Hamlet.rt

Hamlet.rt Trial Processor

User Guide

Version 1.0 (29/01/22)

Raj Jena

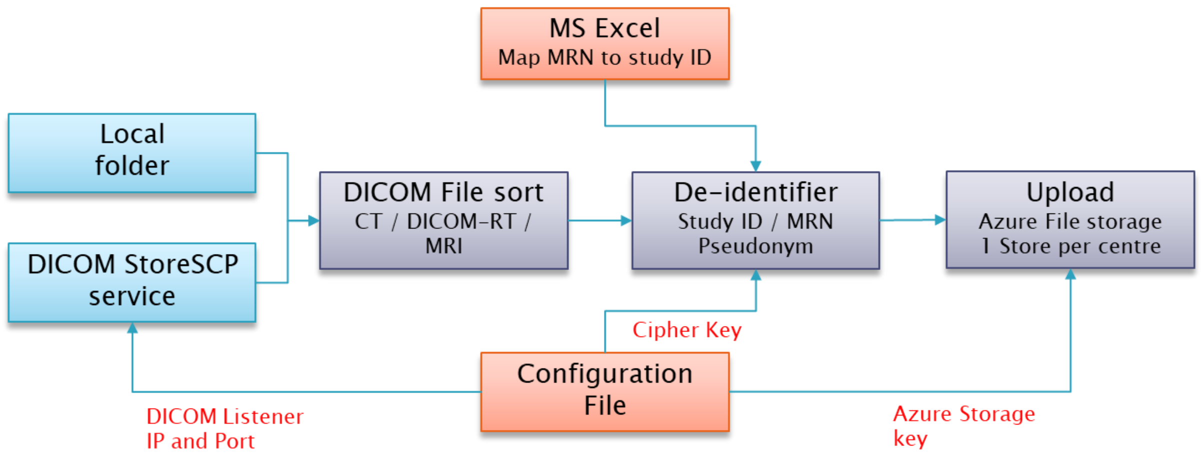
Table of Contents

Intended use

This document provides instructions for installation and use the Hamlet.rt Trial Processor (HTP) Software. The guide assumes some rudimentary working knowledge of the Python language.

Source code for the software is provided for transparency, and you are welcome to re-use any elements of the code, but distribution of the source code beyond the centres participating in the Hamlet.rt study is not permitted.

Overview of the software



The HTP software is designed to perform the following tasks interactively under the control of the user:

- Receive DICOM data into an incoming folder, either manually or via a DICOM receiver
- Sort the data into a hierarchical structure for each patient
- De-identify the data this is one by two possible methods
  - The code looks for the patient MRN in a lookup table and swaps it for the Hamlet.rt study ID
  - If the patient MRN is not listed, it uses a cipher to create a pseudonym and this is used to replace all instances of the patient MRN.
- Compress and package the data ready for upload
- Upload the data to a secure Azure storage account, unique to each participating centre

Prerequisites

HTP can be run on a Windows, Linux or Mac operating systems and requires the Python 3.8 interpreter. It is preferable to use a Python environment manager such as Miniconda to maintain a suitable environment and track dependencies.

Outbound internet access is required for use of the Azure upload functionality. The software will connect to a public internet address to send data, of the form:

```
https://hamletrtxxxxxx.file.core.windows.net
```

Where xxxxxx is the site name. The connection is made via Secure HTTP.

The DICOM receiver is a separate application provided for convenience. It does not require any outbound connection but does require that a user specified port is unblocked for any hardware or software firewall systems. In addition, a fixed IP address is preferable if the DICOM receiver is to be used.

Installation using Miniconda (recommended)

Install Miniconda from https://docs.conda.io/en/latest/miniconda.html

Open the Anaconda Prompt app (Windows) or a terminal (Mac/Linux)

Create an environment for the Hamlet trial processor called **htp** with Python 3.9:

```
conda create --name htp python=3.9
```

Activate the environment by running the following command (you will need to do this each time you wish to use the HTP software)

```
conda activate htp
```

Ensure your installation zip file (created for you by the study team) is available on this computer.

