
FionaDocs

Hauke Bartsch, Marek Kociński

Aug 12, 2025

CONTENTS

1	END USER	5
2	ADMINISTRATION	17
3	ARCHITECTURE	19
4	Contact Information	99
5	Indices and tables	101

FIONA is a comprehensive medical imaging data management and processing platform designed for healthcare research environments. The system handles DICOM medical imaging data throughout its entire lifecycle - from initial acquisition at imaging scanners to final anonymized export for research purposes. Provides DICOM anonymization, quarantine management, and automated transfer from clinical to research PACS systems while ensuring General Data Protection Regulation (GDPR) compliance.

The architecture of the Fiona system can be divided into five layers, including: network layer, processing layer, storage layer, transfer layer and management layer.

- **Network Layer:** DICOM communication services that receive imaging data from scanners and send to research PACS using standard medical imaging protocols.

- **Processing Layer:** Core data processing engines that extract metadata, perform anonymization, and execute containerized analysis workflows on medical imaging studies.
- **Storage Layer:** Organized file system architecture with symbolic links, structured directories, and automated lifecycle management for imaging data and metadata.
- **Transfer Layer:** Secure data distribution system that creates anonymized exports, manages transfer requests, and delivers data to external research repositories.
- **Management Layer:** Administrative services including health monitoring, configuration management, audit logging, and automated maintenance operations.

The system operates as a distributed service with daemon processes, cron jobs, and web applications working together to provide automated medical imaging research data management.

For: Doctors, researchers, medical personnel

1.1 Fiona system

The Fiona system is a comprehensive solution for managing DICOM medical images in a research environment. The system enables automatic reception, processing, anonymization, and transfer of imaging data between Clinical and Research PACS (Picture Archiving and Communication System) systems.



Fig. 1: Fiona front page.

1.2 Research Information System

The research information system (RIS) of the Western Norway Health Authorities (Helse-Vest) also called the “Steve Project” is a secure computer system that stores research data for approved research projects at Haukeland University Hospital and connected hospitals of the Helse Vest region. The project is supported by the radiology department of [Mohn Medical Imaging and Visualization Centre](#) and the [Haukeland University Hospital](#) and approved for research project use by [Helse Vest IKT](#). The physical location of the data is at the premises of IKT Helse Vest Norway. Dedicated storage area and research software (Sectra, IDS7) provides researchers with appropriate permission access to their data. All data is stored in a de-identified format inside the RIS. Maintaining a coupling list is the responsibility of each project and not part of the functionality of the RIS. Based on the REK/DIPA rules for each project a lifetime tracking of the research data per project ensures that data can be anonymized based on data sharing requirements, and that data can be deleted at the end of the project phase - if required. We suggest that research data is allowed to be fully anonymized at the end of the project and remain in RIS for general research access. Key features of the RIS include:•

- Hosted side-by-side with the clinical PACS as an independent installation.
- Accepts de-identified patient identifiers only.
- All data is moved through a de-identification process upon import into RIS.
- All data is assigned to one or more specific research projects and the visibility of data is restricted to individuals with project role access rights.

- Projects require a valid REK approval, such documentation has to be provided at the start of a project by the project owner.
- The project owner can identify additional user accounts that can access the data.
- User access to the research PACS is controlled by IKT and requires a valid Haukeland University Hospital user account.

1.3 New project: creation, setup and access

1.3.1 Apply for a new project

In order to apply for a new project on the research information system (research PACS) please fill out the application form available under “**Apply for a new research project**” here: <https://fiona.local.server.no>.

Additional user access can be requested by the principal investigator of the project under [Apply](https://fiona.local.server.no/applications/Apply/) (<https://fiona.local.server.no/applications/Apply/>) “**Apply for access to an existing project**”.

If you encounter any problems with applying for access, contact admin@your-institution.com.

1.3.2 Access to REDCap for structured data

Our project uses [REDCap](#) as an electronic data capture solution. Projects on the research information system can receive access to their RedCap project as well as access to the image data viewing (see next section).

1.3.3 Access to the “Sectra DMA Forskning” research PACS viewer

Access to the image data is provided by your [IT Department](#). Such access may require a valid local hospital user account and a laptop or PACS workstation that is under control of IT Department. If you contact IT Department ask for the start menu item “**Sectra DMA Forskning**”. With the program and your hospital username and password you will gain access to the research picture archive and communication system (PACS).

Without access to a specific research project you will not see any data in the research PACS. Each research projects requires specific permissions to become accessible for a user.

The research PACS viewer is using a separate clinical PACS software installation (Sectra IDS7). In order to prevent possible interactions between the clinical and the research PACS only one of the application can run at a given time. You will be logged out of the clinical PACS if you start the research PACS viewer.

1.4 Submit data to the Research Information System

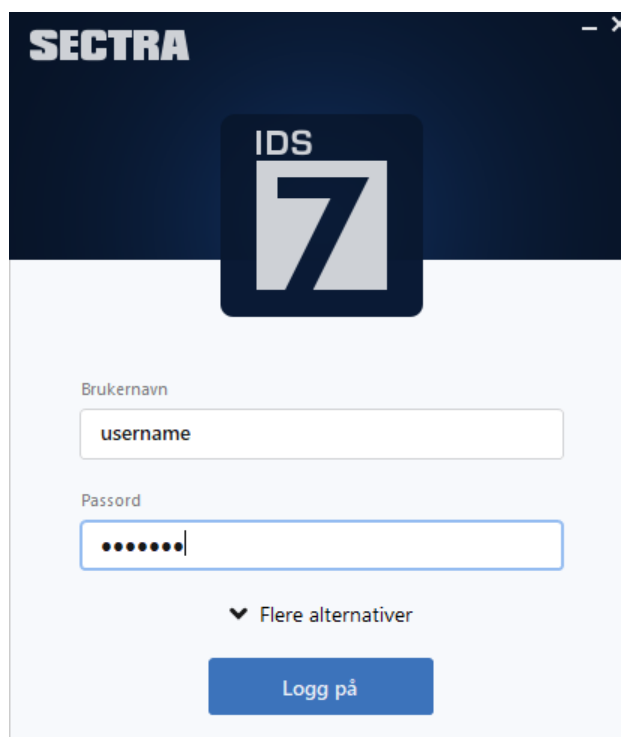
The Research Information System (RIS) contains two components. First, image data is stored in the Sectra DMA Forskning - an image viewer with a vendor neutral archive (VNA). Second, all meta-data is stored in table format in an electronic data capture system REDCap, that is Fiona server on port 4444, (<https://fiona.local.server.no:4444>). Sending image data will create the appropriate entries in [RedCap \(Fiona, port 4444\)](#). Additional data collection instruments can be set up there and used to capture assessments, consent/assent and results from automated image processing. All image data is assigned to a project to allow for project specific data views for each research information user.

The basic steps to submit data are:

1. Send DICOM studies to “HBE Fiona” or “Fiona” (modality station)
2. [Assign](https://fiona.local.server.no/applications/Assign/) to project on <https://fiona.local.server.no/applications/Assign/>.

In step 1 data arrives in a **quarantine** location. In step 2 each DICOM study needs to be **assigned to project**, pseudonymized participant identifier and event name before it will be forwarded to the research PACS and becomes visible to the project users.

Setup of a new project



SECTRA

IDS 7

Brukernavn

username

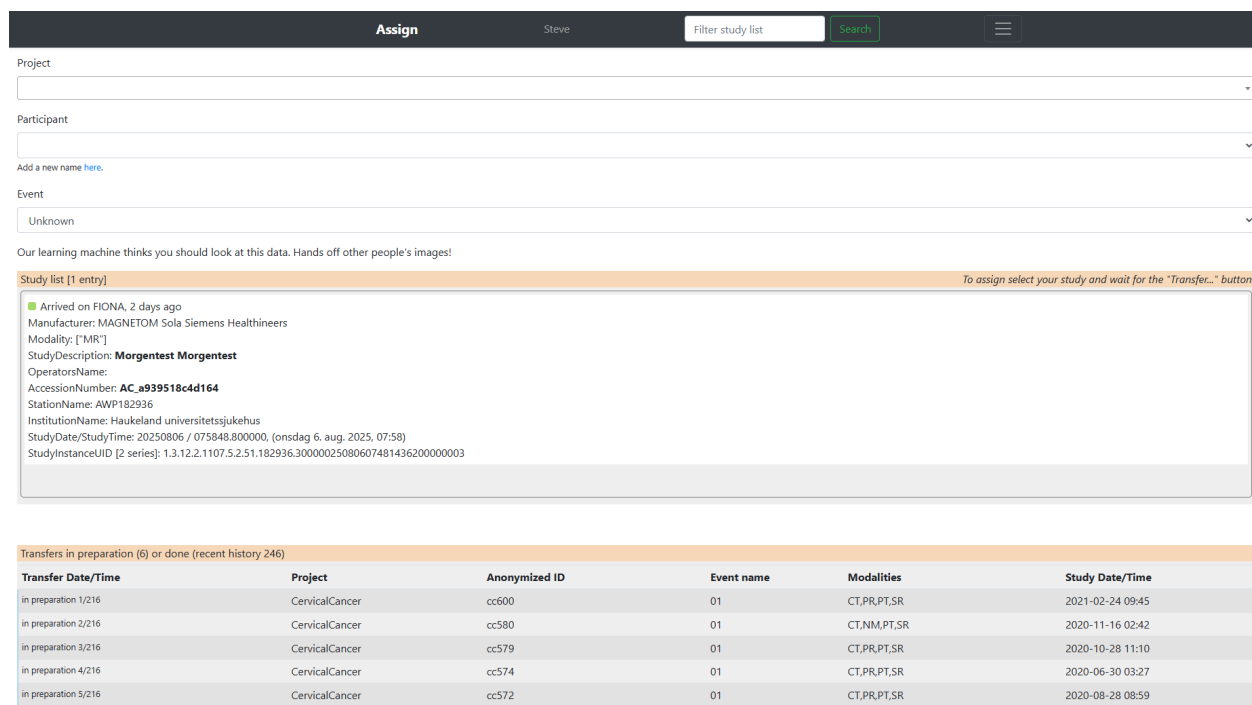
Passord

•••••

▼ Flere alternativer

Logg på

Fig. 2: Forskning PACS log-inn window to view image data by project.



Assign Steve Filter study list Search

Project

Participant

Add a new name [here](#).

Event

Unknown

Our learning machine thinks you should look at this data. Hands off other people's images!

Study list [1 entry] To assign select your study and wait for the "Transfer..." button

■ Arrived on FIONA, 2 days ago
Manufacturer: MAGNETOM Sola Siemens Healthineers
Modality: ["MR"]
StudyDescription: **Morgentest Morgentest**
OperatorsName:
AccessionNumber: **AC_a939518c4d164**
StationName: AWP182936
InstitutionName: Haukeland universitetssjukehus
StudyDate/StudyTime: 20250806 / 075848.800000, (onsdag 6. aug. 2025, 07:58)
StudyInstanceUID [2 series]: 1.3.12.2.1107.5.2.51.182936.30000025080607481436200000003

Transfers in preparation (6) or done (recent history 246)

Transfer Date/Time	Project	Anonymized ID	Event name	Modalities	Study Date/Time
in preparation 1/216	CervicalCancer	cc600	01	CT,PR,PT,SR	2021-02-24 09:45
in preparation 2/216	CervicalCancer	cc580	01	CT,NM,PT,SR	2020-11-16 02:42
in preparation 3/216	CervicalCancer	cc579	01	CT,PR,PT,SR	2020-10-28 11:10
in preparation 4/216	CervicalCancer	cc574	01	CT,PR,PT,SR	2020-06-30 03:27
in preparation 5/216	CervicalCancer	cc572	01	CT,PR,PT,SR	2020-08-28 08:59

The project needs to exist on the research information system before participant data is collected. After a successful setup your project and event names should appear in the Assign application.

1.4.1 How to add image data

The end-point for images is Fiona (<https://fiona.local.server.no>):

- AETitle: Application Entity Title
- IP: xxx.xxx.xxx.xxx
- Port: 11112

Images that arrive at this endpoint are added to a quarantine system (FIONA, <https://fiona.local.server.no:4444>) running the REDCap software. Automatic routing rules (stored in REDCap) are used to anonymize and forward the data to the image storage. If such routing has not been set up, the “Assign” application (see below) needs to be used to forward individual studies based on pre-existing patient ID lists.

From Sectra Production you can send image data to the endpoint “HBE Fiona”. Modality stations might also have the “FIONA” endpoint setup. If the data is already anonymized and has a de-identified PatientName/PatientID entry that indicates the project the FIONA system will attempt to de-identify (pseudonymization) further DICOM tags and forward the images to IDS7 (may take minutes). No further action is needed. If you suspect this did not work, see the corresponding section about the representation of transfers in REDCap.

Image data that contains patient information cannot be automatically assigned to the appropriate project as there is only a single endpoint for FIONA shared by all projects. To assign participants correctly to projects and de-identified participant identifiers a user can perform the assignment to project, participant ID and event name in the “Assign” web application.

If the participant identifiers do not exist yet user may add new project specific identifiers in “Assign”. Such identifiers need to follow the naming rules for a project and are verified using regular expression pattern specific for each project.

The web application for the assignment of scans forwarded to HBE Fiona is available at: <https://fiona.local.server.no/applications/Assign/>.

On the Assign website look for your forwarded study. It should appear in about 15 min. Identify the correct scan using the Accession Number (Undersøkelse-ID) or the date and time of the scan. Select your project from the drop-down. This will fill in the list of patient names and event names. Select the correct patient name and the event this study belongs to. After a couple of seconds a new button appears below the study entry. Use it to select and confirm the assignment. This will forward a de-identified version of the study data to “Sectra Forskning”. If you do not assign your data on Assign they will not be forwarded. After a couple of days (7 days) such data will disappear from the list. Send an email to [Local Fiona Admin](#) to request a resend.

Verification steps

After data arrived at the research PACS a verification step should ensure that all images have been received at the quarantine on FIONA and have been forwarded to research PACS. This can be done by comparing the number of images on the sending station with the number of images in IDS7.

Furthermore the import step will also attempt to de-identify secondary capture images with burned in image information. This process is fully automated and can result in false positive and occasionally false negative results. After a review of the data in IDS7 the user may decide which secondary image series are “safe” to exclude from the pixel rewriting on import. For example a secondary capture series from DTI may not contain any burned in names or identifying numbers or dates. Such image series can be removed in REDCap from further pixel anonymization.

If the number of images on FIONA does not correspond to the number of images available cache previous assignments and automatically forward such images to the research PACS using the previously defined project, patient identifier and event name.

Features for data migration

The [Assign](#) web-application allows users to upload a coupling list that maps the accession number (Undersøkelse-ID) of the study to the pseudonymized participant identifier. Such mappings must be uploaded before the first image study of the project has been forwarded to FIONA. Incoming DICOM studies in FIONA that match entries in the coupling list will automatically be assigned to the project.

How to handle errors?

Correcting errors during data import are not difficult to fix. Try to follow up on such errors on an ongoing basis. The quarantine FIONA station may still have a copy of the data in its cache which simplifies the process. Contact [Local Fiona Admin](#) in such cases and ask for help. This will allow you to fix issues such as:

- Wrong assignment of participant identifiers to DICOM studies
- Wrong assignment of event names to DICOM studies
- Missing images or image series for existing DICOM studies
- Missing entries for DICOM studies on “Assign”.

1.5 Export image data from research PACS

Data in the research PACS is secured by generic procedures during data import that delete or rewrite some DICOM tags, changes dates and replaces unique identifiers. A documentation of this process is available on the GitHub repository of the projects for removal of DICOM meta-tags: [DICOMAnonymizer](#), and for the removal of burned in image information: [RewritePixel](#).

