shown here:

This screenshot is similar to the previous one, but the 'Sessions' tab is not highlighted. Instead, a red circle highlights the 'Subjects' icon (a person silhouette) in the top navigation bar. The 'Sessions' panel now displays a table with columns for 'Subject Label', 'Session Label', and 'Timestamp'. Three sessions are listed: one for Subject 10462@thwjames_OpenScience, one for Subject 085, and one for Subject m80357410. The right panel remains the same, showing the Acquisitions view for Session S2.

1.3.8 Accessing My Acquisitions and Files

Finally, acquisitions are Flywheel containers within a session, and hold any files and metadata associated with a scanner sequence. For example, an acquisition may contain a set of dicoms, the nifti converted file for the same image, and task or behavioral data. As you may recall from earlier, these “containers” in layman’s terms are just like folders or directories that hold relevant files. In Flywheel, we can see acquisitions and files within the project view, as shown below:

Note: It's easy to confuse Acquisitions for Files because of how we use the term colloquially at the scanner. But don't be fooled, Acquisitions are not Files in Flywheel, they're still containers. In other words, an Acquisition's metadata (the information tab) will be different than the File's metadata.

1.3.9 Accessing My Analyses and Provenance

Analyses are Flywheel containers that can be attached to a project, subject, session, or acquisition. For the purpose of exploring the user interface, we will focus on Session level analyses. In the session view within Flywheel, all analyses are visible from the "Analysis" tab as shown below.

Analyses are stored in order from most recent to oldest. Analysis labels may be changed, and analyses may be deleted using the options menu at the right hand side of each analysis object.

Provenance is discussed more in the [Provenance](#) section. As an introduction, its good to be familiar with the provenance tab. You can review a history of all analyses run on your data in this tab, as well as view analysis logs and cancel or

re-run analysis gears.

The screenshot displays the INC Flywheel interface. On the left is a navigation sidebar with categories like DATA, COMPUTE, REPORTING, and ADMIN. The main panel shows the 'Sessions' tab with a table of sessions. The 'Provenance' button in the top right of the session details is circled in red. Below the sessions table, the 'Running Gear' section is visible, showing details for a gear named 'bids-hcp v1.2.5_4.3.0'. Red annotations highlight the gear name, the 'Cancel' button, and the 'View Log' button.

Timestamp	Subject Label	Session Label
2023-04-29 03:44	249	01
2023-04-16 03:23	250	01
2023-04-14 03:49	274	01
2023-04-08 05:29	260	01
2022-12-20 04:34	214	01
2022-12-13 03:59	263	01
2022-12-02 07:50	197	01
2022-12-02 06:27	201	01
2022-11-26 04:41	167	01
2022-11-21 01:33	166	01

What to learn more about how to run gears in flywheel? visit “*Gears*” Basics. Also check out our documentation on running commonly used gears at INC in “*Running Commonly Used Gears*”

1.3.10 Collections

Collections in Flywheel allow users to curate data from a range of projects or based on specific criteria. For example the ‘Radiologist Review’ collection will be used at UCB to curate images requiring incidental finding reviews for a radiologist. Further, collections can have a separate set of users and permissions in order to share specific sessions with users outside your study team. This feature can be found by clicking the collections view as seen here:

The screenshot shows the 'Collections' view in the INC Flywheel interface. The 'Collections' button in the left sidebar is highlighted with a red arrow. The main panel displays a table of collections. The 'Radiologist Review' collection is highlighted, showing it is curated by 'leeh2786@colorado.edu' and contains 2 subjects and 2 sessions.

COLLECTION	CURATOR	SUBJECTS	SESSIONS	ROLE	USERS
Radiologist Review	leeh2786@colorado.edu	2	2	admin	

1.3.11 Data Views and Project Reports

Data Views and Project Reports can be used to compile metadata from any project. Data Views provide the most flexibility to generate tabular views of any metadata within flywheel such as age, race, sex, acquisition info, and more. These views can be shared or exported for 3rd party statistical packages.

Project Reports provide a summary of all sessions collected over a specific time range. Basic descriptive statistics are computed on all demographic information described in each session.

Reporting Project Report Details Print

RRAY May 1, 2022 to May 31, 2022

Group

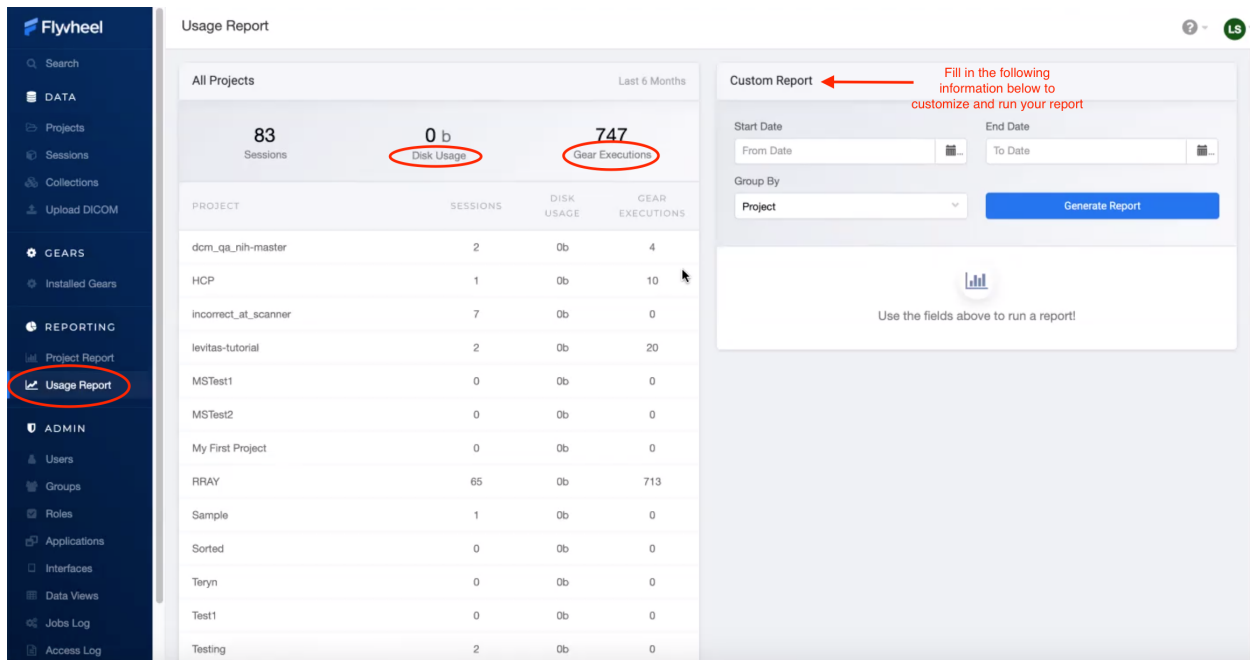
Project Admin

Number of Sessions	6
Unique Subjects	5
Male Subjects	3
Female Subjects	2
Subjects Under 18	3
Subjects Over 18	2

Racial Categories	Ethnic Categories									Total
	Not Hispanic or Latino			Hispanic or Latino			Unknown/Not Reported Ethnicity			
	Female	Male	Unknown	Female	Male	Unknown	Female	Male	Unknown	
American Indian or Alaska Native	0	0	0	0	0	0	0	0	0	0
Asian	0	0	0	0	0	0	0	0	0	0
Black or African American	0	0	0	0	0	0	0	0	0	0
More Than One Race	0	0	0	0	0	0	0	0	0	0
Native Hawaiian	0	0	0	0	0	0	0	0	0	0

1.3.12 Usage Reports

Usage Reports outline overall computing metrics for each project. Basic metrics include disk usage and number of gears (or Analyses) run.



1.4 How To Cite Us

This on-premise Flywheel platform is made possible by the hard work of many groups. Please cite us (Intermountain Neuroimaging Consortium) and the following collaborators if this platform has helped you produce your publication. Collaborators:

- CU Boulder Research Computing (on top of who's platform we have deployed Flywheel)
- Flywheel.io (without whose continued support, this would not be possible)

1.4.1 Contact Us

Interesting in getting started? Contact us [here](#) to request a copy of INC's Memorandum of Use and to set up a one on one consultation.

That's it folks! Tune in for more information and tutorial regarding Flywheel at UCB!

1.5 User Permissions

Within Flywheel (flywheel.rc.colorado.edu, login with university credentials), principal investigators (you) will be assigned an admin role for your group by the INC staff. From there you will have control of adding the rest of your lab members as users. The users will already be enrolled in Flywheel by INC staff, but they will not yet be assigned to any group or project. In this document, you will learn how to add users to your groups and projects, as well as how to assign them permissions, edit those permissions, and remove users if need be. The hierarchy of users in your group or project can largely be designed by you to suit your needs.

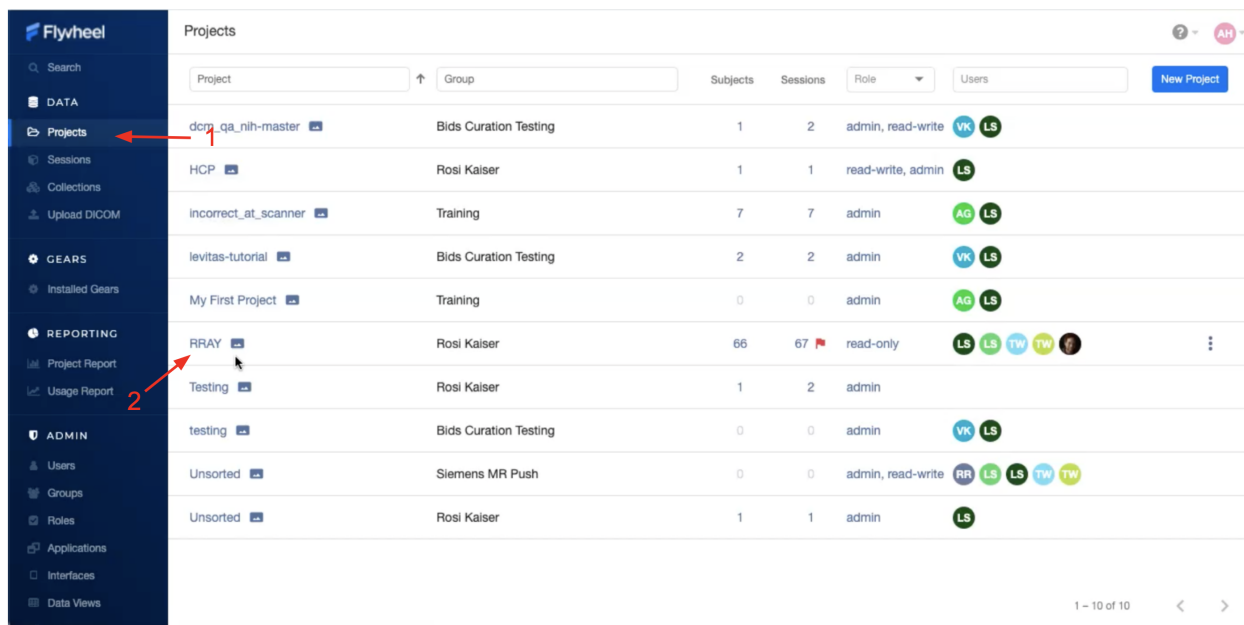
Note: Haven't gotten started with Flywheel yet? Check out documentation about logging in to get started [Logging Into Flywheel](#).

1.5.1 Adding Users

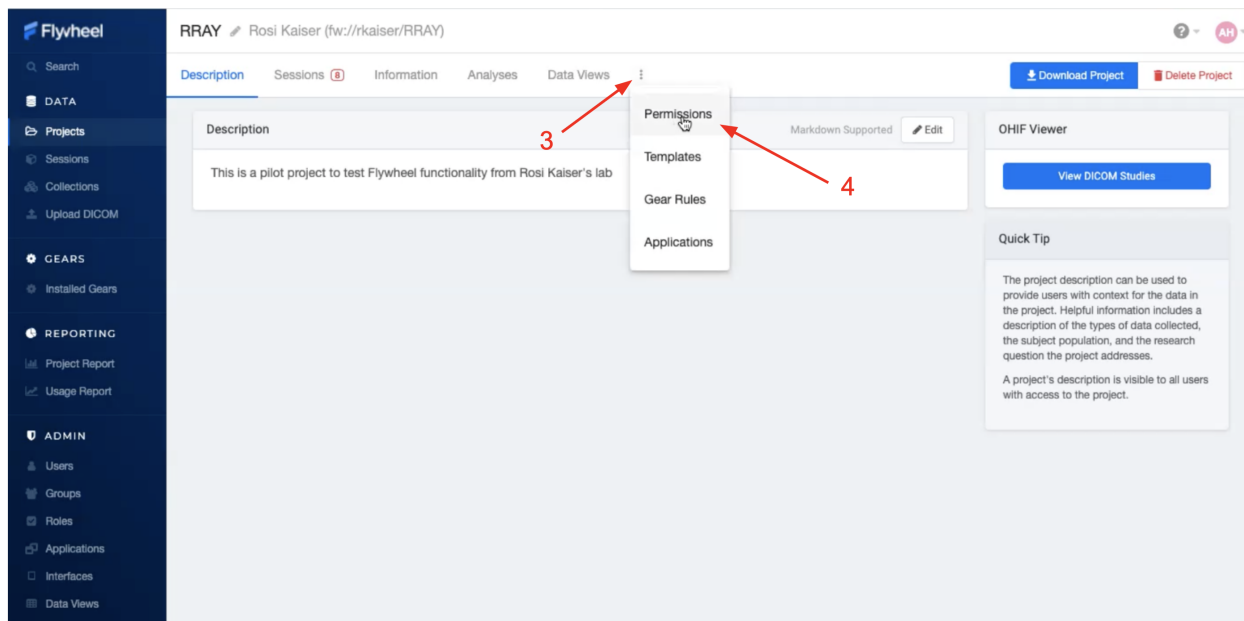
Users may be added to all projects belonging to a specific principal investigator, or may be added on a project by project basis. We will review how you can add users using both methods.

Adding users to projects

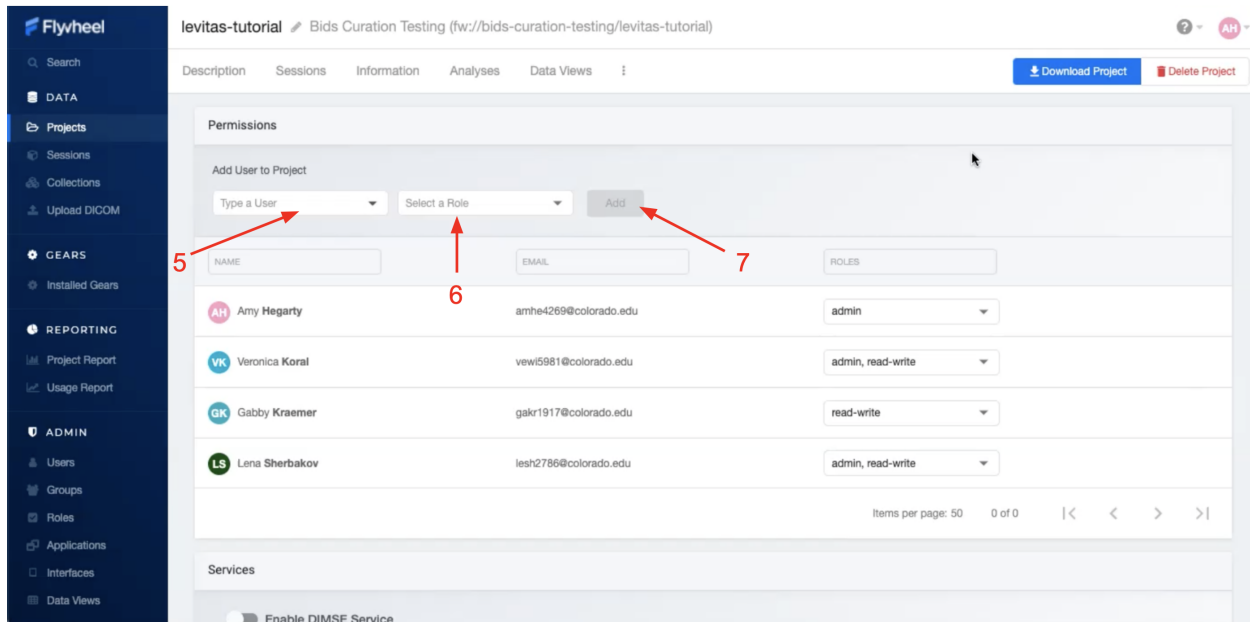
1. Navigate to “Projects” view, on the left hand ribbon
2. Select the project of interest



3. Once in the project, select the three dots along the top bar to expand all project options
4. Select “Permissions” from the dropdown menu



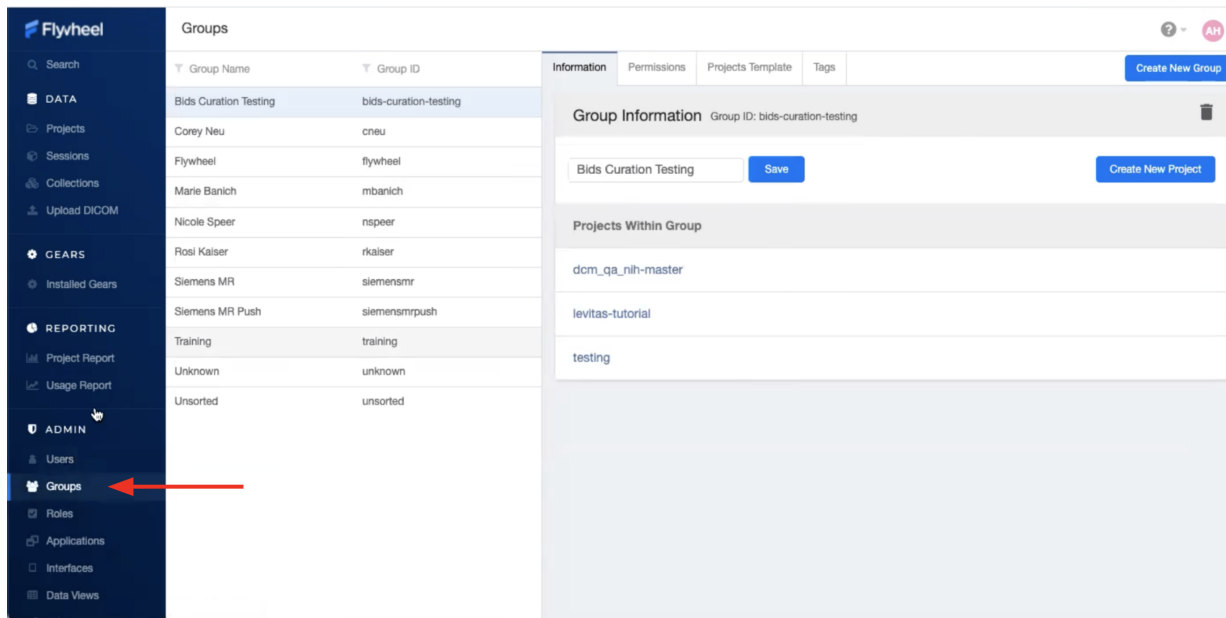
5. Within permissions select a user that has already been added to the UCB Flywheel instance (You may manually type a name or scroll to find the user you're looking for)
6. You must specify the new user's role. If you have discussed any Custom roles with the INC staff you will see them here. Otherwise, you will be able to select from: "Read-only", "Read-write", "admin"
 - a. Multiple roles can be selected for one user by checking several of the drop down boxes next to each possible role; however, we recommend that only one appropriate role is selected. If you're not sure what each role can do, click on the Roles tab on the left-hand ribbon. Then select the role in question, and a long list of actions that can and can't be performed by that Role will appear.
 - b. Looking for more flexibility? Contact INC Staff to set up custom user roles for your Group.
7. Click "Add" to save the changes to your project



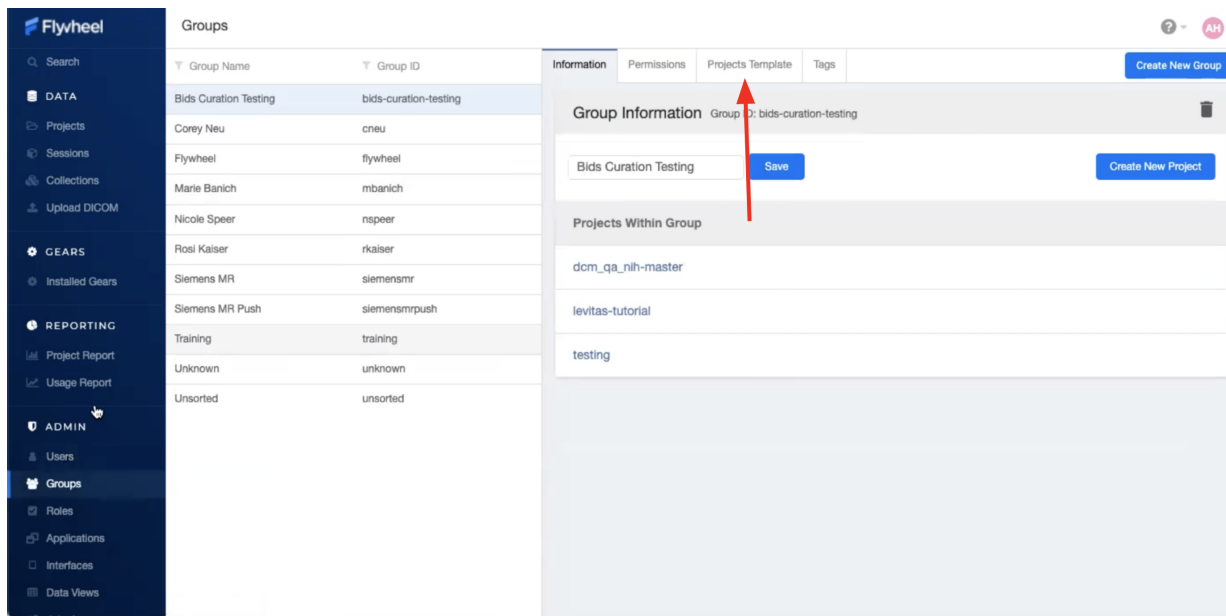
Note: INC staff recommend adding users only to individual projects they're working on. This improves data security, avoids confusion, and prevents access to data for any users other than those who need it.

