COLERIDGE INITIATIVE

COLLABORATION MODULE

Contents

1. Summary

Individuals within and across organizations need to collaborate both within and outside secure research data facilities. The ADRF collaboration module is designed to address such needs. This document provides the following:

(i) a summary of the existing ADRF approach to collaboration,
(ii) features in development, and
(iii) a discussion of future plans.

2. Motivation

Datasets are complex in their own right; when data need to be combined from different sources, across different policy domains, for different research and analysis purposes, it is necessary for teams to work together and communicate effectively. Communication can occur in three different ways. One is team-focused so that people working together can convey real-time information about team activities, such as the development of ideas, schema and code. The second is to tap into a broader community of practice – like a crowd-sourced help desk. The final is asynchronous – so that individuals who learn specific information about data and data fields can provide annotations for subsequent users.

3. Current Approach

The ADRF workspace environment was designed with collaboration and resource sharing as a first order priority. The ADRF environment is set up as a secure, shared workspace with tools that make it easier to work in teams and across organizations around data. The core set of tools we use – Git (and Gitlab), Mattermost, and Jupyter Notebooks – are open source because (i) they minimize agency cost, (ii) they tend to be more innovative than licensed software, and (iii) because they can be highly secure[1]. They can also be used in multiple contexts, as discussed below.

3.1 Team-focused Collaboration

Each team is provided with a shared project workspace designed for collaboration. The intent is to establish a secure and access-controlled environment on the ADRF file system to ensure that teams have a space to experiment, develop, and share project details without worrying about disclosure of sensitive information outside of the team. Each project workspace includes (i) shared and access controlled file system space for multiple users to collaborate on a project, (ii)

[1] http://dodcio.defense.gov/Open-Source-Software-FAQ/

shared and access-controlled database space where project members have full privileges to create and modify project-specific database tables, (iii) shared Gitlab[2] project spaces (see **Appendix A**), (iv) a chat application (Mattermost[3]) that is access controlled at the project level to allow group members to quickly share materials and information, and (v) the flexibility to readily integrate additional shared tools based on client requirements.

Access restrictions for the system are controlled in DF Admin, the custom built ADRF data facility access control management system. DF Admin allows ADRF system administrators to manage and audit which users and groups have access to which projects and project tools and when.

Shared and access controlled file system space

The foundation for allowing teams to collaborate with each other effectively and securely is a single access-controlled project space that all project members can access. It is secured for only those who have explicit access to the data so that inappropriate disclosure does not occur. These shared project spaces then allow users access to the same tools and provides them with the ability share their code, analysis output, and extracts from the data relevant to their project in a safe and intuitive way.

Shared and access controlled database space

Users are provided with controlled access to project databases. They can generate derivative analysis that is most useful for their project and have a single source for storing additional data that all other project members may access. This approach both reduces costly data duplication and increases research replicability across teams.

Tools for teams

Tools that can be used for secure communication of project code and ideas are also available in the ADRF.

One tool is Gitlab, a browser-based open source tool, that makes Git[4] – a popular version control system – both easier to use and more socially interactive. The tool enables researchers to keep their work up-to-date, versioned, and synchronized across project members. At the same time, it keeps this information available only for those who have project access to it; other system users may not have access unless explicitly invited to do so.

Another tool is Mattermost, which is open source chat software similar to the popular communication tool, Slack[5]. Mattermost has the advantage that it can be installed and used on a server (and not centrally hosted over the Internet, like Slack). As such, Mattermost can be used to create and manage team-by-team chat spaces so that project teams can discuss the

[2] https://about.gitlab.com/
[3] https://about.mattermost.com/
[4] https://git-scm.com/
[5] https://slack.com/

data and analysis of their project without sharing any sensitive details outside of the project group or the ADRF system.

Finally, all projects feature Jupyter[6] notebooks integrated into analysis workflows to provide opportunity for project members to create well-commented and shareable code and analyses (with appropriate review from data stewards). Jupyter notebooks are built to view files already located on the file system, in project workspaces. Therefore, users can only view and edit files which they already have access to in the file system based on project permissions (see **Figure 1**).