Data stored in the research PACS is therefore in general suited for data sharing IF pseudonymized data is allowed. In order to support users with the task of data pseudonymization the research information system provides the “Review” web application that lists all existing DICOM tags in a research project (<https://fiona.local.server.no>).

Note

Pseudonymized data is defined here as data for which a coupling list exists somewhere in the universe. This is in contrast to anonymized data where such a list does not exist and can also not be created.

Further de-identification procedures might require changes to image data such as face stripping, removal of outer ear tissue, cortical folding pattern, etc.. Such potential sources of information for re-identification have been proposed in the literature but actual attacks based on them have not recently been documented. Better documented and perhaps more relevant are re-identification using spreadsheet data where external sources are linked to the projects data to discover the supposedly hidden identity of the research participants. For example it might be possible to link Gender, day of birth and the hospital name to a real participant name using a birth or voting registry.

Export using IDS7

The image data from a study can be exported from the research PACS using a right-click menu entry available in the Informasjonsvindu “Exporter til medium”. Such exports will generate either a derived patient ID – if an Anonymization Profile is selected or a faithful copy of the data with all pseudonymized DICOM tags intact.

Note

This export does not prevent re-identification. Specifically the PatientID field is created from the pseudonymized ID used in the research PACS and therefore not random.

The export is also case-by-case, which is tedious if many data need to be exported. The export will also result in directory names that do not reflect the research project structure as participant identifier – event name – modality –

image series. It may be advantageous to export from IDS7 if a single image study needs to be shared without special requirements. Such export folders will also contain an image viewer.

Export to project specific formats, NIfTI and zip-files

The research information system supports a separate export facility that is more suited to implement project specific de-identification. Such export requirements include specific DICOM value changes (replacing underscores with dashes), adding birth date information back, formatting and cleaning of series descriptions, zip-file exports with specific folder structures etc.. This export is appropriate if the receiving institution has specific requirements on how data should be shared.

Request access to the specialized data exports for your project from [Local Fiona Admin](#). Provide your export specification and we will implement your anonymization scheme and make it available to you and other researchers. As an example the “Export” application currently supports the export in NIfTI formats (using dcm2niix) and the export in several zip-file formats.

1.6 End-user contract

Attention

The following text is from the Apply website for the Steve system. Please check this page for updates to the wording.

By creating a project on the research information system you agree to the following:

1. All data stored in the RIS belongs to the research project owner represented by the PI of the project. Adding and verifying added data to the RIS is the responsibility of the project owner. The RIS team will help research projects to automate this process.
2. Sensitive participant information needs to be stored under a separate account and needs to be accessible to authorized (data-manager and above) user accounts only. All other research data is stored as de-identified data (pseudonymized, with external coupling list) or in an anonymized format. This restriction includes sensitive information such as Norwegian identification numbers, real names or parts of real names, other birth certificate information and initials. It is the responsibility of the project to review the result of the de-identification procedures implemented by the RIS team on image meta-data using <https://fiona.local.server.no/applications/ReviewDICOMTags/> and the result of the detection and removal of burned in image data (IDS7). The project will inform the RIS team in a timely manner if the pseudonymization procedure of the RIS team needs to be updated. This restriction is in place to allow for the largest possible user base for the RIS including PhD students and external collaborators.
3. All research data is stored as part of RIS projects identified by a project name of 5-20 characters. Users can gain access to the data upon request from the project PI or an appointed representative of the PI.
4. Projects are expected to utilize best-practises for data handling such as accounts based on roles like data-entry (add data only) and data-manager (change data, export data). Personally identifying fields have to be marked as such (Identifier? field of the instrument designer) and data access groups shall be used for multi-site project setups.
5. Projects will undergo a short review from the RIS team before they are moved by the RIS team from development mode into production mode for data capture. This review may generate suggestions for the project on how to implement best practices for longitudinal data captures, missing validation and the use of additional software features. All research data is collected and stored with a valid REK approval for the time period specified in the REK approval. The REK approval is required at the time that the RIS project is created. Any change of the REK approval start and end dates need to be reported to the RIS team. At the end of the project period data can be either a) deleted or b) fully anonymized (suggested choice). It is up to the project to inform the RIS team about the correct way of handling the data at the end of the project. By default we will assume that data needs to be deleted. Based on the project end date (REK) the RIS team will inform the PI of the project of a pending change of the project status to the archive state. An archive state

project will not allow for further data entry, or changes to captured data. After a period of about 1 year the project data will be exported and provided to the project PI for download. An archived project can be deleted by the RIS team after an unspecified time period. If the project data can be fully anonymized, the RIS team may create a copy of the data with new participant identifiers (without a coupling list). After a re-import a fully anonymized version of the project data can become accessible to other RIS users. The original project data will change to archive state, a copy is provided to the projects PI and the data can be deleted by the RIS team after about 1 year.

1.7 Sensitive Data Projects

1.7.1 Separation of Sensitive Information and Data

A sensitive data project is one that is used to capture human subject data and in general will require REK (regional ethics board approval). In order to setup such a project in REDCap we suggest the follow structure and features of REDCap to be used. These recommendations have been generated based on discussions in relevant risk assessments.

All sensitive data should be stored in a separate REDCap “ID” project including Norwegian Identification Numbers, names or parts of names, addresses and full birth dates (see Figure 1). This project should have its own roles of “Data Manager”, “Data Entry”, and “Controller”. People with permission to access and/or edit this information can use this database to keep contact information up-to-date and to enroll new participants into the study. Each participant should be assigned a pseudonymized ID in the sensitive data project that links the entry to the corresponding participant in the data project. Examples for this ID are:

- <project name>-<site number>-0001,
- <project name>-<site number>-0002,
- etc..

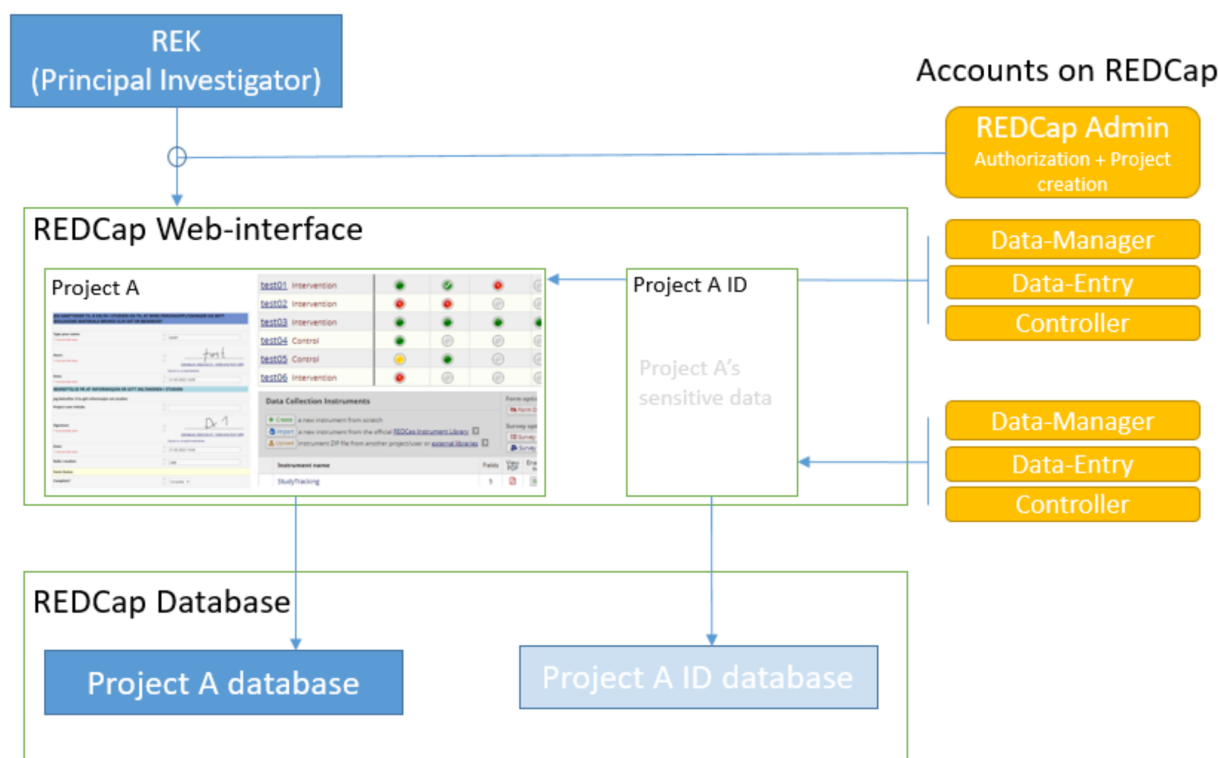


Fig. 3: Sensitive data projects should be split into a REDCap project for data (using pseudonymized ids) and a REDCap project for sensitive data including the coupling list.

All other data should be stored in a separate REDCap “Data” project using the pseudonymized participant ID as a “record_id” (first field in the study).

User rights management

When a project leader / principal investigator (PI) is given a REDCap account and project, they are given “project owner” roles. The project owner can then provide access to project members in “roles”. A role defines a given set of custom permissions which defines the user’s access to data, export permissions and ability to make changes to data.

Each project can have predefined roles. We recommend the predefined roles:

- “Data Manager” (ability to change study design, export),
- “Data Entry” (add, change, or delete data),
- “Controller” to define roles for data viewing, editing, and deleting records.

In more complex cases, different access settings can be given on different forms in the study (see also API access with REDCap). Individual users are assigned to project roles as part of gaining access to one project.

The user rights management is the responsibility of the project owner and/or the users they add to the project with User Rights access. User roles should be set at the lowest access level that is necessary (e.g., export rights only for users who need this permission). Access to the project should be reviewed regularly and personnel who no longer require access need to be removed from the project.

User rights – multi-center projects

In a project where several institutions participate with their own project participants (several hospitals etc.) each group of participants should be assigned to a separate “data access group”. This functionality allows records in a study to be part of the user rights management. A user with access to a single data access group can only see participants that belong to this group. If this user creates a new participant, they will be automatically assigned to the group.

How to handle Email Addresses in Data Projects

Email addresses are special identifying fields that can be stored in data projects for the purpose of creating automated invites for participants to fill out forms from home. In projects that use this feature email fields need to be present in the data project in order to allow for email distribution to participants.

1. Add such email fields to a separate instrument of the REDCap data project and mark the instrument as viewable by specific roles only (like Data Managers).
2. Mark the email field as an “Identifier” field to prevent export of the field’s data by user of roles that cannot view sensitive fields.
3. Add the Action Tag “@PASSWORDMASK” to the field to prevent accidental viewing of the fields values if the instrument is displayed on screen.
4. Add a field validation as “Email” to prevent some miss-typing of emails.

1.7.2 Randomization

Randomized studies have can remove biases caused by selection of participants for specific arms in a study. Such biases can prevent a fair assessment of a treatment option. The randomization feature of REDCap allows users to upload a randomization table that has been externally created before the start of the study – usually by the statistician of the project. After participants are enrolled into the study the randomization entries for that person are “opened” and the choice of the randomization is stored in the project.

Workflow Randomization

Randomization table (from KKF Statistician)

Treatment A,	Intervention
0,	1
1,	0
...	

Setup: Upload to REDCap

Participant A (before Randomization)

During participant enrollment an entry from the randomization table is revealed.

Participant A (after Randomization)

Already randomized
Intervention
based on randomization table

Data History for variable "study_tracking_randomization" for record "test06"

Listed below is the history of all data entered for the variable "study_tracking_randomization" for Study ID "test06". The data history results are sorted from earliest to most recent.

Date/Time of Change	User	Data Changes Made
07/04/2022 11:53:47 <small>(most recent data change)</small>	johndoe	Intervention (0)

Randomization is used to create un-biased groups of participants that may receive different treatment or other interventions. Projects use an externally generated randomization table (project statistician) and "Randomize" uncovers those entries.

STEP 1: Define your randomization model

This step will allow you to define the randomization model you will be implementing and all its parameters, which includes defining strata (if applicable) and optionally randomizing subjects per group/site (if a multi-site study).

NOTE: This section is currently locked and uneditable because the randomization setup process has already been completed. If you wish to modify the randomization setup below, you will need to click the Base Randomization Model button below.

A) Use stratified randomization? ☐
 It is often necessary to ensure equal treatment among a number of factors. Stratified randomization is the solution to achieve balance within one or more subgroups, such as gender, race, diabetes/non-diabetics, etc. By choosing strata (multiple choice criteria fields), you may then be able to ensure balance within those subgroups. [See our docs](#)

B) Randomize by group/site? ☐
 If this is a multi-center/multi-site project (or something similar), you may want to stratify the randomization by each group/site. You can select an existing multiple choice field that represents the groups/sites. Or you can use Data Access Groups to stratify by group/site.

C) Choose your randomization field
 This is the field where the allocated randomization (treatment) group will be saved and stored, and is where the Randomize button will appear on your data collection form.
 study_tracking_randomization (Randomization) for
 Like 0 - baseline Save randomization model Erase randomization model

Randomization Already randomized

Intervention based on randomization table

Randomization errors

If there was an error (assignment intervention/control differs from above) mark it here. ☒ Error, the intervention/choice actually used is different

Randomization (manual choice if automatic randomization error) ☐ select if an error with randomization occurred

☐ control Should only be entered if different from randomization

Error during setup.

Fig. 4: Basics of the randomization workflow in REDCap.

1.7.3 e-Consent

In an e-Consent workflow the basics of the paper informed consent are maintained. An electronic consent document is created based on the approved language and design of the paper consent using HTML features in REDCap. The solution supports signature fields (stored as images) and creates resulting PDF (paper) versions of the consent as well as electronic versions of the consent. The following figures show some of the setup and resulting documentation that is created in the solution.

We suggest exporting e-Consent documents and to store them centrally by the project administration outside of REDCap.

As informed consent document contain the name of the signatory and the one countersigning the informed consent process the e-Consent workflow should be implemented in the sensitive data (ID) REDCap project.

1.7.4 Automatic data exports from REDCap

Data may be exported from REDCap using the REDCap API, a technical interface to automate the export of project and participant information using scripting. To provide such access a dedicated user-account "api_<real username>" should be created which is specific for a single project. Configure the account with a limited set of read permissions to specific fields or instruments using a new API role. The REDCap API will borrow these restrictive permissions for controlled access.

Setup: An administrator can generate an API "token" for this account and share the token and examples of accessing the resource (curl-based access) with the user.

Any change in the role of the <real username> should also apply to the connected API account. Specifically losing access to the project should be implemented for both <real username> and api_<real username>.

What is the e-Consent Framework?

What is e-Consent?

Electronic Consent (e-Consent) is a platform for consenting patients or research subjects either on site or at home using a computer-based consent form rather than traditional paper documentation. Consent forms can be implemented in a REDCap survey via computer, mobile phone, or tablet. **Most importantly, it is highly recommended that you have a discussion with your local Institutional Review Board (IRB) if you wish to do e-Consent in REDCap.**

How the e-Consent Framework works

The 'Auto-Archiver + e-Consent Framework' survey option adds two things to the typical survey-taking process: 1) Before a participant completes the survey, an extra certification page is added to end of the survey that displays an in-line PDF copy of their survey responses in which they will be asked to confirm that all information in the document is correct. Once they confirm all is correct, the survey will then be marked as complete. The survey will not be considered complete until they fulfill the certification step. 2) Upon completion of the survey, a static copy of their responses in the form of a consent-specific PDF will be stored in the projects File Repository. The consent-specific PDF will have the values of the e-Consent Framework Options inserted at the bottom of each page in the PDF. These values (i.e., name, date of birth, etc.) are added to the PDF as extra documentation of the identity of the person who is consenting.