Adding users to groups (adding them to all projects within a group)

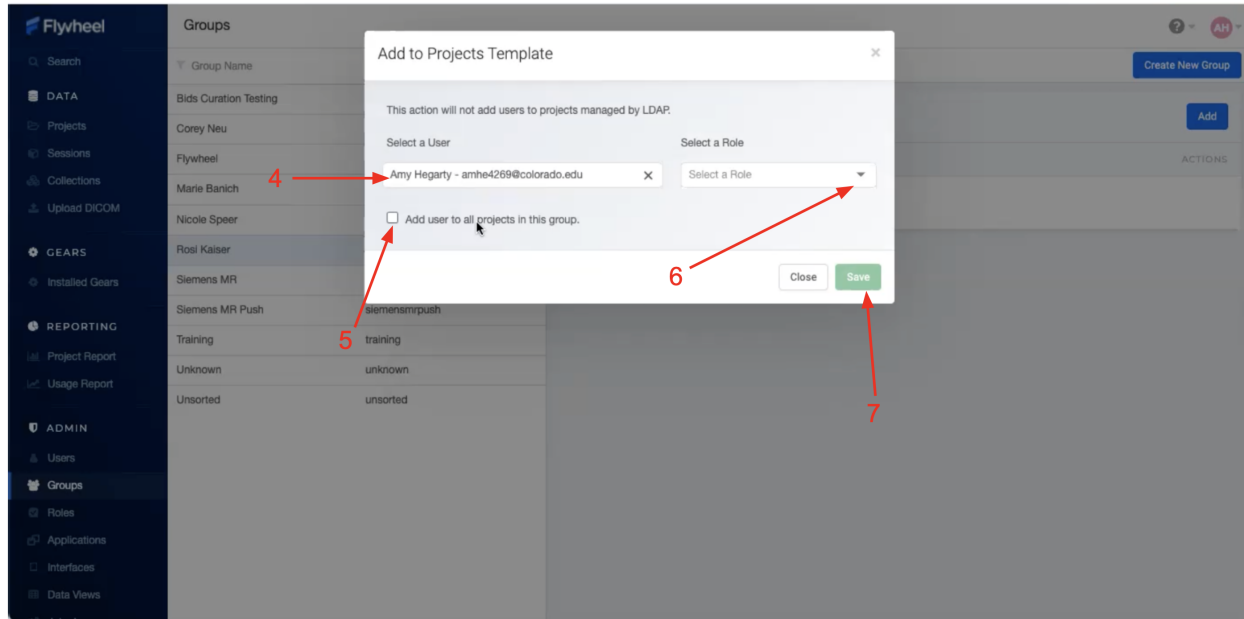
1. Click on the “Groups” tab under the Admin section of the sidebar menu



2. Within “Groups”, navigate to “Project Templates” panel, and select “Add”



3. Select from the list of Users added to UCB Flywheel. If you don’t see the person you are looking for, they haven’t been added as a Flywheel user yet. Contact INC Staff.
4. To add New User to **all** existing and future projects, Check the box that says “Add user to all projects in this group”
5. Select the new user’s role. Multiple roles can be selected for one user by checking several of the drop down boxes next to each possible role. Looking for more flexibility? Contact INC Staff to setup custom user roles for your laboratory.



Note: INC staff recommend using this method only for a small number of users such as lab managers or other staff who will need to access all projects, other users should be added to individual projects to increase data security.

1.5.2 Permissions

Default User roles include: read-write, read-only, and admin. The read-write role allows users to view, create, modify, and delete files or analysis, but does not allow them to delete projects, modify permissions, or project settings. The read-only role allows users to view files and metadata, but prevents adding/modifying files or running analyses. The admin role allows users to view, create, modify, or delete anything within the project, including users and metadata.

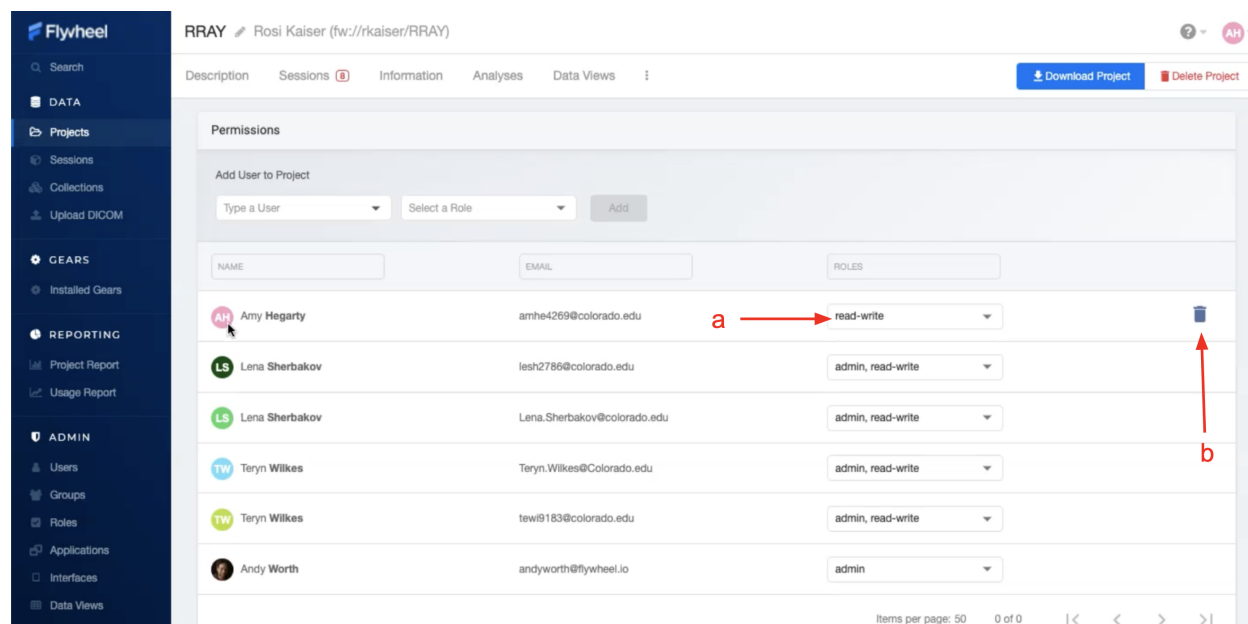
How do I appropriate assign user permissions? Research assistants and students who are working with data and data analysis would fit well into the read-write role, while an external collaborator who is only looking at the data without analysis would fit best in the read-only role.

Modifying assignments

Initial role assignment is done when you first add a user to a group or project, as explained above. In order to change a user's assignment or remove them from a group or project follow the steps below.

1. Follow the instructions above. Navigate to the Permissions panel within your project.
2. From the list of users, select the user for whom you would like to modify permissions.
3. From the drop down menu alter the permissions as desired.

To delete a user, (a) hover the mouse over the user of interest, (b) select the trash icon when it appears. This action will only remove the user from that project, and not from the CU Boulder Flywheel instance.



Creating New Roles

Interested in creating new/custom roles for you Group? This is a simple process, initiated by contacting INC Staff.

1.6 Hierarchy

Flywheel leverages object based storage, meaning all data are represented as objects. These objects are hierarchical, meaning they have a meaningful parent-child or “structured” relationship. A rigid “structure” is enforced for the Flywheel containers: Groups, Projects, Subjects, Sessions, Acquisitions. All Flywheel containers has multiple attributes including metadata, file attachments, and analyses (at Project, Subject, Session levels)

As discussed in more detail within [User Permissions](#) Projects contain additional information regarding User access and permissions for all data stored within a given project.

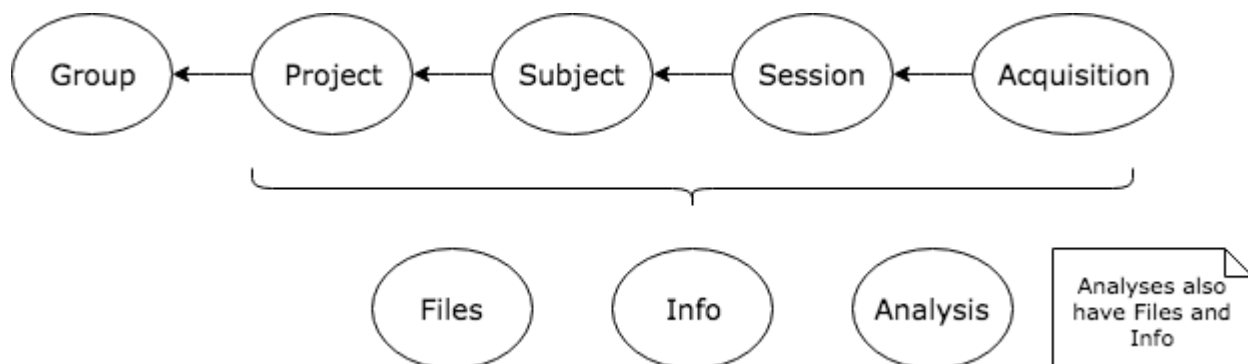
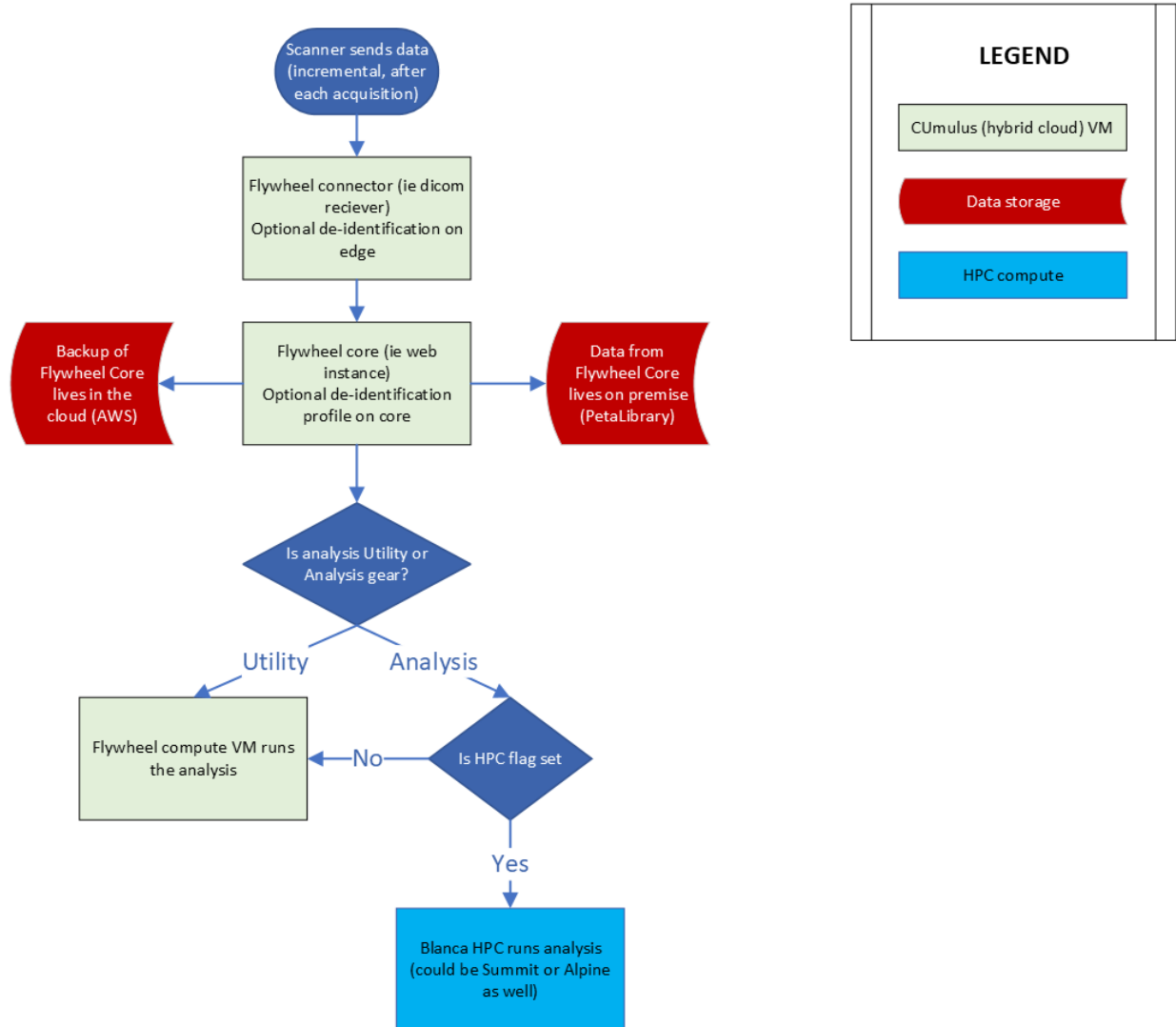


Image originally generated by Penn LINC Neuroinformatics

1.6.1 Flywheel Connector

One important function of Flywheel is its role as a ‘PACS’ system. The Flywheel Core houses the main flywheel database and software. This system *accepts* imaging data from the INC Prisma Fit Scanner via the Flywheel Connector. Imaging data, primarily dicom images, are passed from the scanner to the Flywheel Core. For investigators using our scanning facilities, all sessions will be automatically imported into Flywheel. A set of images collected on a single day for a single participant will be organized into a Session. An Acquisition generally contain a set of dicom images, and any other accompanying data sent directly from the scanner.



Basic schematic of INC on-premise Flywheel infrastructure. Data is sent from the scanner at INC to the Flywheel Connector (a virtual machine running on CU Boulder’s private hybrid cloud, CUMulus). The Flywheel Connector then passes data to the Flywheel Core (which runs a web instance or “user interface” of Flywheel). Different types of analysis requests can then trigger jobs to be run on either CUMulus virtual machines, or on our on-premise HPC resources (Blanca, Alpine, Summit). The Flywheel Core database (where the data is stored) lives on Research Computing’s PetaLibrary, while backups of the data are sent to an INC-managed AWS S3 bucket.

1.6.2 Uploading Files

Files may also be uploaded through the user interface (UI) or command line interface (CLI). We will discuss using the CLI in a later section of this documentation. For now, we will focus on the ways to upload a file from the user interface.

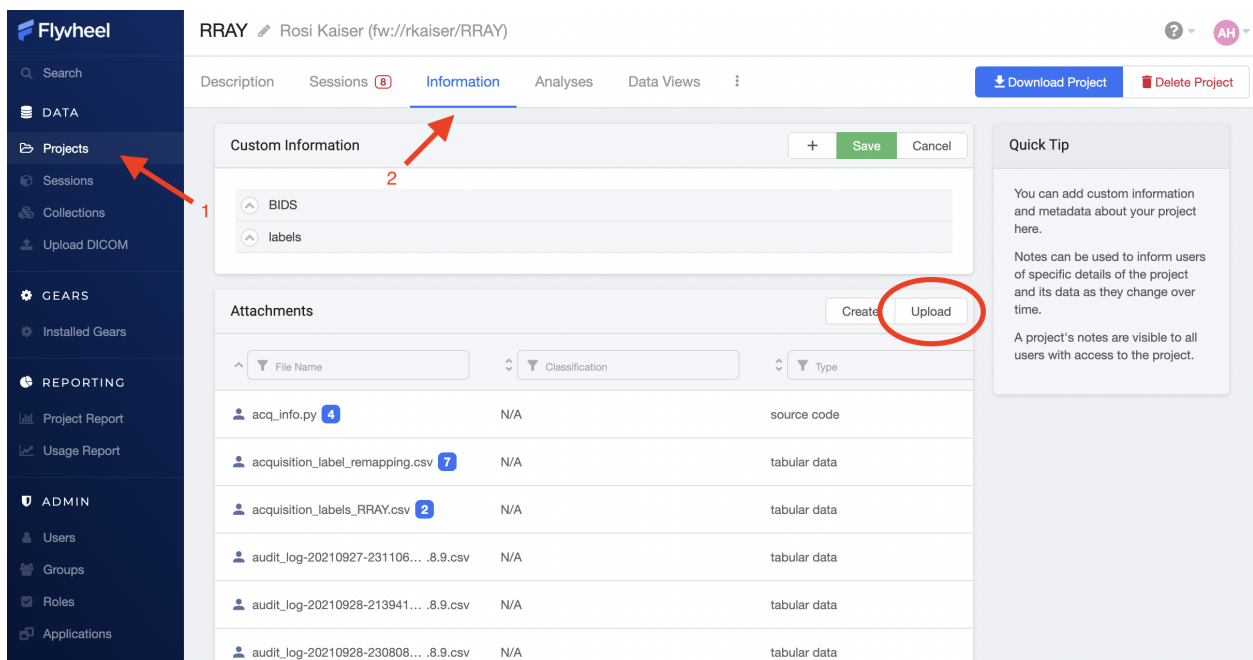
Files may be uploaded into containers including, Projects, Subjects, Sessions, and Acquisitions.

Warning: DICOMS should be uploaded using a special Flywheel feature called “Upload DICOM” and not uploaded as file attachments!

Upload Project Files

Project files can be stored in Flywheel within the Project’s Information attribute. Project files may include files such as a copy of the MR scanning protocol, study protocols, or any other document relevant to the *entire study*.

Note: Remember: Protected health information or personally identifiable information may **NOT** be stored anywhere within Flywheel, including within file attachments. Please be sure to scrub all identifying information before uploading any files.



To upload files, (1) navigate to the Projects view, (2) select the “Information” panel, (3) Select upload. From the upload menu you can either drag and drop files or point to file you wish to upload.

You may wish to check your file has been attached to the project by viewing the list of attachments.

Upload Subject Files

Including file attachments may also be useful at the subject or session level. Both Subject containers and Session containers have unique metadata. For example, Subject containers have key:value pairs for information such as participant Label, Sex, and Cohort (this information may be customized and will be discussed later). In addition, Subject level files may be added as attachments. Examples of these files include behavioral data not tied to a session (such as reading level), consent forms (de-identified ONLY!), and more.

The screenshot shows the Flywheel web application interface. On the left is a dark sidebar with navigation links: Search, DATA, Projects (highlighted with a red arrow and '1'), Sessions, Collections, Upload DICOM, GEARS, Installed Gears, REPORTING, Project Report, Usage Report, ADMIN, Users, Groups, Roles, and Applications. The main content area is titled 'levitas-tutorial' and 'Bids Curation Testing (fw://bids-curation-testing/levitas-tutorial)'. It has tabs for Description, Sessions (selected with a red arrow and '2'), Information, Analyses, and Data Views. Below these are 'Actions' and 'Advanced Filters' dropdowns. A table lists sessions grouped by subject (indicated by a red arrow and '3'). The table has columns for Subject Label, Session Label, and Timestamp. Two rows are visible: one for subject '10462' with session '1' at '2020-01-22 14:29', and another for subject '085' with session '1' at '2021-07-08 12:06'. A red arrow and '4' point to the first row. To the right of the table is a 'Subject' panel for subject '10462' with fields for Label, Subject Type (Human, Animal, Phantom), First Name, Last Name, Sex, and Cohort. A red arrow points to the bottom of this panel with the text 'scroll down to attachments'. Below the Subject panel is a 'pop out' window titled 'Attachments' with 'Create' and 'Upload' buttons. The 'Upload' button is circled in red. The window also contains a paperclip icon and the text 'No attachments have been added yet'.

To upload files, (1) navigate to the Projects view, (2) select the “Sessions” panel, (3) show all sessions grouped by subject, (4) select the subject of interest from the list. Finally, scroll down to “Attachments”, and select Upload.

Upload Session Files

Uploading session files is a similar process used for subject files. Examples of files that may be attached at a session level would include behavioral data collected with a scanning session such as questionnaires, surveys, task responses, etc.

The screenshot shows the Flywheel web application interface. On the left is a dark sidebar with navigation options: Search, DATA, Projects (highlighted with a red arrow and '1'), Sessions, Collections, Upload DICOM, GEARS, Installed Gears, REPORTING, Project Report, Usage Report, ADMIN, Users, Groups, Roles, Applications, Interfaces, and Data Views. The main content area is titled 'levitas-tutorial' and 'Bids Curation Testing (fw://bids-curation-testing/levitas-tutorial)'. It has tabs for Description, Sessions (selected), Information, Analyses, and Data Views. Below the tabs is a table with columns for Actions, Timestamp, Subject, and Session. The table contains two rows: one for 2021-07-08 12:06 with Subject 085 and Session S2, and another for 2020-01-22 14:29 with Subject 10462 and Session S1. The S1 row is selected. To the right of the table is a form for session details, including Label (S1), Timestamp (2020-01-22 14:29), Age (years) (29), Weight (kg) (65.7709), Operator (Enter operator), Custom Information, and Attachments. The Attachments section has a red circle around the 'Upload' button. Red arrows indicate the steps: (1) Projects view, (2) Sessions panel, (3) list view, and (4) selecting the S1 session.

To upload files, (1) navigate to the Projects view, (2) select the “Sessions” panel, (3) show all sessions by list, (4) select the session of interest from the list. Finally, scroll down to “Attachments”, and select Upload.

Upload Acquisition Files

Finally, files can be attached to an acquisition. If your study is using DICOM ingestion from the MR scanner, each acquisition already contains multiple files, such as dicom.zip and nifti formatted files. Examples of additional files that could be uploaded here would include timing files for events presented during the specific acquisition, KSpace Data, and more!

