# RT-HaND\_I SOP for downloading data from the data lake

## Introduction

Due to the need to store unanonymised patients within the data lake to enable harmonisation of patient records, imaging and treatments during future data ingestions, data must be anonymised before it is used by researchers.

XNAT is unable to anonymise scans itself that it already holds in unanonymised form in the way they have been labelled within the RT-HaND\_I lake. Export to a separate location and reimport is required.

In addition to this, because XNAT ingests ***ALL*** DICOM data associated with a particular imaging session, it is necessary to clean the surplus to requirements data from each patient before they are given to the researchers (data minimisation).

Researchers should have been granted permission from the XRAP committee and specified the data they require as well as any anonymisation requirements. A copy of the anonymisation key should be kept with the project paperwork for future reference.

A brief description of the required pipeline is shown below.

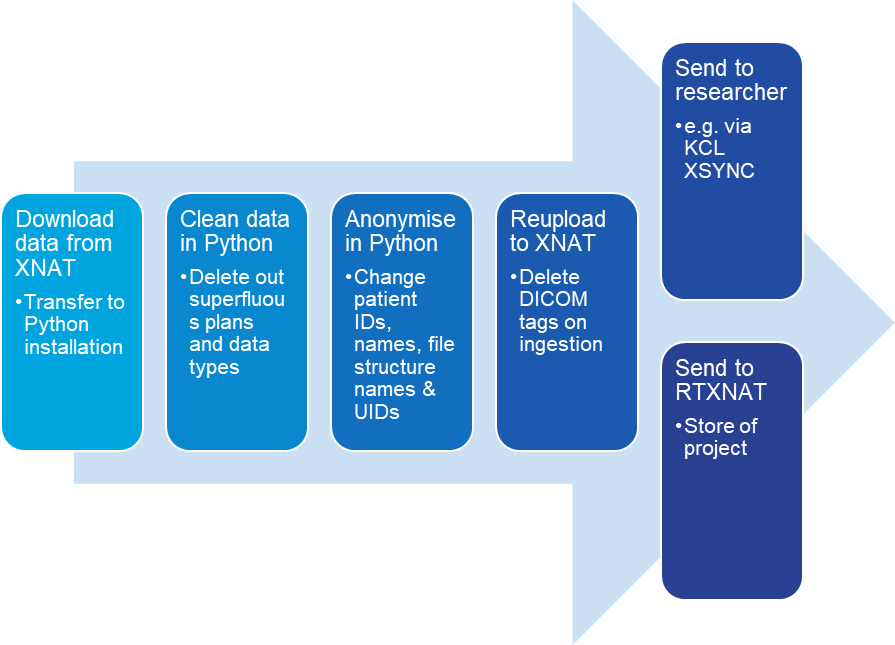


Figure 1: Steps of data download and provision for research.

## Practicalities

### Downloading the data

Some avenues of data transfer within the trust are much faster than others and some are only practicable onsite (e.g. large data transfer (GBs) is too slow from the shared drive to the clinical drive remotely). The data needs to make its way to Citrix10 in order to be manipulated in Python. There is approximately 46GB of spare space on Citrix10 and so care needs to be taken to not copy and manipulate too much data simultaneously. The clinical drive has TBs of spare storage so should be used as the intermediary store. All data should be deleted from the clinical drive and Citrix10 when it has been reuploaded to an instance of XNAT.

There are 2 tested methods of data transfer employed from the RT-HaND\_I lake:

1. Provide CSC a list of all the scans required. This can be created by downloading the name of all the scans available in RT-HaND\_I as per the instructions in RT-HanD 012: Uploading from PACs- Technical Guide and then sorting and filtering for those required. CSC can then zip the scans in batches of 10 and they can be copied from the headnode to the shared drive. Unfortunately, they cannot be copied directly to the clinical drive and so getting these scans to Citrix10 requires the additional (and non-remote) step of shared drive to clinical drive data transfer. Additionally, each Trust user only has the rights to place 14GB of data on the shared drive at any point.
2. Refine search criteria using the XNAT Advanced Search Method, build a pivot query to advanced search based on the subjects and sessions of interest and then under “options” there is a “download” button which enables sets of scans to be downloaded at ~10GB at a time. The download location can be mapped to the clinical drive and from the clinical drive they can be copied to easily be manipulated in Citrix10 Python Installation.