Why is this called a 'Framework'?

The 'e-Consent Framework' is referred to as a 'framework' because enabling this option alone does not provide an e-Consent process but merely provides the general framework or mechanism to allow you to provide e-Consent to patients/subjects. As a survey administrator, you must still create your e-Consent survey and all the questions in it, including name, date of birth, and how you wish to capture the signature. This framework allows you to implement your e-Consent process by providing standardized tools (certification screen + automatic storage of consent form as a 'hard-copy' PDF) while still providing the ability to customize your survey how you wish.

How can the 'signature' process be handled for e-Consent?


Patients/subjects can 'sign' their consents by typing in their name or by utilizing REDCap's 'Signature' field type (i.e., 'wet signature') on the survey. One might also assign PIN numbers to prospective participants to aid in the signature process. **Please note that the signature process will NOT be implemented by REDCap automatically, so it is your responsibility as a survey administrator to construct your survey using one of the methods above in order for the signature to get captured appropriately.**

What is e-Consent version and type?

e-Consent version and type are both free-form text fields whose value will be inserted at the footer of each page in the PDF. Versions are a concept whereby you may give a number or alpha numeric designation to represent the current version or state of the form. So if the form is modified AFTER data collection begins, then it is recommended that a new version be applied. For example, the first version might simply be '1', and after collecting the consent of a few participants, a question is modified or added, which represents a new version of the form, so you might increment the version to '2' (and so forth). The e-Consent type is optional and is another free-form text field that is just a text label that you might want to display in the PDF footer to signify the type of e-Consent that this survey represents (e.g., pediatric). The type is often used to distinguish between multiple e-Consent forms within a project.

Close

100



HELSE BERGEN

Haukeland universitetssjukehus

FORESPØRSEL OM DELTAKELSE I LEGEMIDDELUTPRØVING

BEHANDLING AV BENZODIAZEPINAVHENGIGHET

Dette er en forespørsel til deg om å delta i et forskningsprosjekt for å undersøke ulike muligheter for behandling av avhengighet til benzodiazepiner. Formålet med prosjektet er å få kunnskap om vedlikeholdsbehandling med benzodiazepiner (oksazepam eller diazepam) hos pasienter i legemiddelassisteret rehabilitering (LAR), som har en omfattende og langvarig avhengighet til beroligende medisiner. I studien vil effekt og trygghet av behandlingen bli undersøkt. Du blir spurt om deltakelse i prosjektet fordi du er pasient i LAR i avdeling for rusmedisin (AFR) og har en omfattende og langvarig avhengighet til benzodiazepiner. Det er behov for å inkludere minst 90 deltakere i studien for å kunne stole på resultatene ved studien.

HVA INNEBÆRER STUDIEN?

Når du velger å delta i studien, må du først gi skriftlig samtykke. Deretter vil du få en time hos LAR-lege til første samtale før og kan bli inkludert i studien eller eventuelt ikke. Dersom du velger å delta i studien, vil du bli inkludert i studien og eventuelt ikke. Dersom du velger å delta i studien, vil du bli inkludert i studien og eventuelt ikke. Dersom du velger å delta i studien, vil du bli inkludert i studien og eventuelt ikke.

e-Consent data (in-person)

- Captures data of the participant (typed name + signature + date)
- Captures data of study personnel (who informed, initials + signature + date + role in study)

The screenshot displays the REDCap e-Consent interface. On the left is a 'Confidential Informed Consent' form with fields for Name, Date, and Signature. The signature is captured as an image. To the right, the 'Signatures stored as images' section shows two captured signatures with their respective file names and sizes. Below this, the 'Logging information' table shows a log of updates. At the bottom, the 'Export paper copy of consent' section provides a table of survey completion times and a download link for a PDF copy of the consent form.

Time / Date	Username	Action	List of Data Changes OR Fields Exported
09/05/2022 12:14	admin	Update record test07 (Uke 0 - baseline)	consent_typed_name = 'Hauke Bartsch', consent_name = '722', consent_date = '2022-05-09 12:13', consent_project_initials = 'HB', consent_signature_project = '723', consent_date_project = '2022-05-09 12:13', consent_role = 'tester', informed_consent_complete = '2'

Survey Completion Time	Record	Survey	Event	Identifier (Name, DOB)	IP Address	Version	Type	Download
21/02/2022 16:46	BMX-BAR 003	Informed Consent	Uke 0 - baseline	kjsdfkjsdf	10.84.227.60	Delta:erinformation BMX-BAR V. 1.0 16SEP21	PDF	

Fig. 7: Internal documentation of e-Consent in REDCap. Signatures have been captured and tracked. A paper version for export is available.

1.7.5 Steps at the end of a REDCap project

REDCap is a tool for data collection. At the end of data capture projects using REDCap receive a notification of study end. At this point projects may provide updated REK information (extension of data capture notice). If no such notice is received REDCap projects will:

- Lock all data participants (no further update/add).
- Provide a copy of the REDCap project (CDISC format) to the project's principal investigator or delegate.
- Provide a copy of the project data (CSV) and data dictionary (PDF) to the principal investigator or delegate.
- Request a confirmation that project data (CDISC and CSV) have been received by the project.
- Permanently delete all project data.

This process will be documented in the REDCap project tracking project "DataTransferProjects", the project management tool with information on identity of the person requesting project removal and confirmations for all steps of the project removal process.

1.8 External resources

1.8.1 Git-Hub Repositories

1. Fiona Documentation Repository - <https://github.com/mmiv-center/FionaDocs>
2. DICOMAnonymizer - <https://github.com/mmiv-center/DICOMAnonymizer>
3. RewritePixel - <https://github.com/mmiv-center/RewritePixel>

Project End	
End of project requested	<div><div><div>H</div><div></div></div><div><input type="text"/></div><div><div><div></div><div></div></div><div>Now</div><div>D-M-Y H:M</div></div></div>
Name/email of person requesting project end	<div><div><div>H</div><div></div></div><div><input type="text"/></div></div>
Project end export (CDISC)	<div><div><div>H</div><div></div></div><div>Upload file</div></div>
Project data (spreadsheet format)	<div><div><div>H</div><div></div></div><div>Upload file</div></div>
Project end data dictionary	<div><div><div>H</div><div></div></div><div>Upload file</div></div>
Data forwarded to user (email addressed)	<div><div><div>H</div><div></div></div><div><input type="text"/></div></div>
Received confirmation	<div><div><div>H</div><div></div></div><div><div><input type="radio"/> Yes</div><div><input type="radio"/> No</div></div><div>reset</div></div>
Project end confirmation ok notice from project	<div><div><div>H</div><div></div></div><div><div><div></div></div><div>Expand</div></div></div>
Person performing deletion	<div><div><div>H</div><div></div></div><div>Add signature</div></div>

Fig. 8: End-of-project tracking for REDCap projects

1.8.2 REDCap

1. REDCap - <https://project-redcap.org>

ADMINISTRATION

For: IT administrators, DevOps

ARCHITECTURE

For: Developers, system architects

FIONA System Architecture

A detailed system architecture including all system components is presented below

[Download image as PDF](#)

3.1 Setup

To setup the system some mandatory information must be set in `/data/config/conf.json` file. Fill values appropriate to your server as shown in example json file.

Listing 1: System configuration example settings

```
1 {
2   "DICOMIP": "your_dicom_server_ip",
3   "DICOMPORT": "dicom_port",
4   "DICOMAEITITLE": "FIONA",
5   "SCANNERIP": "scanner_ip",
6   "SCANNERPORT": "scanner_port",
7   "SCANNERAETITLE": "scanner_application_entity_title",
8   "SCANNERTYPE": "scanner_device_type",
9   "MPPSPORT": "modality_performed_procedure_step_port",
10  "SERVERUSER": "server_user_name",
11  "DAICSERVER": "daic_server_ip",
12  "PFILEDIR": "/path/to/your/profile/directory",
13  "CONNECTION": "",
14  "DATADIR": "/path/to/data/dir",
15  "LOCALTIMEZONE": "your_local_zone",
16  "PROCESSING_USER": "processing_user_name",
17  "PROJECTTOKEN": "your_project_token",
18  "Authentication": {
19    "Table-based": { "enabled": 1 },
20    "LDAP": [
21      {
22        "enabled": 0,
23        "connection": "your_value",
24        "certificate": "your_value",
25        "username": "your_value",
26        "password": "your_value",
```

(continues on next page)

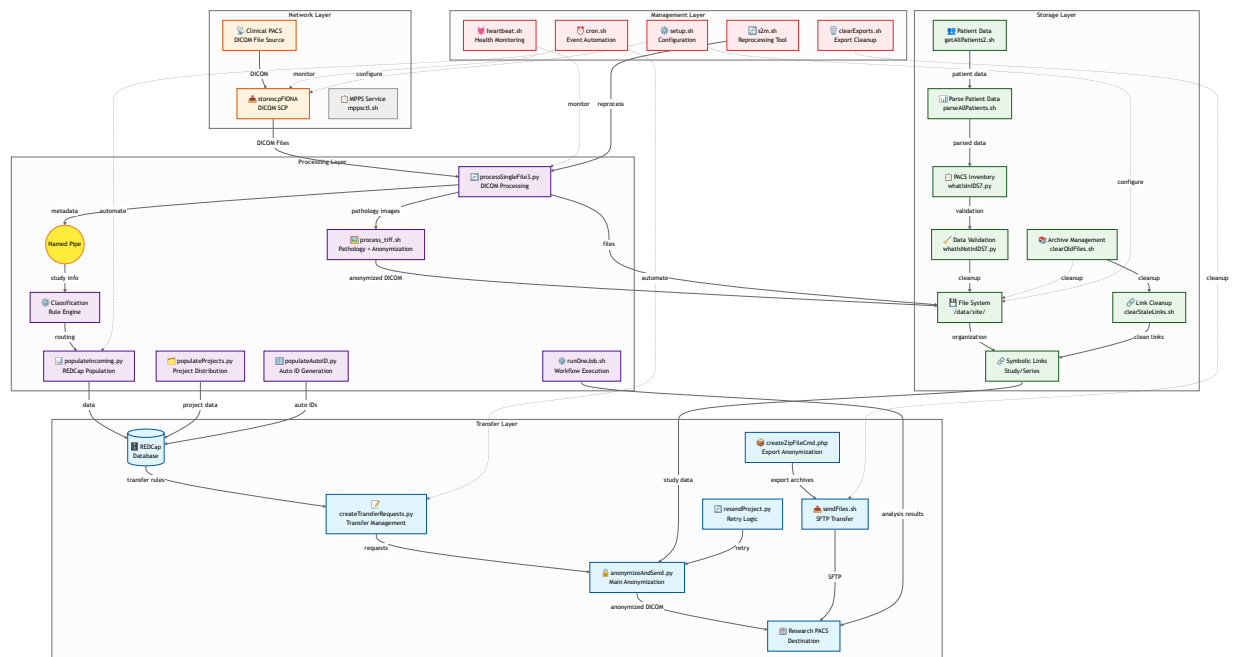


Fig. 1: FIONA layered Architecture

(continued from previous page)

```

27         "query": "your_value",
28         "dn": "your_value"
29     },
30 ]
31 }
32 }

```

3.2 Folder and File structure

```

/home/processing/
├── bin/
│   ├── anonymizeAndSend.py
│   ├── clearExports.sh
│   ├── clearOldFiles.sh
│   ├── clearStaleLinks.sh
│   ├── createTransferRequestsForProcessed.py
│   ├── createTransferRequests.py
│   ├── populateAutoID.py
│   ├── populateIncoming.py
│   ├── populateProjects.py
│   └── utils/
│       ├── getAllPatients2.sh
│       ├── parseAllPatients.sh
│       ├── resendProject.py
│       ├── whatIsInIDS7.py
│       └── whatIsNotInIDS7.py
└── /var/
    ├── www/
    │   └── html/
    │       ├── applications/
    │       │   ├── Assign/
    │       │   │   ├── php
    │       │   │   └── removeOldEntries.sh
    │       │   ├── Attach/
    │       │   │   └── process_tiff.sh
    │       │   ├── Exports/
    │       │   │   ├── php
    │       │   │   └── createZipFileCmd.php
    │       │   ├── User/
    │       │   │   ├── asttt/
    │       │   │   │   └── code/
    │       │   │   │       └── cron.sh
    │       │   └── Workflows/
    │       │       ├── php
    │       │       └── runOneJob.sh
    │       └── server/
    │           ├── bin/
    │           │   ├── heartbeat.sh
    │           │   └── processSingleFile3.py

```

(continues on next page)

(continued from previous page)

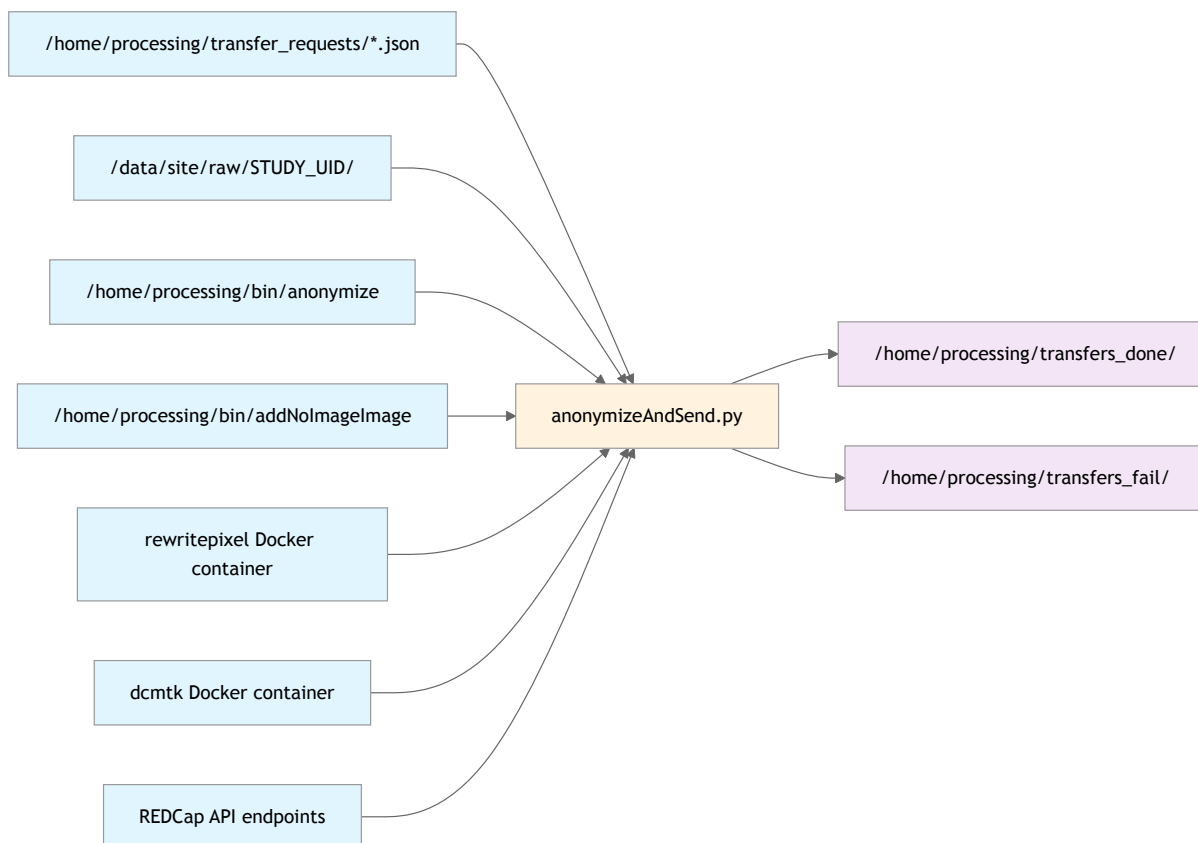
```
|   └─ sendFiles.sh
|   └─ storectl.sh
└─ utils/
    └─ s2m.sh
```

3.3 Components

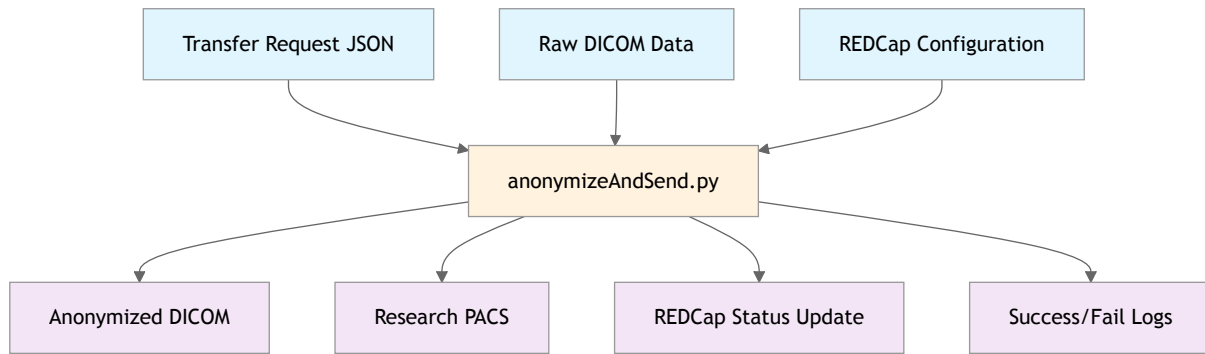
3.3.1 anonymizeAndSend.py

This Python script processes DICOM medical imaging transfer requests by anonymizing patient data and sending anonymized studies to a research PACS system. The script reads JSON transfer requests, applies project-specific anonymization rules including burned-in image detection and pixel data rewriting, then sends the processed data via DICOM protocol while updating REDCap with transfer status. It handles various project features like presentation state removal, secondary capture filtering, and maintains audit trails of all processing operations.