The screenshot shows the Flywheel web application interface. On the left is a dark sidebar with navigation options: Search, DATA, Projects (highlighted with a red arrow and '1'), Sessions, Collections, Upload DICOM, GEARS, Installed Gears, REPORTING, Project Report, Usage Report, ADMIN, Users, Groups, Roles, Applications, Interfaces, and Data Views. The main content area is titled 'levitas-tutorial' and 'Bids Curation Testing (fw://bids-curation-testing/levitas-tutorial)'. It has tabs for Description, Sessions, Information, Analyses, and Data Views. Below the tabs is a table with columns for Actions, Timestamp, Subject, and Session. The table contains two rows: one for 2021-07-08 12:06 with Subject 085 and Session S2, and another for 2020-01-22 14:29 with Subject 10462 and Session S1. The S1 row is selected. To the right of the table is a form for session details, including Label (S1), Timestamp (2020-01-22 14:29), Age (years) (29), Weight (kg) (65.7709), Operator (Enter operator), Custom Information, and Attachments. The Attachments section has a red circle around the 'Upload' button. Red arrows indicate the steps: (1) Projects view, (2) Sessions panel, (3) list view, and (4) selecting the S1 session. The 'Acquisitions' panel is also visible, showing a table of acquisitions. A red arrow points to the 'Acquisitions' tab, and another points to the 'Upload Data to Acquisition' option in the context menu. The context menu is open, showing options: Information, Download Acquisition Files, Delete Acquisition, Upload Data to Acquisition (highlighted with a red arrow and '5'), Create New File, and Notes.

To upload files, (1) navigate to the Projects view, (2) select the “Sessions” panel, show all sessions by list, and select the session of interest from the list, (3) Select the Acquisition from within “Sessions” panel, (4) Identify the Acquisition of interest, scroll to the far right, and select the ellipsis, (5) From the drop down menu, select “Upload Data to Acquisition”.

1.6.3 Required Project Files

AS UCB users of Flywheel, each laboratory has autonomy in the types of data and format to use for their projects. There are a few exceptions. There is a minimum set of Project Files that must be retained in each Flywheel Project. These files include:

1. UCB Memorandum of Use - <Project Label>
2. UCB Prisma Fit Scanner Protocol - <Project Label>
3. acquisition_label_remapping.csv (this file may be excluded in some cases if acquisitions are labeled using a ReproIn compliant naming at the scanner)

1.6.4 Modifying and Creating Files

Creating files from scratch within Flywheel is conducted in a similar way to uploading files, as described in detail above. When creating files in Flywheel, a list of supported file types will be provided including “Plain Text”, “JSON”, “Python”. Once you have entered the information you wish to store in the file, select “Save Changes” and provide a meaningful filename.

Modifying files can be accomplished in 2 ways:

1. Upload a file with the same name as an existing file
2. Edit the file directly with the User Interface in Flywheel (only for select filetypes)

When a file has been modified, a BLUE icon will show up next to the file name with a counter (e.g. 1,2,3). This icon indicates the *version* of that specific file. By selecting the BLUE icon, you can also see all prior versions of the same file. In a subsequent document, we will go into detail about version control and provenance in Flywheel.

1.7 Viewing Data

Flywheel has a range of tools that can be used to view the data within the User Interface. These applications are being regularly updated and improved by the Flywheel Support Team. Please understand with the rapid development of viewers and applications in Flywheel some of the documentation contained below and on Flywheel Docs may be out of date. If you have any questions about current features for viewing data, contact INC Staff.

1.7.1 Viewing DICOMS

Flywheel has an integrated DICOM viewer which allows you to view, interact, and edit DICOMS directly within our instance of Flywheel. Please visit Flywheel Docs for instructions on how to use the viewer [here](#).

Using the “Picture” icon located next to subject, session, or file will automatically launch the dicom viewer and load the active dicom images.

The screenshot shows the Flywheel web interface for a project named 'levitas-tutorial'. The left sidebar contains navigation menus for DATA, PROJECTS, COLLECTIONS, GEAR, REPORTING, and ADMIN. The main content area displays a table of data files. The table has columns for file name, timestamp, subject, and session. A red circle highlights a file named '2_anat_T1w.dicom.zip' with the annotation 'Dicom selection'. Another red circle highlights the 'View' icon next to the same file with the annotation 'Click on the picture frame to pull up the following images'. A red arrow points to the 'View' icon with the annotation 'The Acquisition'.

1.7.2 Viewing NIFTIs

The same viewer can be used to view NIFTI formatted files. Just as above, navigate to the image of interest, then select the “Picture” icon to open the viewer. Want to learn more? Visit the Flywheel [docs](#).

1.7.3 Viewing Other File Formats

Viewing plain text files

Text files can be viewed and edited within Flywheel’s User Interface. To view or edit text files:

1. Navigate to the file of interest
2. Click the vertical ellipsis to expand the options menu for the file
3. Select View

The screenshot shows the Flywheel web interface for a project named 'levitas-tutorial'. The left sidebar contains navigation menus for DATA, PROJECTS, GEARS, REPORTING, and ADMIN. The main content area displays a table of sessions with columns for Actions, Timestamp, Subject, and Session. A red arrow points to the session '19 - dwl_dir80_PA.bval' under the subject '10462'. A context menu is open for this session, showing options like Information, Download, Delete, Version History, Copy File Name, Copy Modality, and View (which is circled in red).

Actions	Timestamp	Subject	Session
<input type="checkbox"/>	2021-07-08 12:06	085	S2
<input type="checkbox"/>	2020-01-22 14:29	10462	S1

In the viewer you can make changes to the text and save, or simply view the contents of the file.

Note: Remember, Flywheel is organized with a rigid *Hierarchy*. All data is managed by membership to a specific “level” in Flywheel.

```
Group --> Project --> Subject --> Sessions --> Acquisitions
files  --^-----^-----^-----^-----^-- files
```

Viewing spreadsheets

Spreadsheets may also be viewed within Flywheel. At this time, spreadsheets cannot be modified in the interface. Instead, if changes are needed, a new version of the spreadsheet must be uploaded to the Flywheel instance.

1.7.4 Adding and Modifying Metadata

Metadata is simply information about your data or “data about data”. Metadata does not include the contents of your data such as the image itself. Instead, metadata is generally descriptive information about the data such as when the file was created or modified, the file owner, etc. Flywheel metadata is stored at each level of the Flywheel hierarchy, with descriptive information about the Project, Subject, Session, Acquisition, Analysis, or File.

For example, each subject has rich metadata associated with that “container”.

The screenshot shows the Flywheel interface for a project named 'levitas-tutorial'. The 'Subject' tab is active, displaying a table of subjects. Subject 10462 is selected, and its metadata is shown in a form on the right. Red arrows indicate the subject selection and the associated metadata form.

Subject Label	Session Label	Timestamp
10462		2020-01-22 14:29
085		2021-07-08 12:06

Metadata for subject 10462:

- Label: 10462
- Subject Type: ☒ Human ☐ Animal ☐ Phantom
- First Name: Subject First Name
- Last Name: OpenSolJan22
- Sex: Male
- Cohort: Select Cohort
- ML Set: Select ML Set
- Race: Select Race
- Ethnicity: Select Ethnicity
- Species: Select Species
- Strain: Subject Strain

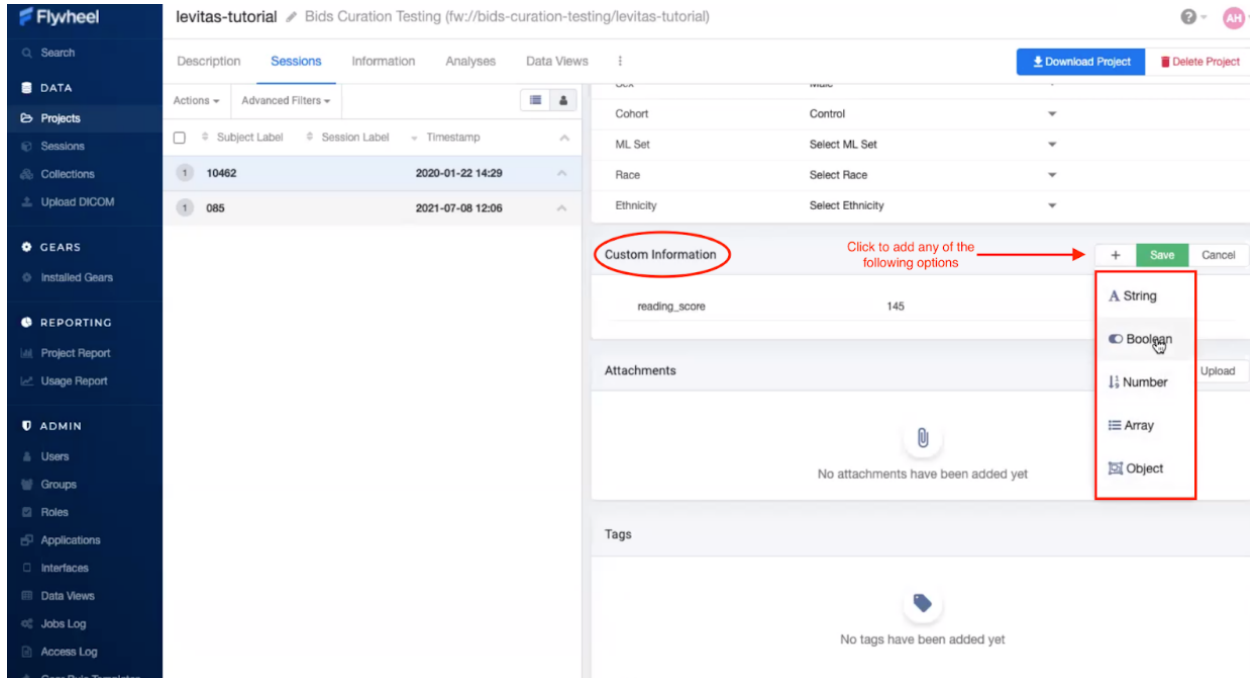
Acquisition and file metadata can be viewed in Flywheel by first expanding the menu options, and selecting Information.

The screenshot shows the Flywheel interface for a project named 'levitas-tutorial'. The 'Acquisitions' tab is active, displaying a table of acquisitions. A dropdown menu is open for the '1-localizer' acquisition, showing options like 'Information', 'Download Acquisition Files', 'Delete Acquisition', 'Upload Data to Acquisition', 'Create New File', and 'Notes'. Red arrows indicate the 'Acquisitions' tab and the 'Information' option in the dropdown menu.

File Description	Created Date	Classification
1-localizer	14:35	
1_localizer_i00003.nii.gz	2022-05-06 09:05	MR: Localizer, T2
1_localizer_i00001.nii.gz	2022-05-06 09:05	MR: Localizer, T2
1_localizer_i00002.nii.gz	2022-05-06 09:05	MR: Localizer, T2
1 - localizer.dicom.zip	2022-05-06 09:05	MR: Localizer, T2
9-localizer	14:44	
9_localizer_i00001.nii.gz	2022-05-06 09:05	MR: Localizer, T2
9 - localizer.dicom.zip	2022-05-06 09:05	MR: Localizer, T2
9_localizer_i00003.nii.gz	2022-05-06 09:05	MR: Localizer, T2
9_localizer_i00002.nii.gz	2022-05-06 09:05	MR: Localizer, T2
anat-FLAIR	14:53	
17 - anat-FLAIR.dicom.zip	2022-05-06 09:05	MR: Structural, FLAIR dicom
17_anat_FLAIR.nii.gz	2022-05-06 09:05	MR: Structural, FLAIR nifti
anat-T1w	14:40	

Adding Custom Metadata

It is also possible to add your own metadata within the “Custom Information” section. Users may add different “types” of metadata such as strings, lists, boolean values, numbers, and complex objects containing additional metadata. For more information about adding your own metadata visit the flywheel documentation ([here](#)).



1.8 Version Control

Version control is used to track all changes made to a file throughout the file’s “lifetime.” Within Flywheel all data are version controlled including all images (e.g. DICOMs, NIFTIs), as well as file attachments (e.g. logs, behavioral data spreadsheets). Flywheel also uses version control for gears to ensure all analyses are reproducible.

Check out Flywheel’s Docs on Version Control to learn more [here](#)!

1.9 Provenance

Provenance is a term we use to describe the history or source of information. For example, if I received a letter in the mail, we might think of that letter’s provenance. The letter was written by someone, and placed in a specific mailbox in a specific city. That letter was then picked up by a mail carrier, went to a central processing facility, and on and on. Finally, the letter was carried by a mail carrier to my door where I opened it. Each step of the letter’s journey is part of the provenance. At the letter’s final destination, we should be able to use the provenance to track exactly what happened to the letter at what time all the way back to when it was placed in the sender’s mail box.

When we think about data provenance, this includes information about all modifications that were made to original or raw data. Each modification should be documented including what modification was taken, the analyst making the modification, the time and date of the modification, and the tools used to make the modification. This information becomes very cumbersome to maintain independently as a researcher. While our computers store some of this information, such as the time and date a file was created or modified, the computer falls far short of what provenance is needed to track each step that was taken on the data’s journey since inception. Luckily, Flywheel makes this very easy and stores a full provenance for all data!

1.9.1 Viewing Provenance

Provenance can be viewed within the Flywheel User Interface. Provenance cannot be modified, or deleted within Flywheel. One important note, Flywheel does not log changes to metadata. It is **critical** to ensure you are not making undesired or accidental changes to metadata fields that cannot be recovered. If accidentally changing metadata is a concern your lab has, please see the section *User Permissions* and examine how each Role can interact with metadata.

To view provenance:

1. From the Sessions List, Select the session you want to view provenance record
2. Navigate to the “Provenance” Tab
3. You will see a list of all jobs using or modifying this session’s data. Each job is labeled with an analysis name, version number, the analyst name, and job status (complete, in progress, failed). To see more information about what modifications were made to the data, select “View Log”

The screenshot shows the Flywheel web application interface. On the left is a dark sidebar with navigation links: DATA, Projects, Sessions, Collections, Upload DICOM, GEARS, Installed Gears, REPORTING, Project Report, Usage Report, ADMIN, Users, Groups, Roles, Applications, Interfaces, and Data Views. The main content area is titled 'levitas-tutorial' and 'Bids Curation Testing (fw://bids-curation-testing/levitas-tutorial)'. It has tabs for Description, Sessions, Information, Analyses, and Data Views. The 'Sessions' tab is active, showing a table with columns: Actions, Timestamp, Subject, and Session. Two sessions are listed: one with timestamp '2021-07-08 12:06' and subject '085' (Session S2), and another with timestamp '2020-01-22 14:29' and subject '10462' (Session S1). A red arrow labeled '1' points to the second session. To the right of the sessions table is the 'Provenance' tab, which shows a list of jobs. At the top of the Provenance tab, it says 'Filter By Gear Name', '12 Jobs Processed', and '19 errors'. A red arrow labeled '2' points to the 'Provenance' tab. The jobs are listed in a table with columns: GEAR, INPUT, OUTPUT, and ACTIONS. Each job entry includes a status icon (green checkmark for success, red triangle for error), the gear name and version, the user, the input file, the output file, and a 'View Log' button. A red arrow labeled '3' points to the 'View Log' button for the 'curate-bids v2.1.3_1.0.7' job.

Viewing Job Details

Each analysis that was run is also assigned a “Job ID”. This is a unique ID within the UCB Flywheel instance that can be used to identify the job across the platform.

To view the Job ID, open an analysis log and look for the Job ID in the bottom left corner of the logs page.

Note: To view the Job ID, start from the provenance tab to look at the logs.

Logs for curate-bids v2.1.3_1.0.7



```

Gear Name: curate-bids, Gear Version: 2.1.3_1.0.7
Executor: compute, CPU: 16 cores, Memory: 34GB, Disk: 250GB, Swap: 0B
Gear starting...
[ run 2022-06-13 19:49:02,241 INFO] Using Saved Config if present
/root/.cache/pypoetry/virtualenvs/curate-bids-n1iZ4KF1-py3.9/lib/python3.9/site-
packages/flywheel/flywheel.py:6300: UserWarning: Client version 15.8.0 does not match server version 16.4.3.
Please update your client version!
  warnings.warn('Client version {} does not match server version {}. Please update your client
version!'.format(SDK_VERSION, release_version))
[ flywheel 2022-06-13 19:49:02,264 WARNING] Use "pip install flywheel-sdk~=16.4.4" to install a compatible version
for this server
[ run 2022-06-13 19:49:02,314 INFO] Curating single session, id 6275392bad4a9a8909fe641d
[ run 2022-06-13 19:49:02,682 INFO] Did not find saved project configuration file
[ flywheel 2022-06-13 19:49:02,701 WARNING] Use "pip install flywheel-sdk~=16.4.4" to install a compatible version
for this server
[ curate_bids 2022-06-13 19:49:03,074 INFO] Using project curation template: Flywheel ReproIn:
https://gitlab.com/flywheel-io/public/bids-client/-/blob/master/flywheel_bids/templates/reproin.json
[ project_tree 2022-06-13 19:49:03,074 INFO] Getting project...
[ rtstat 2022-06-13 19:49:03,099 INFO] Recruiting GPU to draw funny comics
[ curate_bids 2022-06-13 19:49:03,099 INFO] 0: Bidsifying Container: <project> <levitas-tutorial>
[ curate_bids 2022-06-13 19:49:03,100 INFO] Updating BIDS metadata on Flywheel
[ curate_bids 2022-06-13 19:49:03,126 INFO] Getting single session ID=6275392bad4a9a8909fe641d
[ curate_bids 2022-06-13 19:49:03,477 INFO] Getting single subject ID=6275392b71bbaef960c8b84b
[ project_tree 2022-06-13 19:49:03,494 INFO] Getting single session ID=6275392bad4a9a8909fe641d
[ curate_bids 2022-06-13 19:49:04,060 INFO] 1: Bidsifying Container: <session> <S1>
[ curate_bids 2022-06-13 19:49:04,061 INFO] 2: Bidsifying Container: <acquisition> <l-localizer>
[ curate_bids 2022-06-13 19:49:04,062 INFO] 3: Bidsifying Container: <acquisition> <anat-T1w>
[ curate_bids 2022-06-13 19:49:04,064 INFO] 4: Bidsifying Container: <acquisition> <fmap-epi_acq-opphase_dir-AP>
[ curate_bids 2022-06-13 19:49:04,066 INFO] 5: Bidsifying Container: <acquisition> <fmap-epi_acq-opphase_dir-PA>

```

Job ID: 62a794aa26fd1108bc7dd231

[Download](#)

Viewing Analysis Container Provenance

1. From the Sessions list, Select the session you want to view provenance record
2. Navigate to the “Analyses” Tab
3. You will see a list of all analysis containers associated with that session. In addition to Results, Notes, and Custom Information, each analysis container has Gear Configuration, Gear Information, and Gear Logs that track the provenance of the analysis. To see more information about what modifications were made to the data and how the analysis ran, select “Gear Logs”.

1.10 Gears

1.10.1 Overview

In Flywheel, a “Gear” is another name for an analysis or computations run on your data. Flywheel Gears are “containers” that package or “bundle” important metadata with the analysis results. Additional information stored within gears includes all of the information about who ran the analysis, when, with what inputs, the analysis software name, version, maintainer, and of course all of the analysis outputs. Gears keep this critical metadata in one place making analyses trackable and easier to review. Gears will already be installed into Flywheel by INC staff, if you wish to add more, see [Looking For Other Gears?](#) You can view all gears installed on the UCB Flywheel Instance in the “Installed Gears” View in the Gears section of the sidebar menu. Interested in learning more about gears? Check out Flywheel’s documentation on gears [here](#).

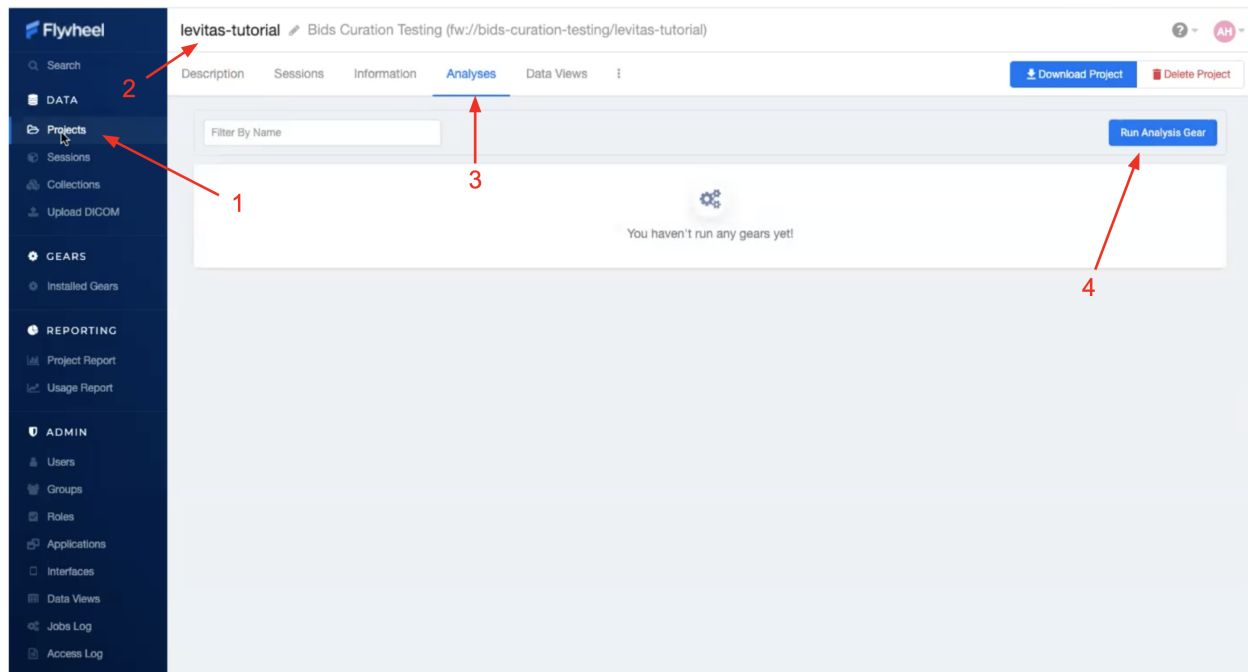
1.10.2 Running Gears

It is important to note, you can start a gear from multiple levels of the Flywheel hierarchy. This provides the flexibility to run a range of software tools that may need any number of inputs, such as only a single acquisition (**acquisition level**), all acquisitions in a session (**session level**), all sessions for a single subject (**subject level**), multiple or all subjects in a given project (**project level**). Being diligent about the Flywheel hierarchy level you use to run your gear is critical. For example, some gears will only execute if ran at a “session” level.

