[Section 1](#_souskvtbnyhm)

[Setting up Python environment:](#_e5xnukx5hzt)

[Section 2](#_jr7ug2cejdq5)

[Creating curation environment and cloning figshare codes from github/cognoma:](#_mmi38zucbdnl)

[Section 3](#_4qy3uc1z8g5)

[Cloning the Virginia Tech Libraries data curation codes from github:](#_erzgsbkntwgv)

[Section 4](#_76s6kqx8tsbb)

[Creating a token on Virginia Tech Data Repository(VTDR) and navigating VTDR:](#_2w05e7zbfldx)

[Section 5](#_pagtyy9fcsac)

[Setting up configurations on your local computer:](#_rzyg1nkzgxjd)

[Setting up APTrust Repo registry configurations and Partner Tools:](#_b8vfdrp4s1bp)

[Section 6](#_2bv9vtu61mcp)

[Google sheets API automation:](#_24j39dx59x2z)

[Section 7](#_2p1t24qonw)

[Filling ingest/published article information in VTDR v7 spreadsheet and setting up DART:](#_zdeagpqpofms)

[Section 8](#_dp8iba34pr38)

[Running the VTDR Workflow scripts:](#_2s3wfbig6f4x)

[Creating README.rtf:](#_ny54qf57ggii)

[Creating Ingest bag creation and depositing it to APTrust-Repo/APTrust-Demo/ VT-Library S3:](#_dgzwbzjh4bwu)

[Steps for Publication bag creation and deposition to APTrust-Demo/APTrust-Repo/VT-Library S3:](#_fgptfd4eroh)

[Section 9](#_uu5heor7v9az)

[Checking the bags created by DART and bag validation (need to fill information later on bag validation):](#_bzukejx3ayg2)

[Section 10](#_a7zqeuwd92fy)

[Accessing information from deposited bags through VT library S3, non-disseminated content folder, LibCRM:](#_70rtpv3xg4rr)

[Section 11](#_6g1u6hhy6lf5)

[Changing thumbnails:](#_npzk3mdrusvq)

[Section 12](#_wccb6vr3fwqp)

[Possible errors:](#_1hf8ng32jgi0)

These directions are written for installation on Mac

# Section 1

## Setting up Python environment:

-Download [Anaconda | Anaconda Distribution](https://www.anaconda.com/products/distribution#Downloads) for Mac. **NOTE**: Pick Destination Folder as “/Users/padma/opt/anaconda3” in the process.

-Use a python source code editor preferably [visual studio code](https://code.visualstudio.com/) for Windows/Mac

-For Mac install git at <https://git-scm.com/download/mac>, below are detailed instructions:

-To open terminal on Mac, click Launchpad and under “Search” type terminal

-In the open terminal type /bin/bash -c "$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh>)"

-Next type ‘brew install git’

-Install GoogleDrive: To access bags/administration content on shared google drive install <https://www.google.com/drive/download/>. Upon completion of installation a google drive icon will appear in the launchpad or open finder and google drive will appear on the left side under favorites.

# Section 2

## Creating curation environment and cloning figshare codes from github/cognoma:

Open Visual Studio Code, Open Terminal->New Terminal, open bash and type the following commands-

(Some of the commands in this section are copy pasted from [UAL-RE/LD-Cool-P: Python tool to enable data curation (github.com)](https://github.com/UAL-RE/LD-Cool-P) with some changes)

(Following command creates a folder called “Curation” at the following path: /Users/padma/opt/anaconda3/envs)

$ conda create -n curation python=3.9.0

Click y when Proceed ([y]/n) option pops up

(Following command activates the curation folder)

$ conda activate curation

(The above command creates a folder curation whose path is at /Users/padma/opt/anaconda3/envs/curation)

$ cd /Users/padma/opt/anaconda3/envs/curation

-Next open curation folder: Go to File->OpenFolder on VS Code and select curation folder at the above path

-Go to extensions in the left-hand corner on VS Code and enter python, select Python IntelliSense (Pylance) and install this python interpreter. After installation, click shift+ctrl+P, type Python interpreter and select Python 3.9.0 (‘curation: conda)

On VS Code, go to “Terminal” tab and open new terminal, under this terminal on the right column click the drop-down menu, next to “+” tab and select “ **bash**”

Next, clone figshare from cognoma at [GitHub - cognoma/figshare: A package for downloading and uploading figshare data](https://github.com/cognoma/figshare), in the **bash** terminal type:

(curation)$git clone <https://github.com/cognoma/figshare.git>

(Alternately, this can be cloned by shift+ctrl+p and pick Git: Clone, go to the github URL above , click code, copy https link and paste it in VSC Git: Clone, press enter, pick curation directory)

Now, run the setup.py script as follows in the **bash** terminal:

(The path below will be /Users/padma/opt/anaconda3/envs/curation, where username is what you have to enter, alternately you could go to the folder where curation folder was created, and copy paste the path below)

(curation) $ cd /Users/padma/opt/anaconda3/envs/curation/

(The path below will be/Users/padma/opt/anaconda3/envs/curation/figshare, where username is what you have to enter, alternately you could go to the folder where curation folder was created and copy paste the path below)

(curation) $cd /Users/padma/opt/anaconda3/envs/curation/figshare

(curation) $python setup.py develop

(**Errors:** If setup doesn’t develop or conda is not recognized then activate it from the terminal, type “conda activate base” from the powershell followed by “code” to open VS Code from the shell,detailed instructions: [Running Visual Studio Code on macOS](https://code.visualstudio.com/docs/setup/mac#:~:text=Launching%20from%20the%20command%20line,code%27%20command%20in%20PATH%20command.))

# Section 3

## Cloning the Virginia Tech Libraries data curation codes from github:

-Clone [VT Figshare repository](https://github.com/PadmaCarstens/VTechDataServices-Figshare) from [Virginia Tech University Libraries github repository](https://github.com/VTUL):

Open bash on VS Code, clone VT repository in the curation folder using the following command:

$ cd /Users/padma/opt/anaconda3/envs/curation/

$ git clone <https://github.com/VTUL/VTDR_RepositoryServices.git>

# Section 4

## Creating a token on [Virginia Tech Data Repository](https://data.lib.vt.edu/)(VTDR) and navigating [VTDR](https://data.lib.vt.edu/):

Short description on VTDR: VTDR is Virginia Tech’s platform for sharing the research datasets provided by the researchers. The researcher is provided with a citation and DOI upon publishing their dataset. The research can then include this in a journal or manuscript or thesis/dissertation etc. This platform is based off of figshare and customized to fit VTDR’s requirements. On VTDR, curators are encouraged to read information in the tabs:“About, Preparing Data for Deposit tap and Deposit Agreement” found at the bottom of VTDR. This will help the curator have a better understanding of VTDR and how it operates.

**Navigating VTDR:**

**Creating a token:** Click on the name initials on the top right corner of VTDR, this is displayed in a circle. Pick “Applications”, scroll down to “Personal Tokens”, click on “Create Personal Token”, in the “Desc

ription” box enter any description e.g. “token for VTDR workflow”, click “Save”. On the next pop up box, copy the token created and paste it in the token tag in generate\_config.py.

**Impersonating author’s account:** Click on the name initials on the top right corner of VTDR, this is displayed in a circle. Go to “Administration”, pick the tab “USERS”, under “search users” box type the authors name, click the gear account and pick “Access this account”, click the pen icon that pops up when scrolling on the item, here you can edit the metadata of the item or add/drop files in the “Manage” box.

Other figshare functionalities can also be explored following figshare documentation found at <https://help.figshare.com>. Documentation for interacting with Figshare’s API can be found at: [Figshare documentation](https://docs.figshare.com/)

# Section 5

## Setting up configurations on your local computer:

Go to File->Open File-> Open generate\_config\_example.py located at Users/username/Anaconda3/envs/curation/VTechDataRepo/generate\_config\_example.py, go to File->Save As-> (Go to curation/VTechDataRepo folder) generate\_config.py

Open .gitignore in Users/username/Anaconda3/envs/curation/VTechDataRepo/ and make sure generate\_config.py appears in this list.

Open generate\_config.py and fill in all the following fields, example for test user “Padma” is :

FigshareArticleID: This is the number found towards the end of the DOI link, click the red "Cite" button on the item under review/to be published

PubVerNum: Publication Version Number to be downloaded, found at the end of "Cite"

VTDRToken: VTDR token created under applications

CurName: Name of the curator, this shows up in ArchivalReadme Package created in the publication folder

CurationDir: Curation directory where Ingest/Publication folder/README file will be created

NonDissContentDir: Directory where Non disseminated content will be stored, non disseminated content includes provenance log, email correspondence and archival readme package created in the publication folder

FileID: This is used only in the ChangeThumbnail.py script. File ID is the number at the end of the file link, this is the ID of the file that the curator wants to change the thumbnail of the published article to.