Related Files



Data flow diagram



Data paths

- **Input directories:**

- /home/processing/transfer_requests/
 - /data/site/raw/
 - **Output directories:**
 - /home/processing/transfers_done/
 - /home/processing/transfers_fail/
 - **Temporary processing:**
 - System temp directories via `tempfile.TemporaryDirectory()`
-

Script docstring starts here —>>> ->> ->>

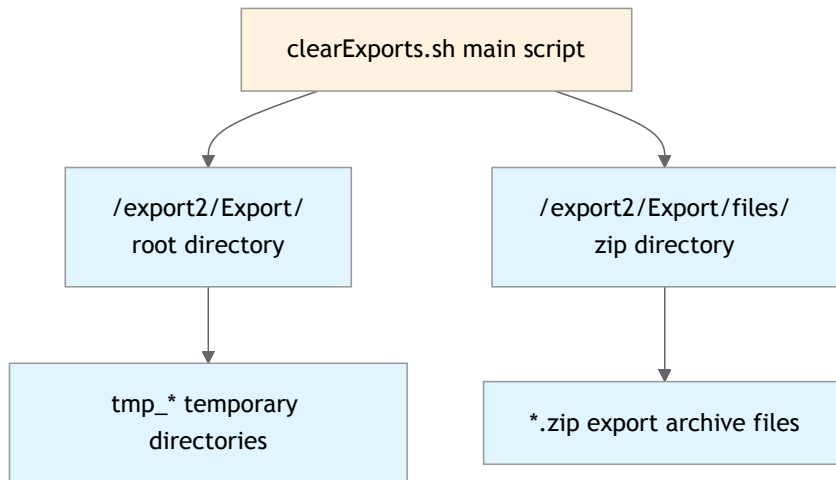
There is no docstring.

2025.06.17 mk

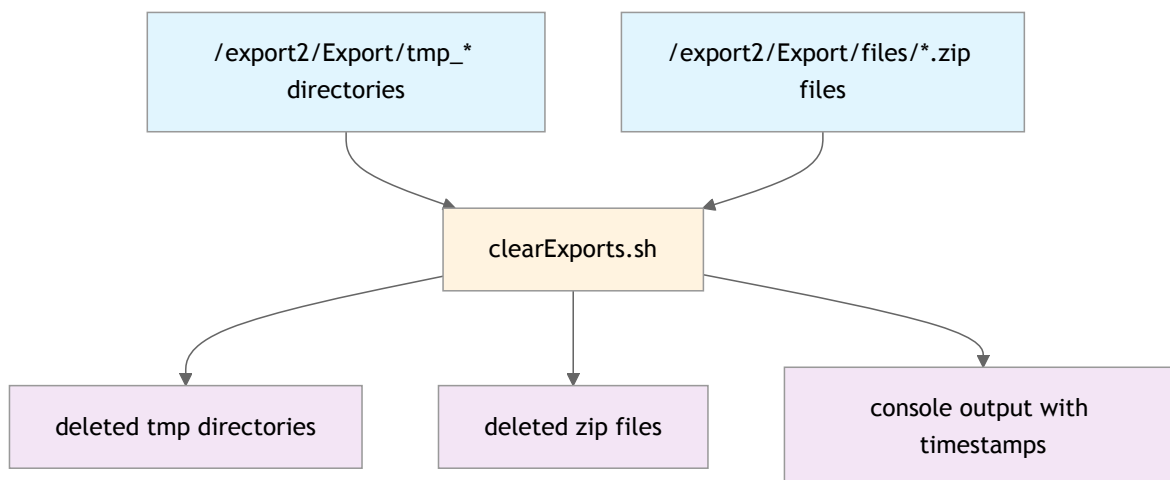
3.3.2 clearExports.sh

This bash script manages disk space by automatically deleting old export files when disk usage exceeds 80%. It first removes temporary directories older than 1000 minutes, then systematically deletes the oldest ZIP files in `/export2/Export/files/` until disk usage drops to 80% or below. The script provides timestamped logging throughout the cleanup process to track which files are being removed.

Related File



Data Flow Diagram



Data paths

- **Input directories:**

- /export2/Export/,
 - /export2/Export/files/
 - **Output directories:**
 - Files are deleted (no output directories)
-

Script docstring starts here —>>> ->> ->>

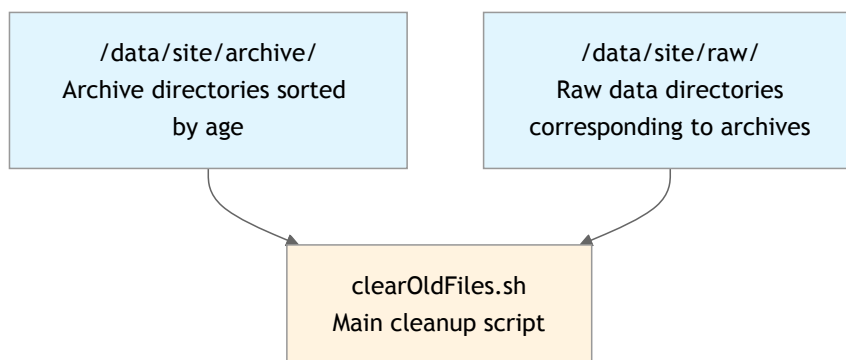
```
find . -type d -maxdepth 1 -printf '%T+ "%p"n' | sort | less
```

delete oldest files until we have at least 20% free space on this partition

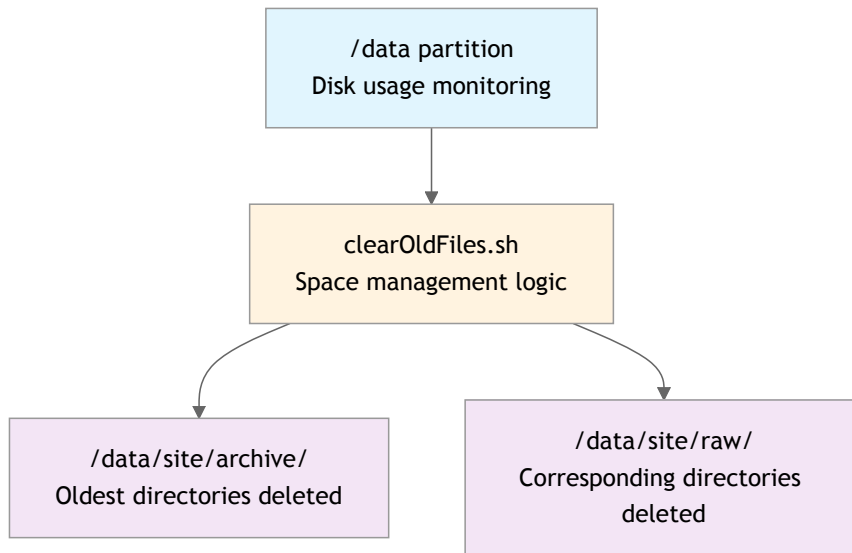
3.3.3 clearOldFiles.sh

This bash script automatically manages disk space by deleting oldest archive directories when partition usage exceeds 80%. It monitors the /data/ partition and systematically removes directories from /data/site/archive/ in chronological order (oldest first) until free space reaches at least 20%. The script also removes corresponding directories in /data/site/raw/ based on extracted UIDs from archive directory names.

Related Files



Data Flow Diagram



Data paths

- **Input folder:**

- /data/site/archive/,
- /data/site/raw/,
- /data/

- Output folder: No output directories, delete only.
-

Script docstring starts here —>>> ->> ->>

find . -type d -maxdepth 1 -printf '%T+ "%p"n' | sort | less

delete oldest files until we have at least 20% free space on this partition

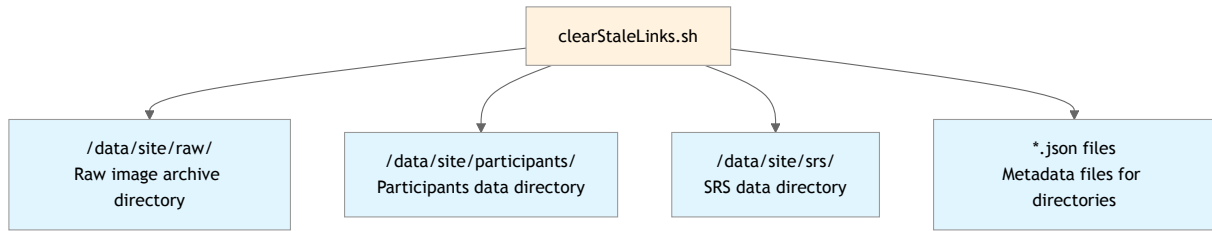
TODO: delete links in /data/site/participants/

TODO: delete links in /data/site/srs/

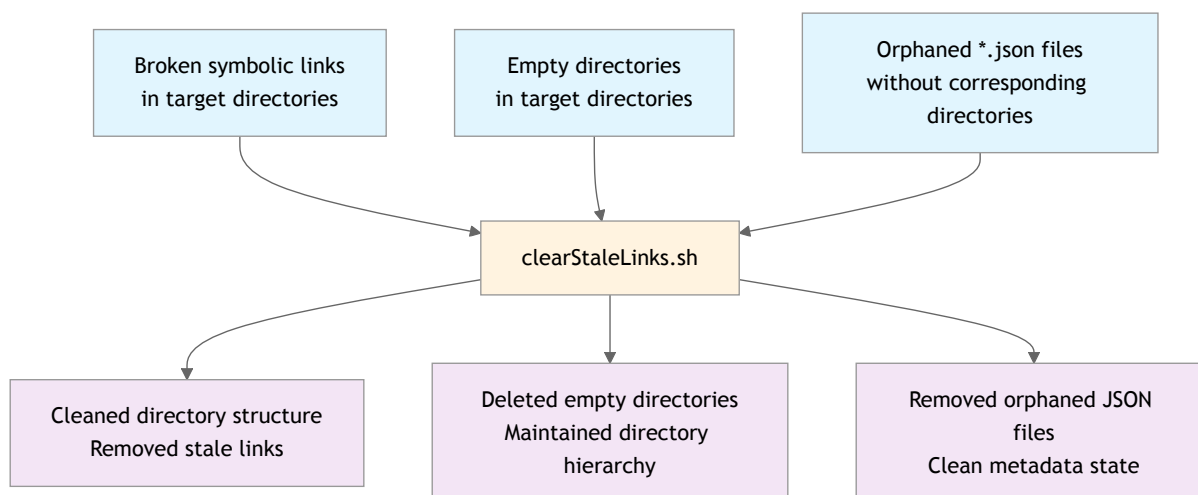
3.3.4 clearStaleLinks.sh

This bash script performs maintenance on image archive directories by cleaning up broken symbolic links and empty folders. It processes three specific directories (/data/site/raw/, /data/site/participants/, /data/site/srs/) to remove stale symbolic links that no longer point to valid files, delete empty directories, and clean up orphaned JSON metadata files. The script ensures data integrity by maintaining a clean directory structure in the archive system.

Related Files



Data Flow Diagram



Data paths

- **Input/Processing Directories:**

- /data/site/raw/,
 - /data/site/participants/,
 - /data/site/srs/
- Output: The same directories (cleaned and maintained)
-

Script docstring starts here —>>> ->> ->>

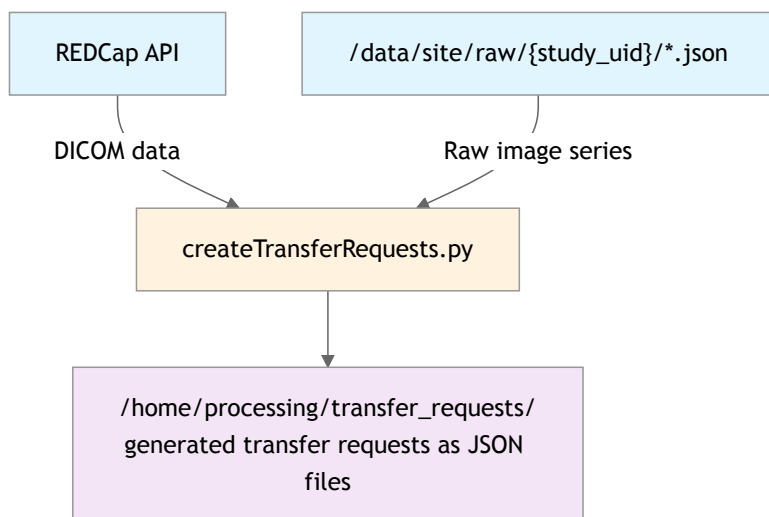
Delete links that do not point to real images in archive.

Just to be sure we have a clear /data/site/raw folder we should remove broken symbolic link and empty folders.