Running Gears at the Project Level

Running gears at a project level is most commonly used for running group analyses. In order to run a gear on an entire project, follow the steps below.

1. Select “Projects” on the sidebar menu.
2. Select your active project.
3. Navigate to the “Analyses” panel in that project.
4. Click the “Run Analysis Gear” button in the upper right corner.

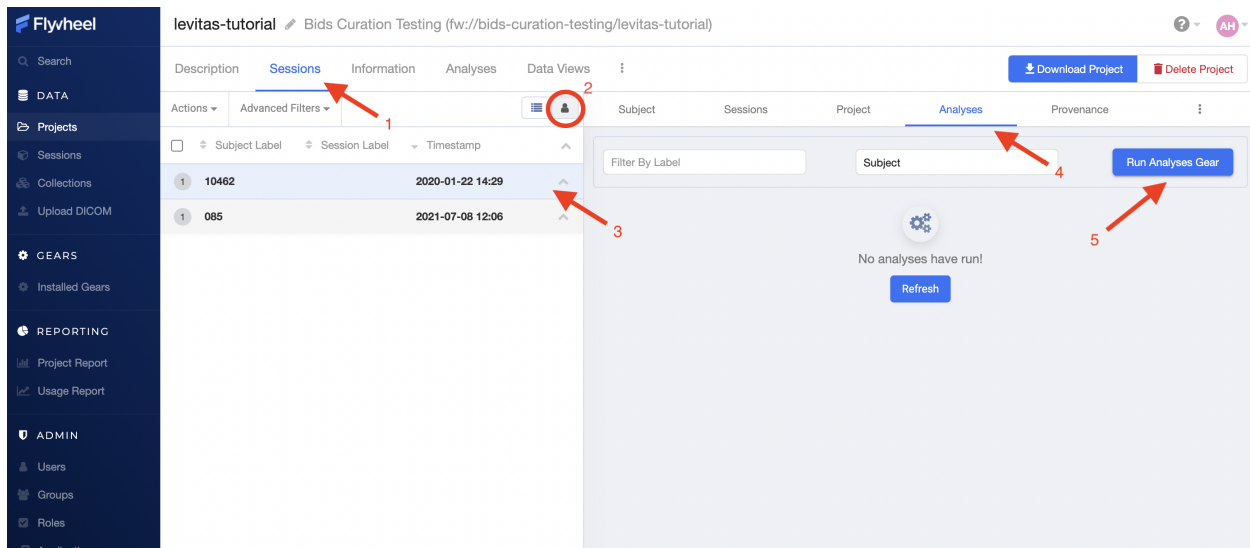


5. Select the gear and version
6. Click “Run Gear”

Running Gears at the Subject Level

Running gears at a subject level is most commonly used for analyses that require multiple or all sessions from a single subject to be accepted as inputs at the same time. Many BIDS formatted analyses may be run at a subject or session level. In order to run a gear on an entire subject, follow the steps below.

1. From your project, select “Sessions” panel
2. Toggle the View to “Subject View”
3. Select a single subject for your analysis
4. Select the “Analyses” panel inside the Sessions Menu
5. Click the “Run Analysis Gear” button in the upper right corner



Running Gears at the Session Level

Running gears at a session level is the most common method for running analysis gears. Most gears including curation, re-naming, and preprocessing gears can be run on a single session. In order to run a gear on a single session, follow the steps below.

1. From your project, select “Sessions” panel
2. Make sure the view is toggled to “List View”
3. Select a single session for your analysis
4. Select the “Analyses” panel inside the Sessions Menu
5. Click the “Run Analysis Gear” button in the upper right corner

The screenshot shows the Flywheel web interface for a project named 'levitas-tutorial'. The left sidebar contains navigation menus for DATA, Projects, GEARS, REPORTING, and ADMIN. The main content area has tabs for Description, Sessions, Information, Analyses, and Data Views. The 'Sessions' tab is active, displaying a table of sessions. Red arrows and numbers indicate the following steps: 1. Click the 'Sessions' tab. 2. Click the 'List View' icon. 3. Click the session 'S1' (timestamp 2020-01-22 14:29). 4. Click the 'Acquisitions' tab. 5. Click the 'Run Analyses Gear' button in the top right corner of the 'Acquisitions' tab.

Running Gears at the Acquisition Level

Running gears on a single acquisition is generally reserved for light weight “Utility” Gears and “Gear Rules”, which will be discussed later. In order to run a gear on a single acquisition, follow the steps below.

1. From your project, select “Sessions” panel
2. Make sure the view is toggled to “List View”
3. Select a single session for your analysis
4. Select the “Acquisitions” tab
5. Click the “Run Gear” button in the upper right corner
6. Select either “Utility Gear” or “Analysis Gear” (gear methods discussed below)

Important: Running an “Analysis Gear” will default to running at a Session Level even from this menu!

The screenshot shows the Flywheel web interface for the same project. The 'Acquisitions' tab is now active, displaying a table of acquisitions. Red arrows and numbers indicate the following steps: 1. Click the 'Sessions' tab. 2. Click the 'List View' icon. 3. Click the session 'S1'. 4. Click the 'Acquisitions' tab. 5. Click the 'Run Gear' button in the top right corner of the 'Acquisitions' tab.

Gear Versions

Flywheel Gears are version controlled, meaning both the underlying software version and the version of Flywheel's container are recorded and stored in the analysis. In addition, INC often makes custom changes to the gears on this instance and is identified with INC versioning.

Listing 1: Flywheel Gear Versions

```
Example Gear Version:
1.2.5_20.0.6_inc1.0
The meaning of the versioning broken down piece by piece
1.2.5 --> Flywheel Gear Version
20.0.6 --> Underlying Pipeline / Software Version
inc1.0 --> INC Customized Version
```

If you find that a gear version is missing the `inc` version tag, that simply means that INC did not modify the gear from what Flywheel originally produced.

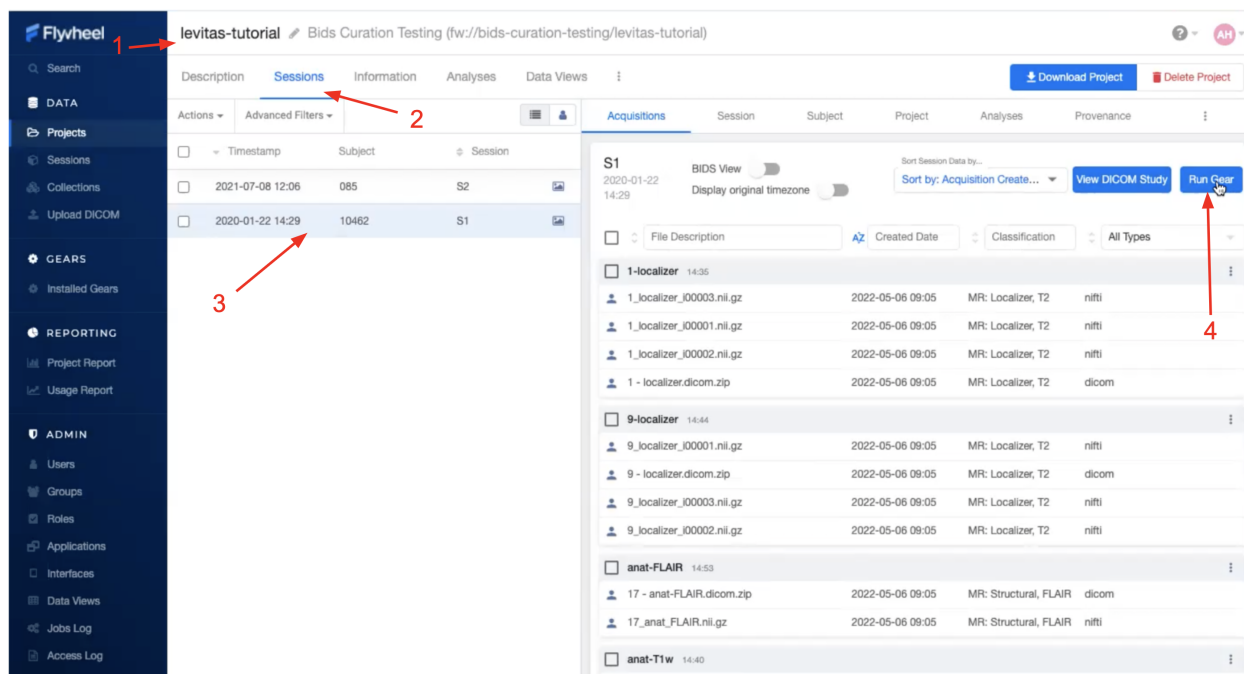
Note: When starting a new analysis, it is best practice to select the most updated version of the gear. Once a version is selected for a project, we recommend to use the same version for the duration of your project.

1.10.3 Gear Methods

Running Utility Gears

Utility Gears are lightweight analyses usually used only for converting file types or performing quality assurance checks. The outputs of these gears are saved directly with the input data.

1. Navigate to the desired project
2. Select the “Sessions” panel
3. Select the desired session from the list of sessions
4. From the “Acquisitions” tab, Click “Run Gear” in the upper right corner
5. Select Utility Gear



6. Select a gear and version

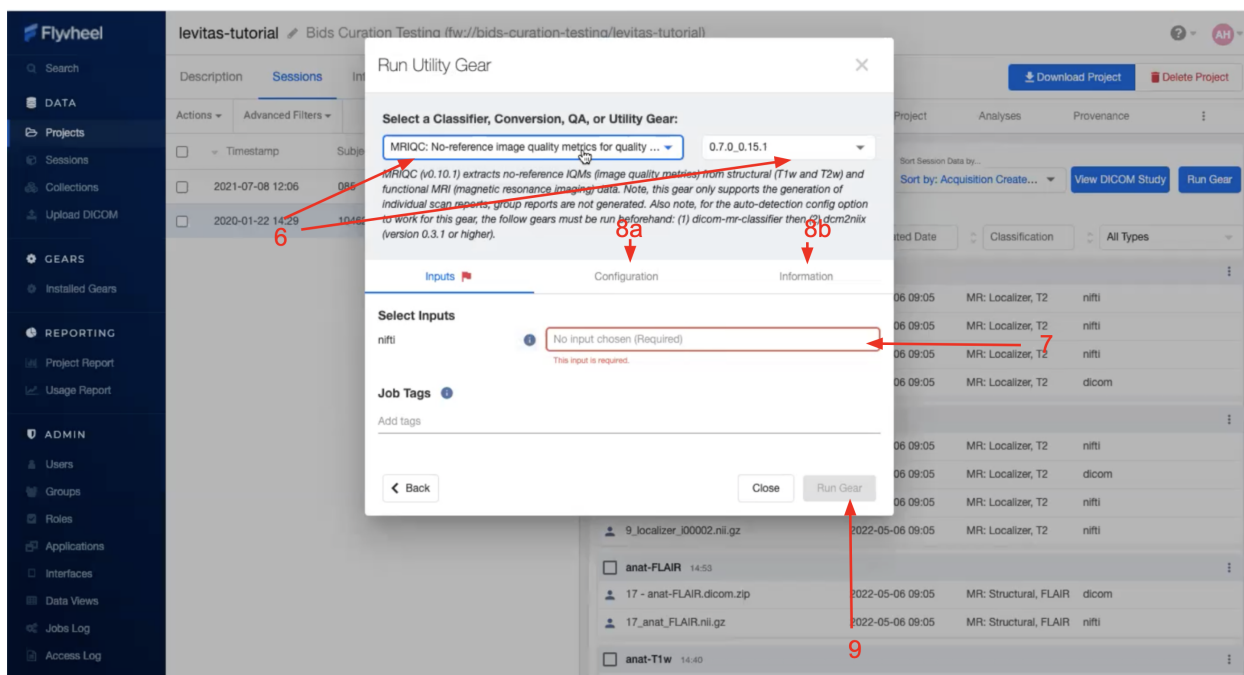
7. Choose a file for the input

8. Under the configuration tab, select the necessary options for your job

a. For more information about each configuration setting, hover the mouse over the info icon next to the configuration

b. The information tab tells you all the data and metadata that will be stored about the job

9. Click “Run Gear”



Running Analysis Gears

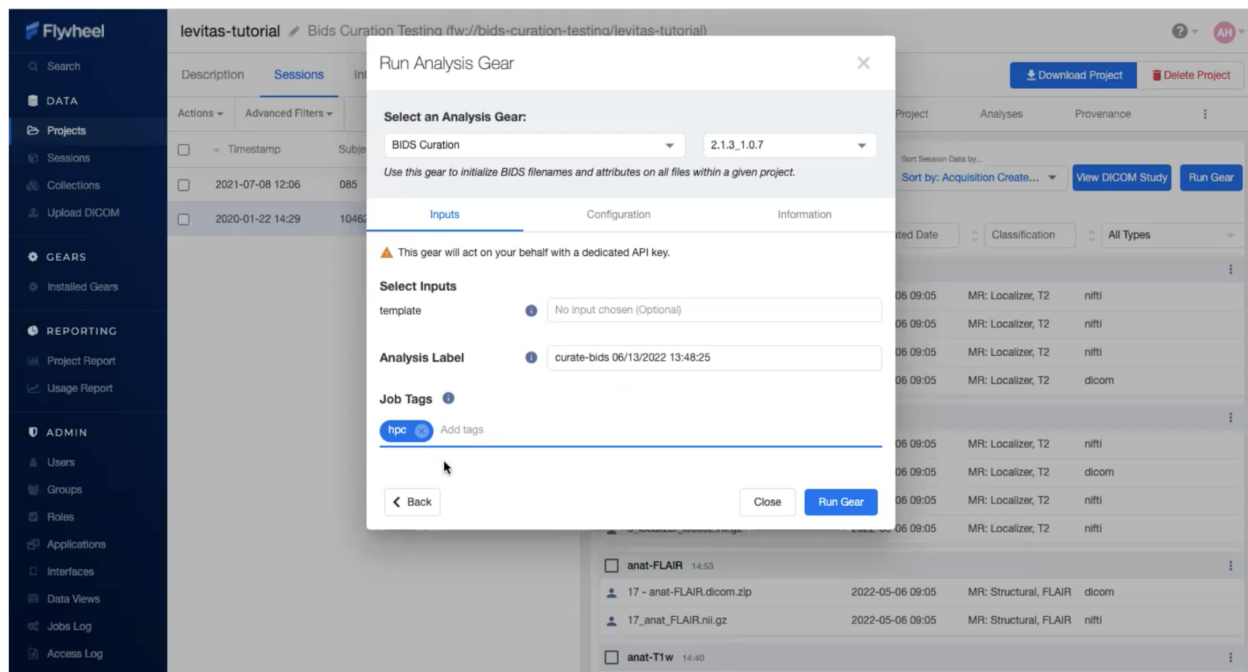
Analysis Gears are used for larger analyses and are organized as unique containers (or “folders”) in Flywheel. These containers are conveniently called Analyses in your Session view. Analysis Gears package additional information including the gear inputs, configuration, version, logs, and outputs.

To run an Analysis Gear, follow the steps below:

1. Navigate to the desired project
2. Select the “Sessions” tab
3. Select the desired session from Session’s List
4. From the “Acquisitions” panel, Click “Run Gear” in the upper right corner
5. Select Analysis Gear
6. Select the gear and version of the gear you want to run
7. Select all the necessary input files
8. **Change any options under the configuration tab that are needed**
 - a. For more information about each configuration setting, hover the mouse over the info icon next to the configuration
 - b. The information tab tells you all the data and metadata that will be stored about the job
9. Select “Run Gear”

Running High Performance Compute (HPC) Analysis Gears

In order to run Gears on the HPC environment (instead of the CUMulus virtual machines), add a job tag “hpc” when setting up the analysis. This job tag is case sensitive. Additional settings including SLURM resources configurations may be available in the gear’s configuration settings.



1.10.4 View Gear Status

To view the progress of your gear, navigate to the desired session, and then select the “Provenance” tab. Provenance shows a list of all gears for a specific session. To view the results of your *Analysis* Gears, navigate to the “Analyses” tab.

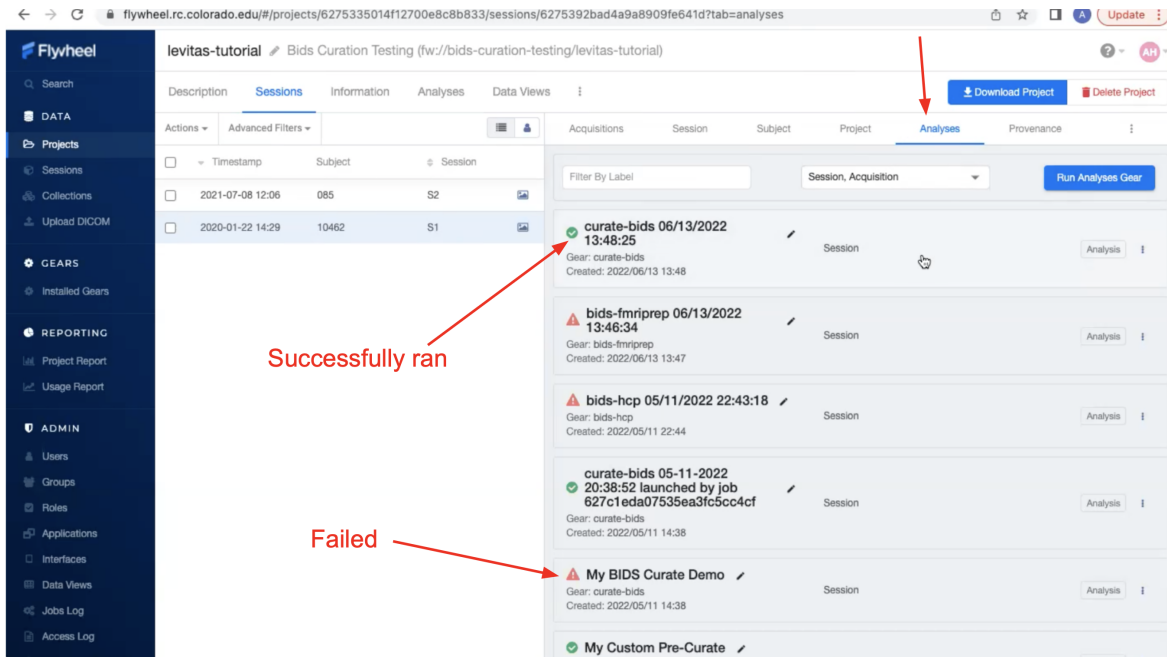
The screenshot displays the Flywheel web application interface. On the left is a dark sidebar with navigation links: Search, DATA, Projects, Sessions, Collections, Upload DICOM, GEARS (Installed Gears), REPORTING (Project Report, Usage Report), and ADMIN (Users, Groups, Roles, Applications, Interfaces, Data Views, Job Log, Access Log). The main header shows the project name 'levitas-tutorial' and a breadcrumb 'Bids Curation Testing (fw://bids-curation-testing/levitas-tutorial)'. Below the header is a tabbed interface with 'Sessions' selected. A table lists sessions with columns for Actions, Timestamp, Subject, and Session. The 'Provenance' tab is highlighted in the top right navigation bar, indicated by a red arrow. The Provenance view shows a summary bar with 'Processing', '10 Jobs Processed', and '18 errors', along with a 'Refresh' button. Below this is a table of gears with columns: GEAR, INPUT, OUTPUT, and ACTIONS. The table lists several gears, including 'mriqc v0.7.0.0.15.1', 'bids-hcp v1.2.5.4.3.0', 'curate-bids v2.1.3.1.0.7', and 'custom-pre-curate v0...'. Each gear entry shows its input and output files and provides 'Cancel', 'Retry', or 'View Log' buttons.

What To Do If Your Gear Failed?

If your gear fails, first check the job log. If there is an obvious error message such as missing or incorrect input, make that change and run the gear again. If a problem persists or you cannot determine why the gear failed, reach out to the [INC staff](#) for assistance.

What To Do If Your Gear Succeeded?

After your analysis gear completes successfully you can inspect the results in the “Analyses” tab. Select the analysis container of interest. You should see a “Results” tab with all data created during the analysis. From this container, you may also review the records of all inputs, configurations, gear information, and logs.



1.10.5 Looking For Other Gears?

The [Flywheel Gear Exchange](#) is a place where you can see other gears created for Flywheel (usually by Flywheel staff). If you wish to add another gear to our Flywheel instance, look on the Gear Exchange and contact a member of the INC staff to request the gear be added to UCB Flywheel instance. If the gear is not in the Gear Exchange, contact INC staff as well for a consultation. [INC staff](#) can work with Flywheel to develop the gear, or train/direct you on how to develop your own Flywheel gears.

1.11 CLI Basics

1.11.1 CLI From Local Computer

Follow the steps below to interact with Flywheel from the command line interface on your own personal computer.