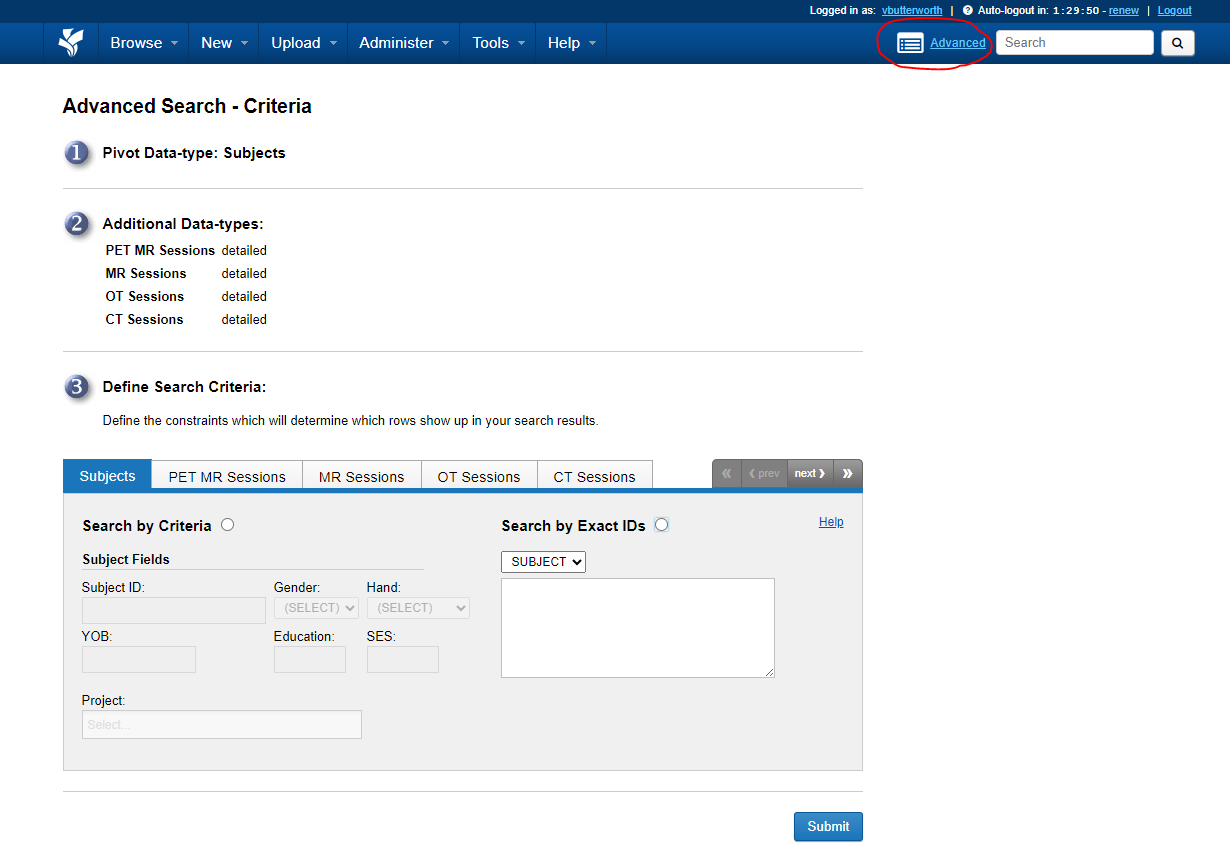


Figure 1: Advanced Search builder

### **Cleaning the data**

XNAT ingests all dicom data associated with a particular imaging session with no discrimination. Monaco datasets are relatively clean because mostly only contain 1 RTPlan, 1 RTDose, 1 CT and 1 RTSS. Eclipse datasets are messier however. Examples of surplus to requirement data within Eclipse patients include verification plan RTPlan and RTDose files, RTImages from DDRs, CBCTs for projects not requiring CBCT data, RTRecords for projects not requiring RT delivery information and RTPlans, structure sets (RTSS) and RTDose files for non-clinical treatment plans, for example training plans or CBCT assessments.

Currently, a v0.1 script exists which identifies plans and copies the corresponding CT, RTSS and dose file. This enables immediate deletion of obviously non-clinical plans. This does not work for the 165 incorrectly reconstructed Monaco patients who do not have an associated dummy treatment field BUT these patients should not have any surplus data anyway. An extension to the script is planned which would use a look-up table of clinical plan names per patient to enable autodeletion of non-clinical plans. These clinical plan names are stored within the explanatory .csv file given with radiotherapy patients. The script in its current form is in Appendix 1 and stored on the RTPHYS GitHub.

### **Anonymising the data**

Depending on the path the data is taking, different anonymisation requirements will need to be met and different methods will need to be used to achieve these requirements. All cleaned and anonymised datasets should be stored within RTXNAT for future use.

The 4 anonymisation requirements for working with data from XNAT are as follows:

1. Rename patient ID and name as per the project identification key.
2. Rename XNAT file structure to strip out all patient ID information so that XNAT does not “burn” patient identifiable information into the session IDs when reuploaded.
3. Anonymise UIDs (for projects where patients will be put back into Eclipse).
4. Strip surplus to requirement DICOM tags.

Currently steps 1-3 would be performed using Python and step 4 would be performed upon reupload to XNAT using the CSC’s custom script given in Appendix 3. A script for renaming patient IDs as per a project identification key is given in Appendix 2 and a script for anonymising UIDs is currently being written.

### **Future work**

Ideally, this data cleaning and anonymising would take place on the XNAT server itself to minimise many GB of data being transferred (some steps impossible when NOT physically on site at CTC or QMH) but would require understanding of the head-node XNAT python set-up and require administrative rights. This might be a useful undertaking when a script covering every eventuality has been refined.

The current data cleaning script will need to be updated for projects requiring CBCTs or registered pre-treatment imaging.