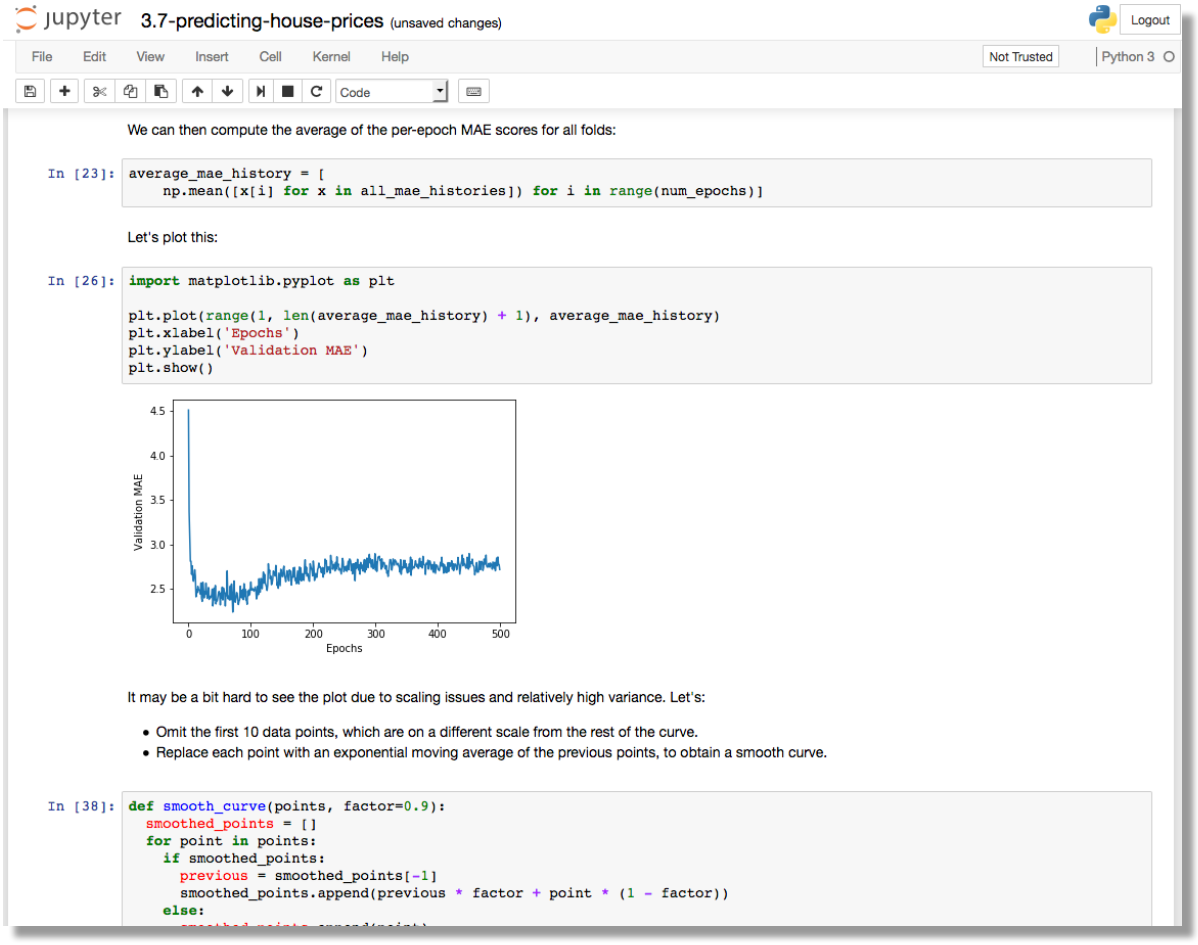


Figure 1. Screenshot of Jupyter notebook showing combination of code, documentation, and data visualization all in one place (source: https://github.com/fchollet/deep-learning-with-python-notebooks/blob/master/3.7-predicting-house-prices.ipynb)

[6] http://jupyter.org/

These are the tools we have found to be most useful for teams to work together in the ADRF, but we do not limit team collaboration exclusively to these tools. If other tools promise to provide secure and effective communication between team members in the ADRF, they could be added following an appropriate security and feature review by ADRF system administrators.

3.2 Community Collaboration

The ADRF Explorer and the three core tools installed in the ADRF also provide a centralized set of community and social tools for ADRF users. Our experience is that these tools allow researchers to communicate and share insights and code throughout the data analysis lifecycle. More importantly, these tools also allow users to share their data and analysis with other groups where appropriate and allowed by the terms of use of the data they are analyzing.