1. Install the CLI on the machine from which you'd like to transfer data to/from Flywheel using the [Flywheel CLI Guide](#)
2. Now that the CLI is installed, if you'd like to ingest BIDS data from your local machine into Flywheel, follow the [Flywheel BIDS Ingestion Guide](#). For example, the command to ingest a single BIDS session (for subject 101, session S1) for a project called TEST, which is in group **sample** is:

```
fw ingest bids -p TEST --subject 101 --session S1
<path_to_your_BIDS_folder> sample
```

3. If you'd like to ingest raw dicom data from your local machine into Flywheel, follow the [Flywheel Dicom Ingestion Guide](#). For example, the command to ingest the raw dicoms for a single session (for subject 101, session S1) for a project called TEST, which is in group **sample** is:

```
fw ingest dicom <path_to_your_dicom_folder> sample TEST --subject 101
--session S1
```

4. For a full list of other Flywheel command line features (including checking login status, managing server jobs, exporting and downloading data, and much more), review the [Flywheel CLI Overview](#)

1.11.2 CLI From Blanca Compute Node

For your convenience, the Flywheel CLI is already installed in a Conda environment on PetaLibrary. To use the Flywheel CLI from a compute node, follow the steps below:

1. Log in to a Blanca compute node (interactive session, JupyterHub, or even a Summit compute node will work)
2. Load the latest Anaconda3 module:

```
source /curc/sw/anaconda3/latest
```

3. Activate the *flywheel* conda environment:

```
conda activate flywheel
```

4. Check the flywheel status to ensure you are logged in as the correct user:

```
fw status
```

5. If you are not logged with yourself as the user, log in to the Flywheel API with your Flywheel API key:

```
fw login <your_api_key>
```

6. Now you are ready to run any FW API command (refer to the section above for specific commands)

Note: Special Instructions for Large Uploads: Even though the Flywheel CLI lives on PetaLibrary in the conda environment called *flywheel*, the cache for all temporary data is hard coded to go to your *home* directory on PetaLibrary. However, because your *home* directory is so small, this cache can quickly fill up resulting in the upload failing and your *home* directory full. To avoid this issue, you should create a soft link between your *home* cache directory and your *projects* cache directory, which is much larger, ie:

```
/home/<identiKey>/cache/flywheel -> /projects/<identiKey>/cache/flywheel
```

1.12 SDK Basics

If you do not yet have experience using Open OnDemand to launch Jupyter Notebooks, please first review the documentation [High Performance Compute Portal](#). Be sure to follow all instructions, including [Setting Up Conda Environments](#)

Follow our example tutorial using the Flywheel SDK to [Flywheel Software Developer Kit \(SDK\) Basics and Helpful Hints](#).

1.13 Deidentification

1.13.1 Deidentification Overview

There are many instances in which you would want to de-identify your data before ingesting it into Flywheel. The main reason is if your acquired data from the scanner is not already de-identified (ie, if you don't scan at INC). Below is a brief summary of several ways to de-identify your data before it enters Flywheel, or once it's already in Flywheel.

Warning: The UCB instance of Flywheel relies on storage (PetaLibrary) that can have **no personal identifying information** on it. Therefore, it is crucial to de-identify your data if it was collected somewhere outside of INC.

1.13.2 Deidentification on Edge

De-identification on edge refers to wiping out any identifying information in dicom headers at the source (before those dicoms are ever uploaded to Flywheel). That source can be your institution's data storage cluster, or your institution's scanner. In either case, Flywheel will have no record of any personal identifying information (PII) in your dicom headers if de-identification happens on edge.

Fortunately, through the Flywheel command line interface tool (CLI), Flywheel can assist in the de-identification process without “knowing” about your data or registering it in the Flywheel database.

Pre-requisites

To get started, you'll need to download the Flywheel CLI on the machine where your data resides, or on a machine that has mounted access to where your data resides: visit the [CLI Basics](#) page.

Before getting started, you'll also need some way to read the existing header information in your dicom files (for example, [DCMTK's dcmdump tool](#)). Alternatively, if you already know the header codes and header information, you can forgo looking it up with the dicom header reader.

Once the Flywheel CLI is downloaded (and your personal API key generated), you are ready to start ingesting de-identified data into Flywheel.

Simple De-identification

If the only identifiable information in your dicoms exists in the Patient ID, Patient Name, and Patient Birthdate, Flywheel has a pre-built flag that simply removes those dicom header fields from every dicom file that is uploaded.

Suppose we want to upload a dicom acquisition from our local computer onto Flywheel. We first inspect the header information in that acquisition (just checking one dicom in the acquisition is enough). All of the following commands are run in a Terminal or DOS prompt (aka command line interface):

```
dcmdump <path_to_single_dicom>
```

For example:

```
dcmdump just_t1_dummy_ids/ics/deid/deid02/inc01/anat-T1w_acq-mpr08_run-01/1.3.12.2.1107.
→ 5.2.43.67087.2022101911212761306002309.0.0.0.dicom/1.3.12.2.1107.5.2.43.67087.
→ 2022101911284077703103114.MR.dcm
```

The returned output below (a partial dump of the dicom header) makes it clear that there is indeed identifying information:

Dicom Code	Field Name	Value
(0010,0010)	Patient's Name	John
(0010,0020)	Patient ID	Doe
(0010,0030)	Patient's Birth Date	19010101
(0010,0040)	Patient's Sex	F
(0010,1010)	Patient's Age	121Y

To upload this acquisition but *without* the Patient's Name, Patient ID, or Patient's Birth Date, the Flywheel command is simply to add the `--de-identify` flag onto the `fw ingest dicom` command.

```
fw ingest dicom --de-identify --subject <subject_ID> --session <session_ID> <path_to_
↳dicom_directory> <Flywheel Group> <Flywheel Project>
```

For example:

```
fw ingest dicom --de-identify --subject deid01 --session inc01 just_t1_dummy_ids/ics/
↳deid/deid02/inc01/anat-T1w_acq-mpr08_run-01/1.3.12.2.1107.5.2.43.67087.
↳2022101911212761306002309.0.0.0.dicom ics deid
```

More details about the `fw ingest dicom` command are given on our [CLI Basics](#) page.

If you now check your upload in Flywheel by clicking on the information button next to the dicom, you'll see that the Patient's Name, Patient ID, and Patient Birth Date simply don't exist as metadata.

Note: While the Patient's Name, Patient ID, and Patient's Birthdate have been de-identified, if you had any other identifying information in your dicom header, it is now in Flywheel. Not good. For scrubbing other identifying metadata from the dicom header, read on.

De-identification Profile

Creating a de-identification profile allows you to have maximum control over how *each and every* piece of metadata in your dicom header is handled. Creating a de-identification profile can be a daunting task at first - not because it's difficult - but because of how many different options you have at every decision. Some of the basic options include: do nothing, increment all age-related numbers by some given amount, delete the data entirely, replace the data with something different, jitter the data if it's a number, and hash the data.

Flywheel has extensive information about how to create a de-identification profile on their [de-id documentation website](#).

In short, this de-identification profile is nothing more than a YAML file. The example below shows a de-identification profile that does the following:

- Sets up a place to store the de-id log (a csv file that allows you to re-identify your data, should you need to)
- Increments each date field by subtracting 17 days
- Sets the patient's age to use Year units
- Calculates the Patient's Age from the Patient Birth Date dicom tag
- Removes the PatientID dicom tag
- Replaces the dicom tag StationName with XXXXX
- Hashes the AccessionNumber and ConcatenationUID dicom tags

```
# the de-id updates before uploading
# The option is ignored in ingest, you can use --save-deid-logs PATH to save the log.

deid-log: ./deid_log.csv

# Sets the filetype to DICOM

dicom:
```

(continues on next page)

(continued from previous page)

```

# Date-increment controls how many days to offset each date field
# where the increment-date (shown below) is configured.
#Positive values will result in later dates, negative
# values will result in earlier dates.

date-increment: -17

# patient-age-from-birthdate sets the DICOM header as a 3-digit value with a suffix
# be 091D, and that same age in months would be 003M. By default, if
# the age fits in days, then days will be used,
# otherwise if it fits in months, then months
# will be used, otherwise years will be used

patient-age-from-birthdate: true

# Set patient age units as Years. Other options include months (M) and days (D)

patient-age-units: Y

# The following are field transformations.
# Remove, replace-with, increment-date, hash, and hashuid can be used with any DICOM
# field. Replace name with the DICOM field "keyword" by the DICOM standard
fields:

# Use remove Remove a dicom field Removes the field from the DICOM entirely.
# If removal is not supported then this field will be blank.
# This example removes PatientID.

- name: PatientID
  remove: true

# Replace a dicom field with the value provided.
# This example replaces "StationName" with "XXXX" in Flywheel

- name: StationName
  replace-with: XXXX

# Offsets the date by the number of days defined in
# the date-increment setting above, preserving the time
# and timezone. In this example, StudyDate appears as 17 days earlier

- name: StudyDate
  increment-date: true

# One-Way hash a dicom field to a unique string

- name: AccessionNumber
  hash: true

# Replaces a UID field with a hashed version of that
# field. The first four nodes (prefix) and last node

```

(continues on next page)

(continued from previous page)

```
# (suffix) will be preserved, with the middle being  
# replaced by the hashed value
```

```
- name: ConcatenationUID  
  hashuid: true
```

Testing the De-identification Profile

Once you’ve created your de-identification profile, Flywheel also has a command line interface tool that allows you to test your profile on sample data before using it more broadly for the final dicom upload.

Extensive documentation on testing your de-id profile exists on the [Flywheel site](#) as well as a brief summary below.

In the previous section, we created a de-identification profile that we called `deid_profile.yaml`. Suppose we now want to test how this profile transforms one of our example dicom directories, and store the results of this transformation in a directory we call `deid_test_dir`. Below is the Flywheel command line call that performs the aforementioned steps:

```
fw deid test <path_to_dicom_directory_to_deid> <path_to_deid_yaml_profile> <path_to_  
→directory_for_test_results> --session <session_ID> --subject <subject_ID>
```

For example:

```
fw deid test just_t1_dummy_ids/1.3.12.2.1107.5.2.43.67087.2022101911212761306002309.0.0.  
→0.dicom deid_profile.yaml deid_test_dir --session inc01 --subject deid02
```

The result of this call from the terminal creates a csv file called `deid_log.csv` in the directory `deid_test_dir`. The CSV file shows a before and after (what each dicom header field was before the transformation, and what it became after the transformation). When you first build a de-id profile, it’ll be an iterative process of testing the profile to make sure you have captured all the desired transformations and haven’t left any identifying information in the dicom header.

Uploading the De-ID Profile To Your Flywheel Project

Once you have created and tested your de-identification profile, you can ask the INC team to upload your profile to the relevant Flywheel Project. Once the profile exists as an attachment in your Project settings, any upload you perform (via the GUI, SDK, or CLI) for that Project will first be de-identified based on the rules you laid out in your profile.

Alternatively, if you’d rather keep your de-id profile secret, want to apply different de-id profiles for different subjects, etc, it’s best to continue to the next section which describes how to upload data while the de-id profile remains local to your personal computer (not in Flywheel).

The Ingest Config Template

Now that you’ve put in the hard work into making the perfect de-id profile, you’d like to use it for an actual data upload. However, if you opted not to upload the de-id profile to Flywheel, there’s one more step: the Ingest Config Template.

The ingest config template is a broad topic in and of itself (best described in the [Flywheel template documentation](#)).

Briefly, the ingest template is a configuration YAML file that allows you to control every part of the Flywheel upload process. For example, the ingest template defines the relationship between your local folder structure (where the data exists) and how you want that data to be labelled and mapped onto the Flywheel data hierarchy. Critically, the ingest config template also defines the de-identification profile(s) for the given Project.

The ingest template is its own topic and won't be covered in this section; however, to apply the de-id profile you created and tested, you simply need to paste it into the ingest config template. Below is an example of an ingest config template titled `config.yaml`. Notice the copied and pasted de-identification profile we worked on in previous sections.

```
#####
# Template and Group/Project Settings
#####

template:
  - pattern: "{group}"
  - pattern: "{project}"
  - pattern: "{subject}"
  - pattern: "{session}"
  - pattern: "{acquisition}"
  packfile_type: dicom

#####
# Optional includes/excludes for directories and files
#####

# Patterns of directories to include
# include-dirs:
# - "*.dicom"

# Patterns of filenames to exclude
# exclude:
# - "*.txt"
# - "*.xml"

#####
# De-identification Settings
#####
deid-profiles:
- name: Anschutz

  # Indicates where you want to place the de-id log. You will use this log file to
  ↪ preview
  # the de-id updates before uploading
  # The option is ignored in ingest, you can use --save-deid-logs PATH to save the log.

  deid-log: ./deid_log.csv

  # Sets the filetype to DICOM

  dicom:

    # Date-increment controls how many days to offset each date field
    # where the increment-date (shown below) is configured.
    # Positive values will result in later dates, negative
    # values will result in earlier dates.

    date-increment: -17
```

(continues on next page)

(continued from previous page)

```
# patient-age-from-birthdate sets the DICOM header as a 3-digit value with a suffix
# be 091D, and that same age in months would be 003M. By default, if
# the age fits in days, then days will be used,
# otherwise if it fits in months, then months
# will be used, otherwise years will be used
```

```
patient-age-from-birthdate: true
```

```
# Set patient age units as Years. Other options include months (M) and days (D)
```

```
patient-age-units: Y
```

```
# The following are field transformations.
# Remove, replace-with, increment-date, hash, and hashuid can be used with any DICOM
# field. Replace name with the DICOM field "keyword" by the DICOM standard
```

fields:

```
# Use remove Remove a dicom field Removes the field from the DICOM entirely.
# If removal is not supported then this field will be blank.
# This example removes PatientID.
```

```
- name: PatientID
  remove: true
```

```
# Replace a dicom field with the value provided.
# This example replaces "StationName" with "XXXX" in Flywheel
```

```
- name: StationName
  replace-with: XXXX
```

```
# Offsets the date by the number of days defined in
# the date-increment setting above, preserving the time
# and timezone. In this example, StudyDate appears as 17 days earlier
```

```
- name: StudyDate
  increment-date: true
```

```
# One-Way hash a dicom field to a unique string
```

```
- name: AccessionNumber
  hash: true
```

```
# Replaces a UID field with a hashed version of that
# field. The first four nodes (prefix) and last node
# (suffix) will be preserved, with the middle being
# replaced by the hashed value
```

```
- name: ConcatenationUID
  hashuid: true
```


Putting it All Together

The last step once the de-id profile and the template config YAML are ready, is to make the actual call to Flywheel to upload your dicoms. This is done either with a call to `fw ingest dicom` or with `fw ingest template`.

To use `fw ingest dicom` to upload the example data to Flywheel Group ics, Flywheel Project deid, Flywheel Subject deid03, and Flywheel Session inc01, using our created config file `config.yaml` which includes the de-id profile named `Anschutz`, we use the following command line call:

```
fw ingest dicom --config-file config.yaml --de-identify --deid-profile Anschutz --
↳subject deid03 --session inc01 ./just_t1_dummy_ids/ics/deid/deid02/inc01/anat-T1w_acq-
↳mpr08_run-01/1.3.12.2.1107.5.2.43.67087.2022101911212761306002309.0.0.0.dicom ics deid
```

More options are available with `fw ingest template`, but to accomplish the same upload as above, the command line argument is:

```
fw ingest template -C config.yaml ./just_t1_dummy_ids --group ics --project deid --de-
↳identify --deid-profile Anschutz
```

1.13.3 Deidentification From the Scanner

It is also possible to create a profile that applies to data coming directly from the scanner. If this is of interest, please contact Lena or Amy.

1.13.4 Deidentification Gear in Flywheel

Lastly, there's an additional option to de-identify the data once it's already in Flywheel by running the de-id gear. However, Flywheel has version control on files (including DICOM files), so a copy of your "identifiable" data before the de-id gear was run will exist somewhere in Flywheel (even if it is not accessible to all users). Since the CUB instance of Flywheel can't have any identifiable information at any time, running the de-id gear is not an option we advertise on our site.

1.14 Running Commonly Used Gears

If you have not already reviewed the "*Gears*" Basics, please go back and work through the introduction to Gears before continuing. In the following documentation, step-by-step instructions to run commonly used gears are included. If you have any questions, please contact INC staff. To contribute, please use the INC [github repository](#)

For all gears described below, begin by launching a session level analysis gear.

1. Select the session for the new analysis from the list of sessions on the left hand side. Once the correct session is selected, click "Run Gear" in the top right corner.

The screenshot shows the INC Flywheel interface. On the left, a table lists sessions with columns for Timestamp, Subject, and Session. The session with ID 137 is selected. On the right, a detailed view of this session is shown, including a list of acquisitions. A red circle highlights the 'Run Gear' button in the top right corner. A red arrow points from this button to the 'Analysis Gear' option in the 'Select Gear Type' dialog.

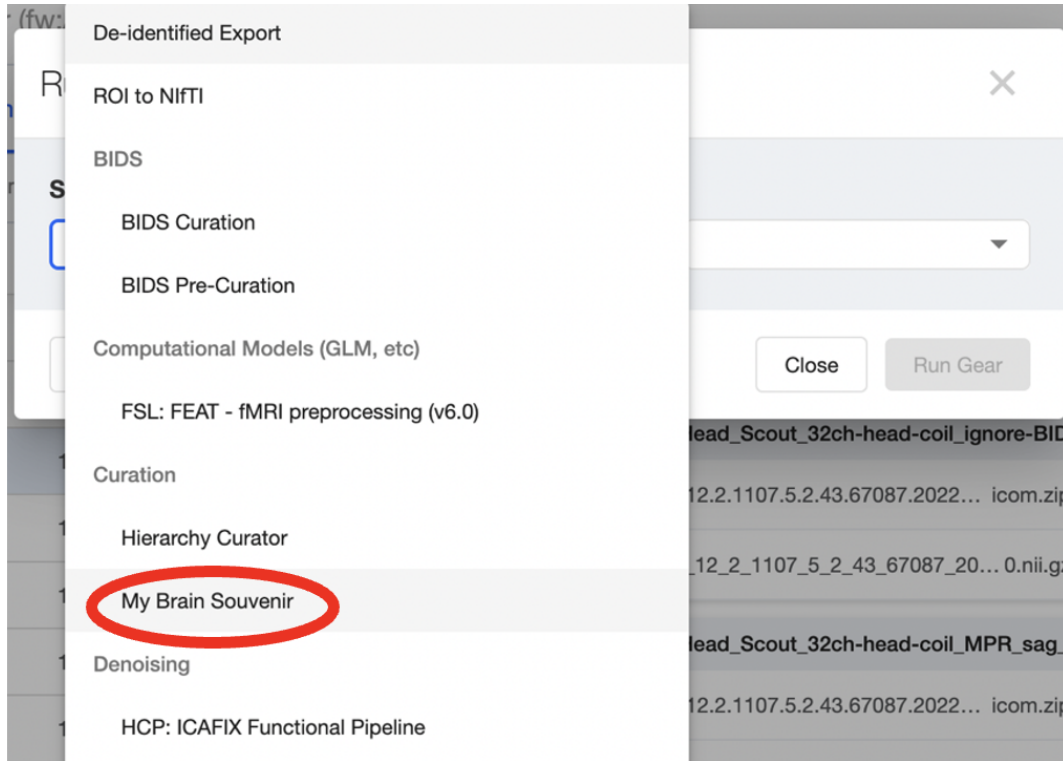
2. From the popup window, select “Analysis Gear”

The 'Select Gear Type' dialog box is shown. It has two options: 'Utility Gear' and 'Analysis Gear'. The 'Analysis Gear' option is highlighted with a red arrow. The 'Utility Gear' option is described as performing basic data conversion and QA tasks, with outputs saved to the input container. The 'Analysis Gear' option is described as performing advanced preprocessing and data analysis, with outputs saved as discrete analysis containers.

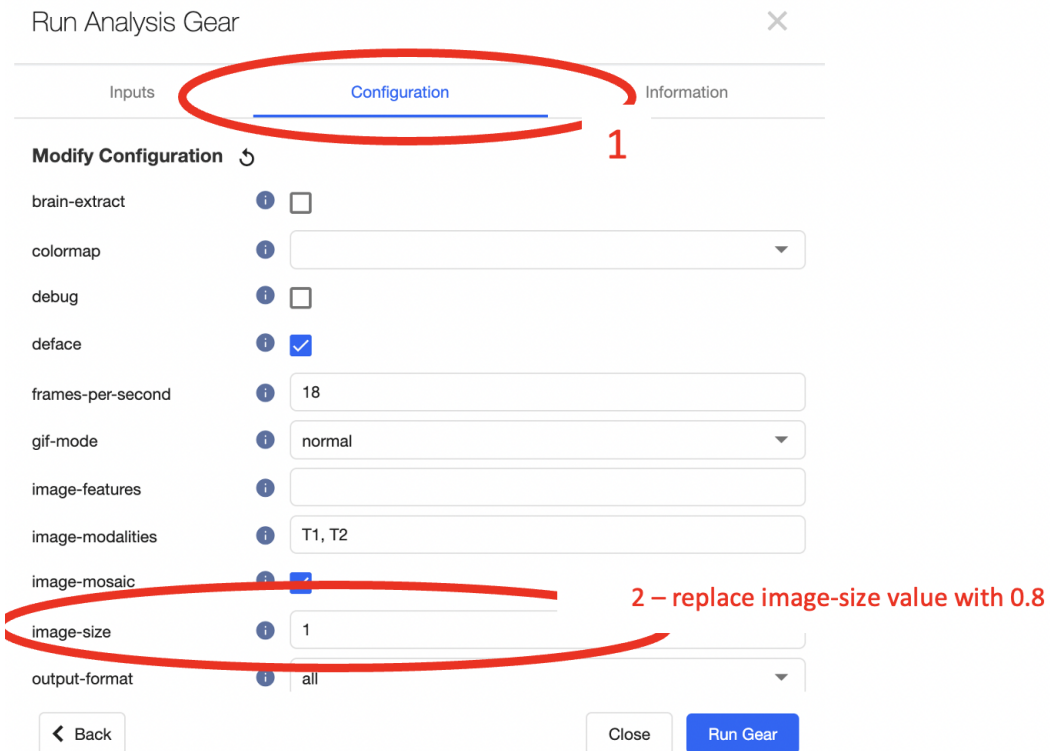
1.14.1 MY BRAIN SOUVENIR

My Brain Souvenir is a custom Flywheel Gear used to generate electronic souvenirs for study participants. Image formats (GIF | JPEG | MOV) can be specified and output images can be configured to store defaced, brain extracted, or native images. Each study should check with the IRB coordinator to confirm appropriate sharable image formats and venues (e.g. email, secure link, etc).