Extract the contents of the ZIP file into a suitable folder that will be accessible to the user and navigate to this folder in the terminal.

Next step is to install all dependencies using pip. As with many Python packages, there is a requirements.txt file lists all the dependencies. Type the following to install the dependencies:

```
pip install -r requirements.txt
```

You will see the package dependencies being downloaded and installed into this virtual environment. The code only needs the following packages

- `azure_storage`
- `Pillow`
- `pydicom`
- `pynetdicom`
- `PyYAML`

Installation using existing Python installation

You can of course also install the code directly to your base python installation. This is not recommended in case of any dependency issues that arise.

Ensure your installation zip file (created for you by the study team) is available on this computer.

Extract the contents of the ZIP file into a suitable folder that will be accessible to the user, and navigate to this folder using your terminal shell of choice.

Next step is to install all dependencies using pip. As with many Python packages, there is a requirements.txt file lists all the dependencies. Type the following to install the dependencies:

```
pip install -r requirements.txt
```

You will see the package dependencies being downloaded and installed into this virtual environment. The code only needs the following packages

- azure_storage
- Pillow
- pydicom
- pynetdicom
- PyYAML

Create data folders

The HTP software needs three separate folders to be created to process data. The folder structure is also important for manual verification of data for de-identification.

The **incoming** folder is the folder to which you can copy DICOM files from your data network. It is also the folder that is used by the DICOM receiver to store DICOM files via a DICOM StoreSCP service.

The **sorted** folder is the folder where data is sorted and psudonymised. Data in this folder should no longer contain patient identifiers.

The **outgoing** folder is the folder where compressed files are stored ready for upload. These files can be sent using the in-built Azure storage tools, or manually by any other agreed process.

Please create these 3 folders in an appropriate location on the computer. Here is an example from my own installation on Windows:

```
Directory of C:\data\htp

21/01/2022  11:56    <DIR>          incoming

21/01/2022  11:57    <DIR>          outgoing

21/01/2022  11:57    <DIR>          sorted
```

You will need the paths to these folders for the configuration file.

Complete the configuration file

HTP uses a YAML file in the code folder to store all necessary configuration. It is very simple. Open Config,yaml using a text editor of your choice and set the following configuration values:

APP section

> SITE – Enter the Site Name (preferably one word). This is **only** used to create archive filenames
>
> USER – Enter user's initials if you have HTP installed on more than one computer. This is only used to create archive file names.

DATAPATH section

> INCOMING – Enter path of the incoming folder you created
>
> SORTED – Enter path of the folder you created for sorted data
>
> OUTGOING– Enter path of the outgoing folder you created

Important – please ensure that the paths are terminated with a '\' for windows paths, or a '/' for Linux and Mac paths.

DICOMNODE section (only needed if you are using the DICOM receiver, otherwise leave default values)

> **AET** – You can set the name of the DICOM SToreSCP service AET here, incase you need to specify an AET for the software that is used to send data
>
> **PORT** – This is the port on which the DICOM receiver will listen for incoming DICOM data

CIPHER section

> **FOLDERKEY** – this is the cipher key that will be used to encode the patients MRN if it is not found in the lookup table. It can be a mixture of upper case characters, lower case characters and numbers. Please verify that the cipher is longer than the longest MRN in your institution. Please keep your cipher key secure.

AZURE section

> Please copy over the Azure storage settings provided to you by the PI. If you are not using Azure upload, please leave the default test account settings.