FigshareArticleID="21408450"

PubVerNum="01"

VTDRToken="1234"

CurName="Padma Carstens"

CurationDir="/Users/padma/opt/anaconda3/envs/curation"

NonDissContentDir="/Volumes/GoogleDrive/Shared drives/CurationServicesGoogleDriveArchive/BAGS/NonDisseminatedContent"

DartExePath="/Volumes/DART\ 2.0.21/DART.app/Contents/MacOS/DART"

ReadmeDir="/Users/padma/opt/anaconda3/envs/curation/README\_FILES"

FileID="38937515"

**Finding DartExePath on Mac**:

1. Download DART app from [Download - DART Documentation (aptrust.github.io)](https://aptrust.github.io/dart-docs/download/) and do the DART setup
2. Right click DART app icon in the dock (this looks like 2 black circles with an arrow pointing to the center), click ‘show in finder’
3. This brings up a DART box with a DART icon pointed to the ‘Applications’ folder. Right click on the DART icon and click “Show Package Contents”
4. This brings up a ‘Contents’ folder. Go to ‘Contents’->MacOS->DART
5. Drag this “DART” folder into the terminal window, this gives the path to the DART executable, copy this path to ‘DartExePath’ in generate\_config.py

**Finding NonDisseminatedContentFolder on Mac:**

To get the path to NonDisseminatedContentFolder (for entering in generate\_config.py) follow the steps below:

click ‘google drives’-> shared drives->CurationServicesGoogleDriveArchive->BAGS->NonDisseminatedContetn

Open a terminal window and drag ‘NonDisseminatedContent’ folder into this window, this will provide the path to NonDisseminatedContent folder which can be copied into generate\_config.py, remove “\” used for spaces. For example, non disseminated content path looks something like this:

NonDissContentDir="/Volumes/GoogleDrive/Shared drives/CurationServicesGoogleDriveArchive/BAGS/NonDisseminatedContent"

**Ingest and Publication version numbers**:

Since ingest record is created before any changes are made to the article, this is always a first version (01). Only one version for ingest will be created and this version will correspond to the first time the client requests publication. If a client requests updates to their published dataset, a new version for ingest bag **WILL NOT** be created, only the publication bags will reflect the versioning updates.

Publication number will remain the same for updated publications. The version number and date (if applicable) will change for updated versions. This allows all updated versions to be connected to the same publication number. For eg: if P00235 is updated and original published bag is VTDR\_P00235\_I00268\_DOI\_22227883\_AbaidN\_v01\_202300310, the updated versions will look like VTDR\_P00235\_I00268\_DOI\_22227883\_AbaidN\_v03\_20230310, VTDR\_P00235\_I00268\_DOI\_22227883\_AbaidN\_v03\_20230706 i.e. only the version number and date change will be reflected in future versions, publication number P00235 will remain the same for all the updated versions.

**Note:** The config path “LargeBagsPath”,” F:/VTechbags” is for downloading huge datasets to a different location (in this case to Sandisk), this option can be ignored since it requires uncommenting a section of the script in the IngFolder\_Download\_TransferBagAPTrust.py and PubFolder\_Download.py in order to enable downloading to a different path.

After these configurations are saved, run generate\_config.py, this creates configurations.ini file (this is hidden if scripts are pushed to github through .gitignore)

## Setting up APTrust Repo registry configurations and [Partner Tools](https://aptrust.github.io/userguide/partner_tools/):

Fill in the following credentials:

—-------------------------------------------------------------------—

Following is copy pasted from [Partner Tools - APTrust User Guide](https://aptrust.github.io/userguide/partner_tools/)

APTRUST\_AWS\_KEY Access Key ID to access S3. Required only for S3 operations. Works with any S3-compatible service.

APTRUST\_AWS\_SECRET Secret access key to access S3. Required only for S3 operations. Works with any S3-compatible service.

APTRUST\_REGISTRY\_URL URL of the APTrust registry you want to access. Production is https://repo.aptrust.org. Demo is https://demo.aptrust.org. Required only for registry operations.

APTRUST\_REGISTRY\_API\_VERSION Version of the current registry API. For now, this should be v3. Required only for registry operations.

APTRUST\_REGISTRY\_EMAIL The email address associated with your APTrust registry account. Required only for registry operations.

APTRUST\_REGISTRY\_API\_KEY The API key associated with your APTrust registry account. Required only for registry operations.