## Appendix 1: Script to sort plans by name

#this script does an okay job at creating folders of plan, dose cube, structure set and CT based on the plan name. It does a good job of removing extraneous CBCTs and irrelevant dose files. It does not work for the Monaco patients who were reconstructed incorrectly and as such do not have a plan with a field associated with them however these patients should not have large swathes of surplus dicom data.

import os

import shutil

import pydicom

def extract\_uids\_and\_plan\_name\_from\_rt\_plan(plan\_file\_path):

try:

dicom\_file = pydicom.dcmread(plan\_file\_path, stop\_before\_pixels=True)

# Extract the RT Plan Name

plan\_name = dicom\_file.get((0x300a, 0x0002), None).value if dicom\_file.get((0x300a, 0x0002), None) else "UnknownPlan"

# Extract the Study Instance UID

study\_instance\_uid = dicom\_file.get("StudyInstanceUID", None)

# Extract the Referenced Structure Set UID

referenced\_structure\_set\_uid = None

if hasattr(dicom\_file, "ReferencedStructureSetSequence"):

referenced\_structure\_set\_uid = dicom\_file.ReferencedStructureSetSequence[0].ReferencedSOPInstanceUID

# Extract the Patient ID

patient\_id = dicom\_file.get("PatientID", "UnknownPatient")

return plan\_name, study\_instance\_uid, referenced\_structure\_set\_uid, patient\_id

except Exception as e:

print(f"Error reading RT Plan file: {e}")

return None, None, None, None

def find\_dicom\_files(root\_dir, study\_instance\_uid=None, modality=None, sop\_instance\_uid=None):

found\_files = []

for dirpath, \_, filenames in os.walk(root\_dir):

for filename in filenames:

file\_path = os.path.join(dirpath, filename)

try:

dicom\_file = pydicom.dcmread(file\_path, stop\_before\_pixels=True)

if modality and dicom\_file.Modality != modality:

continue

if study\_instance\_uid and dicom\_file.StudyInstanceUID != study\_instance\_uid:

continue

if sop\_instance\_uid and dicom\_file.SOPInstanceUID != sop\_instance\_uid:

continue

found\_files.append(file\_path)

except Exception as e:

print(f"Error reading file {file\_path}: {e}")

return found\_files

def copy\_and\_rename\_files(files, destination\_dir, new\_filename\_prefix=None):

if not os.path.exists(destination\_dir):

os.makedirs(destination\_dir)

for file\_path in files:

# Determine the new filename if a prefix is provided

if new\_filename\_prefix:

new\_filename = f"{new\_filename\_prefix}.dcm"

destination\_path = os.path.join(destination\_dir, new\_filename)

else:

destination\_path = os.path.join(destination\_dir, os.path.basename(file\_path))

shutil.copy2(file\_path, destination\_path)

print(f"Copied {file\_path} to {destination\_path}")

def main():

root\_dir = input("Enter the root directory to search for RT Plan files: ")

destination\_root = input("Enter the root destination directory for the copied files: ")

for dirpath, \_, filenames in os.walk(root\_dir):

for filename in filenames:

file\_path = os.path.join(dirpath, filename)

try:

dicom\_file = pydicom.dcmread(file\_path, stop\_before\_pixels=True)

if dicom\_file.Modality == "RTPLAN":

print(f"\nProcessing RT Plan file: {file\_path}")

plan\_name, study\_instance\_uid, referenced\_structure\_set\_uid, patient\_id = extract\_uids\_and\_plan\_name\_from\_rt\_plan(file\_path)

if not study\_instance\_uid:

print("Critical UIDs are missing, skipping this plan.")

continue

# Create destination directory for this plan with both Patient ID and Plan Name

plan\_destination\_dir = os.path.join(destination\_root, f"{patient\_id}\_{plan\_name}")

os.makedirs(plan\_destination\_dir, exist\_ok=True)

# Copy the RT Plan file

copy\_and\_rename\_files([file\_path], plan\_destination\_dir)

# Find and copy the CT files

ct\_files = find\_dicom\_files(root\_dir, study\_instance\_uid=study\_instance\_uid, modality="CT")

if ct\_files:

copy\_and\_rename\_files(ct\_files, plan\_destination\_dir)

else:

print("No CT files found.")

# Find and copy the Dose file

dose\_files = find\_dicom\_files(root\_dir, study\_instance\_uid=study\_instance\_uid, modality="RTDOSE")

if dose\_files:

copy\_and\_rename\_files(dose\_files, plan\_destination\_dir)

else:

print("No Dose files found.")

# Find and copy the Structure Set file, and rename it

if referenced\_structure\_set\_uid:

structure\_set\_file\_path = find\_dicom\_files(root\_dir, sop\_instance\_uid=referenced\_structure\_set\_uid, modality="RTSTRUCT")

if structure\_set\_file\_path:

structure\_set\_name = dicom\_file.get((0x3006, 0x0002), None).value if dicom\_file.get((0x3006, 0x0002), None) else "UnknownStruct"

new\_filename\_prefix = f"{patient\_id}\_{plan\_name}\_{structure\_set\_name}"

copy\_and\_rename\_files(structure\_set\_file\_path, plan\_destination\_dir, new\_filename\_prefix)

else:

print("No Structure Set files found.")

except Exception as e:

print(f"Error processing file {file\_path}: {e}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

## Appendix 2: Manipulating dicom tags with known values from .csv file e.g. changing patient ID & name and structure set label.