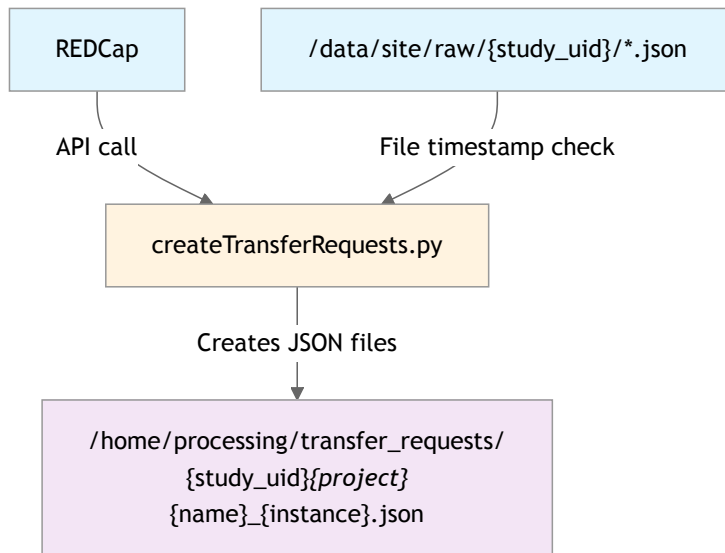
3.3.5 createTransferRequests.py

This Python script retrieves medical imaging transfer records from a REDCap database via API and creates JSON transfer request files for studies that need to be transferred. It checks for new image series by comparing transfer dates with file modification times in the raw data directory. The script only creates transfer requests for records that have complete information (project name, transfer name) and haven't been processed yet.

Related Files



Data Flow Diagram



Data paths

- **Input directories:**

- /data/site/raw/{study_instance_uid}/
 - **Output directories:**
 - /home/processing/transfer_requests/
-

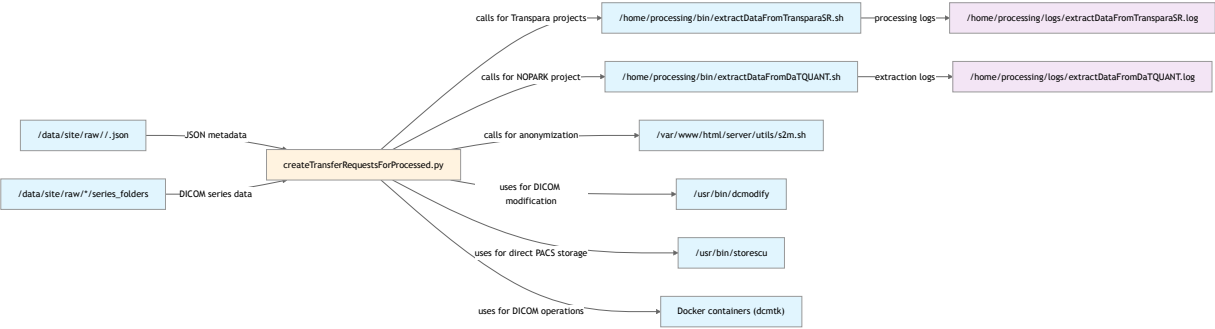
Script docstring starts here —>>> ->> ->>

There is no docstring.

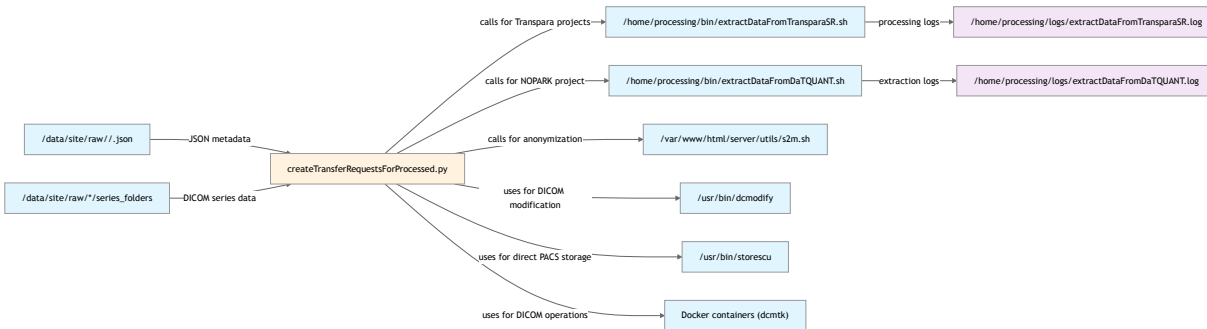
3.3.6 createTransferRequestsForProcessed.py

This script automatically detects processed medical imaging series that need to be forwarded to research PACS systems by identifying series with StudyInstanceUIDs that contain dots (indicating they're new/processed data). It retrieves transfer requests from REDCap, modifies DICOM metadata to match original study identifiers, and forwards the data either directly to storage or through anonymization processes depending on the project type. The script also extracts structured report data for specific projects like Transpara studies and NOPARK.

Related Files



Data Flow Diagram



Data paths:

- **Input Paths:**

- /data/site/raw/*/
 - **Output Paths:**
 - Research PACS server,
 - /home/processing/logs/,
 - Temporary directories (created dynamically).
-

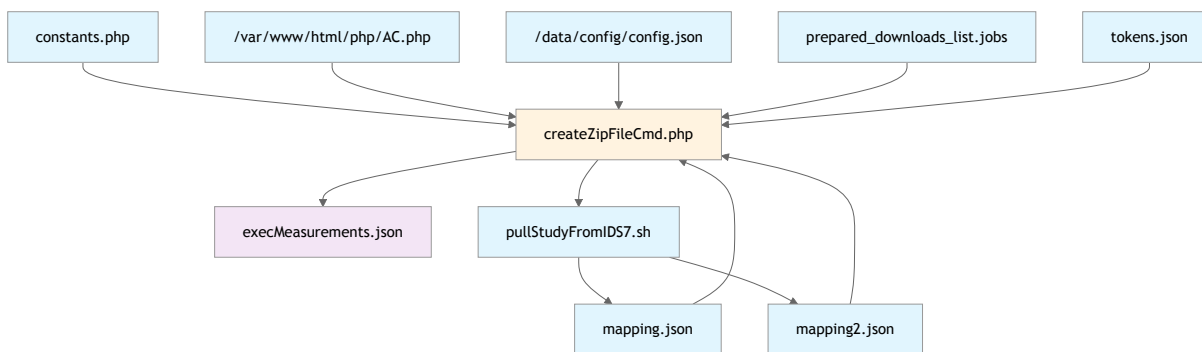
Script docstring starts here —>>> ->> ->>

There is no docstring.

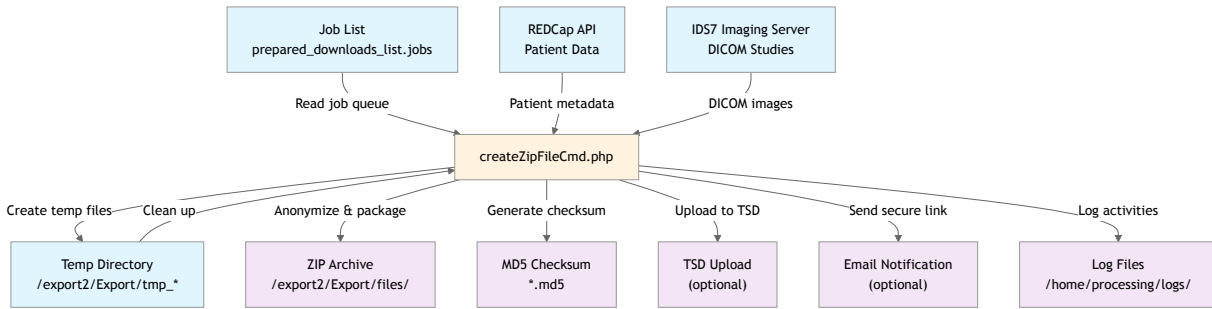
3.3.7 createZipFileCmd.php

This PHP script is a command-line cron job that processes medical imaging export requests by creating ZIP archives of DICOM studies. It pulls DICOM data from an imaging server, applies anonymization transformations based on project-specific rules, optionally converts to NIFTI format, and packages the results into downloadable ZIP files. The script handles secure email delivery with password protection and supports export to TSD (Norwegian research data storage).

Related Files



Data Flow Diagram



Data paths

- **Input paths:**

- /var/www/html/applications/Exports/php/prepared_downloads_list.jobs,
- /data/config/config.json,
- /var/www/html/applications/Exports/php/tokens.json

- **Output paths:**

- /export2/Export/,
 - /export2/Export/files/,
 - /home/processing/logs/,
 - /var/www/html/applications/Exports/php/execMeasurements.json
-

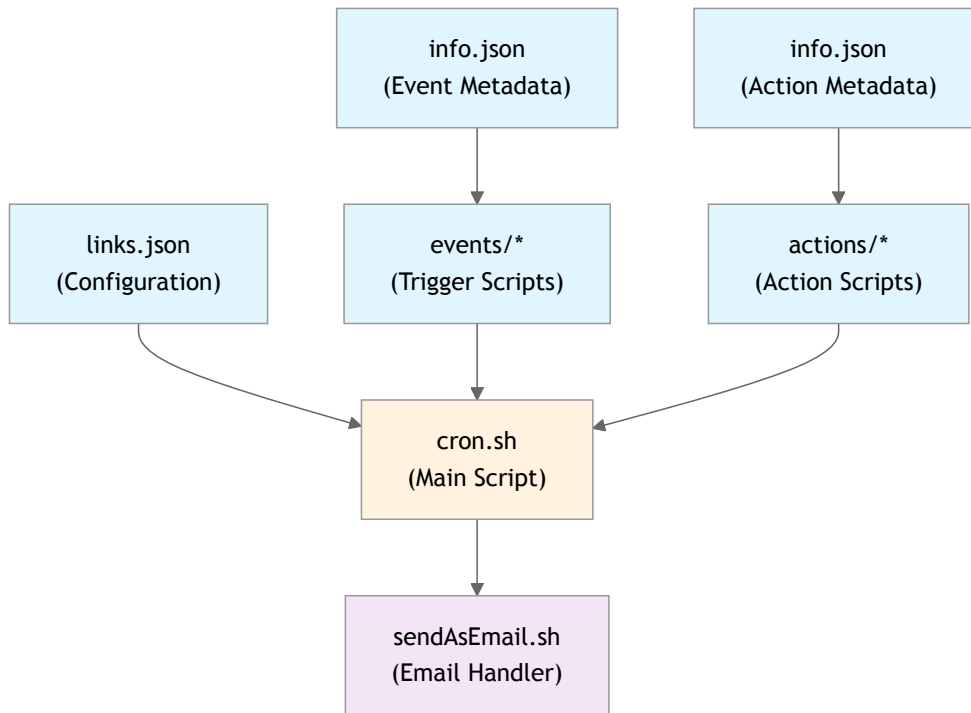
Script docstring starts here —>>> ->> ->>

This is a docstring.

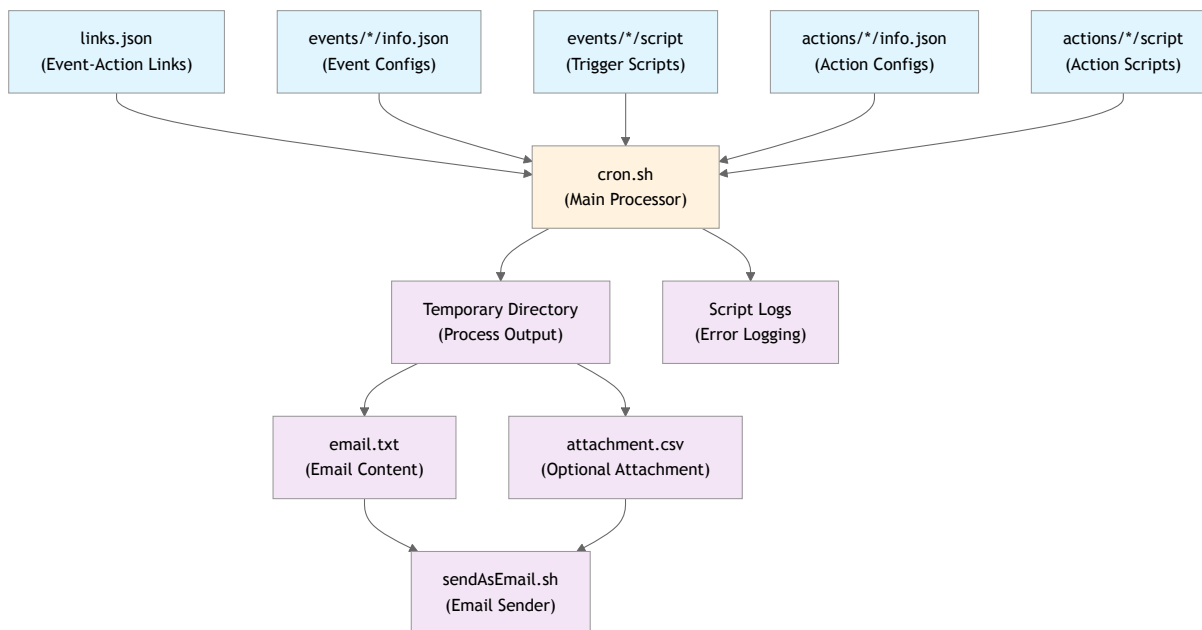
3.3.8 cron.sh

This script is an automation scheduler that reads event-action links from a JSON configuration file (/var/www/html/applications/User/asttt/code/links.json) and executes corresponding triggers and actions. It processes each link by checking if the associated event trigger conditions are met, and if so, executes the corresponding action script. The script supports email notifications with optional CSV attachments and uses temporary directories for processing outputs. It's designed to run as a cron job for automated task execution based on predefined event-action mappings.

Related Files



Data Flow Diagram



Data Paths

- **Input Paths:**

- /var/www/html/applications/User/asstt/code/links.json
- /var/www/html/applications/User/asstt/code/events/*
- /var/www/html/applications/User/asstt/code/actions/*
- events/*/info.json` and `actions/*/info.json

- **Output Paths:**

- /tmp/asst_cronXXXXX/
 - /tmp/asst_cronXXXXX/email.txt
 - /tmp/asst_cronXXXXX/attachment.csv
 - \${script}_log
-

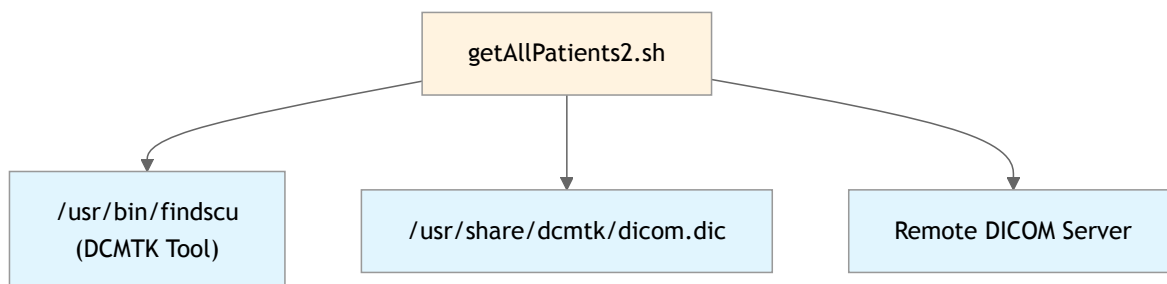
Script docstring starts here —>>> ->> ->>

There is no docstring.