—---------------------------------------------------------------------------

APTRUST\_REGISTRY\_URL = 'https://repo.aptrust.org'

APTRUST\_REGISTRY\_API\_VERSION='v3'

APTRUST\_REGISTRY\_EMAIL='xyz@vt.edu'

APTRUST\_REGISTRY\_API\_KEY=''

APTRUST\_AWS\_KEY=''

APTRUST\_AWS\_SECRET=''

**APTRUST\_REGISTRY\_API\_KEY:** Go to the name icon on repo.aptrust.org, click “GET API KEY”, generate your key and use it in the configuration file. Save the generate\_configurations.py in your “Curation” folder.

In order to access the aptrust repo registry, please download apt-cmd tools. Save apt-cmd app in your curation folder.

Please note that the registry list shows the ingested content only after the ingest status is “SUCCESS”. When the ingest content status is “STARTED” or “PENDING”, the registry list request does not find the file name and registers it as not ingested yet. Therefore, the file can still be overwritten when it is in “STARTED” or “PENDING” state.

# Section 6

## Google sheets API automation:

Copy client\_secret.json from Curation Workflow folder and paste it to the curation folder which is located in the anaconda environments eg: C:\Users\users\Anaconda3\envs\curation

# Section 7

## Filling ingest/published article information in VTDR v7 spreadsheet and setting up DART:

Before running these scripts, the following steps need to be followed:

1. Please make sure the ingest/published record information for the article in review/published is entered in the VTDR V7 spreadsheet in the corresponding sheet “Ingest” or “Published”
2. Please make sure DART app is installed and set up for uploading bags to APTrust following instructions in CurationWorkflow\_SetupDART\_APTrustDeposit.docx in the Curation Workflow folder

# Section 8

## Running the [VTDR Workflow](https://github.com/PadmaCarstens/VTechDataRepo) scripts:

Note for Errors:

If “module not found error” is received then try “pip install modulename”, examples below:

-For “PyRTF module not found error” do “pip install PyRTF3”

-For “bagit module not found” do “pip install bagit”

Run the workflow scripts in the following order:

When article is in review:

1. **Run AutomatedREADMErtf.py**
2. **Run IngFolder\_Download\_TransferBagAPTrust.py**

After article is published:

1. **Run PubFolder\_Download.py**
2. **Run PubBagDart\_TransferBagAPTrust.py (after transferring provenance log and email interaction documents)**

The details needed to be filled in while running the scripts are as follows:

### Creating README.rtf:

-Enter the article id whose README.rtf needs to be created in generate\_config.py and **run generate\_config.py**

-Make sure ingest information in the “Ingest” sheet in the VTDR V7 spreadsheet is filled in as described in section 6

-**Run the script AutomatedREADMErtf.py** located at Curation/VTechDataRepo/Figshare-APTrust

-A README.rtf will be created at Curation/README\_FILES\_timestamp\_authorname

### Creating Ingest bag creation and depositing it to APTrust-Repo/APTrust-Demo/ VT-Library S3:

1. Open the script: IngFolder\_Download\_TransferBagAPTrust.py
2. Pick a workflow for depositing bag to APTrust-Demo/APTrust-Repo/VT-Library S3 (enter 1, 2, 3 or 4 from the options below):

Option 1:Enter “1” (without the quotations) to deposit the ingest bag to APTrust-Demo storage, this option is for depositing test bags to APTrust-Demo storage system. Bags deposited here will be deleted after a certain amount of time

Option 2: Enter “2” to deposit the ingest bag to APTrust-Repo and VT library S3 storage systems (this is the standard option). APTrust-Repository does periodic integrity checks on the bags deposited. Depositing bags to the VT library S3 storage system will allow us to access the bags if needed. Bags deposited to VT library S3 storage will be deleted every few months.

Option 3: Enter “3” to deposit the ingest bag to VT library S3 storage system only

Option4: Enter “4” to deposit the ingest bag to APTrust-Repo only

1. **Run the script IngFolder\_Download\_TransferBagAPTrust.py**

Running this script after picking the workflow above will create an ingest folder in the “Curation” folder which looks like VTDR\_I00NBR\_lastnamefirstnameinitial\_lastnamefirstnameinitial\_v0X\_date. An ingest bag (ingest folder in tar format and with tag values in it) will also be created at Username/. dart/bags or at the “Output Path” picked in “Application Setting” in DART app. The bag name will have the same naming convention but with a .tar at the end. This bag will be uploaded to APTrust-Demo/APTrust-Repo/VT-Library s3 bucket depending on the workflow selected in step 2. The bags on demo or repo can be checked for upload at demo.aptrust.org or repo.aptrust.org