#This code manipulates patient name and ID from a .csv input with the format: old\_patient\_id, new\_patient\_id, new\_patient\_name and new\_patient\_sex. This code was successfully used to rename and reinput the genders for patients for the Xerostomia project.

import os

import pydicom

import csv

from datetime import datetime

def load\_patient\_data\_from\_csv(csv\_file):

"""Load patient data from the CSV file into a dictionary."""

patient\_data = {}

with open(csv\_file, newline='') as csvfile:

csv\_reader = csv.DictReader(csvfile)

for row in csv\_reader:

old\_patient\_id = row['old\_patient\_id']

patient\_data[old\_patient\_id] = {

'new\_patient\_id': row['new\_patient\_id'],

'new\_patient\_name': row['new\_patient\_name'],

'new\_patient\_sex': row['new\_patient\_sex']

}

return patient\_data

def modify\_dicom\_for\_patient(dicom\_data, new\_patient\_id, new\_patient\_name, new\_patient\_sex, new\_structure\_set\_name,

new\_structure\_set\_label, operator\_name="Unknown"):

"""Modify DICOM metadata for the given patient."""

# Modify the relevant tags

dicom\_data.PatientID = new\_patient\_id

dicom\_data.PatientName = new\_patient\_name

dicom\_data.PatientSex = new\_patient\_sex

# Check if the file is an RT Structure Set

if dicom\_data.Modality == "RTSTRUCT":

print(f"Processing RT Structure Set for new PatientID: {new\_patient\_id}")

# Add or modify StructureSetName (3006,0002)

dicom\_data.StructureSetName = new\_structure\_set\_name

# Add or modify StructureSetLabel (3006,0002)

dicom\_data.StructureSetLabel = new\_structure\_set\_label

# Add Operators' Name (0008,1070) if missing

if not hasattr(dicom\_data, 'OperatorsName'):

dicom\_data.OperatorsName = operator\_name

# Add Structure Set Date (3006,0008) and Time (3006,0009)

current\_date = datetime.now().strftime("%Y%m%d")

current\_time = datetime.now().strftime("%H%M%S")

if not hasattr(dicom\_data, 'StructureSetDate'):

dicom\_data.StructureSetDate = current\_date

if not hasattr(dicom\_data, 'StructureSetTime'):

dicom\_data.StructureSetTime = current\_time

def process\_dicom\_files(input\_folder, output\_folder, patient\_data, new\_structure\_set\_name, new\_structure\_set\_label,

operator\_name="Unknown"):

"""Process DICOM files in the folder and modify them based on the patient data."""

for root, dirs, files in os.walk(input\_folder):

for file in files:

# Check if the file is a DICOM file

if file.endswith(".dcm"):

dicom\_path = os.path.join(root, file)

# Read the DICOM file

dicom\_data = pydicom.dcmread(dicom\_path)

old\_patient\_id = dicom\_data.PatientID

# Check if the current PatientID is in the patient\_data dictionary

if old\_patient\_id in patient\_data:

patient\_info = patient\_data[old\_patient\_id]

new\_patient\_id = patient\_info['new\_patient\_id']

new\_patient\_name = patient\_info['new\_patient\_name']

new\_patient\_sex = patient\_info['new\_patient\_sex']

print(f"Modifying DICOM for PatientID: {old\_patient\_id} -> {new\_patient\_id}")

# Modify the DICOM data

modify\_dicom\_for\_patient(dicom\_data, new\_patient\_id, new\_patient\_name, new\_patient\_sex,

new\_structure\_set\_name, new\_structure\_set\_label, operator\_name)

# Create the patient's folder inside the output directory

patient\_output\_folder = os.path.join(output\_folder, new\_patient\_id)

if not os.path.exists(patient\_output\_folder):

os.makedirs(patient\_output\_folder)

# Save the modified DICOM file in the new patient's folder

output\_dicom\_path = os.path.join(patient\_output\_folder, f"{new\_patient\_id}\_{file}")

dicom\_data.save\_as(output\_dicom\_path)

print(

f"Saved modified file for PatientID: {old\_patient\_id} as {new\_patient\_id} in {patient\_output\_folder}")

# Example usage

input\_folder = "XERO/XERO6"

output\_folder = "XERO/XERO6EDITED" # Parent output folder where each patient folder will be created

csv\_file = "XERO/newid\_sex.csv"

new\_structure\_set\_name = "StructureSet"

new\_structure\_set\_label = "StructureSet"

operator\_name = "DEFAULT\_OPERATOR" # Optional: Set the operator's name

# Load patient data from the CSV file

patient\_data = load\_patient\_data\_from\_csv(csv\_file)

# Process the DICOM files based on the loaded patient data

process\_dicom\_files(input\_folder, output\_folder, patient\_data, new\_structure\_set\_name, new\_structure\_set\_label, operator\_name)

## Appendix 3: RT-specific XNAT project ingestion anonymisation script for patients assuming pre-altered names and IDs.

// GSTT XNAT Anonymisation/De-identification Script

// Copyright (c) 2024, Guys and St Thomas NHS Trust

// All Rights Reserved

// This script was written in September 2023 2020 to suit the needs of GSTT and conforms to the DICOM Supplement 142

// It aims to de-identify all DICOM data uploaded to XNAT.

// It retains patients age, size, weight, ethnic group, smoking status, pregnancy status and sex neutered status.

// The manufacturer-specific DICOM data is not altered by this script.

version "6.5"

retainPrivateTags["(0101,{TEST}9B)"] // Remove all private tags except those in the TEST block of group 0101.