1. From the drop down list of available gears, select “Brain Souvenir”



2. This gear requires no inputs, and the default configuration is pretty good. I recommend changing the image size config setting (image size = 0.8), to ensure the output images are small enough to be sent as email attachments. From the “Run Analysis Gear” Menu, select the Configuration panel.

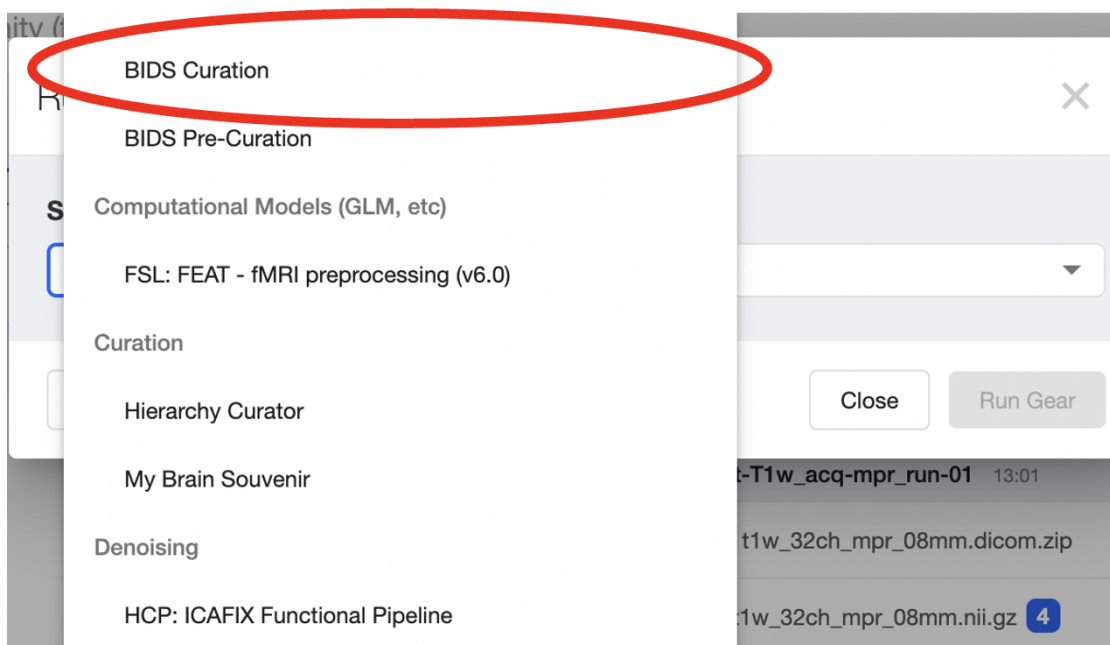


Filename	Conte
<anat-acq>.gif	Native Anatomical Scan (GIF Format)
<anat-acq>.jpg	Native, single midplane slice (JPEG Format)
<anat-acq>.mov	Native Anatomical Scan (MOV Format)
<anat-acq>_defaced.gif	Defaced Anatomical Scan (GIF Format)
<anat-acq>_defaced.jpg	Defaced, single midplane slice (JPEG Format)
<anat-acq>_defaced.mov	Defaced Anatomical Scan (MOV Format)

1.14.2 BIDS CURATE

BIDS Curate gear should be run for all neuroimaging MRI sessions. Before running (or re-running) bids curate, all acquisitions should be in reproin standard naming format. If you are unsure if your data can be BIDS curated, contact INC staff.

1. From the popup window, select “Analysis Gear” and scroll through the available gears to “BIDS Curation”



2. Add the appropriate inputs and configurations for the gear as shown below.

Inputs Configuration Information

⚠ This gear will act on your behalf with a dedicated API key.

Select Inputs

template

Analysis Label

Job Tags

[< Back](#) [Close](#) [Run Gear](#)

If study has a "reproin_template.json" file, add it here.

In the configuration tab, you may need to add following options:

(if re-running bids-curate) Reset : True

(if diffusion tensor in study) intendedfor_regex : `.*fmap.* nii`

Configuration Inputs Information

Modify Configuration ↺

base_template

intendedfor_regexes

reset ☒

use_or_save_config

verbosity

[< Back](#) [Close](#) [Run Gear](#)

If study includes dti, add the regular expression here

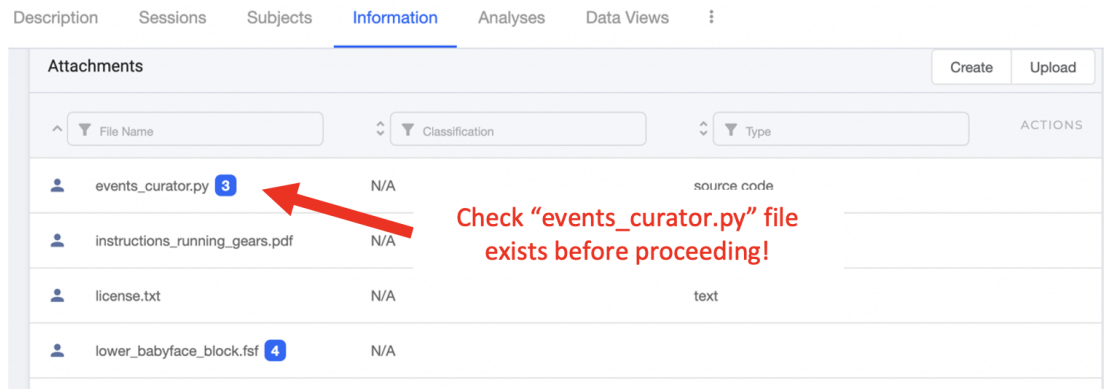
Check the box to reset any incorrect BIDS naming

- That's it! Select the "Run Gear" button. Once the gear starts you should see the analysis in the list of session analyses. The icon next to the analysis will change from a spinning gear to a green check once it completes successfully. Check out the outputs of the gear to confirm BIDS naming convention was correctly applied.

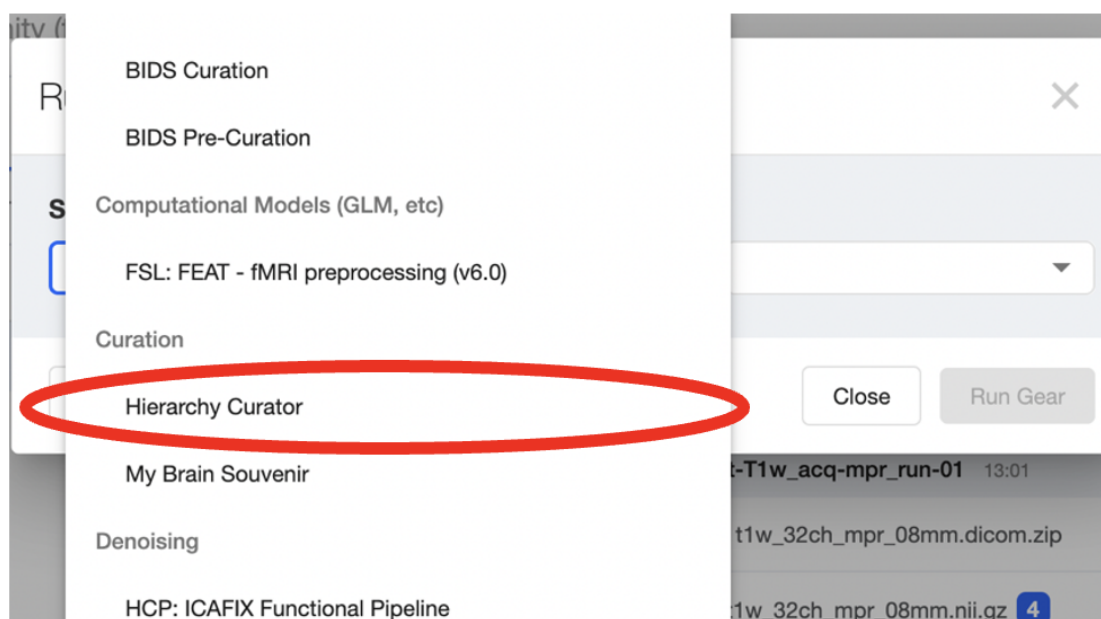
1.14.3 EVENT CURATOR

The event curator is a derivative analysis using the [hierarchy curator](#). A custom python script should be written to handle event creation from either an eprime or PsychoPy stimulus file. After uploading the recordings (see ["Uploading Event Files"](#)), this gear is used to generate derivative event data from the stimulus recordings. For help generating a python script for use with the hierarchy curator, contact INC staff.

- After uploading the raw timing files for all applicable acquisitions, you can generate event data using the "Hierarchy Curator" gear. You must have supporting curator code uploaded to your project. If you do not have an "events_curator.py" file in your project contact the INC Data and Analysis Team for assistance!



- Return to the session of interest and select "Run Gear". From the drop down menu, select "Hierarchy Curator"



- In the Gear "Inputs" select the "events_curator.py" file as the curator.

Run Analysis Gear

Hierarchy Curator 2.1.4_inc0.1

Curates a container in the flywheel hierarchy given a python HierarchyCurator class. Using an implementation of the HierarchyCurator Class (provided as an input file (e.g., curator.py)) this gear is able to curate an entire project, walking down the hierarchy through project, subject, session, acquisition, analysis, and file containers.

Inputs Configuration Information

⚠ This gear will act on your behalf with a dedicated API key.

Select Inputs

additional-input-one No input chosen (Optional)

additional-input-three No input chosen (Optional)

additional-input-two No input chosen (Optional)

curator events_curator.py

Analysis Label hierarchy-curator 02/13/2023 12:16:57

⏪ Back Close Run Gear

Select the project level “events_curator.py” file as the curator input

4. That's it! After the gear finishes, check that new event files have appeared in the sessions acquisitions named "...events.tsv"

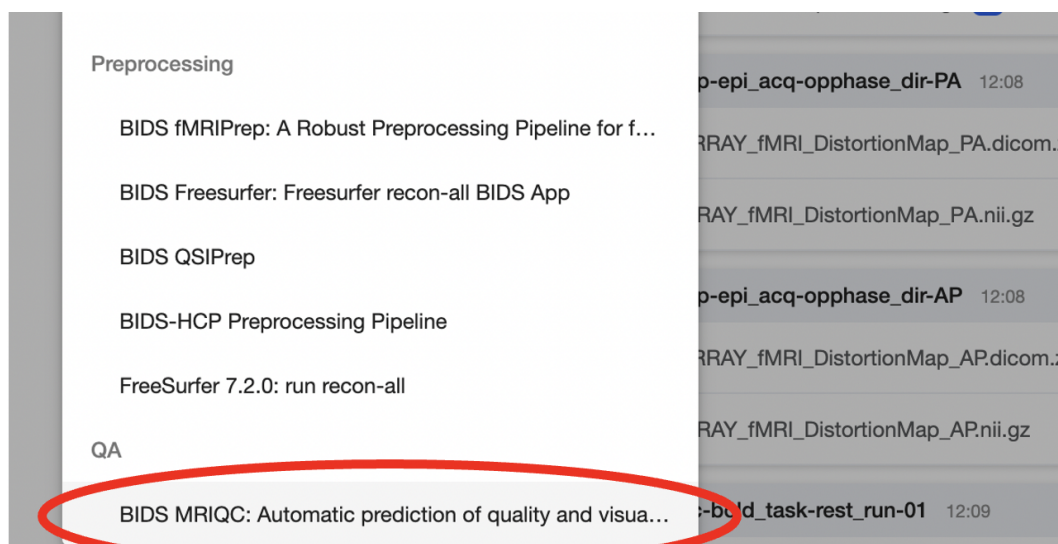
1.14.4 BIDS-MRIQC

`bids-mriqc` is a Flywheel wrapper for `mriqc`. MRIQC is used to extract image quality metrics (IQMs) from structural and functional magnetic resonance imaging data. Please cite the use of MRIQC as follows:

Esteban O, Birman D, Schaer M, Koyejo OO, Poldrack RA, Gorgolewski KJ; MRIQC: Advancing the Automatic Prediction of Image Quality in MRI from Unseen Sites; PLOS ONE 12(9):e0184661; doi:10.1371/journal.pone.0184661.

MRIQC is automatically run for all new sessions on INC Flywheel. If you need to change configuration settings, or re-run a failed job follow the instructions below.

1. From the drop down list of available gears, select “BIDS MRIQC”



2. IMPORTANT!! This gear is HPC Compatible. Please always launch this gear on hpc to keep other compute resources free. To run the job on our HPC clusters, add a “Job Tag”: hpc

Run Analysis Gear

BIDS MRIQC: Automatic prediction of quality and visu... 1.2.2_22.0.6_inc0.7

MRIQC (0.15.2 - April 6, 2020) extracts no-reference image quality metrics (IQMs) from T1w and T2w structural and functional magnetic resonance imaging data. Note: arguments --n_procs --mem_gb and --ants-nthreads are not available to configure because they are set to use the maximum available as detected by MRIQC.

Inputs Configuration Information

⚠ This gear will act on your behalf with a dedicated API key.

Select Inputs

bidsignore No input chosen (Optional)

Analysis Label bids-mriqc 12/08/2022 13:28:46

Job Tags

hpc Add tags

Back Close Run Gear

add "hpc"

If study has a "bidsignore" file, add it here.

3. If your project has a .bidsignore file stored in the project information, add this file as the optional gear input.
4. Next, move to the Configuration panel. You will see configuration options for mriqc performance as well as CU Boulder’s HPC slurm scheduler. In general, these configuration settings can be set to default values. You can always find out more information about each configuration setting by hovering the mouse over the information circle next to each setting.

Run Analysis Gear Configuration

session-id

slurm-cpu 1

slurm-nodes 1

slurm-ntasks 1

slurm-partition blanca-ics

slurm-qos blanca-ics

slurm-ram 12G

slurm-time 1428

start-idx 0

stop-idx 0

task-id

verbose v

Back Close Run Gear

Alpine Settings:

- Account: ucb-general
- Partition: amilan
- qos: normal

Blanca Settings:

- Account: blanca-ics-<study>
- Partition: blanca-ics
- qos: blanca-ics

- Set the HPC slurm scheduler settings as needed. By default the gear will run on blanca-ics with appropriate memory, RAM and wall time.
- That's it! Select the "Run Gear" button. Once the gear starts you should see the analysis in the list of session analyses. The icon next to the analysis will change from a spinning gear to a green check once it completes successfully.

1.15 Uploading Event Files

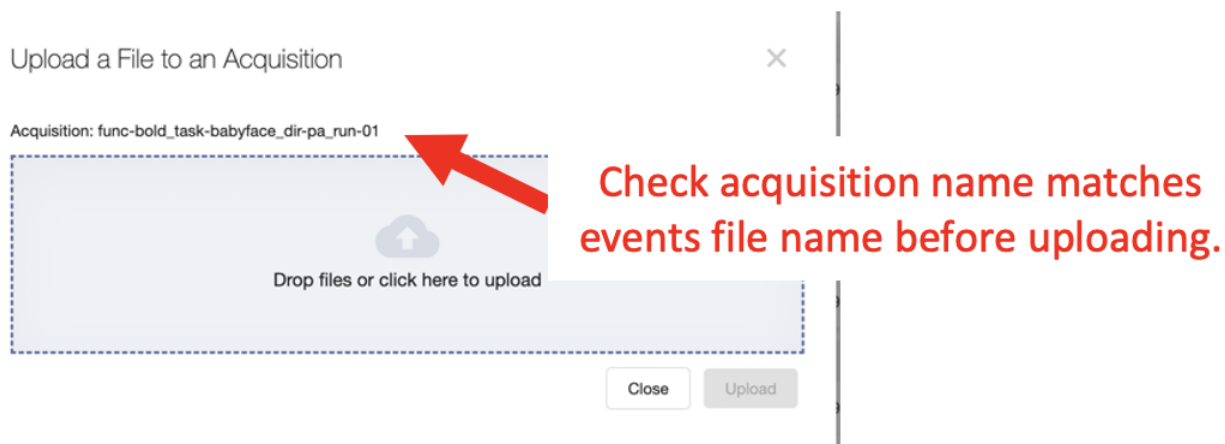
UPLOAD RAW EVENT TIMING DATA (PYSCHOPY or EPRIME)

- From the stimulus computer, use the experimentation software (e.g. psychopy or eprime) to export event data in a spreadsheet format (e.g. *.csv, *.tsv).
- You will need to upload the event file to Flywheel, to do so: navigate to your project, and go the sessions panel. Select the session of interest.
- In the acquisition view, find the correct acquisition to match the event data. For example, an acquisition "func-bold_task-babyface_dir-pa_run-01"
- BEFORE uploading the event data, the file should be renamed to the correct naming convention. All event data must follow this STRICT BIDS naming convention: <acquisition>_recording-[psychopy/eprime]_stim.tsv
 file_from_stim_computer.tsv -> func-bold_task-[task]_dir-[dir]_run-[index]_recording-eprime_stim.tsv
- After renaming the raw data file, upload the file to the Flywheel acquisition of interest.

1. Locate the acquisition which was collected with matching event data files

2. Select "Upload Data to Acquisition" for the matching scanner acquisition

- Locate the file for upload. BE SURE the file is named using the correct filename convention (STEP 4). Select "Upload".



7. Repeat this process for all acquisitions with task related events files.

1.16 The Basics

Don't have a research computing account yet? Please request an account [here](#).

1.16.1 What is Petalibrary?

PetaLibrary is a location where data is stored (similar to an external hard drive) and is regularly backed up to ensure data integrity. More accurately, PetaLibrary is a Research Computing managed storage cluster where scanner data and derivative data (i.e. analysis data) live for various projects. Petalibrary is managed using allocations, which are predefined chunks of space allotted for storage to a given lab or group (eg: banich lab, kaiser lab, etc).

All of INC's scanner data and **Flywheel** database data goes into an allocation titled ics. May labs also choose to purchase storage on a PI-sepcific allocation (eg: the banich allocation). We advise you restrirt the storage on CURC's Petalibrary to data you actively use on the high performance compute cluster. Archivable data should be moved to a different "cold storage" system, so you can retain these data at a lower cost.

It is therefore very important to not treat PetaLibrary like you may treat your personal laptop. We are on shared space: adding to one directory on the ICS allocation will decrease the total amount of free space available to everyone else.

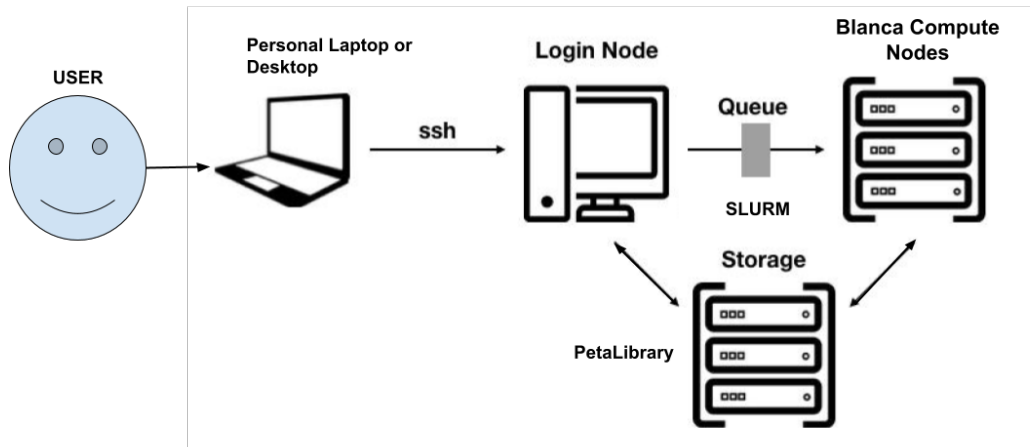
1.16.2 What is Blanca?

Blanca is one of three high performance compute clusters managed by Research Computing ([alpine](#) and [summit](#) are the other HPC clusters) . Most of us are familiar with hard drives, but perhaps less familiar with a computer's processor. While the hard drive is what stores large amounts of data, the central processing unit (or CPU) is what acts on the data (fetches it, decodes it, performs analysis, etc). You may think of a CPU as the thing in the computer that does all the thinking!

In simplified terms, a compute infrastructure is nothing more than a bunch of these CPUs and the networking structure that supports them.

As INC, we have 12 nodes (essentially 12 dedicated "computers") on which we can run analyses, collectively known as the blanca-ics cluster. Each node has between 28 and 56 CPUs. The reason we put "computers" in quotation marks is because these nodes don't have hard drives, unlike computers. In the analogy above, think of these nodes as the processors, the thinkers, but they can't actually store much information - storage is delegated to PetaLibrary.

INC also has 2 dedicated Blanca login nodes onto which PetaLibrary is mounted/accessible, and through which we can launch compute jobs onto our 12 compute nodes. The image below illustrates how a user interacts with our Blanca nodes.



1.17 High Performance Compute Portal

CU Boulder Research Computing (CURC) supports a browser based HPC Portal, **Open OnDemand** [here](#). This browser based portal allows users to navigate and interact with CURC's HPC from anywhere with an internet connection! It's fast and easy one stop shop for your HPC needs.

Please visit CURC documentation on [Open OnDemand](#) for a step by step guide.