Slack and Mattermost

Interactive, real-time "chat" can be facilitated between project groups by Slack outside of the ADRF and Mattermost inside of the ADRF (to share details about data that may not leave the ADRF) for direct or group-based messaging. While Mattermost can be setup for in-group discussions only, it can also be used to host discussions with the larger community around datasets and methods.

ADRF Explorer

ADRF Explorer – the ADRF's data cataloging, documentation, and discovery engine – allows users to add annotations to dataset metadata, as well as add feedback and code snippets to share ideas and functioning analysis with other users (see **Figure 2**), provided each user has access to view detailed information about any given dataset in the Explorer.

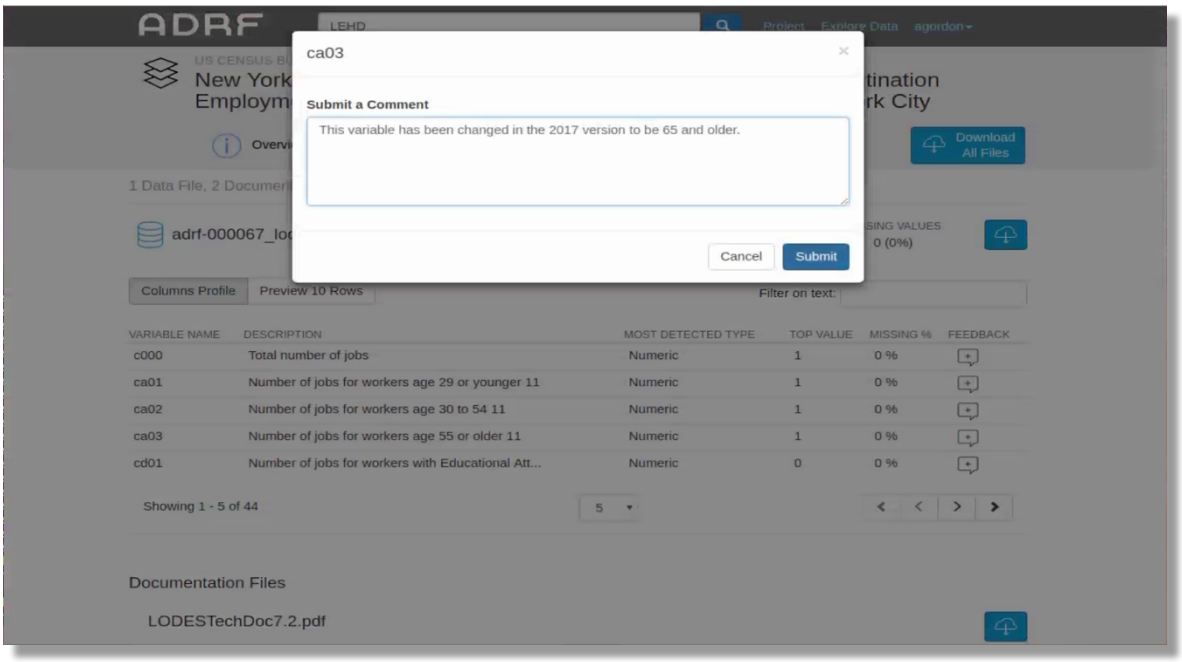


Figure 2. Screenshot of user adding an annotation to a variable for a dataset in the ADRF Explorer

Git and Gitlab

Git and Gitlab's primary role in the ADRF is to allow users to share and conduct version control on code and analyses within their project group. However, it also provides users with issue tracking, the ability to fork and create derivative versions of a codebase, and to write project documentation via a wiki. This functionality can be used to share project work with other users in the ADRF following an appropriate export review process conducted by data stewards familiar with the terms of use for a project's datasets.

Jupyter

Finally, Jupyter notebooks are a useful tool for sharing well-commented code and analyses with others outside of the project group and the wider ADRF community. Provided that users have access to a specific directory or set of files that can be navigated to in the Jupyter browser, they may share code that they are using to analyze data with ease while also providing in-line documentation for what that code is doing, making it easier for those coming from outside of the project to understand the work being conducted there.

3.3 Asynchronous Collaboration

The ability to provide details on datasets and projects for future users is also important. Currently under development is a tool to be integrated into the ADRF that enhances users' ability to annotate datasets, attach projects and project explanations, code, and publications to

datasets, all which will provide a deeper level of understanding about what makes a dataset useful (see **Figure 3**).