// USER-DEFINED VARIABLES

NUMBEROFDAYSTOBESHIFTED := "11" // Set number of days for dates to be shifted (this should be updated between projects)

NUMBEROFSECONDSTOBESHIFTED := "950400" // Set number of seconds for datetimes to be shifted based on NUMBEROFDAYSTOBESHIFTED (this should be updated between projects)

// TAGS TO BE ASSIGNED PROJECT-SPECIFIC VALUES

(0008,1030) := project // Set Study Description as XNAT Project Name

(0008,0050) := session // Set Accession Number as Session ID given at DQR

(0020,0010) := session // Set Study ID to Session ID

// BELOW TAGS ARE COMMENTED OUT FOR PROJECT SINCE THESE HAVE ALREADY BEEN SET WITH pydicom

//(0010,0010) := subject // Set Patient Name as Subject ID given at DQR

//(0010,0020) := subject // Set Patient ID as Subject ID given at DQR

// TAGS TO BE REMOVED

//-(0000,1000) // Remove Affected SOP Instance UID

-(0008,0012) // Remove Instance Creation Date

-(0008,0013) // Remove Instance Creation Time

-(0008,002A) // Remove Acquisition Datetime

-(0008,0024) // Remove Overlay Date

-(0008,0025) // Remove Curve Date

-(0008,0031) // Remove Series Time

-(0008,0032) // Remove Acquisition Time

-(0008,0034) // Remove Overlay Time

-(0008,0035) // Remove Curve Time

-(0008,0080) // Remove Institution Name

-(0008,0081) // Remove Institution Address

-(0008,0082) // Remove Institution Code Sequence

-(0008,0092) // Remove Referring Physicians Address

-(0008,0094) // Remove Referring Physicians Telephone Numbers

-(0008,0096) // Remove Referring Physicians Identification Sequence

-(0008,0201) // Remove Timezone Offset From UTC

-(0008,1010) // Remove Station Name

-(0008,1048) // Remove Physician(s) of Record

-(0008,1049) // Remove Physician(s) of Record Identification Sequence

-(0008,1050) // Remove Performing Physicians Name

-(0008,1052) // Remove Performing Physicians Identification Sequence

-(0008,1060) // Remove Name of Physician(s) Reading Study

-(0008,1062) // Remove Physician Reading Study Identification Sequence

-(0008,1070) // Remove Operators Name

-(0008,1072) // Remove Operators Identification Sequence

-(0008,1080) // Remove Admitting Diagnoses Description

-(0008,1084) // Remove Admitting Diagnoses Code Sequence

-(0008,1111) // Remove Referenced Performed Procedure Step Sequence

-(0008,1120) // Remove Referenced Patient Sequence

-(0008,2111) // Remove Derivation Description

-(0008,2112) // Remove Source Image Sequence

-(0008,4000) // Remove Identifying Comments

-(0010,0021) // Remove Issuer of Patient ID

-(0010,0032) // Remove Patients Birth Time

-(0010,0050) // Remove Patients Insurance Plan Code Sequence

-(0010,0101) // Remove Patients Primary Language Code Sequence

-(0010,0102) // Remove Patients Primary Language Modifier Code Sequence

-(0010,1000) // Remove Other Patient IDs

-(0010,1001) // Remove Other Patient Names

-(0010,1002) // Remove Other Patient IDs Sequence

-(0010,1005) // Remove Patients Birth Name

-(0010,1040) // Remove Patient Address

-(0010,1050) // Remove Insurance Plan Identification

-(0010,1060) // Remove Patients Mothers Birth Name

-(0010,1080) // Remove Military Rank

-(0010,1081) // Remove Branch of Service

-(0010,1090) // Remove Medical Record Locator

-(0010,2000) // Remove Medical Alerts

-(0010,2110) // Remove Allergies

-(0010,2150) // Remove Country of Residence

-(0010,2152) // Remove Region of Residence

-(0010,2154) // Remove Patients Telephone Number

-(0010,2180) // Remove Occupation

-(0010,21B0) // Remove Additional Patients History

-(0010,21D0) // Remove Last Menstrual Date

-(0010,21F0) // Remove Patients Religious Preference

-(0010,2297) // Remove Responsible Person

-(0010,2299) // Remove Responsible Organization

-(0010,4000) // Remove Patient Comments

-(0018,1000) // Remove Device Serial Number

-(0018,1004) // Remove Plate ID

-(0018,1005) // Remove Generator ID

-(0018,1007) // Remove Cassette ID

-(0018,1008) // Remove Gantry ID

-(0018,1030) // Remove Protocol Name

-(0018,1200) // Remove Date of Last Calibration

-(0018,1400) // Remove Acquisition Device Processing Description

-(0018,4000) // Remove Acquisition Comments

-(0018,7006) // Remove Detector Description

-(0018,700A) // Remove Detector ID

-(0018,700C) // Remove Date of Last Detector Calibration

-(0018,9424) // Remove Acquisition Protocol Description

-(0018,A001) // Remove Contribution Equipment Sequence Attributes