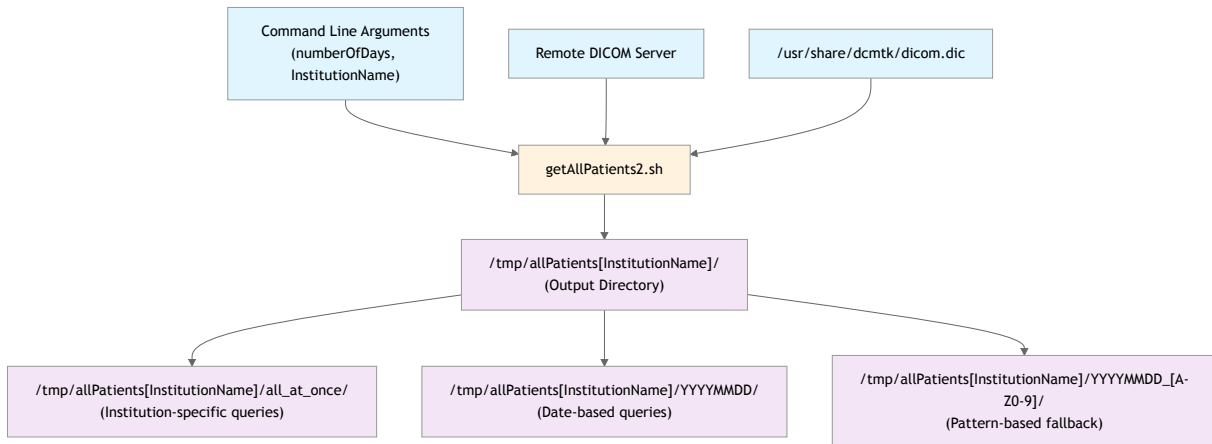
3.3.9 getAllPatients2.sh

This script retrieves DICOM patient study information from a remote DICOM Query/Retrieve SCP server using the findscu command. It can pull data for a specified number of days (default 7000) and optionally filter by institution name. The script handles large datasets by implementing fallback strategies including date-based queries and patient ID pattern matching when bulk queries fail. Results are stored as individual DICOM files in organized directory structures under /tmp/allPatients[InstitutionName]

Related Files



Data Flow Diagram



Data paths

- **Input Paths:**

- /usr/share/dcmtdk/dicom.dic

- **Output Paths:**

- /tmp/allPatients[InstitutionName]/ - Main output directory
- /tmp/allPatients[InstitutionName]/all_at_once/
- /tmp/allPatients[InstitutionName]/YYYYMMDD/
- /tmp/allPatients[InstitutionName]/YYYYMMDD_[pattern]/

Script docstring starts here —>>> ->> ->>

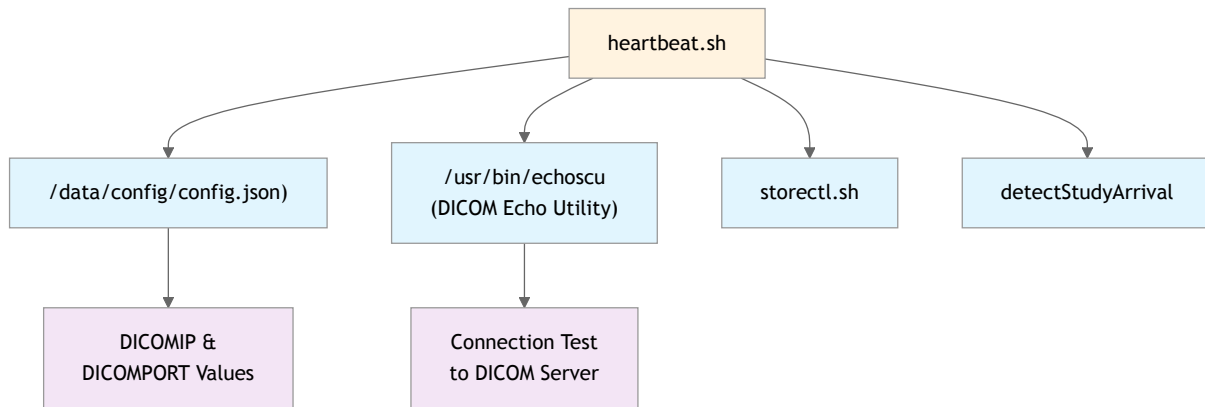
We can provide an argument to this program, the maximum number of days we would like to pull. In general we might get away with a very short period because new scans will come in as recent scans. But some test data might be very old. So we should do one long run at night and short runs during the day.

As a second argument allow a specific project name. The whole thing takes too long right now. Treat some project as special here.

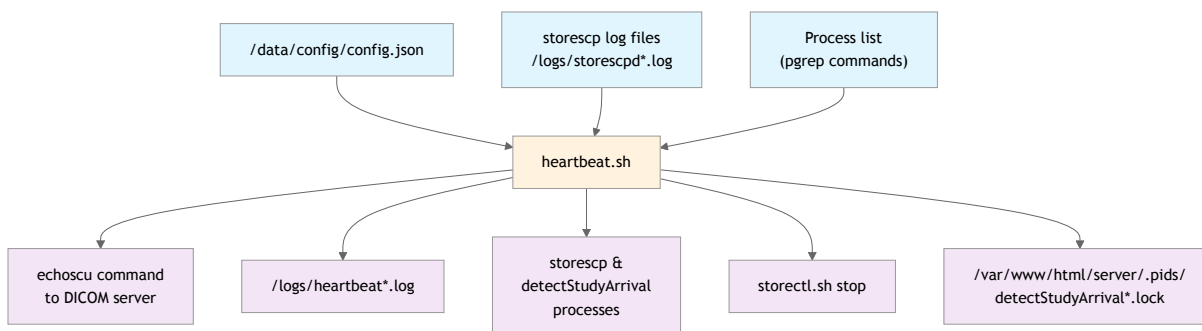
3.3.10 heartbeat.sh

heartbeat.sh is a monitoring script that performs health checks on DICOM storescp services by testing connectivity with echoscu. If the connection test fails, it terminates unresponsive storescp processes and cleans up stuck detectStudyArrival.sh jobs that have been running for over an hour. The script is designed to run via cron every minute and relies on external process managers like monit to restart killed services automatically.

Related Files



Data Flow Diagram



Data paths:

- **Input Folders/Files:**

- /data/config/config.json
- /var/www/html/server/logs/storescpd*.log

- **Output Folders/Files:**

- /var/www/html/server/logs/heartbeat*.log
 - /var/www/html/server/.pids/detectStudyArrival*.lock
-

Script docstring starts here —>>> ->> ->>

create a heart beat for the storescp

One way it can fail is if multiple associations are requested.

If the timeout happens the connection will be unusable afterwards.

Here we simply use echoscu to test the connection and if that fails we will kill a running storescp (hoping that monit will start it again).

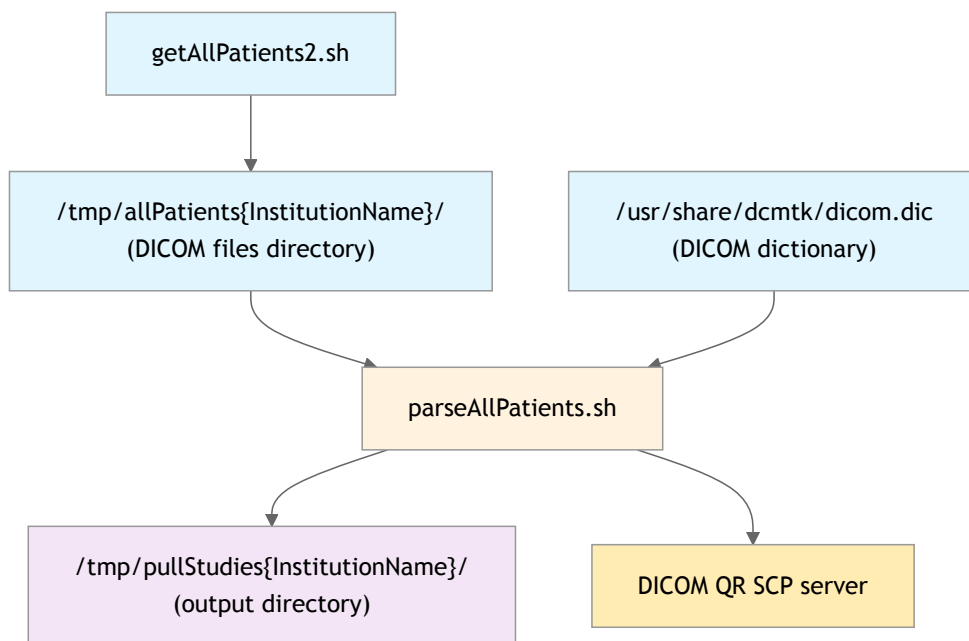
In order to activate put this into the crontab of processing (every minute):

```
*/1 * * * * /usr/bin/nice -n 3 /var/www/html/server/bin/heartbeat.sh
```

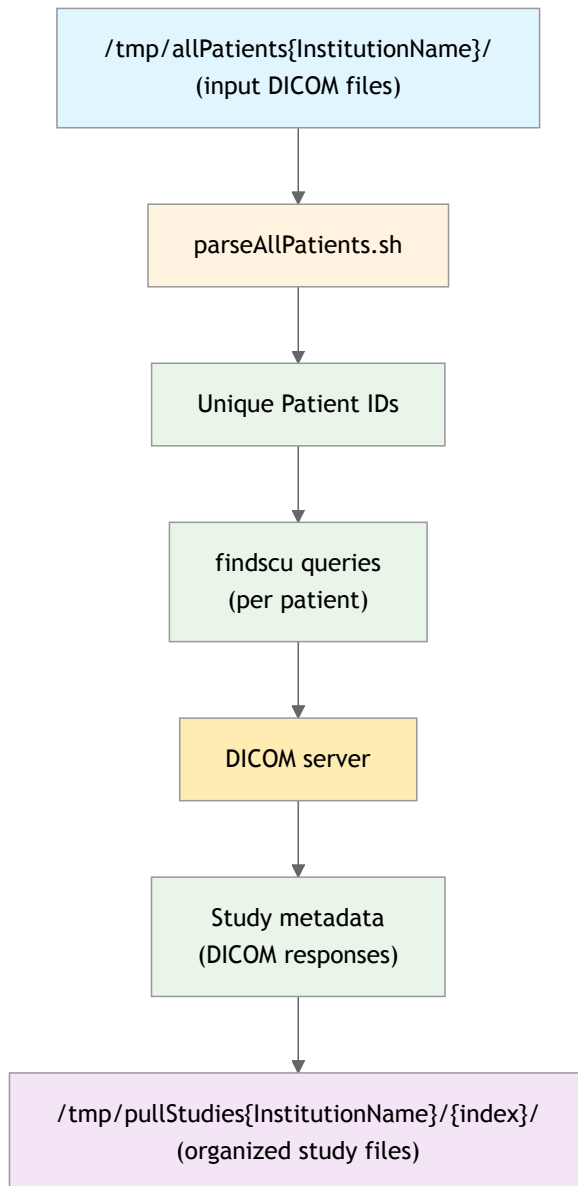
3.3.11 parseAllPatients.sh

This script processes DICOM files to extract unique patient IDs and retrieves study-level information for each patient from a DICOM Query/Retrieve service. It depends on output generated by `getAllPatients2.sh` and uses DCMTK tools to parse DICOM files and query a remote PACS server. The script creates a directory structure organizing study information by patient ID and connects to a DICOM server to pull comprehensive study metadata including modalities, dates, and institutional information.

Related Files



Data Flow Diagram



Data paths

- Input Directories:

- /tmp/allPatients{InstitutionName}/
- /usr/share/dcmtdk/dicom.dic

Output Directories:

- /tmp/pullStudies{InstitutionName}/
 - /tmp/pullStudies{InstitutionName}/{index}/
-

Script docstring starts here —>>> ->> ->>

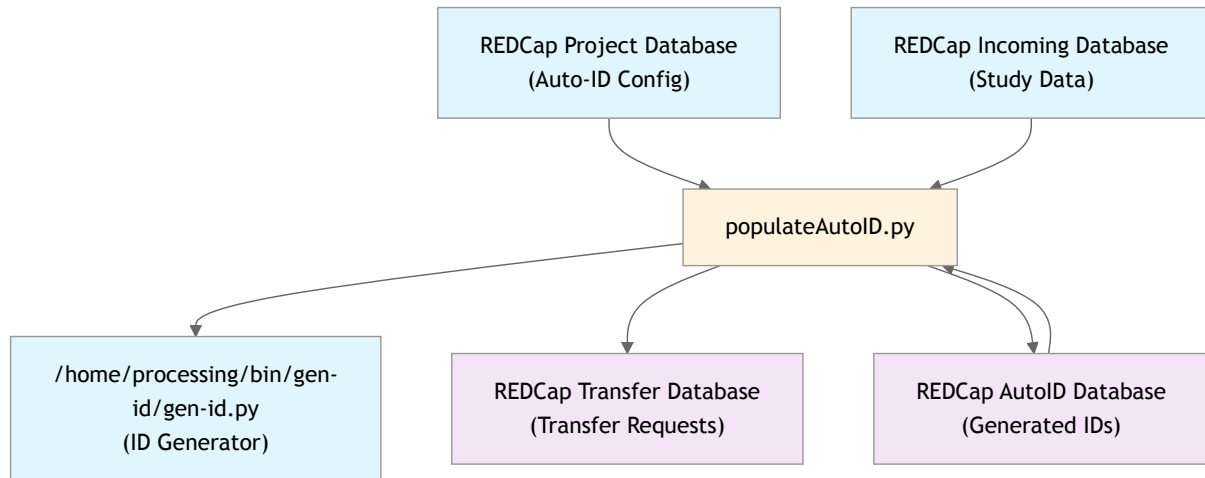
depends on output generated by getAllPatients2.sh

get list of optional findscu entries from http://dicom.nema.org/medical/dicom/current/output/html/part04.html#sect_C.6.1.1

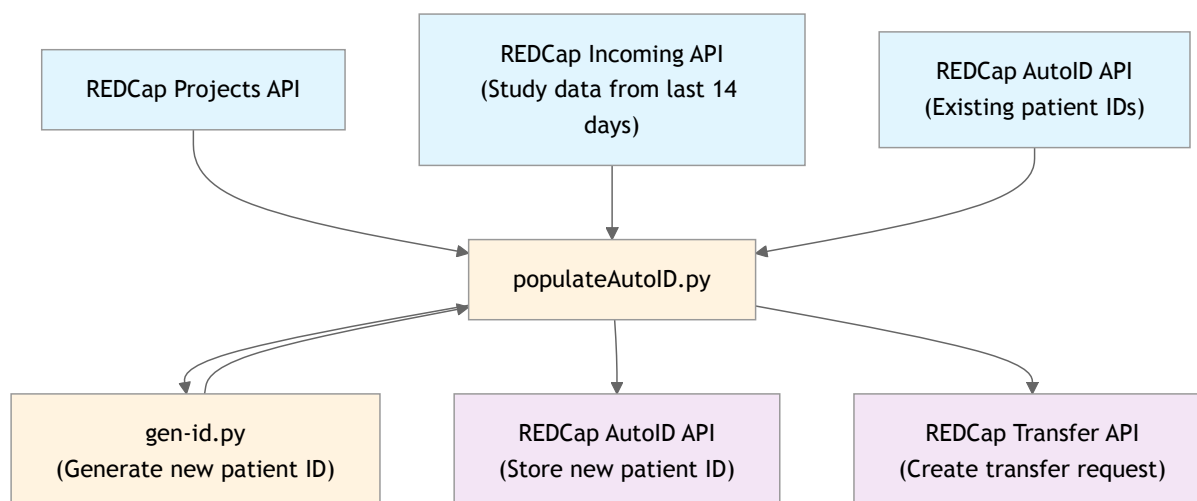
3.3.12 populateAutoID.py

This script automates the creation of transfer requests for medical imaging studies by checking auto-ID enabled projects in REDCap. It retrieves incoming DICOM data, generates or retrieves existing patient IDs using configurable naming patterns, and creates transfer requests for studies that don't already have them. The script interfaces with multiple REDCap databases using different API tokens and calls an external `gen-id.py` utility to generate new patient identifiers when needed.