In the document below we will talk through the methods you need to access the HPC environment and get started with your own neuroimaging analyses. First, let's consider your "use case." How will you be using HPC? Once you have that answer in mind, let us work through the following questions.

1.17.1 Do you need a Desktop?

In your *use case* will you need to interact with a graphical user interface (GUI)? Some examples where you may need a desktop include using [CONN: Functional Connectivity Toolbox](#), viewing spreadsheets, or html reports, viewing data or images such as nifti images or surface meshes. Most other cases (such as moving/copying/deleting files, running bash or python code, etc.) can be achieved without a User Desktop.

If you answer “Yes”

You will use Open OnDemand to start a new “Core Desktop” Interactive session. Follow the instructions in [Using Core Desktop](#) to get started.

If you answer “No”

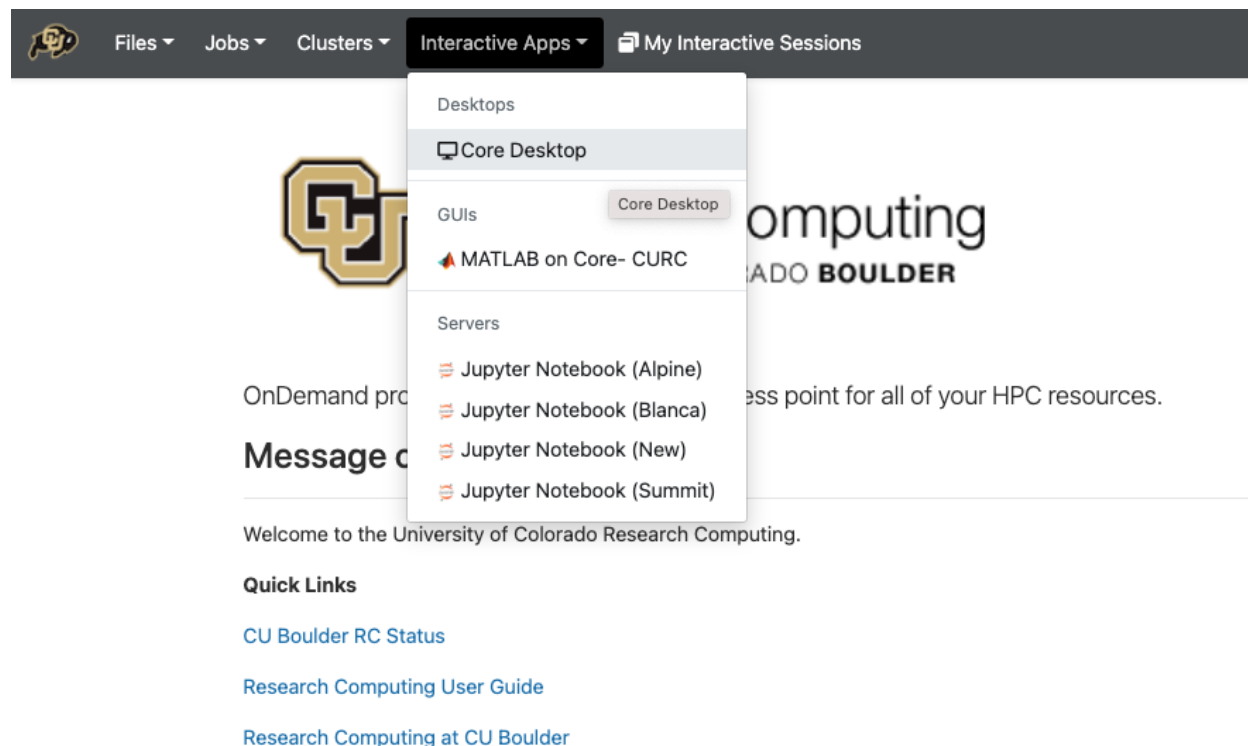
Without a desktop, we recommend you use an interactive “Jupyter Notebook” session. Follow the instructions in [Using Jupyter Notebook \(New\)](#) to get started.

Using Core Desktop


Core Desktop is used to open a “Desktop” view into the HPC. The desktop is run on CURC’s **viz nodes** which are special GPU accelerated login nodes designed to support running desktop viewers for multiple users.

To get started, go to [open ondemand](#).

1. From the dropdown list of “Interactive Sessions”, Select **Core Desktop**



2. Set the compute options for your **Core Desktop** session.

 Files ▾ Jobs ▾ Clusters ▾ Interactive Apps ▾ My Interactive Sessions

Home / My Interactive Sessions / Core Desktop

Interactive Apps

Desktops

Core Desktop

GUIs

MATLAB on Core- CURC

Servers

Jupyter Notebook (Alpine)

Jupyter Notebook (Blanca)

Jupyter Notebook (New)

Jupyter Notebook (Summit)

Core Desktop

This app will launch an interactive desktop on one or more compute nodes. You will have full access to the resources these nodes provide. This is analogous to an interactive batch job.

Account

You can leave this blank if **not** in multiple projects.

Number of hours

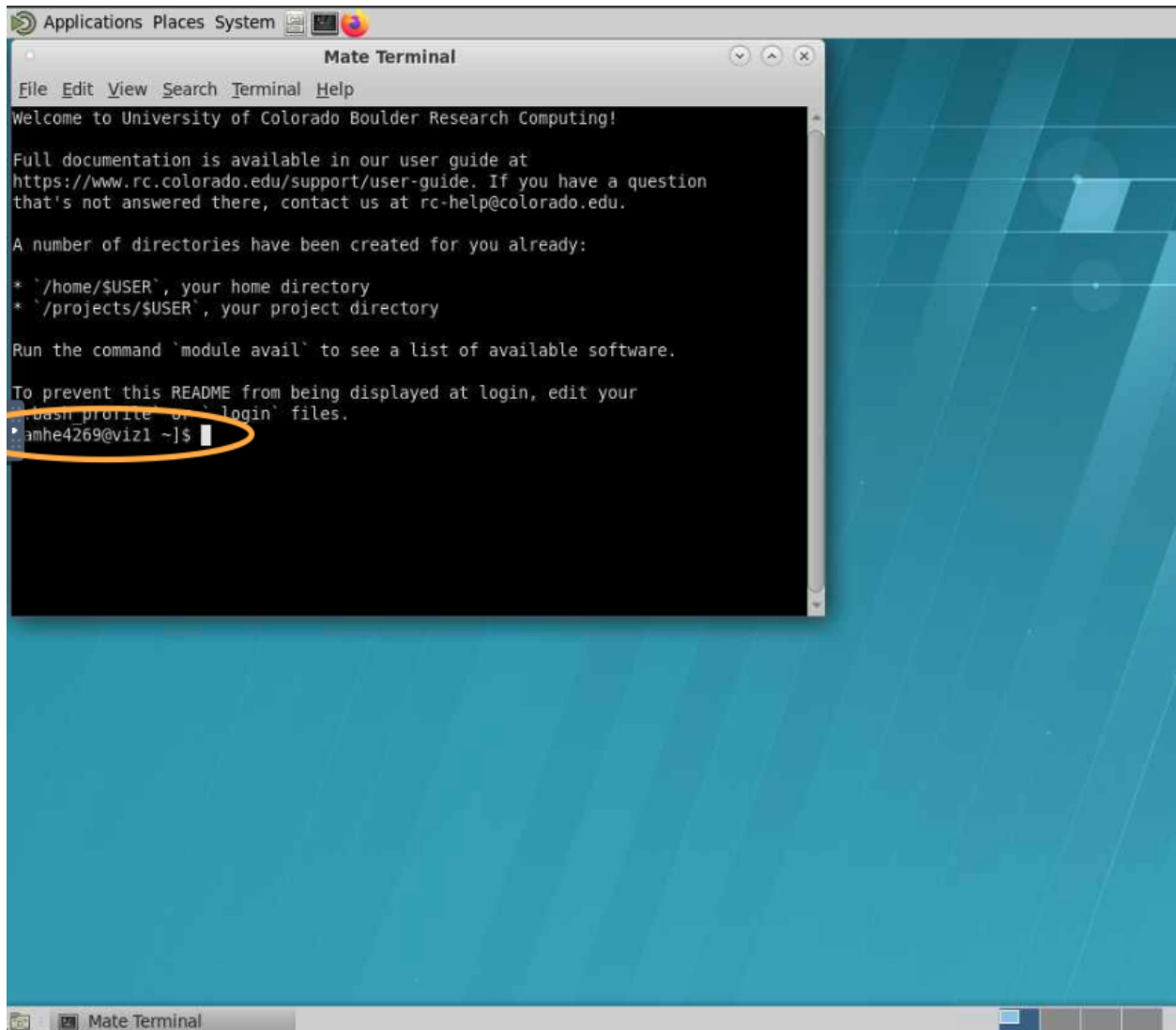
Number of cores

Number of cores on node.

Launch

* The Core Desktop session data for this session can be accessed under the [data root directory](#).

3. Launch the new Desktop. Here you can start a terminal, or another application.



Note: the Desktop is running on a **viz node**. To access filesystems and compute nodes exclusive to Blanca or alpine, will need to load the slurm module.

Alpine (and Blanca) Scratch Filesystem

Research computing supports a large gpfs [scratch filesystem](#). Scratch space should be used for all compute jobs run on Alpine or Blanca. These high-performance scratch directories are not backed up, and are not appropriate for long-term storage. Data may be purged at any time subject to overall system needs. Files are automatically removed 90 days after their initial creation. Once an analysis is complete, please move all data files to Flywheel or a Petalibrary Allocation for permanent storage.

Alpine Scratch

/scratch/alpine/<identikey>/

Blanca Scratch

/scratch/blanca/<identikey>/

Important

Alpine Scratch is mounted to all Blanca compute nodes using a sshfs mount at the file path `/scratch/blanca/`. This means you can access the same scratch files from both alpine and blanca computing using these two filepaths.

Alpine scratch may also be viewed from the **vis nodes** on open ondemand.

Alpine (and Blanca) Compute

To run compute jobs on Alpine and Blanca compute nodes, we use a SLURM job scheduler. From **vis nodes**, simply load the correct slurm module to get started.

```
# to launch alpine compute jobs
module use /curc/sw/modules/slurm
module load slurm/alpine

# to launch blanca compute jobs
module use /curc/sw/modules/slurm
module load slurm/blanca
```

Some filesystems and software should only be used on compute nodes (the “workhorses” of HCP). You may start an interactive session in order to access these resources using Slurm’s interactive session.

```
# to start an interactive session (blanca)
sinteractive --partition=blanca-ics --qos=blanca-ics -c <number of cores> --mem <memory_
↪ 1K/1M/1G> -t <time> --export=NONE

# to start an interactive session (alpine)
sinteractive --partition=amilan -c <number of cores> --mem <memory 1K/1M/1G> -t <time> --
↪ export=NONE
```

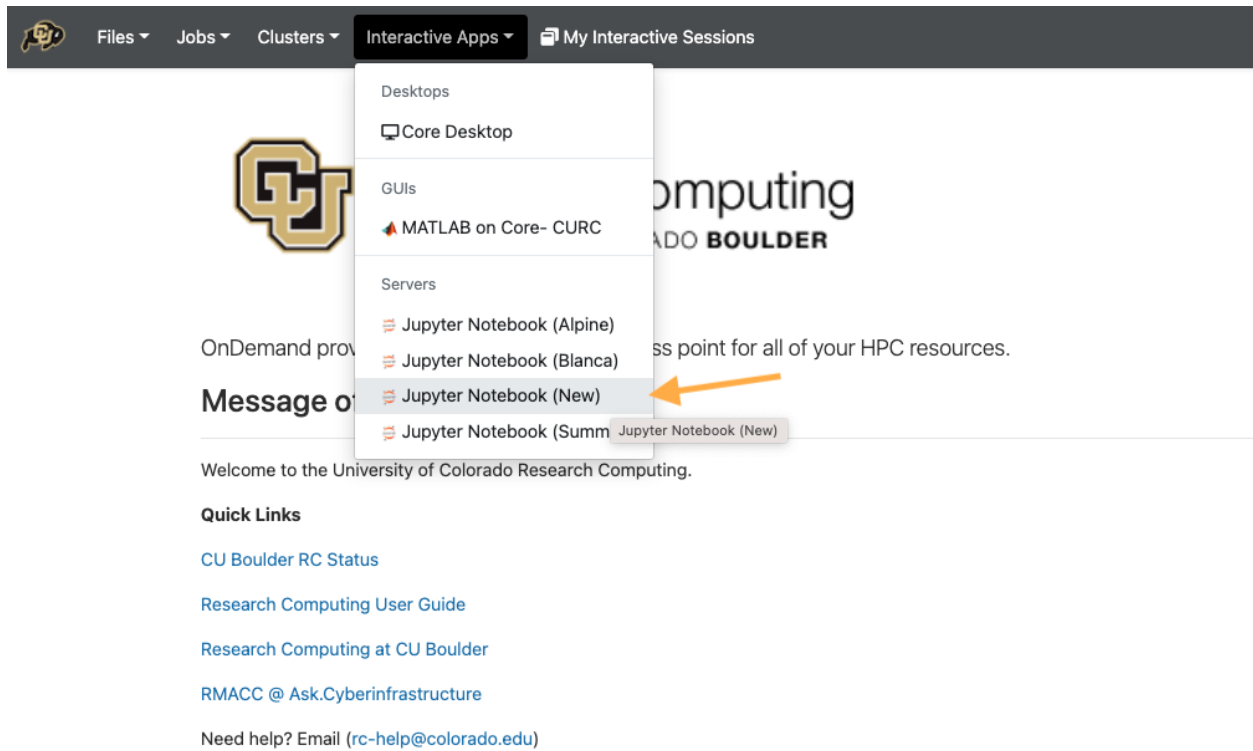
Looking for more information on using Slurm scheduler, blanca priority accounts, and more check out <placeholder>

Using Jupyter Notebook (New!)

In most cases, actions you need to take on the HPC can be done using a terminal or within python using a Jupyter Notebook or Console. Want to learn more about Jupyter Notebooks? read up [here](#).

To get started, go to [open ondemand](#).

1. From the dropdown list of “Interactive Sessions”, Select **JupyterHub 2 (Custom)**



2. Set the compute options for your **Jupyter Notebook** session. Be careful to correctly set both the partition and QOS section, to match the desired cluster. Here are examples of correct partition/QOS settings:

Blanca Cluster

Partition: blanca-ics

QOS: blanca-ics

Alpine Cluster

Partition: amilan

QOS: <leave blank>

Home / My Interactive Sessions / Jupyter Notebook (New)

Interactive Apps

Desktops

Core Desktop

GUIs

MATLAB on Core- CURC

Servers

Jupyter Notebook (Alpine)

Jupyter Notebook (Blanca)


Jupyter Notebook (New)

Jupyter Notebook (Summit)

Jupyter Notebook (New) version: 2345bda

This app will launch a Jupyter Notebook server on one or more nodes.

Cluster


Blanca  Select blanca or alpine cluster ▾

Only CU Boulder Users can use Alpine at this time.


Account

You can leave this blank if not in multiple projects.


Partition

blanca-ics 

Number of hours


1 

Number of cores

1 

Number of cores on node. Note: If using the default shas-interactive queue, your job will not start if you request more than one core.

Qos

blanca-ics 

You can leave this to normal unless you need a custom qos.

☒ Use JupyterLab instead of Jupyter Notebook?

Enter correct partition and QOS, Example here shows entry for Blanca Cluster

Note: The new jupyter notebook session will be launched from a Blanca or Alpine compute node depending on your preference.

- You can use the **Terminal** feature to run any shell scripts, as well as the Python **Console** and Python **Jupyter Notebook** for any python based scripts.

1.18 Setting Up Conda Environments

Our data and analysis team at INC have generated several custom python environments to support the needs of our researchers. Anaconda makes it easy to build and interchange virtual environments for running your python code. If you are new to anaconda, please check out research computing documentation on the [subject](#).

To get started using the INC anaconda environments, please do the following:

- On reserach computing's HPC (summit, blanca, or alpine), create a text file `/home/$USER/.condarc` with the contents:

```
pkgs_dirs:
- /projects/$USER/.conda_pkgs
envs_dirs:
- /projects/$USER/software/anaconda/envs
- /projects/ics/software/anaconda/envs
```

2. Next, load the anaconda software using module then confirm you “see” all the shared conda environments:

```
[amhe4269@bnode0301 ~]$ module load anaconda
(base) [amhe4269@bnode0301 ~]$ conda env list
# conda environments:
#
base                * /curc/sw/anaconda3/2020.11
atocblm_22          /curc/sw/anaconda3/2020.11/envs/atocblm_22
globus              /curc/sw/anaconda3/2020.11/envs/globus
synoptic_f22       /curc/sw/anaconda3/2020.11/envs/synoptic_f22
datalad            /projects/ics/software/anaconda/envs/datalad
dcm2BIDS           /projects/ics/software/anaconda/envs/dcm2BIDS
flywheel           /projects/ics/software/anaconda/envs/flywheel
incenv             /projects/ics/software/anaconda/envs/incenv
jupyter            /projects/ics/software/anaconda/envs/jupyter
nda-tools          /projects/ics/software/anaconda/envs/nda-tools
pysurfer           /projects/ics/software/anaconda/envs/pysurfer
sahahn_neurotools  /projects/ics/software/anaconda/envs/sahahn_neurotools
```

Note: The output of *conda env list* may look slightly different from what is shown above, but you should see paths that point to `/projects/ics/software/anaconda/envs/`

3. Finally, you need to create a **kernel** to be able to access any of these environments in jupyterhub:

```
[amhe4269@bnode0301 ~]$ module load anaconda
(base) [amhe4269@bnode0301 ~]$ conda activate flywheel
(flywheel) [amhe4269@bnode0301 ~]$ python -m ipykernel install --user --name flywheel --
↳display-name flywheel
```

Note: You need to create a python kernel for *every* conda environment you want to be accessible through jupyterhub. This has to be run for each new user.

That’s it! You are ready to start developing!

Keep Reading to learn more about leveraging the large Scratch filesystem with Flywheel for local analyses. Go to [Get to Know Your Scratch Space](#)

1.19 Get to Know Your Scratch Space

We encourage all users to utilize the large scratch filesystem mounted on Blanca and Alpine. Scratch filesystems offer a fast, safe environment for quick development and troubleshooting.

Warning: The scratch filesystem purges files more than 90 days old! This is not a permanent file storage. Any files you wish to retain permanently should be moved to **Petalibrary** or uploaded to **Flywheel**.

Where to find Scratch for each Cluster.

Cluster	Scratch Filesystem	Size	Mounted Locations
Summit	/summit/scratch/ <user>/	10 TB	Mounted read-write on all SUMMIT nodes (login, compute)
Blanca	/scratch/blanca/ <user>/	15 TB	Mounted read-write on all BLANCA (compute only) nodes
Alpine	/scratch/alpine/ <user>/	15 TB	Mounted read-write on all ALPINE nodes (compile, compute)

1.19.1 Using my Scratch Space

If you are not sure how to log on to CU Boulder Research Computing resources, please go back to read about the [High Performance Compute Portal](#). Be sure to access the scratch filesystem from the appropriate compute or compile nodes (see breakdown above).

Leverage Scratch Space with Flywheel Storage

Some neuroimaging analyses and workflows are not yet supported as Flywheel gears. For these workflows, the current workaround is to download your Flywheel analyses (or workflow inputs) run the workflow on your local machine (or HPC), then upload the results (workflow outputs) back to flywheel as a new analysis.

Follow our example tutorial using the Flywheel SDK to [Download and Run Analysis Locally](#).

Uploading Scratch Analyses to Flywheel

Once you generate an analysis you wish to permanently store. Be sure to move the analysis to Petalibrary, or upload the analysis to Flywheel.

Follow our example tutorial using the Flywheel SDK to [Upload My Analysis to Flywheel](#).

1.20 Working With Your Data Outside Flywheel

Some workflows are not currently supported in Flywheel, and need to be implemented on a local compute environment. Please check out our python SDK tutorials [here](#) <notebooks/download-and-run-locally.ipynb> to learn more.

1.20.1 CONN Functional Connectivity Toolbox

CONN Functional Connectivity Toolbox. is an open-source Matlab/SPM-based cross-platform software for the computation, display, and analysis of functional connectivity Magnetic Resonance Imaging (fcMRI).

CONN Analyses can generate *very* large output directories and therefore special care must be taken whenever generating analyses in the platform to ensure responsible data stewardship. Below we have outlined our recommendation for best practices for developing CONN analyses as well as archiving these analyses after publication.

Setting Up Your Analysis

CONN analyses require extensive computational and disk size resources. To ensure responsible use of computational resources, it is best practice to:

1. Test your CONN analysis design with 3-5 subjects
2. Start simple, build out the complexity of the model design one step at a time
3. Once you are satisfied with the model design run the final full model design only once
4. All CONN *development* models should be stored on a scratch filesystem and removed when complete
5. When possible generate a single preprocessed dataset for all CONN analyses
6. Current CONN limitations limit single file import size < 1GB. For input file sizes greater than 1GB, downgrading the datatype can reduce nifti file sizes for conn import.

Archiving Analysis When Complete

Once a CONN analysis is completed, we recommend users archive their conn analysis. Removing intermediate data files can help manage the size of the archived analysis.

1. Intermediate preprocessing files can be removed. Before removing any preprocessing files, check which dataset files are loaded in CONN. This includes checking the file paths defined in the “Primary Dataset”, “Unsmoothed Dataset”, ...

Important: Do not remove any input files that are referenced in the “Setup” panel of CONN. Removing these files may result in irreversible errors in the conn menu.

2. Approximately 60%-80% of the data storage requirements of the CONN outputs directory are contained in the “results” subdirectory. These *.mat and *.matc files contain information generated from the denoising, first-level, and second-level modeling steps of the CONN analysis.