-(0018,A003) // Remove Contribution Description

-(0020,3401) // Remove Modifying Device ID

-(0020,3404) // Remove Modifying Device Manufacturer

-(0020,3406) // Remove Modified Image Description

-(0020,4000) // Remove Image Comments

-(0020,9158) // Remove Frame Comments

-(0028,4000) // Remove Image Presentation Comments

//-(0029,1009) // Remove Siemens-specific tags

//-(0029,1019) // Remove Siemens-specific tags

-(0032,0012) // Remove Study ID Issuer

-(0032,1020) // Remove Scheduled Study Location

-(0032,1021) // Remove Scheduled Study Location AE Title

-(0032,1030) // Remove Reason for Study

-(0032,1032) // Remove Requesting Physician

-(0032,1033) // Remove Requesting Service

-(0032,1060) // Remove Requested Procedure Description

-(0032,1070) // Remove Requested Contrast Agent

-(0032,4000) // Remove Study Comments

-(0038,0010) // Remove Admission ID

-(0038,0011) // Remove Issuer of Admission ID

-(0038,001E) // Remove Scheduled Patient Institution Residence

-(0038,0020) // Remove Admitting Date

-(0038,0021) // Remove Admitting Time

-(0038,0040) // Remove Discharge Diagnosis Description

-(0038,0050) // Remove Special Needs

-(0038,0060) // Remove Service Episode ID

-(0038,0061) // Remove Issuer of Service Episode ID

-(0038,0062) // Remove Service Episode Description

-(0038,0300) // Remove Current Patient Location

-(0038,0400) // Remove Patients Institution Residence

-(0038,0500) // Remove Patient State

-(0038,1234) // Remove Referenced Patient Alias Sequence

-(0038,4000) // Remove Visit Comments

-(0040,0001) // Remove Scheduled Station AE Title

-(0040,0002) // Remove Scheduled Procedure Step Start Date

-(0040,0003) // Remove Scheduled Procedure Step Start Time

-(0040,0004) // Remove Scheduled Procedure Step End Date

-(0040,0005) // Remove Scheduled Procedure Step End Time

-(0040,0006) // Remove Scheduled Performing Physician Name

-(0040,0007) // Remove Scheduled Procedure Step Description

-(0040,000B) // Remove Scheduled Performing Physician Identification Sequence

-(0040,0010) // Remove Scheduled Station Name

-(0040,0011) // Remove Scheduled Procedure Step Location

-(0040,0012) // Remove Pre-Medication

-(0040,0241) // Remove Performed Station AE Title

-(0040,0242) // Remove Performed Station Name

-(0040,0243) // Remove Performed Location

-(0040,0244) // Remove Performed Procedure Step Date

-(0040,0245) // Remove Performed Procedure Step Start Time

-(0040,0248) // Remove Performed Station Name Code Sequence

-(0040,0253) // Remove Performed Procedure Step ID

-(0040,0254) // Remove Performed Procedure Step Description

-(0040,0275) // Remove Request Attribute Sequence

-(0040,0280) // Remove Comments on Performed Procedure Step

-(0040,0555) // Remove Acquisition Context Sequence

-(0040,1004) // Remove Patient Transport Arrangements

-(0040,1005) // Remove Requested Procedure Location

-(0040,1010) // Remove Names of Intended Recipient of Results

-(0040,1011) // Remove Intended Recipients of Results Identification Sequence

-(0040,1102) // Remove Person Address

-(0040,1103) // Remove Person Telephone Numbers

-(0040,1400) // Remove Requested Procedure Comments

-(0040,2001) // Remove Reason for Imaging Service Request

-(0040,2008) // Remove Order Entered By

-(0040,2009) // Remove Order Enterer Location

-(0040,2010) // Remove Order Callback Phone Number

-(0040,2400) // Remove Imaging Service Request Comments

-(0040,3001) // Remove Confidentiality Constraint on Patient Data Description

-(0040,4025) // Remove Scheduled Station Name Code Sequence

-(0040,4027) // Remove Scheduled Station Geographic Location Code Sequence

-(0040,4030) // Remove Performed Station Geographic Location Code Sequence

-(0040,4034) // Remove Scheduled Human Performers Sequence

-(0040,4035) // Remove Actual Human Performers Sequence

-(0040,4036) // Remove Human Performers Organization

-(0040,4037) // Remove Human Performers Name

-(0040,A027) // Remove Verifying Organization

-(0040,A078) // Remove Author Observer Sequence

-(0040,A07A) // Remove Participant Sequence

-(0040,A07C) // Remove Custodial Organization Sequence

-(0040,A730) // Remove Content Sequence

-(0040,A123) // Remove Person Name

-(0054,0200) // Remove Context Group Version

-(0054,0410) // Remove Context Group Version

-(0070,0086) // Remove Content Creators Identification Code Sequence

-(0088,0200) // Remove Icon Image Sequence (see Note 12)