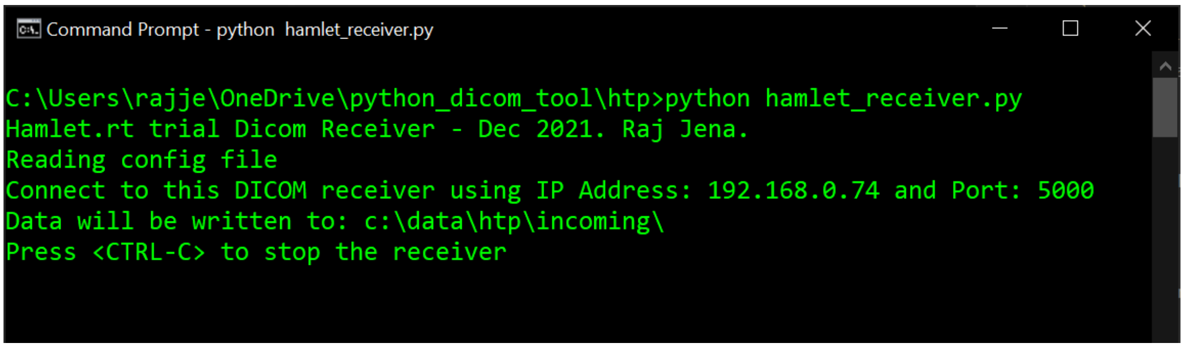
Using the provided DICOM receiver

The HTP software will look for DICOM files in the INCOMING folder and subfolders. You can either copy datafiles directly to this folder, or you can use the provided DICOM listener and push data from the listener across your data network.

To run the DICOM receiver, ensure that your config.yaml file has the correct values for your setup. Then open a terminal screen, navigate to the code folder and type:

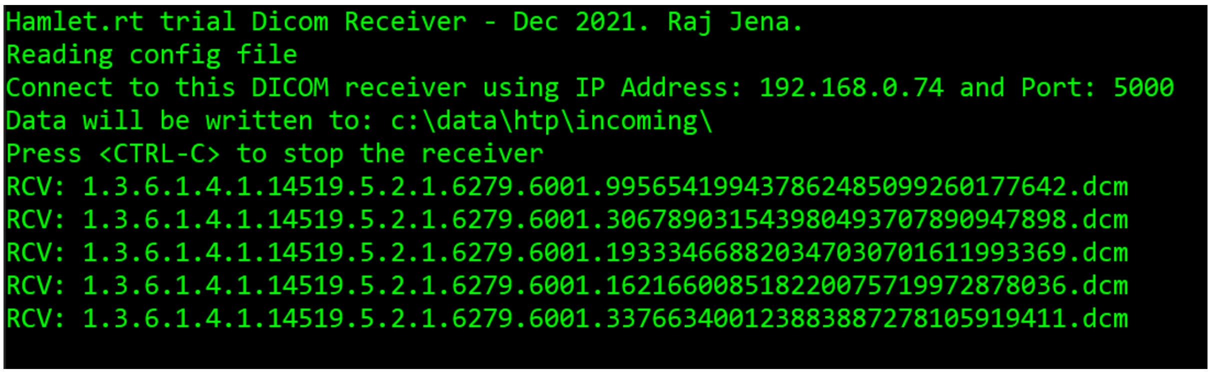
```
python hamlet_receiver.py
```

You should see output confirming the IP address and port that are being used by the listener:



```
C:\Users\rajje\OneDrive\python_dicom_tool\htp>python hamlet_receiver.py
Hamlet.rt trial Dicom Receiver - Dec 2021. Raj Jena.
Reading config file
Connect to this DICOM receiver using IP Address: 192.168.0.74 and Port: 5000
Data will be written to: c:\data\htp\incoming\
Press <CTRL-C> to stop the receiver
```

You will see confirmation of incoming files on the screen, like this:



```
Hamlet.rt trial Dicom Receiver - Dec 2021. Raj Jena.
Reading config file
Connect to this DICOM receiver using IP Address: 192.168.0.74 and Port: 5000
Data will be written to: c:\data\htp\incoming\
Press <CTRL-C> to stop the receiver
RCV: 1.3.6.1.4.1.14519.5.2.1.6279.6001.995654199437862485099260177642.dcm
RCV: 1.3.6.1.4.1.14519.5.2.1.6279.6001.306789031543980493707890947898.dcm
RCV: 1.3.6.1.4.1.14519.5.2.1.6279.6001.193334668820347030701611993369.dcm
RCV: 1.3.6.1.4.1.14519.5.2.1.6279.6001.162166008518220075719972878036.dcm
RCV: 1.3.6.1.4.1.14519.5.2.1.6279.6001.337663400123883887278105919411.dcm
```

To exit the receiver press <CTRL-C>.