This functionality will support data users in the ADRF as a social community who can create custom collections of datasets and explain for what purposes datasets in the ADRF may be linked. Users will also be able to tag datasets, comment on other users' dataset collections, and earn points by contributing improved documentation to dataset – benefiting other users. As such, this tool would provide both community-wide and asynchronous collaboration allowing users to share information about potential uses and linkages between data as well crowdsource a deeper understanding of individual dataset variables over time.

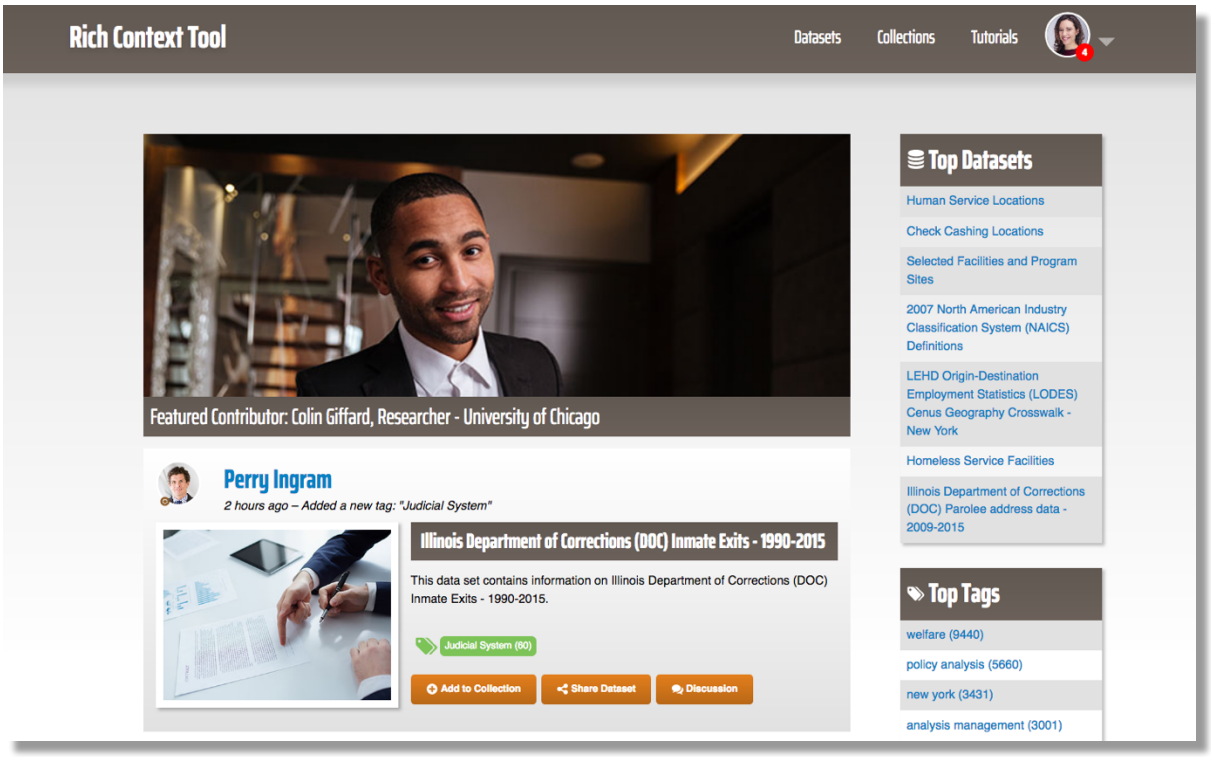


Figure 3. Screenshot of Community-based data exploration tool currently under development

3.4 Technical Details

In the ADRF, the single point of truth about Users, Projects and Datasets is DF Admin. DF Admin is a Django web application where the ADRF administrators can create and update those entities. DF Admin syncs this information with LDAP, a well-known application widely used that shares user and group information across network applications and systems, to store users and their passwords. The ADRF also stores the Projects and Datasets as LDAP groups to make this information available to systems that support LDAP. For example, GitLab is aware of the ADRF Projects and their members through LDAP.

The workspace servers are Linux Servers and they authenticate against LDAP. Inside Linux, the LDAP Projects and Datasets become groups. There is a script running every 5 minutes collecting new LDAP projects and creating a folder for them in NFS (Network File System). Linux sets read and write permissions to this folder for all users inside the project group, which is how security is enforced in the projects folders in the ADRF environment.