-(0088,0904) // Remove Topic Title

-(0088,0906) // Remove Topic Subject

-(0088,0910) // Remove Topic Author

-(0088,0912) // Remove Topic Key Words

-(0400,0100) // Remove Digital Signature UID

-(0400,0402) // Remove Referenced Digital Signature Sequence

-(0400,0403) // Remove Referenced SOP Instance MAC Sequence

-(0400,0404) // Remove MAC

-(0400,0550) // Remove Modified Attributes Sequence

-(0400,0561) // Remove Original Attributes Sequence

-(2030,0020) // Remove Text String

-(300E,0008) // Remove Reviewer Name

-(4000,0010) // Remove Arbitrary

-(4000,4000) // Remove Text Comments

-(4008,0042) // Remove Results ID Issuer

-(4008,0102) // Remove Interpretation Recorder

-(4008,010A) // Remove Interpretation Transcriber

-(4008,010B) // Remove Interpretation Text

-(4008,010C) // Remove Interpretation Author

-(4008,0111) // Remove Interpretation Approver Sequence

-(4008,0114) // Remove Physician Approving Interpretation

-(4008,0115) // Remove Interpretation Diagnosis Description

-(4008,0118) // Remove Results Distribution List Sequence

-(4008,0119) // Remove Distribution Name

-(4008,011A) // Remove Distribution Address

-(4008,0202) // Remove Interpretation ID Issuer

-(4008,0300) // Remove Impressions

-(4008,4000) // Remove Results Comments

// brats2024 SPECIFIC TAGS TO BE REMOVED

-(3006,0002) // Remove Structure Set Label

-(3006,0009) // Remove Structure Set Time

-(0070,0080) // Remove Content Label

-(300E,0005) // Remove Review Time

-(300A,0007) // Remove RT Plan Time

// Removal of private tags amended for XNAT 1.8.3

delete[ "(0040,1001)" ] // Removes Philips date tags

delete[ "(0040,2004)" ] // Removes Philips date tags

delete[ "(2001,10CB)" ] // Removes private Philips tags

delete[ "(2001,1100)" ] // Removes private Philips tags

delete[ "(2001,9001)" ] // Removes private Philips tags

delete[ "(2005,140F)" ] // Removes private Philips tags

delete["(0033,100E)"] // TODO: Which manufacturer's private tag is this? See project-specific anon script on repo.

delete["(0033,1013)"] // TODO: Which manufacturer's private tag is this? See project-specific anon script on repo.

// Removal of MRI-specific data in case it isn't removed at first iteration

-(0008,0012) // Remove Referenced Texture Sequence Attribute

-(0008,0013) // Remove Referenced Surface Data Sequence

-(0008,1140) // Remove Referenced Image Sequence Attribute

-(0040,0244) // Remove Procedure Step Start Date in Siemens

-(0040,0250) // Remove Procedure Step Start Date in Phillips

-(0040,0251) // Remove Performed Procedure Step Start Time in Phillips

-(0040,0245) // Remove Performed Procedure Step Start Time in Siemens

-(0040,0253) // Remove Performed Procedure Step ID

-(0040,0254) // Remove Performed Procedure Step Description

-(0040,0275) // Remove Request Attribute Sequence

-(0054,0220) // Remove Code Sequence Attribute

-(0070,0082) // Remove Presentation Creation Date

-(0070,0083) // Remove Presentation Creation Time

// Removal of PET-specific data in case it isn't removed at first iteration

-(0054,0016) // Radio-pharmaceutical details -- time and date

// Removal of DX-specific data in case it isn't removed at first iteration

-(0054,0410) // Remove Context Group Version

// Removal of modality-specific tags that, depending on manufacturer, often contain dates, times, etc.

-(0008,0106) // Remove Context Group Version (date)

-(0018,1201) // Remove Time of Last Calibration

-(0040,0250) // Remove Performed Procedure Step End Date

-(0040,0251) // Remove Performed Procedure Step End Time

-(0070,0082) // Remove Presentation Creation Date

-(0070,0083) // Remove Presentation Creation Time

-(0040,030E) // Remove Exposure Dose Sequence

-(0008,1115) // Remove Referenced Series Sequence

-(0040,2004) // Remove Issue Date of Imaging Service Request

-(0040,2005) // Remove Issue Time of Imaging Service Request

-(0040,0241) // Remove Performed Station AE Title

-(0080,0050) // Remove Secondary Accession Number Tag

-(0054,0220) // Remove View Code Sequence

-(0018,700E) // Remove Time of Last Detector Calibration

-(0008,1040) // Remove Institutional Department Name

-(0008,1140) // Remove Referenced Image Sequence

-(0008,1110) // Remove Referenced Study Sequence

-(0018,1012) // Remove Date of Secondary Capture

-(0018,1014) // Remove Time of Secondary Capture

// TODO: Find out why empty strings isn't accepted in DicomEdit in XNAT v1.8.7

-(0040,A073) // Remove Verifying Observer Sequence

-(0040,A075) // Remove Verifying Observer Name

-(0040,A123) // Remove Person Name

-(0070,0001) // Remove Graphic Annotation Sequence

-(0008,0030) // Remove Study Time

//-(0008,0090) // Remove Referring Physicians Name

-(0010,0030) // Remove Patients Birth Date

-(0040,2016) // Remove Placer Order Number of Imaging Service Request

-(0040,2017) // Remove Filler Order Number of Imaging Service Request

-(0040,A088) // Remove Verifying Observer Identification Code Sequence

-(0070,0084) // Remove Content Creators Name

-(0008,0033) // Remove Content Time

-(0018,0010) // Remove Contrast Bolus Agent

// TAGS TO BE ASSIGNED A BLANK VALUE

// TODO: Find out why empty strings isn't accepted in DicomEdit in XNAT v1.8.7

//(0040,A073) := "" // Empty field replaces the Verifying Observer Sequence

//(0040,A075) := "" // Empty field replaces the Verifying Observer Name

//(0040,A123) := "" // Empty field replaces the Person Name

//(0070,0001) := "" // Empty field replaces the Graphic Annotation Sequence

//(0008,0030) := "" // Replace Study Time with a zero length value

////(0008,0090) := "" // Replace Referring Physicians Name with a zero length value

//(0010,0030) := "" // Replace Patients Birth Date with a zero length value

//(0040,2016) := "" // Replace Placer Order Number of Imaging Service Request with a zero length value

//(0040,2017) := "" // Replace Filler Order Number of Imaging Service Request with a zero length value

//(0040,A088) := "" // Replace Verifying Observer Identification Code Sequence with a zero length value

//(0070,0084) := "" // Replace Content Creators Name with a zero length value

//(0008,0033) := "" // Replace Content Time with a zero length value

//(0018,0010) := "" // Replace Contrast Bolus Agent with a zero length value

// TAGS TO BE ASSIGNED HASHED VERSIONS OF THEIR ORIGINAL VALUES

// DO NOT HASH SOP Instance UID (0008,0018) AS THIS MAY BREAK SCAN WHEN USERS TRY TO OPEN/PROCESS IT.