Related Files



Data Flow Diagram



Data path

- Input Sources:

- `/home/processing/bin/gen-id/gen-id.py`
 - Output Destinations:
 - Temporary files (created and cleaned up automatically)
-

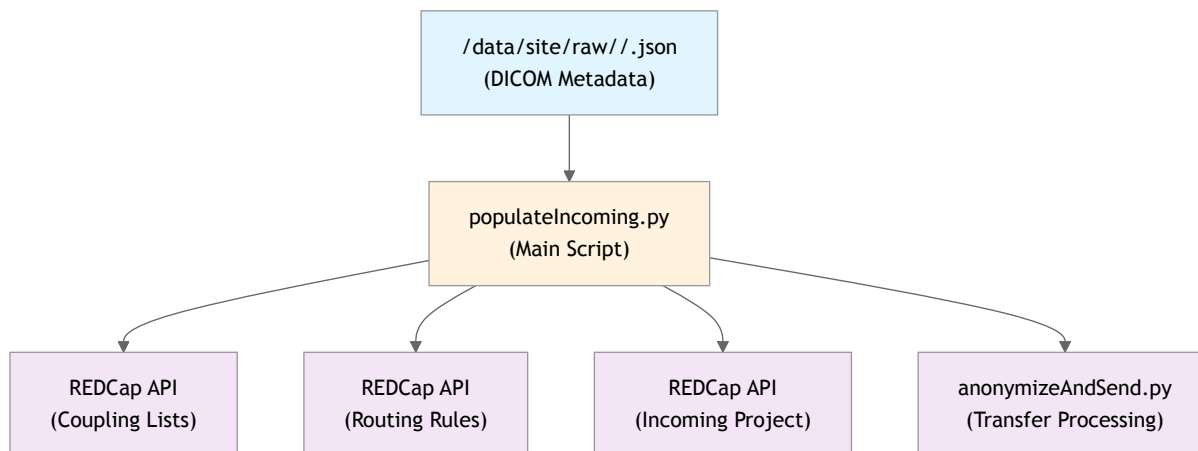
Script docstring starts here —>>> ->> ->>

Check all auto-id projects and create new transfer requests for each

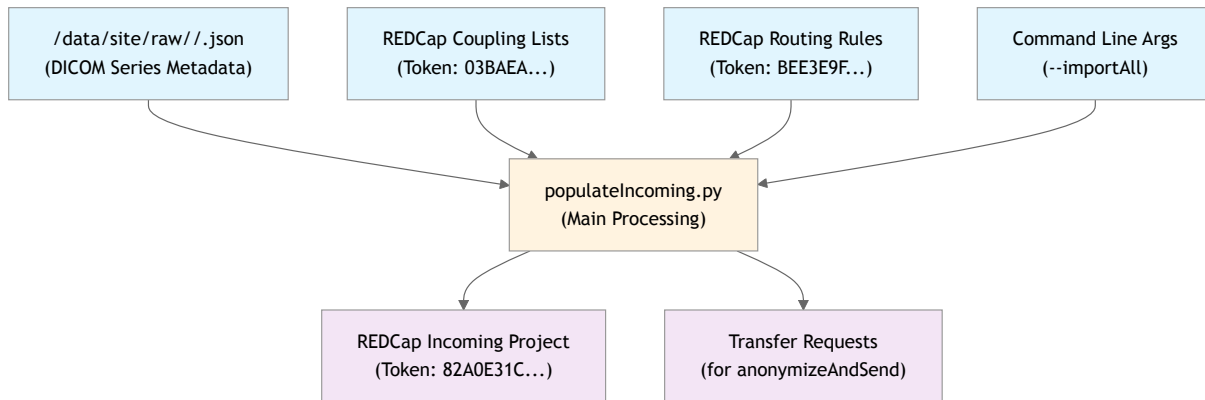
3.3.13 populateIncoming.py

This Python script populates the Study and Series information in the Incoming table in REDCap by processing DICOM metadata JSON files. It reads imaging series data from the filesystem, matches them against routing rules and coupling lists, and creates transfer requests for appropriate research projects. The script communicates with REDCap via API calls to store study/series metadata and generate transfer requests for data anonymization and distribution. It supports both incremental updates for new series and full project reimports via command-line arguments.

Related Files



Data Flow Diagram



Data patsh

- Input Paths

- /data/site/raw/*/*.json
 - Output Paths
 - RedCap endpoint
-

Script docstring starts here —>>> ->> ->>

This program fills in the Study and Series information in the Incoming table in REDCap.

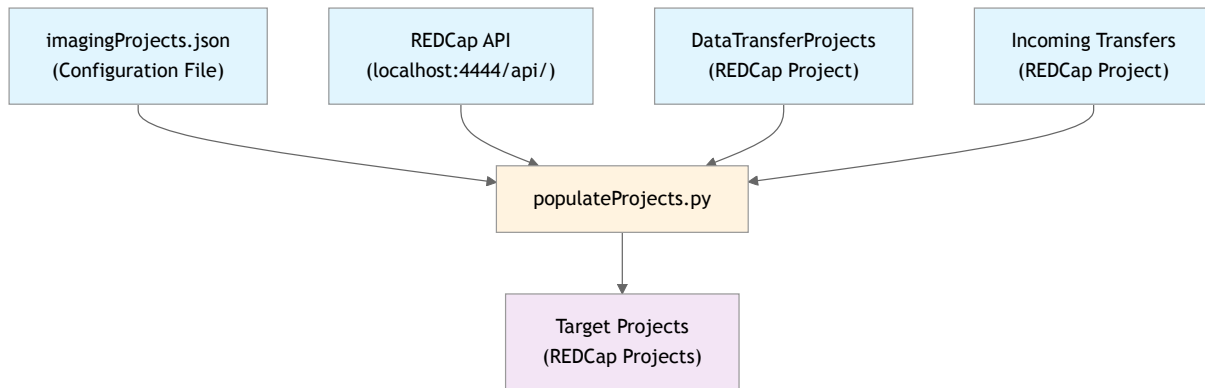
If a CouplingList entry exists its also adding a TransferRequest so that anonymizeAndSend can do its job.

TODO: support a new CouplingList entry even if there is already a TransferRequest done.

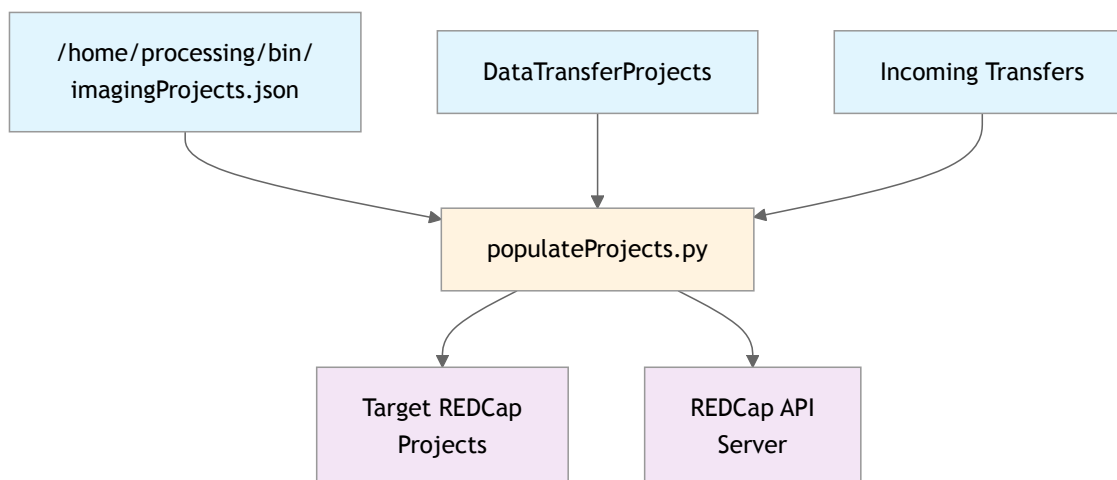
3.3.14 populateProjects.py

This Python script populates REDCap projects with imaging study information by retrieving transferred studies from incoming transfers and creating corresponding entries in target projects. The script fetches transfer data from a central REDCap database, matches it with project tokens, and creates participant records with associated imaging instrument data for each study that has been forwarded to PACS. It processes transfers either for all active projects or a specific project when the `-project` parameter is provided, ensuring each study appears in its own REDCap project with proper repeat instance management.

Related Files



Data Flow Diagram



Data paths

- Input Paths:

- `/home/processing/bin/imagingProjects.json`
 - Output Destinations:
 - Multiple REDCap projects (determined dynamically from `DataTransferProjects`)
-

Script docstring starts here —>>> ->> ->>

Each study that has been forwarded to PACS should appear in its own REDCap project.

We can get a list of all transferred studies from incoming transfers. We get a token for the project and add the entry - if it does not exist yet.

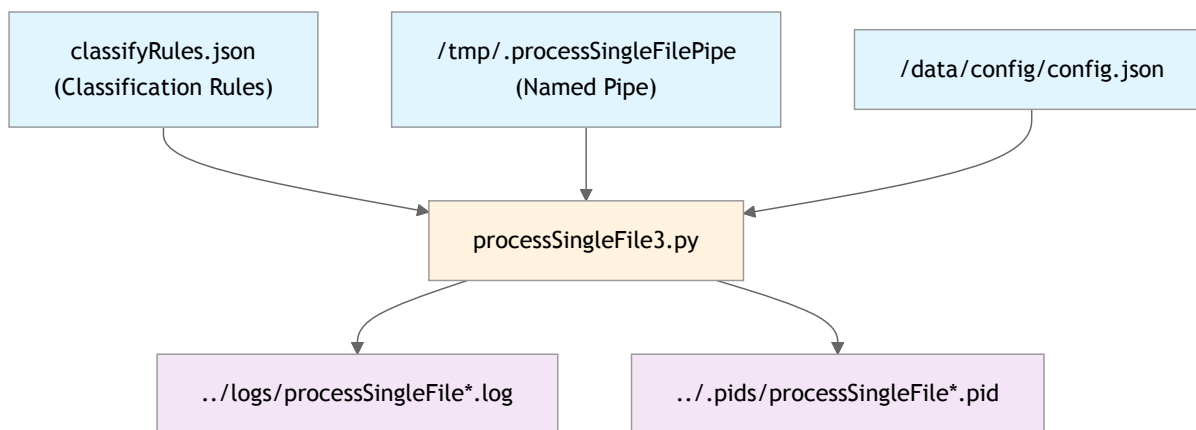
For informatino to appear in the Imaging instrument you need to set it up as a repeating instrument for “Event 1” (not a repeating event!).

TODO: Without calling for specific projects does not work anymore. We need to get a list of all imaging projects and run them project by project.

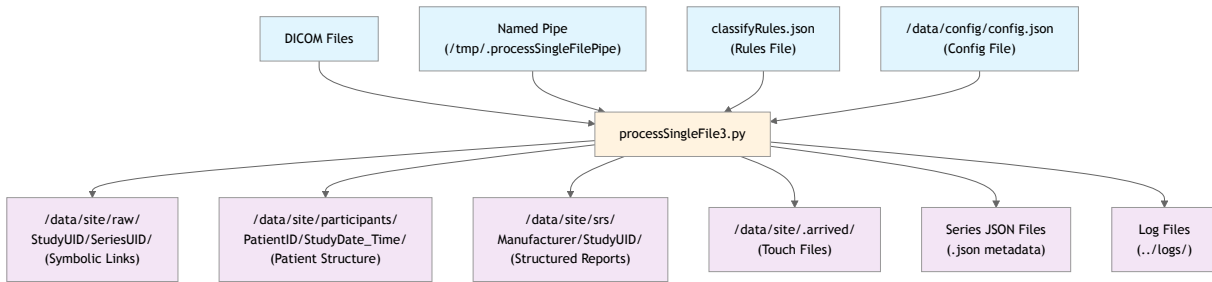
3.3.15 processSingleFile3.py

`ProcessSingleFile3.py` is a daemon process that monitors a named pipe for DICOM file processing requests and creates organized directory structures with symbolic links. The daemon reads DICOM files, extracts header information including Siemens CSA headers, and organizes them into Study/Series hierarchies while generating JSON metadata files. It supports classification rules for automatic categorization of medical imaging data and handles structured reports separately from regular imaging data.

Related Files



Data Flow Diagram



Data paths

- Input Paths:

-
- /data/config/config.json - Configuration file containing project settings classifyRules.json - Classification rules file (same directory as script)
 - /tmp/.processSingleFilePipe[projname] - Named pipe for receiving file processing requests
 - Source DICOM files (paths received via named pipe)
 - Output Paths:
 - /data/site/raw/[StudyUID]/[SeriesUID]/ - Organized DICOM structure with symbolic links
 - /data/site/participants/[PatientID]/[StudyDate_StudyTime]/ - Patient-oriented directory structure
 - /data/site/srs/[Manufacturer]/[StudyUID]/ - Structured reports directory
 - /data/site/.arrived/ - Touch files for series arrival detection
 - /data/site/temp/ - Temporary directory for atomic JSON file operations
 - ../logs/processSingleFile[projname].log - Log files
 - ../.pids/processSingleFile[projname].pid - Process ID files
 - Series JSON metadata files (.json files alongside DICOM directories)
-

Script docstring starts here —>>> ->> ->>

Create a daemon process that listens to send messages and reads a DICOM file, extracts the header information and creates a Study/Series symbolic link structure.