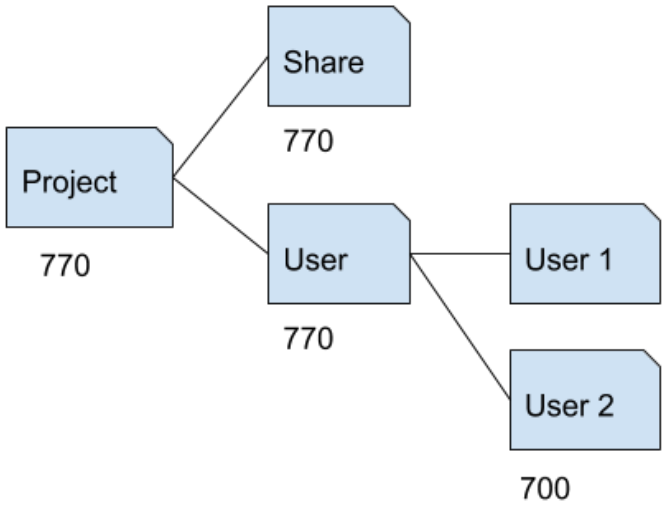


Figure 4. Diagram of Project directory structure and permissions

The Project folder, the Share folder, and the User folder have the root as owner and the project group as the group owner. The owner and the group owner have read and write permissions in these folders (770). The user folders inside the folder User have the members of the project as owners. Each member owns a folder, and inside this folder only they have read and write permissions.

The ADRF has a Postgres Relational Database Management System (RDBMS). The RDBMS has a central database and every ADRF Project has a schema. There is a python script pulling the Project information from LDAP and creating the Project schemas in the database and setting read and write permissions for the project's members.

In GitLab, each ADRF Project has a repository. There is a python script pulling the information from LDAP and creating the repository. Once created, GitLab updates the members of the repository without the help of any scripts. The members of the groups will receive developer permissions inside the project repository, which basically gives to them the ability to pull and push, but not the ability to manage the repository.

Mattermost permissions work similarly to the other applications. A python script pulls the information from LDAP and creates a Mattermost team and channels for each project. Each project will have a team associated in Mattermost. All the members of the project will be members of the Mattermost team. Inside the team, the members of a project will be able to chat with one another. They are free to create their own channels (usually project topics) inside the team.

Keycloak is the Identity and Access Management System in the ADRF. Keycloak pulls all user and group information from LDAP. Keycloak is the Oauth Identity Provider. All Web applications available in the ADRF use Keycloak as the authentication system. It applies this authentication to Mattermost, ADRF Explorer, GitLab, and JupyterHub.