If you navigate to the INCOMING folder, you will be able to see the received DICOM files waiting for processing.

Although it is possible to run the receiver and the HTP software at the same time (in two different terminal shells) it is not recommended.

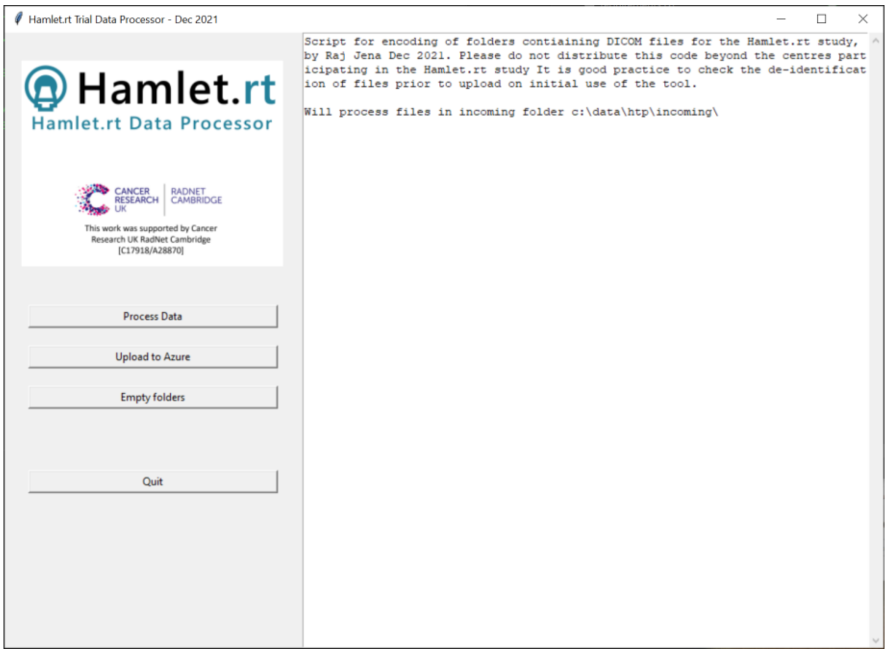
Running the HTP software for the first time (local processing only)

It is recommended you try out the software and check functionality before uploading any data. A sample dataset containing 10 DICOM files is available from the study team. Please download the zip file and extract the contents into your incoming folder or use test files of your own.

Now navigate to the folder containing the HTP software, open a terminal and run

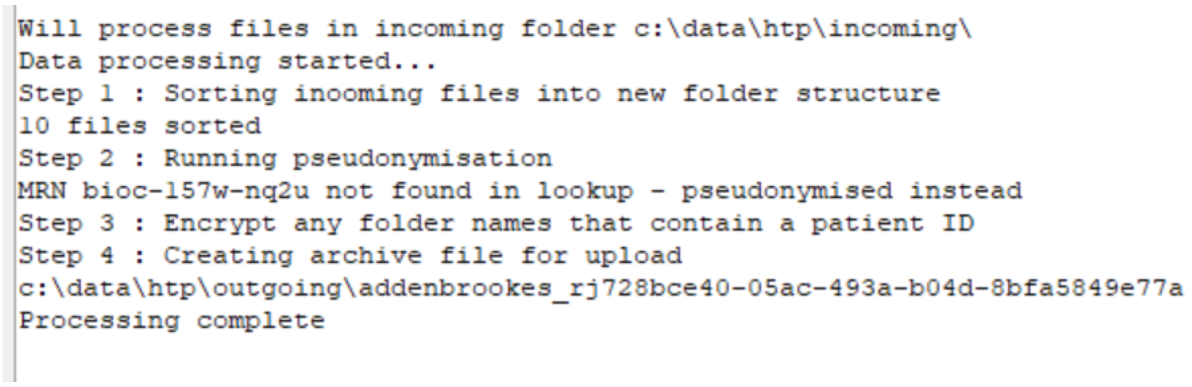
```
        python hamlet_processor_v6.py
```

After a brief delay, you should see a simple GUI window appear:


Hamlet.rt Trial Data Processor - Dec 2021
Hamlet.rt
Hamlet.rt Data Processor
CANCER RESEARCH UK
RADNET CAMBRIDGE
This work was supported by Cancer Research UK RadNet Cambridge [C17918/A28870]
Process Data
Upload to Azure
Empty folders
Quit

```
Script for encoding of folders contiaining DICOM files for the Hamlet.rt study,
by Raj Jena Dec 2021. Please do not distribute this code beyond the centres part
icipating in the Hamlet.rt study It is good practice to check the de-identificat
ion of files prior to upload on initial use of the tool.

Will process files in incoming folder c:\data\htp\incoming\
```

Now click the process data button and look at the output on the text panel. It should show something like this:



```
Will process files in incoming folder c:\data\htp\incoming\
Data processing started...
Step 1 : Sorting inooming files into new folder structure
10 files sorted
Step 2 : Running pseudonymisation
MRN bioc-157w-nq2u not found in lookup - pseudonymised instead
Step 3 : Encrypt any folder names that contain a patient ID
Step 4 : Creating archive file for upload
c:\data\htp\outgoing\addenbrookes_rj728bce40-05ac-493a-b04d-8bfa5849e77a
Processing complete
```

The steps explain what the HTP software is doing:

- It creates its own folder structure from the data.
- Then it works through each file, finds the patient MRN, and checks it against the lookup in the Excel CSV file
- If it finds a match, it will automatically replace the MRN with the study ID and anonymise all other tags.

- If it does not find a match (as in this case) it uses the cipher function to pseudonymise the MRN.
- The code checks that the MRN does not appear in a folder name
- Finally, an archive ZIP file is created for upload

Please check the contents of the incoming, sorted and outgoing folders to verify the software's operation.

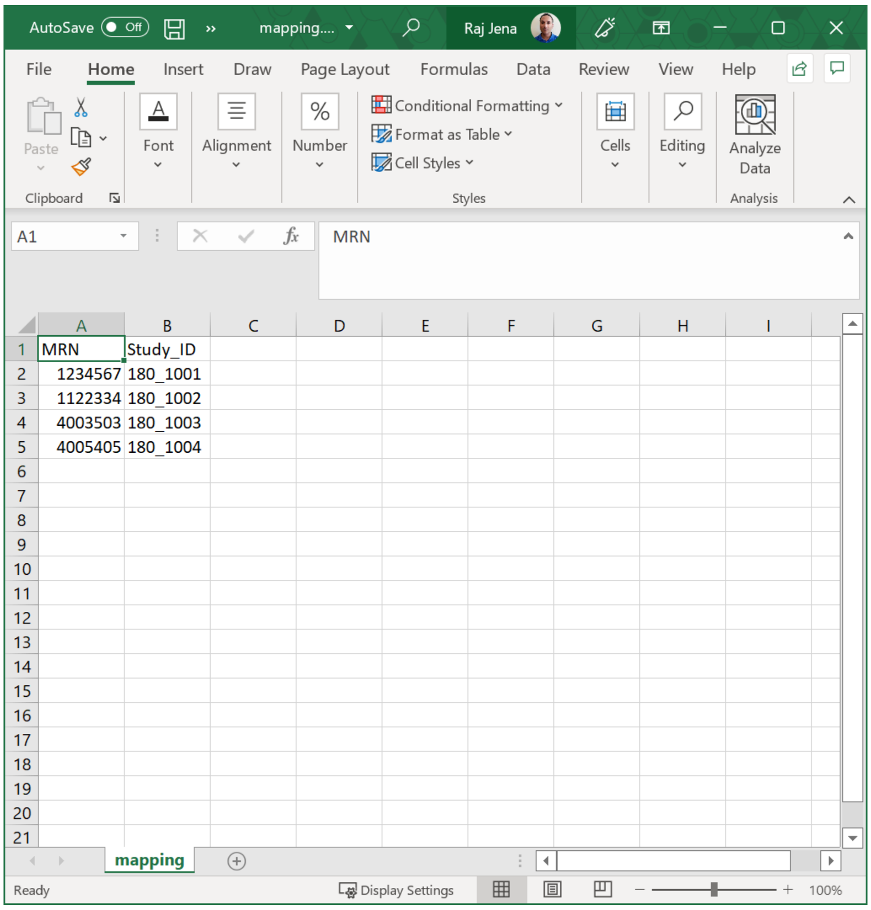
Now click the Empty folders button to clear all the folders. Note that the archive files are not removed from the outgoing folder. This must be done manually, because end-users may choose to use another option to transfer the compressed archive files.