Users **can** remove the contents of “results/preprocessing” and “results/firstlevel” without disrupting ability to recover the second level model results already generated for the project. To generate new second level model results, the denoising and first-level modelling step would need to be regenerated.

Zippping Archived Outputs

Once users have removed all intermediate result files, the conn analysis can be archived by first zipping the conn outputs directory and mat file, then uploading the analysis to the flywheel platform. Here is an example below:

```
# zip outputs directory and mat file
!zip -R conn_analysis conn_analysis/ conn_analysis.mat

# zip preprocessed inputs **only if not already stored on platform
!zip -R conn_inputs conn_inputs/

# upload zips to flywheel
fw = flywheel.Client()

project = fw.lookup('<group>/<project>')

# create new project level analysis
analysis = project.add_analysis(label='CONN Analysis: ' + datetime.now("%x %X"))

# upload output zipped directories
analysis.upload_output('conn_analysis.zip')
```

1.21 Resources

1.21.1 Flywheel Documentation Resources

- **Main Flywheel Documentation Page**
 - Good starting place for high level overview
- **Flywheel Webinars**
 - Webinars that Flywheel has hosted on various topics
- **Flywheel Gear Exchange**
 - Different analysis gears that have been developed by the Flywheel team that can be installed on our INC instance
- **Flywheel Python SDK ReadTheDocs Full Documentation**
 - Documentation on the Python Software Development Kit (SDK) that enables the user to write Python code that can be interpreted by Flywheel
- **Flywheel Python SDK Quickstart**
 - The Quickstart is all the useful SDK commands in one place
- **Flywheel CLI Documentation**
 - A guide to using the Flywheel Command Line Interface (CLI) tool for everything from data ingestion to running gears
- **Flywheel CLI Beta Version Documentation**
 - A beta (second) version of the CLI that included additional useful tools

1.21.2 Flywheel Code Resources

- **GitLab of Flywheel Tutorials**
 - Code (mostly Jupyter Notebook) tutorials covering CLI, Python, and different code Webinars for the Flywheel SDK
- **GitLab of Flywheel Gears**
 - The code behind the Flywheel gears. This code can be downloaded, modified, and rebuilt for the purpose of customizing gears
- **GitHub of Flywheel Gears**
 - Some Flywheel gears are still stored on GitHub versus GitLab
- **Templates for building out Flywheel Gears**
 - Particularly useful is the bids-app-template if creating gears from scratch
- **Docker Hub for Flywheel Images**
 - All Flywheel gears run as either docker or singularity containers. This dockerhub allows you to see (and download) the docker images for those gears directly (without having to build them locally).

1.21.3 INC Flywheel-related Resources

- **INC Github**
 - The INC Github page contains INC-modified Flywheel Gears as well as other singularity containers and analysis code that INC staff have built

1.22 FAQs

1.23 Flywheel Software Developer Kit (SDK) Basics and Helpful Hints

Date modified: 28-July-2022 **Authors:** Amy Hegarty, Lena Sherbakov, Intermountain Neuroimaging Consortium

Description: The following includes a set of basic SDK commands for Flywheel. Examples included below:

- Basic Flywheel Hierarchy (finding a session)
- Pulling Data To Scratch workspace
- Uploading Analyses to Flywheel

Links

- Flywheel SDK documentation [here](#)
- Flywheel python SDK examples [here](#)

```
[3]: print("Welcome to Intermountain Neuroimaging Consortium!")
```

```
Welcome to Intermountain Neuroimaging Consortium!
```

1.23.1 Import Python Dependencies

```
[ ]: from pathlib import Path
import sys, subprocess, os, nipype, datetime, logging
sys.path.append('/projects/ics/software/flywheel-python/bids-client/')
sys.path.append('/projects/ics/software/flywheel-python/')
from getpass import getpass
import flywheel, flywheel_gear_toolkit
from flywheel_gear_toolkit.interfaces.command_line import (
    build_command_list,
    exec_command,
)
from flywheel_gear_toolkit.utils.zip_tools import unzip_archive, zip_output
from utils.zip_htmls import zip_htmls
from flywheel_bids.export_bids import export_bids
from flywheel_bids.export_bids import download_bids_dir
import subprocess
from datetime import datetime
from zipfile import ZipFile
```

1.23.2 Define Supporting Functions

```
[ ]: # supporting functions
def analysis_exists(session, analysis_name, exclude_list=[]):
    # Returns True if analysis already exists with a running or complete status, else
    ↪ false
    # make sure to pass full session object (use fw.get_session(session.id))
    #
    #Get all analyses for the session
    flag=False
    for analysis in session.analyses:
        #only print ones that match the analysis label
        if analysis_name in analysis.label:
            #filter for only successful job
            analysis_job=analysis.job
            if any(analysis_job.state in string for string in ["complete","running",
            ↪ "pending"]):
                if analysis_job.failure_reason is None:
                    flag=True
            #check if session is in exclude list
            if any(session.id in string for string in exclude_list):
                flag=True

    return flag
```

1.23.3 Flywheel API Key and Client

An API Key is required to interact with the datasets on flywheel db. More on this in the Flywheel SDK doc [here](#). Here we pull the API key directly from the user's os environment, set within the flywheel command line interface.

```
[ ]: # Instantiate a logger
logging.basicConfig(level=logging.INFO, format='%(asctime)s %(levelname)s %(message)s')
log = logging.getLogger('root')

[ ]: # Create client, using CLI credentials
fw = flywheel.Client()

# who am I logged in as?
log.info('You are now logged in as %s to %s', fw.get_current_user()['email'], fw.get_
↪config()['site']['api_url'])
```

1.23.4 Viewing a Flywheel Project

The `fw.lookup` function can be used to find any container in flywheel by path. This is an easy way to find projects, subjects, or sessions. Lets view a Flywheel project and list all sessions and their scan date.

```
[ ]: project = fw.lookup('<group>/<project>')
project.reload()
for session in project.sessions.find():
    # Loop through all sessions in the subject container.
    dt = session.timestamp
    if not dt:
        dt = "NAN"
    else:
        dt = dt.strftime("%x %X")
    print('%s: Subject: %s Session: %s\tScanning Date: %s' % (session.id, session.
↪subject.label, session.label, dt))
```

1.23.5 Viewing a Flywheel Analysis

Similarly, we can view analyses that have been either uploaded to Flywheel or generated by running flywheel gears. Here, lets look for all analyses attached to a single session.

```
[ ]: # start by finding a session of interest
session = fw.lookup('<group>/<project>/<subject>/<session>')

# we need to take an extra step to get the full session object, not just basic info
session_object = fw.get_session(session.id)

# print all the analyses for this session
for analysis in session_object.analyses:
    if hasattr(analysis.job, 'state'):
        print('%s: %s %s' % (analysis.id, analysis.label, analysis.job.state))
    else:
        print('%s: %s' % (analysis.id, analysis.label))
```


1.23.6 Download Data From Flywheel to Local Scratch (Part 1, Analyses)

Often times you need to use data in flywheel as inputs for group level analyses. At present this is still most easily accomplished by exporting the data from file to a scratch filesystem.

```
[ ]: # path to scratch directory
username=os.getenv('USER')
scratch='/scratch/summit/'+username+'/'
os.chdir(scratch)

# find the analysis you wish to download
analysis_id='629f8f5cbffdbee5eb7dcb37'
analysis=fw.get(id=analysis_id)

# Download the data to scratch
for fl in analysis.files:
    fl.download(scratch+fl['name'])

# unzip files
if '.zip' in fl['name']:
    zipfile = ZipFile(scratch+fl['name'], "r")
    zipfile.extractall(scratch)

# Run your computations! Here lets just list the contents of the directory and store
↳ that in a text file
os.system('mkdir -p output ; ls -l '+scratch+' > output/out.log')
```

1.23.7 Upload Results to Flywheel Analysis (Part 1, Session Analyses)

```
[ ]: # Once you have finished your analysis, we need to upload your files and store some
↳ critical metadata in flywheel

# session add analysis, zip outputs, upload
# ...#zip output except for scratch
log.info('Zipping contents of directory %s', scratch+'output')
zip_output(scratch, 'output', "output.zip", exclude_files=['scratch'])

#create an analysis for that session
timestamp=os.path.getmtime(scratch+'output')
dtobject = datetime.fromtimestamp(timestamp)
analysis = session_object.add_analysis(label='Sample Upload '+dtobject.strftime(" %x %X
↳ "))

#upload analysis files
analysis.upload_output(scratch+'output.zip')
```

1.23.8 Upload Results to Flywheel Analysis (Part 2, Project Analyses)

```
[ ]: # same thing for project

#create an analysis for that project
timestamp=os.path.getmtime(scratch+'output')
dtobject = datetime.fromtimestamp(timestamp)
analysis = project.add_analysis(label='Sample Upload '+dtobject.strftime(" %x %X"))

#upload analysis files
analysis.upload_output(scratch+'output.zip')
```

1.23.9 Some Examples Looping Through All Sessions

Here is some example code where we loop through all sessions to check if an analysis exists. We will use a function we defined at the top of the notebook.

```
[ ]: for session in project.sessions.find():
    full_session = fw.get_session(session.id)
    flag = analysis_exists(full_session, "hcp")
    print('%s: %s %s' % (session.subject.label, session.label, flag))
```

Thats all Folks!

1.24 Download and Run Analysis Locally

Date: 26-Sept-2022

Description:

- This notebook provides a walk through to download analyses stored in Flywheel to a local file system. Some neuroimaging analyses and workflows are not yet supported as Flywheel gears. For these workflows, the current workaround is to download your Flywheel analyses (or workflow inputs) run the workflow on your local machine, then upload the results (workflow outputs) back to flywheel as a new analysis.
- In this example, we will be downloading data into a CURC scratch filesystem and run a group CONN analysis on our high performance compute.
- It should be possible to run this notebook in any jupyter-compatible third party-platforms such as [google collab](#) or [mybinder.org](#).

1.24.1 Requirements

- University of Colorado at Boulder Research Computing (CURC) account
- Access to University of Colorado Flywheel Instance

The following workbook should be run on CURC Blanca Compute. If you are unsure you have the correct permission or access to these resources please contact INC Data and Analysis team: Amy Hegarty [amy.hegarty@colorado.edu] or Lena Sherbakov [lena.sherbakov@colorado.edu].

CURC Jupyterhub

Before launching this jupyter notebook, users should launch this session using Open OnDemand.

TIP: Follow the instructions on INC Documentation to get started with Jupyter Notebooks.

We will be working on a large scratch system mounted only on Blanca compute nodes in this tutorial. If you do not have access to this filesystem you should select a similar large capacity scratch environment for analysis.

1.24.2 Setup

TIP: Please use the “flywheel” kernel for this tutorial. If you do not see a “flywheel” kernel, contact INC Data and Analysis team to install this environment.

```
[2]: print("Welcome to Intermountain Neuroimaging Consortium!")
```

```
Welcome to Intermountain Neuroimaging Consortium!
```

```
[ ]: # Python standard package come first
import logging
import os, platform, sys
from zipfile import ZipFile

# Third party packages come second
import flywheel

# add software paths
sys.path.append('/projects/ics/software/flywheel-python/bids-client/')
sys.path.append('/projects/ics/software/flywheel-python/')
```

Lets initialize a logger to keep track of the progress of our job (e.g. useful to keep track of runtime).

```
[ ]: # Instantiate a logger
logging.basicConfig(level=logging.INFO)
log = logging.getLogger('root')
```

Lets check we are on the correct computing system.

```
[ ]: host = os.getenv('HOSTNAME', os.getenv('COMPUTERNAME', platform.node())).split('.')[0]

if "bnode" not in host:
    log.error("Tutorial should be run on CURC high performance compute nodes: blanca")
```

1.24.3 Flywheel API Key and Client

You can get your API_KEY by following the steps described in the Flywheel SDK doc [here](#).

DANGER: Do NOT share your API key with anyone for any reason - it is the same as sharing your password and may break human subject participant confidentiality. ALWAYS obscure credentials from your code, especially when sharing with others/committing to a shared repository.

```
[ ]: API_KEY = getpass('Enter API_KEY here: ')
```

Instantiate the Flywheel API client either using the API_KEY provided by the user input above or by reading it from the environment variable FW_KEY.

```
[ ]: fw = flywheel.Client(API_KEY if 'API_KEY' in locals() else os.environ.get('FW_KEY'))
```

You can check which Flywheel instance you have been authenticated against with the following:

```
[ ]: log.info('You are now logged in as %s to %s', fw.get_current_user()['email'], fw.get_
↪config()['site']['api_url'])
```

1.24.4 Constants

Often you will have to define a few constants in your notebook which serve as the inputs. Such constant for instance is the API_KEY that was used to instantiate the Flywheel client. Other examples could be a PROJECT_ID or PROJECT_LABEL that will be used to identify a specific project.

```
[ ]: PROJECT_LABEL = 'MyProject'
```

1.24.5 Helper functions

Here are all the custom helper functions we have developed for use in this example.

```
[ ]: def get_project_id(fw, project_label):
    """Return the first project ID matching project_label

    Args:
        fw (flywheel.Client): A flywheel client
        project_label (str): A Project label

    Returns:
        (str): Project ID or None if no project found
    """
    project = fw.projects.find_first(f'label={project_label}')
    if project:
        return project.id
    else:
        return None
```

1.24.6 Main script

We will be using the Flywheel SDK to identify and retrieve specific analysis files stored in Flywheel for download. Importantly, since the original analysis files are still retained in Flywheel we can use our local copy of the data as a temporary or scratch workspace and remove all files at the end of this workflow.

First, let's point to a project in Flywheel.

```
[ ]: project_id = get_project_id(fw, PROJECT_LABEL)
    if project_id:
        print(f'Project ID is: {project_id}.')
    else:
        print(f'No Project with label {PROJECT_LABEL} found.')
```

Let's start by getting some information about the analyses in our flywheel project. We will loop through all the sessions in our desired project, and log the number of complete, failed, and cancelled jobs. This structure will be the same that we will use when downloading the list of analyses next.

```
[ ]: # get project object
project = fw.get_project(project_id)

gear_name = 'bids-fmriprep'

icomplete = 0
ifailed = 0
icancelled = 0
isessions = 0

# loop through all sessions in the project. More detailed filters could be
# used to specify a subset of sessions
for session in project.sessions.find():

    full_session = fw.get_session(session.id)
    isessions += 1
    for analysis in full_session.analyses:

        analysis_job=analysis.job

        #only print ones that match the analysis label
        if gear_name in analysis.label:
            if any(analysis_job.state in string for string in ["complete"]):
                icomplete += 1

            elif any(analysis_job.state in string for string in ["failed"]):
                ifailed += 1
                log.info("subject: %s session: %s %s job: %s %s", session.subject.label,
↪ session.label, session.id, analysis.id, analysis_job.state)

            elif any(analysis_job.state in string for string in ["cancelled"]):
                icancelled += 1
                log.info("subject: %s session: %s %s job: %s %s", session.subject.label,
↪ session.label, session.id, analysis.id, analysis_job.state)

log.info('%s Sessions, gear %s: %s complete, %s failed, %s cancelled', str(isessions),
↪ gear_name, str(icomplete), str(ifailed), str(icancelled))
```

(continues on next page)

(continued from previous page)

Lets point to a file location on our local machine (in this case Blanca Compute) to store the analyses locally.

TIP: Point to a large scratch filesystem for fast read and write operations.

```
[ ]: # path to scratch directory
username=os.getenv('USER')
scratch='/scratch/blanca/'+username+'/'
os.chdir(scratch)
```

1.24.7 Download Analyses to Local Filesystem

Next we are going download analyses of interest. In this example, we will download all fmriprep output directories. There is plenty of customization you can use here to be sure you are downloading only the sessions and analyses of interest. Check out some filtering examples in our tutorial [here](#) .

WARNING: This will take some time!

```
[ ]: # loop through all sessions in the project. More detailed filters could be
#    used to specify a subset of sessions
for session in project.sessions.find():

    full_session = fw.get_session(session.id)
    isessions += 1
    for analysis in full_session.analyses:

        analysis_job=analysis.job

        #only download ones that match the analysis label
        if gear_name in analysis.label:
            if any(analysis_job.state in string for string in ["complete"]):

                # Download the data to scratch
                for fl in analysis.files:
                    fl.download(scratch+fl['name'])

                # unzip files
                if '.zip' in fl['name']:
                    zipfile = ZipFile(scratch+fl['name'], "r")
                    zipfile.extractall(scratch)

    log.info('Downloaded analysis: %s for Subject: %s Session: %s', analysis.
↪label, session.subject.label, session.label)
```

Now that your data is stored in local scratch, its time to run your analysis as you always would, for example... CONN analysis. To get started with your CONN analysis, you will need to return to Open OnDemand and start a new core desktop session. For more information on how to do this, please visit our [documentation](#)

After you are happy with your analysis don't forget to upload your analysis results to flywheel. Follow the instructions in this tutorial to get started: [upload-my-analysis.ipynb](#)

1.25 Upload My Analysis to Flywheel

Date: 26-Sept-2022

Description:

- This notebook provides a walk through to upload analyses generated on a local filesystem to Flywheel. Some neuroimaging analyses and workflows are not yet supported as Flywheel gears. For these workflows, the current workaround is to download your Flywheel analyses (or workflow inputs) run the workflow on your local machine, then upload the results (workflow outputs) back to flywheel as a new analysis.
- It should be possible to run this notebook in any jupyter-compatible third party-platforms such as [google collab](#) or [mybinder.org](#).

1.25.1 Requirements

- University of Colorado at Boulder Research Computing (CURC) account
- Access to University of Colorado Flywheel Instance

The following workbook should be run on CURC Blanca Compute. If you are unsure you have the correct permission or access to these resources please contact INC Data and Analysis team: Amy Hegarty [amy.hegarty@colorado.edu] or Lena Sherbakov [lena.sherbakov@colorado.edu].

CURC Jupyterhub

Before launching this jupyter notebook, users should launch this session using Open OnDemand.

TIP: Follow the instructions on INC Documentation to get started with Jupyter Notebooks.

We will be working on a large scratch system mounted only on Blanca compute nodes in this tutorial. If you do not have access to this filesystem you should select a similar large capacity scratch environment for analysis.

1.25.2 Setup

TIP: Please use the “flywheel” kernel for this tutorial. If you do not see a “flywheel” kernel, contact INC Data and Analysis team to install this environment.

```
[1]: print("Welcome to Intermountain Neuroimaging Consortium!")
```

```
Welcome to Intermountain Neuroimaging Consortium!
```

```
[ ]: # Python standard package come first
import logging
import os, platform, sys
from zipfile import ZipFile
from datetime import datetime

# Third party packages come second
```

(continues on next page)

(continued from previous page)

```
import flywheel

# add software paths
sys.path.append('/projects/ics/software/flywheel-python/bids-client/')
sys.path.append('/projects/ics/software/flywheel-python/')
```

Lets initialize a logger to keep track of the progress of our job (e.g. useful to keep track of runtime).

```
[ ]: # Instantiate a logger
logging.basicConfig(level=logging.INFO)
log = logging.getLogger('root')
```

Lets check we are on the correct computing system.

```
[ ]: host = os.getenv('HOSTNAME', os.getenv('COMPUTERNAME', platform.node())).split('.')[0]

if "bnode" not in host:
    log.error("Tutorial should be run on CURC high performance compute nodes: blanca")
```

1.25.3 Flywheel API Key and Client

You can get you API_KEY by following the steps described in the Flywheel SDK doc [here](#).

DANGER: Do NOT share your API key with anyone for any reason - it is the same as sharing your password and may break human subject participant confidentiality. ALWAYS obscure credentials from your code, especially when sharing with others/committing to a shared repository.

```
[ ]: API_KEY = getpass('Enter API_KEY here: ')
```

Instantiate the Flywheel API client either using the API_KEY provided by the user input above or by reading it from the environment variable FW_KEY.

```
[ ]: fw = flywheel.Client(API_KEY if 'API_KEY' in locals() else os.environ.get('FW_KEY'))
```

You can check which Flywheel instance you have been authenticated against with the following:

```
[ ]: log.info('You are now logged in as %s to %s', fw.get_current_user()['email'], fw.get_
    ↪config()['site']['api_url'])
```

1.25.4 Constants

Often you will have to define a few constants in your notebook which serve as the inputs. Such constant for instance is the API_KEY that was used to instantiate the Flywheel client. Other examples could be a PROJECT_ID or PROJECT_LABEL that will be used to identify a specific project.

```
[ ]: PROJECT_LABEL = 'MyProject'
```


1.25.5 Helper functions

Here are all the custom helper functions we have developed for use in this example.

```
[ ]: def get_project_id(fw, project_label):
    """Return the first project ID matching project_label

    Args:
        fw (flywheel.Client): A flywheel client
        project_label (str): A Project label

    Returns:
        (str): Project ID or None if no project found
    """
    project = fw.projects.find_first(f'label={project_label}')
    if project:
        return project.id
    else:
        return None
```

1.25.6 Upload My Analysis

First, lets point to a project in Flywheel.

```
[ ]: project_id = get_project_id(fw, PROJECT_LABEL)
if project_id:
    print(f'Project ID is: {project_id}.')
else:
    print(f'No Project with label {PROJECT_LABEL} found.')
```

Next we need to zip the contents of our analysis then upload those zipped directories to a flywheel analysis.

```
[ ]: project = fw.get_project(project_id)

# create new project level analysis
analysis = project.add_analysis(label='CONN Analysis: ' + datetime.now("%x %X"))

# zip outputs...
os.system('zip -R conn_analysis conn_analysis/ conn_analysis.mat')

# upload output zipped directories
analysis.upload_output('conn_analysis.zip')

if os.path.exists('conn_inputs.zip'):
    os.system('zip -R conn_inputs conn_inputs/ ')
    analysis.upload_output('conn_inputs.zip')
```

Finally, you can open Flywheel GUI to check your analysis was sucessfully uploaded.