4. Future

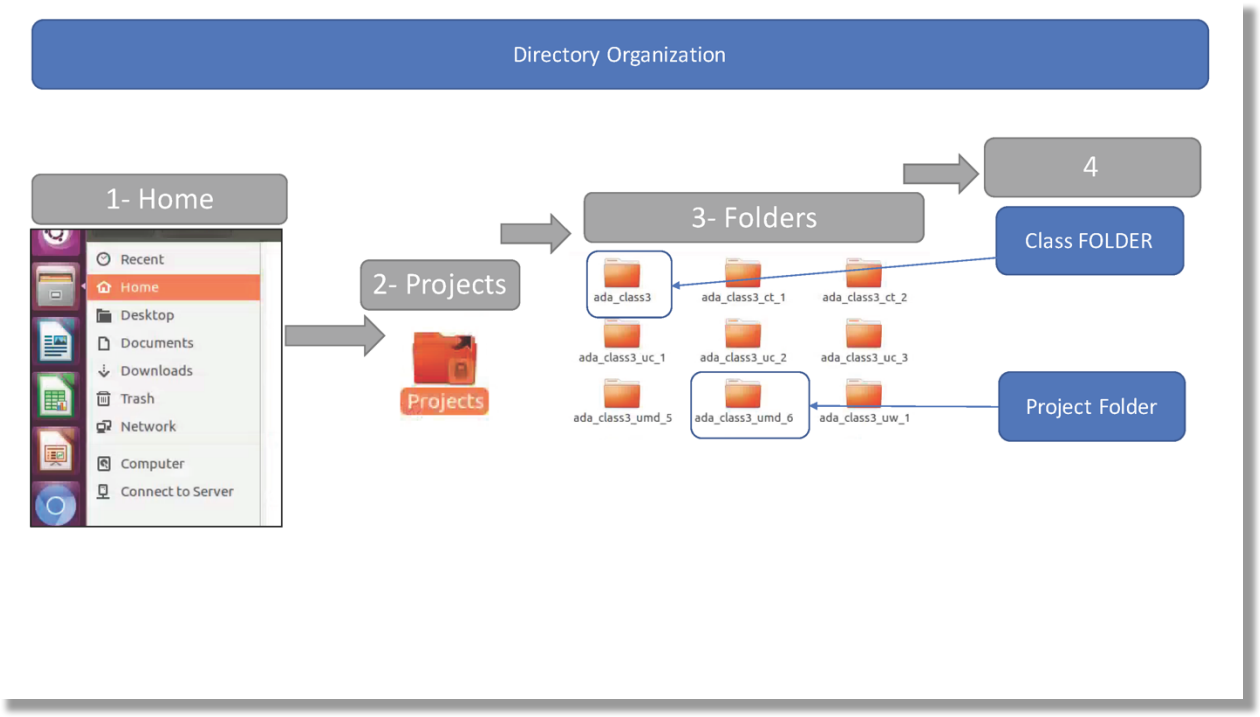
4.1 Overview of Customizations

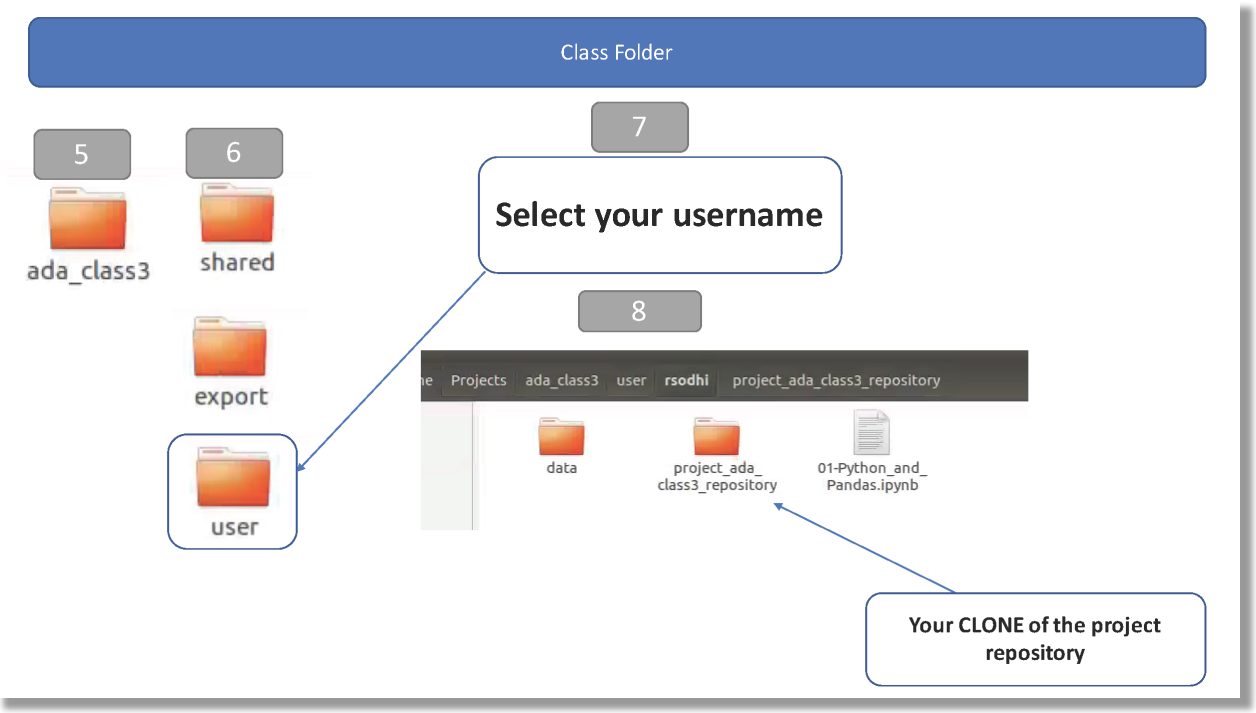
The collaboration module for the ADRF, with appropriate training and documentation, can be customized to conduct their work within teams in a secure data facility. Determining what to customize depends on what is needed for collaboration or if there are any changes that must be made to the project workspaces. The following elements are necessary:

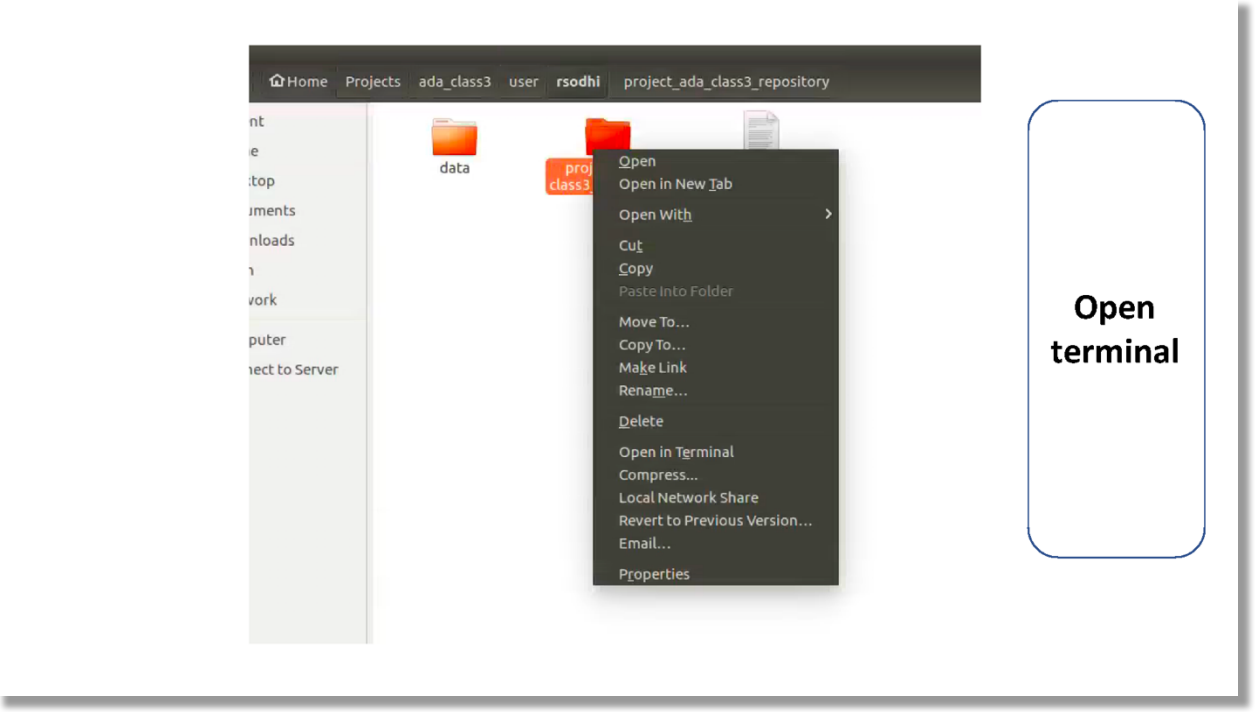
1. Understand what collaboration tools organizations might need and review for suitability within the secure data facility environment.
2. Understand what changes might need to take place with respect to project workspaces either for legal or organizational purposes.
3. Continue to develop functionality of social tools around datasets as needed.

Appendix A

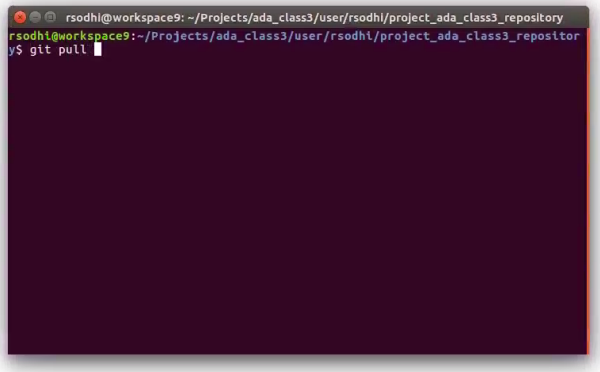
Sample Applied Data Analytics course slides instructing system users on the appropriate usage of Git within their projects.