### Steps for Publication bag creation and deposition to APTrust-Demo/APTrust-Repo/VT-Library S3:

1. Open and **run the script PubFolder\_Download.py**, this script downloads the published dataset and creates a publication folder in the “Curation” folder
2. Open the publication folder created, this folder will have a naming convention like: VTDR\_P00XYZ\_I00XYZ\_DOI\_XYZ\_lastnamefirstinitial\_v0X\_date.
3. Fill in the [Provenance Log](https://docs.google.com/document/d/1AOQsr0GP00mp3ZfpeW9oJE-wLY48oKWYZ_NR2cLz8wI/edit) and save Email interactions and save them as ProvenanceLog.rtf and Email Correspondence (or Email\_Correspondence1, Email\_Correspondence2 etc. in case of multiple email threads) in VTCurationServicesActions folder found at the path: C:\Users\username\anaconda3\envs\curation\VTDR\_P00XYZ\_I00XYZ\_DOI\_XYZ\_lastnamefirstinitial\_v0X\_date\VTCurationServicesActions
4. Open and **run the script PubBagDART\_TransferBagAPTrust.py**
5. Pick a workflow for depositing bag to APTrust-Demo/APTrust-Repo/VT-Library S3:

**Option 1:** Enter “1” (without the quotations) to deposit the ingest bag to APTrust-Demo storage, this option is for depositing test bags to APTrust-Demo storage system. Bags deposited here will be deleted after a certain amount of time

**Option 2:** Enter “2” to deposit the ingest bag to APTrust-Repo and VT library S3 storage systems (this is the standard option). APTrust-Repository does periodic integrity checks on the bags deposited. Depositing bags to the VT library S3 storage system will allow us to access the bags if needed. Bags deposited to VT library S3 storage will be manually deleted every few months.

**Option 3:** Enter “3” to deposit the ingest bag to VT library S3 storage system only

**Option 4:** Enter “4” to deposit the ingest bag to APTrust-Repo only

This will create a publication bag (publication folder in tar format and with tag values in it) “Username/.dart/bags” and has the same naming convention but with a .tar at the end. This bag will be uploaded to APTrust-Demo/APTrust-Repo/VT-Library s3 bucket based on what was selected in step 5. The bags on demo or repo can be checked for upload at demo.aptrust.org or repo.aptrust.org

1. Check the bag created following the instructions in section 9

# Section 9

## Checking the bags created by DART and bag validation (need to fill information later on bag validation):

To check the contents of the bag created in tar format created at the output path, go to the Start->Command Prompt and type:

>cd <Output Path> (*For Example*: if bag is created at an output path C:\Users\username\Documents\DART then change directory to

cd C:\Users\username\Documents\DART)

>tar -xvf VTDR\_I00NNN\_XYZ\_XYZ\_v01\_xof8\_date.tar

This will extract the contents of the bag above in the same directory as the bag in tar format.

# Section 10

## Accessing information from deposited bags through VT library S3, non-disseminated content folder, LibCRM:

-Set up VT library S3 following the instructions on Accessing Bags S3.

-**Opening the stored bags through webdrive**: Right click the webdrive icon, and select “Connect to S3”, this opens the S3 window where the uploaded bags can be accessed. Extract the bags following instructions under section 9.

-**Opening the stored bags through cyberduck:** click start windows icon, go to cyberduck app and the bags will be found under ESRI\_storage. Extract the bags following instructions under section 9.

# Section 11

## Changing thumbnails:

This section describes how to change the thumbnail on [**VTDR**](https://data.lib.vt.edu/) for a published article with a thumbnail that does not reflect the desired thumbnail. This can be done using ChangeThumbnail.py. Open generate\_config.py and enter the article ID (FigshareArticleID), published version number(PubVerNum) and FileID of the desired thumbnail file, this is found at the end of the file link after selecting it on [**VTDR**](https://data.lib.vt.edu/)). Run generate\_config.py, then run ChangeThumbnail.py. The status code “205” means the thumbnail change is successful.