Creating the Study ID lookup table

The lookup table is simply a comma separated variable file, where each line contains the local MRN and the corresponding study ID. It is stored in a file called mapping.csv in the installation folder. You can use Excel or a text editor of your choice to edit this file.

Note that this file provides a lookup to overcome pseudonymisation of study data. It should not be shared with any other centre.

Here is an example of the lookup table:



MRN	Study_ID
1234567	180_1001
1122334	180_1002
4003503	180_1003
4005405	180_1004

Simple add rows to this file as you accrue study patients. Note that if you attempt to process data from a patient where the MRN and study ID are not listed in the lookup, then the processor will create a pseudonym based on the patient's MRN. A Vigenere cipher is used for this purpose, using a cipher key specified in the `config.yaml` file. Note however that this is not a strong form of encryption – it simply to ensure that the automated processor would never transmit an MRN over a public internet connection.

Running the HTP software for data upload

Let us review the full data processing workflow, including azure upload. I will use the same sample data for this example, and I will be uploading data to

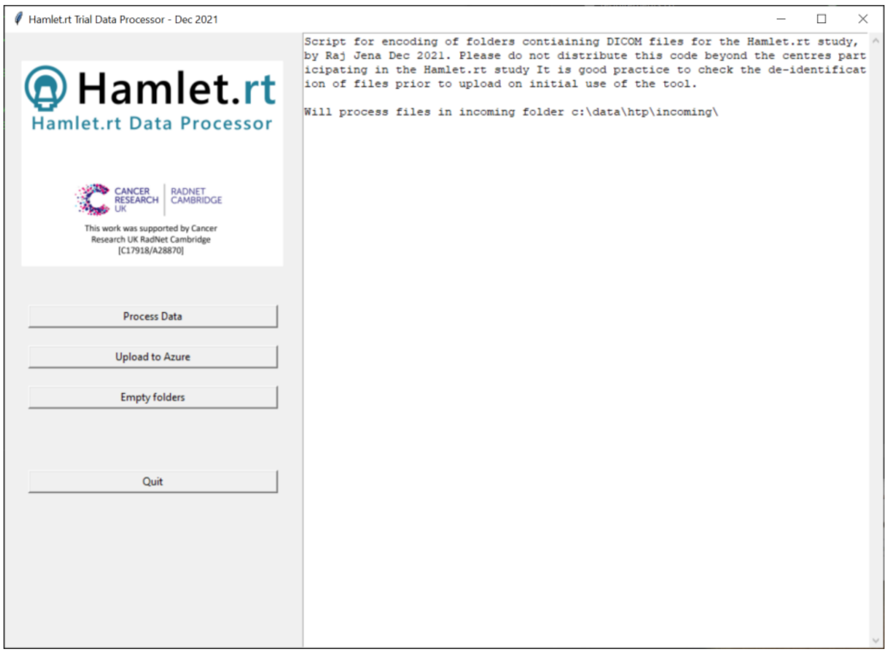
```
https://hamletrtcambridge.file.core.windows.net
```

And I will use the folder **`cambridge-incoming`** for data storage. These values are stored in the **`config.yaml`** file on my computer.