//(0002,0002) := hashUID[(0002,0002)] // HashUID used to replace Media Storage SOP Class UID

//(0002,0003) := hashUID[(0002,0003)] // HashUID used to replace the Media Storage SOP Instance UID

//(0004,1511) := hashUID[(0004,1511)] // HashUID used to replace the Referenced SOP Instance UID in File

(0008,0014) := hashUID[(0008,0014)] // HashUID used to replace the Instance Creator UID

//(0008,0016) := hashUID[(0008,0016)] // HashUID used to replace SOP Class UID

//(0008,0058) := hashUID[(0008,0058)] // HashUID used to replace the Failed SOP Instance UID List

(0008,0090) := hashUID[(0008,0090)] // HashUID used to replace Referring Physicians Name

(0008,010D) := hashUID[(0008,010D)] // HashUID used to replace the Context Group Extension Creator UID

//(0008,1140) := hashUID[(0008,1140)] // HashUID used to replace the Referenced Image Sequence

(0008,1195) := hashUID[(0008,1195)] // HashUID used to replace the Transaction UID

(0008,3010) := hashUID[(0008,3010)] // HashUID used to replace the Irradiation Event UID

(0018,1002) := hashUID[(0018,1002)] // HashUID used to replace the Device UID

(0020,000D) := hashUID[(0020,000D)] // HashUID used to replace the Study Instance UID

(0020,000E) := hashUID[(0020,000E)] // HashUID used to replace the Series Instance UID

(0020,0052) := hashUID[(0020,0052)] // HashUID used to replace the Frame of Reference UID

(0020,0200) := hashUID[(0020,0200)] // HashUID used to replace the Synchronization Frame of Reference UID

(0020,9161) := hashUID[(0020,9161)] // HashUID used to replace the Concatenation UID

(0020,9164) := hashUID[(0020,9164)] // HashUID used to replace the Dimension Organization UID

(0028,1199) := hashUID[(0028,1199)] // HashUID used to replace the Palette Color Lookup Table UID

(0028,1214) := hashUID[(0028,1214)] // HashUID used to replace the Large Palette Color Lookup Table UID

(0040,4023) := hashUID[(0040,4023)] // HashUID used to replace the Referenced General Purpose Scheduled Procedure Step Transaction UID

(0040,A124) := hashUID[(0040,A124)] // HashUID used to replace the UID

(0040,DB0C) := hashUID[(0040,DB0C)] // HashUID used to replace the Template Extension Organization UID

(0040,DB0D) := hashUID[(0040,DB0D)] // HashUID used to replace the Template Extension Creator UID

(0070,031A) := hashUID[(0070,031A)] // HashUID used to replace the Fiducial UID

(0088,0140) := hashUID[(0088,0140)] // HashUID used to replace the Storage Media File-set UID

(3006,00C2) := hashUID[(3006,00C2)] // HashUID used to replace the Related Frame of Reference UID

(300A,0013) := hashUID[(300A,0013)] // HashUID used to replace the Dose Reference UID

// TAGS TO BE ASSIGNED SHIFTED DATE VALUES (NUMBER OF DAYS SET ABOVE)

// BELOW TAGS COMMENTED OUT FOR braTS2024 PROJECT AS THESE ARE ALREADY SHIFT WITH pydicom

//(0008,0020) := shiftDateByIncrement[(0008,0020), NUMBEROFDAYSTOBESHIFTED] // Change Study Date

//(0008,0021) := shiftDateByIncrement[(0008,0021), NUMBEROFDAYSTOBESHIFTED] // Change Series Date

//(0008,0022) := shiftDateByIncrement[(0008,0022), NUMBEROFDAYSTOBESHIFTED] // Change Acquisition Date

//(0008,0023) := shiftDateByIncrement[(0008,0023), NUMBEROFDAYSTOBESHIFTED] // Change Content Date

(0040,0244) := shiftDateByIncrement[(0008,0244), NUMBEROFDAYSTOBESHIFTED] // Change Performed Procedure Step Start Date

//These tags do not work for Monaco patients

//(3006,0008) := shiftDateByIncrement[(3006,0008), NUMBEROFDAYSTOBESHIFTED] // Change Structure Set Date

//(300E,0004) := shiftDateByIncrement[(300E,0004), NUMBEROFDAYSTOBESHIFTED] // Change Review Date

//(300A,0006) := shiftDateByIncrement[(300A,0006), NUMBEROFDAYSTOBESHIFTED] // Change RT Plan Date

//These are dangerous tags which remove labels BUT ideally, the label should be changed in pydicom because they are required tags for importing data in.

//-(3006,0002) // Remove Structure Set Label

//-(3006,0009) // Remove Structure Set Time

//-(0070,0080) // Remove Content Label

//-(300E,0005) // Remove Review Time

//-(300A,0007) // Remove RT Plan Time