# Section 12

## Possible errors:

—---------------------------------------------------------------------------------------------

**Error:**

“File "C:\Users\username\Anaconda3\lib\site-packages\redata-0.4.1-py3.9.egg\redata\commons\logger.py", line 3, in <module>

from os import path, uname, chmod, mkdir

ImportError: cannot import name 'uname' from 'os' (C:\Users\padma\Anaconda3\lib\os.py)”

**Possible solution:**

Then, open logger.py in the folder above and replace

“from os import path, uname, chmod, mkdir” on line 3 with the following 2 lines:

from os import path, chmod, mkdir

from platform import uname

Now, save the logger.py and run again

—---------------------------------------------------------------------------------------------

**Error:**

“Module gspread not found”

Then install gspread:

-pip install gspread

—---------------------------------------------------------------------------------------------

**Error:**

“ModuleNotFoundError: No module named 'oauth2client'”:

Then, install oath2client:

-pip install oauth2client

—---------------------------------------------------------------------------------------------

**Error:**

“ModuleNotFoundError: No module named 'PyRTF'”

Then, install PyRTF:

-pip install PyRTF

—---------------------------------------------------------------------------------------------

**Error:**

ModuleNotFoundError: No module named 'bagit'

Then, install bagit:

-pip install bagit

—---------------------------------------------------------------------------------------------

**Error:**

ModuleNotFoundError: No module named 'rdflib'

Then, install rdflib:

-pip install rdflib

—-----------------------------------------------------------------------------------------

**Error:**

Python/conda/pip command not found/recognized in VS code:

open anaconda prompt and type following:

>cd Anaconda

>where conda

>conda install pip

>conda activate curation

>code

If pip/conda/python is still not recognized, then follow the suggestions below:

1. Make sure conda is added to the path. There are several ways to do this, one of them is to add the anaconda paths to the environment variables as described here:

[python - 'Conda' is not recognized as internal or external command - Stack Overflow](https://stackoverflow.com/questions/44515769/conda-is-not-recognized-as-internal-or-external-command/51996934#51996934)

1. Update anaconda to the latest version, either by opening ‘anaconda navigator’ from the start menu or by following the instructions listed below:

[Fix: 'conda' is not recognized as an internal or external command, operable program or batch file (appuals.com)](https://appuals.com/fix-conda-is-not-recognized-as-an-internal-or-external-command-operable-program-or-batch-file/)

1. Make sure python is selected as the default interpreter on vs code: On vs code go to settings, type “Python: Select Interpreter” in the search, make sure the box in “Python: Default Interpreter Path” has “python” in it.

—------------------------------------------------------------------------------------------------------

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

**Error:**

HOKIES+padma@PADMA504404 MINGW64 ~/anaconda3/envs/curation/LD-Cool-P (master)

$ python setup.py develop

Download error on https://pypi.org/simple/: [SSL: CERTIFICATE\_VERIFY\_FAILED] certificate verify failed: unable to get local issuer certificate (\_ssl.c:1122) -- Some packages may not be found!

No local packages or working download links found for ldcoolp-figshare>=0.2.2

error: Could not find suitable distribution for Requirement.parse('ldcoolp-figshare>=0.2.2')

(curation)

—----------------------------------------------------------------------------------------------

HOKIES+padma@PADMA504404 MINGW64 ~/anaconda3/envs/curation/LD-Cool-P (master)

$ pip3 install --trusted-host pypi.org --trusted-host files.pythonhosted.org ldcoolp-figshare>=0.2.2

ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts.

ldcoolp 1.1.8 requires html2text==2020.1.16, which is not installed.

ldcoolp 1.1.8 requires jinja2==2.11.3, which is not installed.

figshare 0.3.5 requires requests==2.20.0, but you have requests 2.25.1 which is incompatible.

ldcoolp 1.1.8 requires numpy==1.21.0, but you have numpy 1.24.3 which is incompatible.

—---------------------------------------------------------------------------------

HOKIES+padma@PADMA504404 MINGW64 ~/anaconda3/envs/curation/LD-Cool-P (master)

$ pip install numpy==1.21.0

$ pip install jinja2==2.11.3

$ pip install html2text==2020.1.16

$ pip install requests==2.25.1

$ pip install requests==2.20.0

$ pip3 install --trusted-host pypi.org --trusted-host files.pythonhosted.org

$ python setup.py develop

—----------------------------------------------------------------------------------------

(Check if package is installed correctly:)

HOKIES+padma@PADMA504404 MINGW64 ~/anaconda3/envs/curation/LD-Cool-P (master)

$ conda list ldcoolp

# packages in environment at C:\Users\padma\anaconda3\envs\curation:

#

# Name Version Build Channel

ldcoolp 1.1.8 dev\_0 <develop>

ldcoolp-figshare 0.3.2 pypi\_0 pypi

(curation)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%