Class Folder
5
6
7
ada_class3
shared
Select your username
8
export
user
Projects
ada_class3
user
rsodhi
project_ada_class3_repository
data
project_ada_
class3_repository
01-Python_and_
Pandas.ipynb
Your CLONE of the project repository


Home
Projects
ada_class3
user
rsodhi
project_ada_class3_repository
data
Open
Open in New Tab
Open With
Cut
Copy
Paste Into Folder
Move To...
Copy To...
Make Link
Rename...
Delete
Open in Terminal
Compress...
Local Network Share
Revert to Previous Version...
Email...
Properties
Open terminal

Class Folder



After this 'git pull'– if you do get an error about a merge conflict--- you can follow the below commands.

However, do this ONLY IF YOU ARE TRYING TO TAKE THE LATEST INSTRUCTIONAL MATERIAL. DO NOT DO THIS IN YOUR PROJECT FOLDERS OR ELSE YOU WILL LOSE ALL CHANGES

Type–
"git fetch origin"

Followed by
"git reset –hard origin/master"

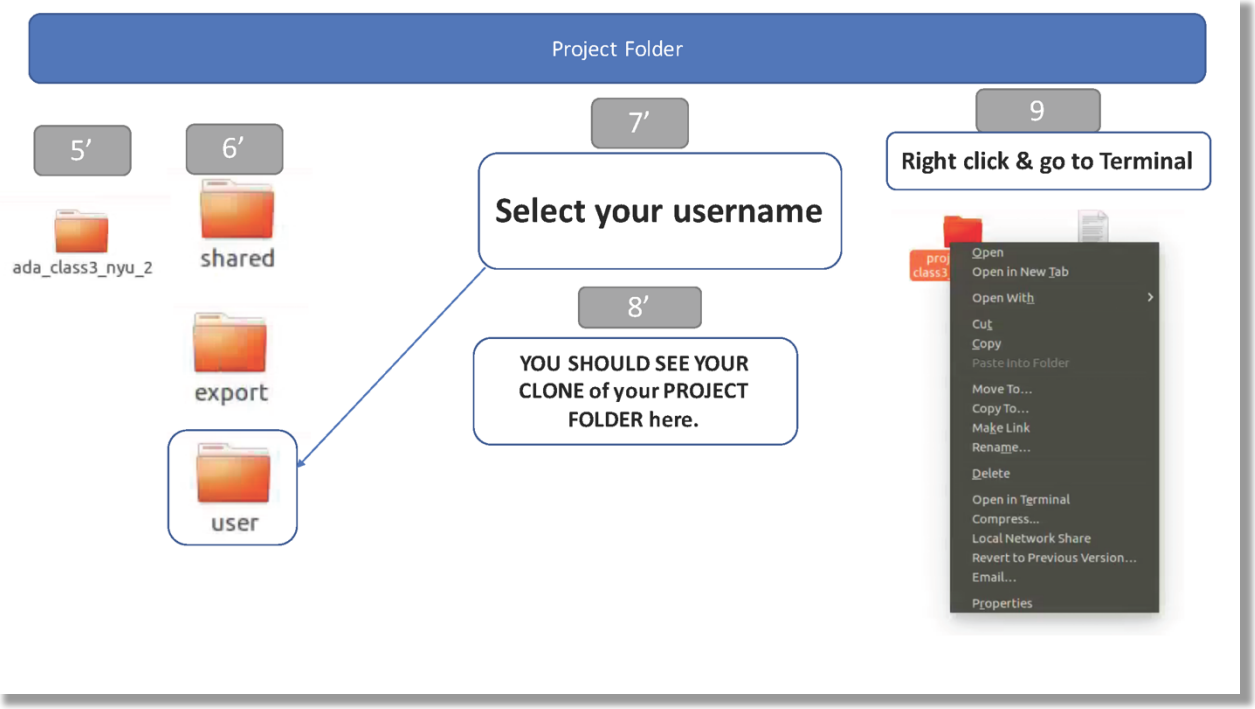
This will ensure that your local changes are dropped.

Class Folder

An ideal success message will either be:

a) Already up-to-date

b) (e.g. ~ 1 file changed, 99 assertions made)



GIT

1. If you have made changes or created a new file---- type: **git status**

2. You will see that your changes are not staged for commit--- now type **git add <filename>**

3. Now- type **git commit –m "commit message"**

4. **git pull:** This is to pull changes your team members have made

5. **git push:** Finally you can push your committed changes (which you did in STEP 3).

Now your team members can do the same thing and pull your changes to their directories as well.