As before, navigate to the folder containing the HTP software, open a terminal and run

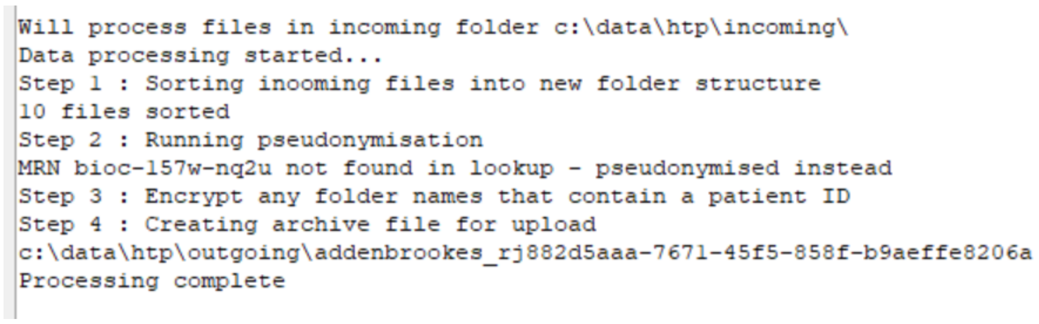
```
python hamlet_processor_v6.py
```

After a brief delay, you should see a simple GUI window appear:


Hamlet.rt Trial Data Processor - Dec 2021
Hamlet.rt
Hamlet.rt Data Processor
CANCER RESEARCH UK
RADNET CAMBRIDGE
This work was supported by Cancer Research UK RadNet Cambridge [C17918/A28870]
Process Data
Upload to Azure
Empty folders
Quit

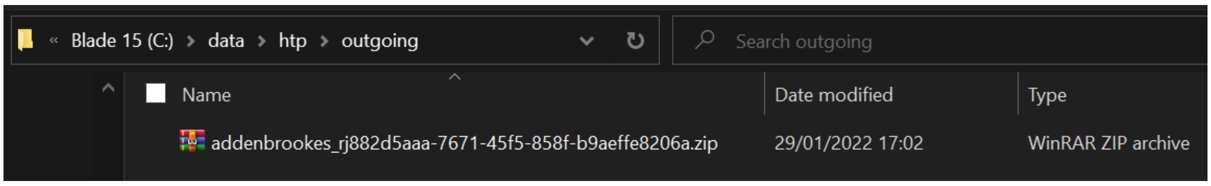
```
Script for encoding of folders contiaining DICOM files for the Hamlet.rt study,
by Raj Jena Dec 2021. Please do not distribute this code beyond the centres part
icipating in the Hamlet.rt study It is good practice to check the de-identificat
ion of files prior to upload on initial use of the tool.

Will process files in incoming folder c:\data\htp\incoming\
```

Now click the process data button and look at the output on the text panel. On my computer it looks like this:



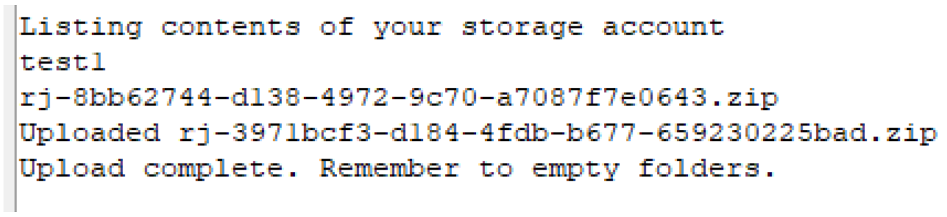
```
Will process files in incoming folder c:\data\htp\incoming\
Data processing started...
Step 1 : Sorting inooming files into new folder structure
10 files sorted
Step 2 : Running pseudonymisation
MRN bioc-157w-nq2u not found in lookup - pseudonymised instead
Step 3 : Encrypt any folder names that contain a patient ID
Step 4 : Creating archive file for upload
c:\data\htp\outgoing\addenbrookes_rj882d5aaa-7671-45f5-858f-b9aeffe8206a
Processing complete
```

Note the outgoing filename – it is made up the site name, the user initials, and a unique hash key. You can check the file is in the outgoing folder:



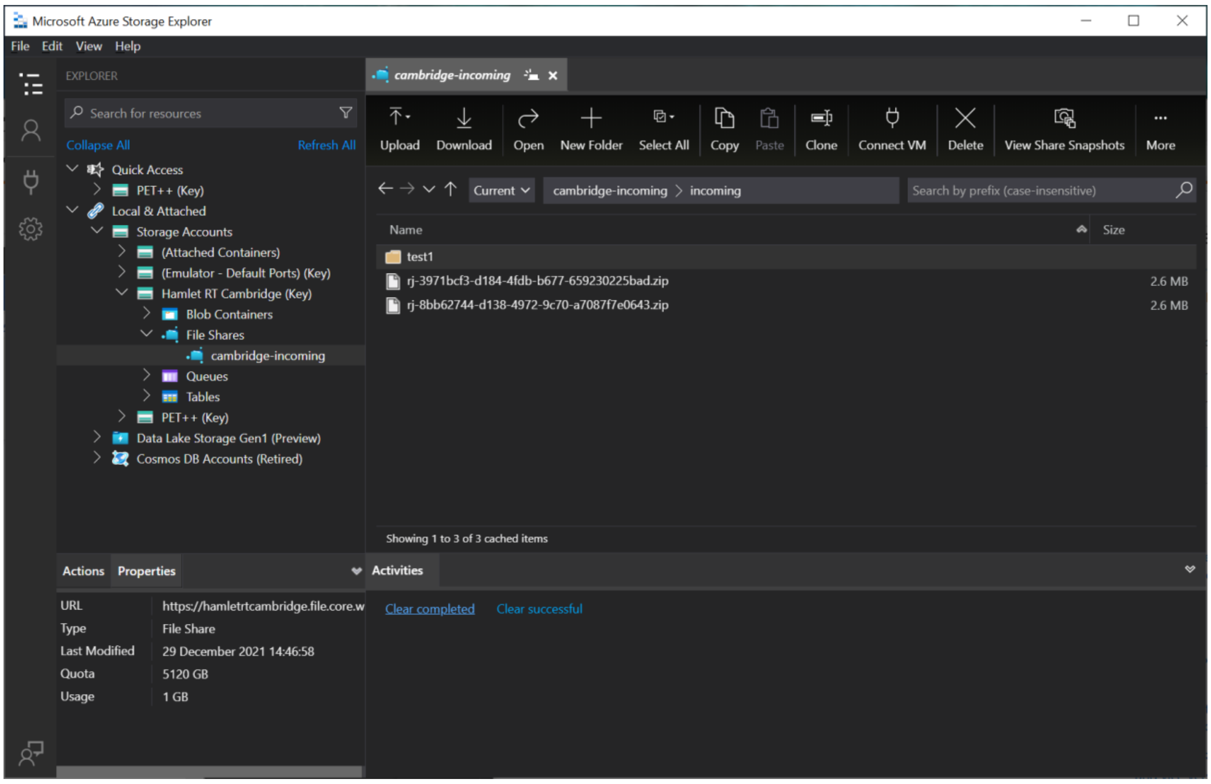
Name	Date modified	Type
addenbrookes_rj882d5aaa-7671-45f5-858f-b9aeffe8206a.zip	29/01/2022 17:02	WinRAR ZIP archive

Now press Upload to Azure, and this file will be uploaded to you Azure storage account. There will be a slight delay whilst the connection is made. When the file is uploaded, you should see something like this:



```
Listing contents of your storage account
test1
rj-8bb62744-d138-4972-9c70-a7087f7e0643.zip
Uploaded rj-3971bcf3-d184-4fdb-b677-659230225bad.zip
Upload complete. Remember to empty folders.
```

I am now going to use Azure Storage Explorer to verify the upload. See the data transfer guide for more details on how to set up and use the application.



Name	Size
test1	
rj-3971bcf3-d184-4fdb-b677-659230225bad.zip	2.6 MB
rj-8bb62744-d138-4972-9c70-a7087f7e0643.zip	2.6 MB

Showing 1 to 3 of 3 cached items

URL	https://hamletrtcambridge.file.core.w
Type	File Share
Last Modified	29 December 2021 14:46:58
Quota	5120 GB
Usage	1 GB

By navigating to the incoming folder in the Cambridge Incoming storage account, I can now verify the upload.

The final step is to click the **empty folder** button which will clear the incoming and sorted folders. This is important because if you don't clear the folders, data from the previous upload will be included in subsequent uploads.

Questions and support

If you have any questions about any of the options listed for data transfer, please contact Dr Raj Jena via email at rjena@nhs.net