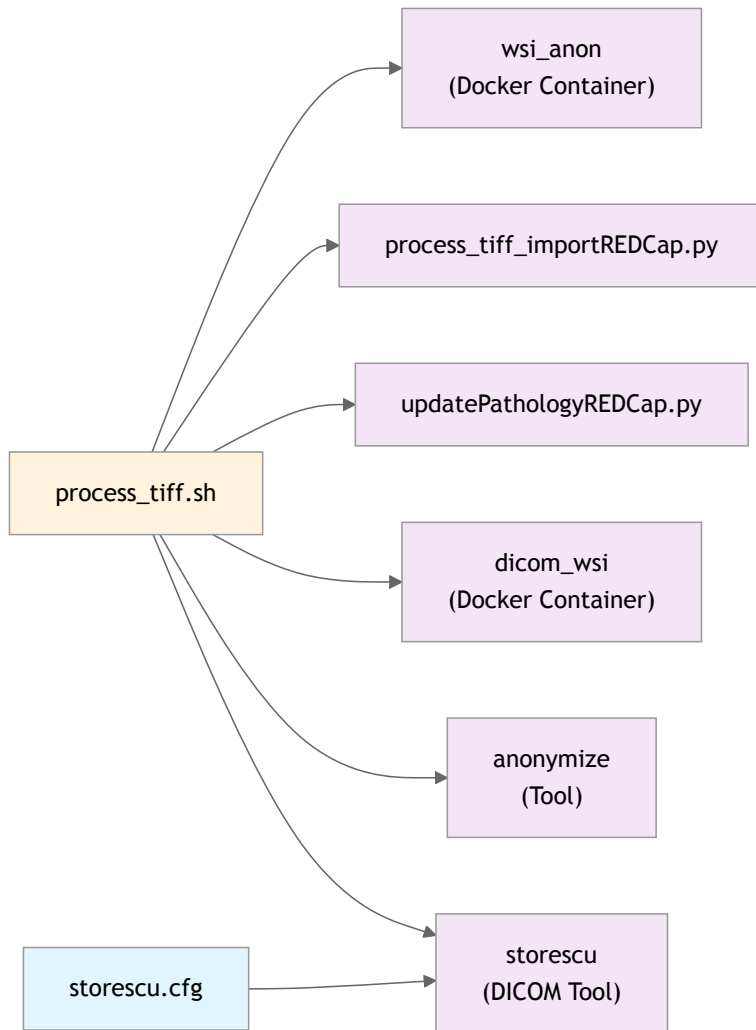
The parser for the Siemens CSA header have been adapted from

https://scion.duhs.duke.edu/svn/vespa/tags/0_1_0/libduke_mr/util_dicom_siemens.py

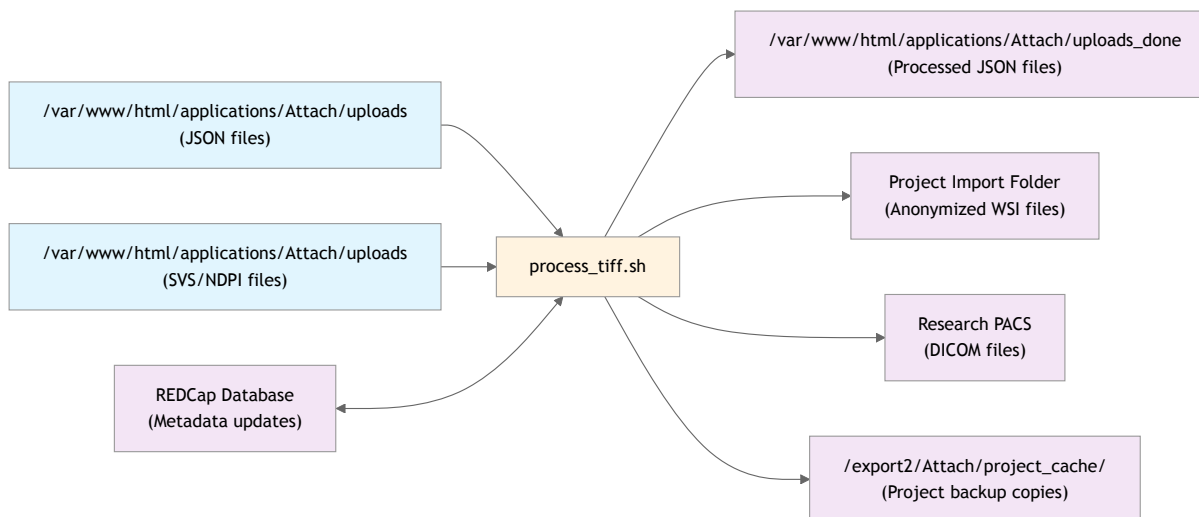
3.3.16 process_tiff.sh

The `process_tiff.sh` script processes pathology image files (SVS and NDPI formats) by extracting metadata, anonymizing the images, and either importing them directly to a PACS system or storing them in a designated project folder. It handles two workflow paths: path-based imports where files are pseudonymized and stored locally, and DICOM conversion where files are converted to DICOM format, anonymized, and sent to a research PACS. The script processes JSON metadata files from REDCap uploads, extracts scanner information, and updates the database with processing results.

Related Files



Data Flow Diagram



Data pats

- Input Paths:

- /var/www/html/applications/Attach/uploads - Source JSON and image files
 - /var/www/html/applications/Attach/storescu.cfg - DICOM configuration file
 - Output Paths:
 - /var/www/html/applications/Attach/uploads_done - Processed JSON files Project-specific import folders (defined in JSON: project_pat_import_folder)
 - /export2/Attach/project_cache/{InstitutionName}/ - Optional project backup cache Research PACS server - DICOM storage destination
 - Temporary Paths:
 - /tmp/ - Temporary processing files Dynamically created temp directories for DICOM conversion and anonymization
-

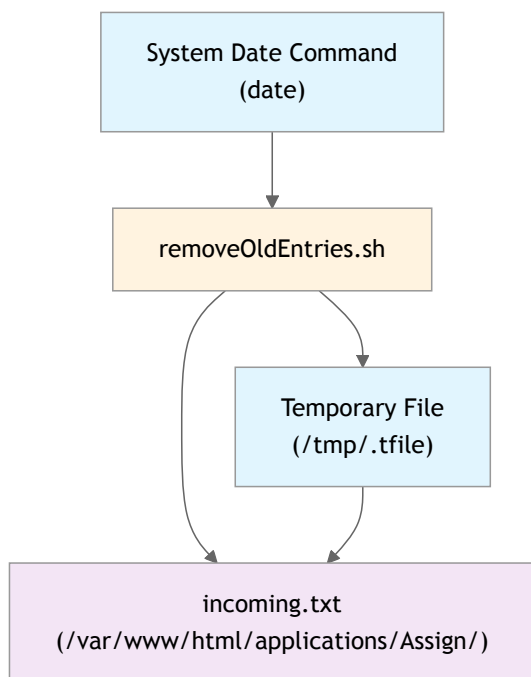
Script docstring starts here —>>> ->> ->>

To be updated.

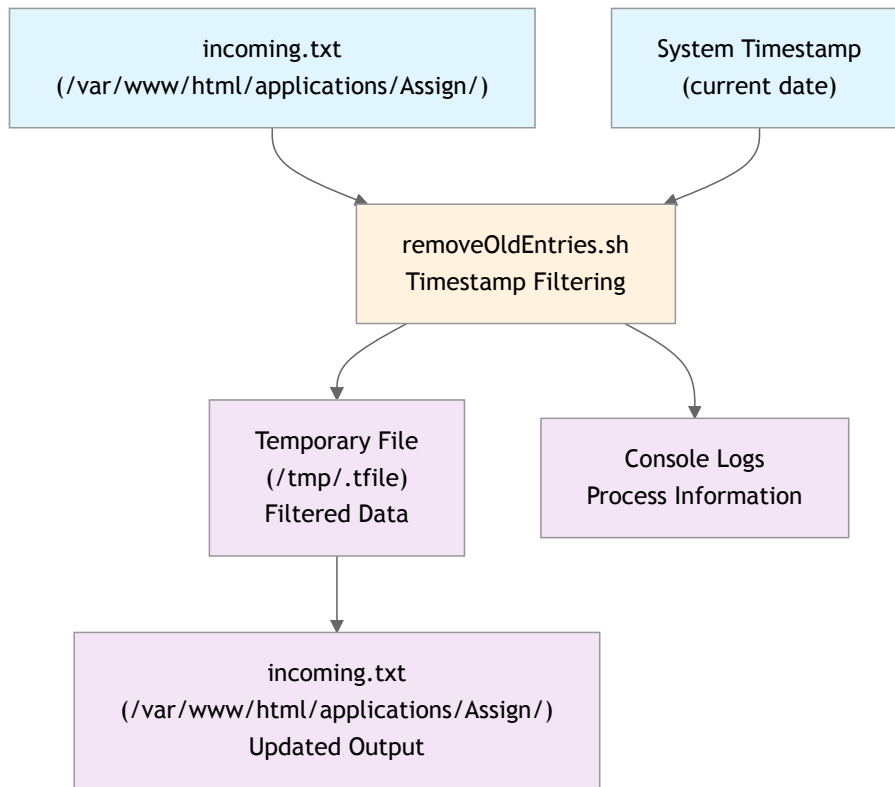
3.3.17 removeOldEntries.sh

This bash script removes old entries from the `incoming.txt` file based on timestamp comparison. It reads each line from the file, extracts the first field as a timestamp, and keeps only entries that are newer than 7 days (604800 seconds). The script uses a temporary file for atomic file operations and provides detailed logging of the cleanup process.

Related Files



Data Flow Diagram



Data paths

- Input/Output Directories:

- /var/www/html/applications/Assign/incoming.txt
 - /tmp/.tfile
-

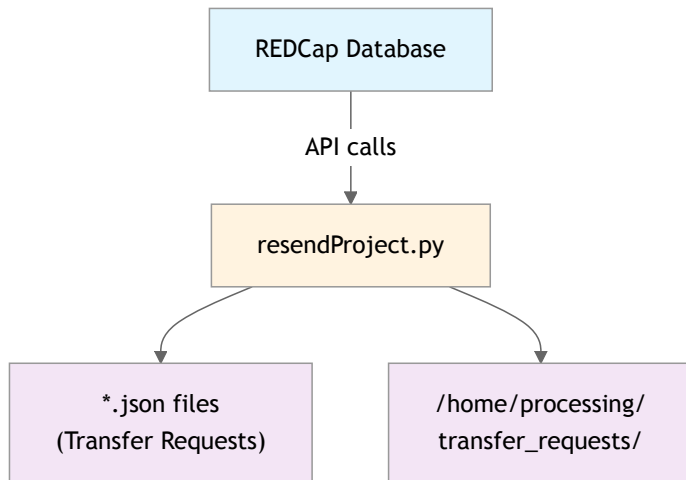
Script docstring starts here —>>> ->> ->>

Remove any old entries from incoming.txt

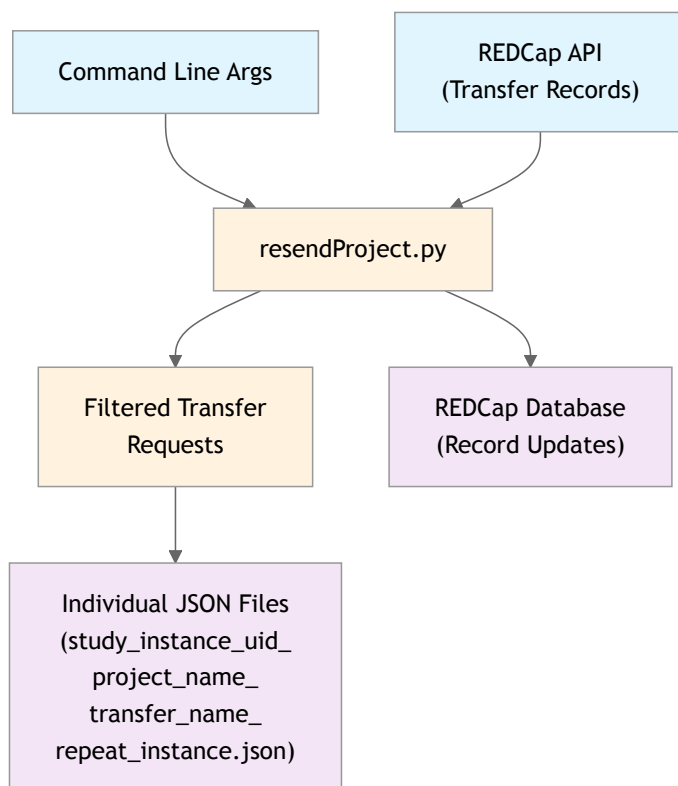
3.3.18 resendProject.py

This Python script manages medical imaging transfer requests by checking REDCap database records for studies where the transfer date occurred before the request date. It retrieves transfer requests from a REDCap API, filters them based on date logic to identify studies that need to be resent, and generates JSON transfer request files for reprocessing. The script supports filtering by specific project names and creates uniquely named JSON files in a designated transfer requests directory.

Related Files



Data Flow Diagram



Data paths

- Input Paths:

- REDCap database
 - Output Paths:
 - ``/home/processing/transfer_requests/`
 - `{study_instance_uid}_{transfer_project_name}_{transfer_name}_{redcap_repeat_instance}.json`
-

Script docstring starts here —>>> ->> ->>

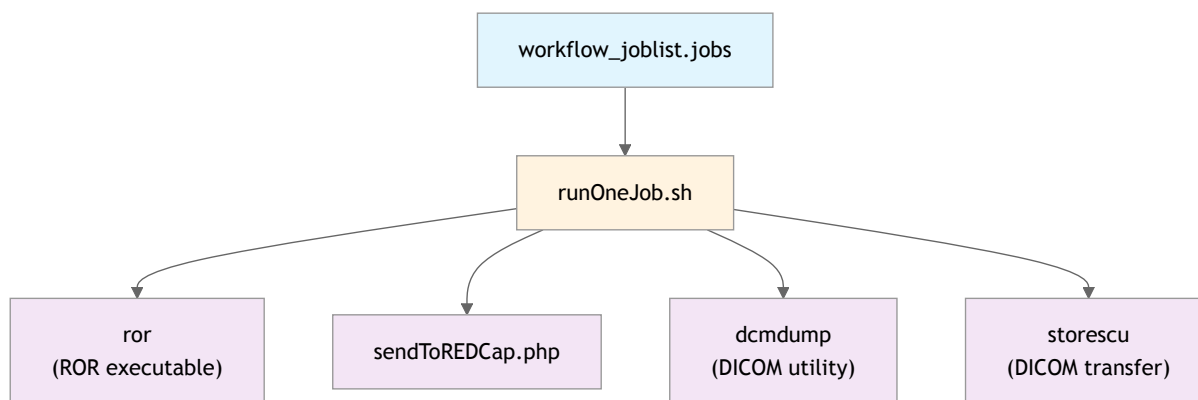
check all transfer requests that have already been done

if the send date is before the request date send again

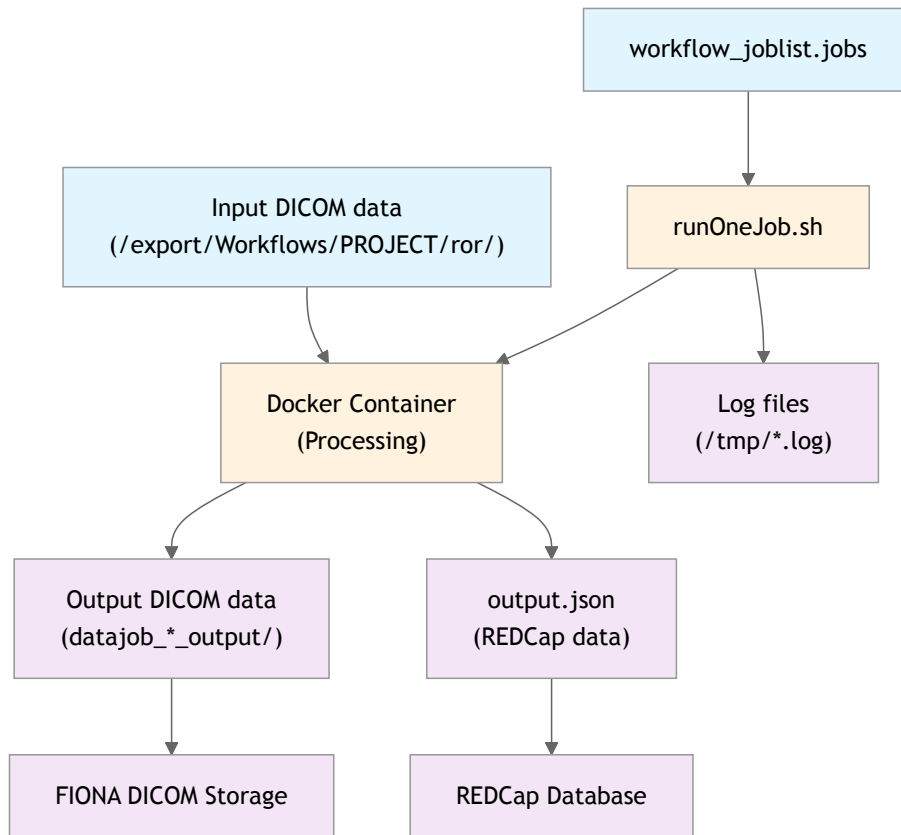
3.3.19 runOneJob.sh

The `runOneJob.sh` script processes workflow jobs by reading JSON job definitions from a queue file and executing containerized processing tasks using the ROR (Run on Request) system. It validates job parameters, creates working directories, executes Docker containers with specified images, and handles output validation by comparing DICOM StudyInstanceUIDs between input and output folders. Upon successful completion, it sends processed DICOM data to a FIONA storage system and optionally transfers results to REDCap, then removes completed jobs from the queue.

Related Files



Data Flow Diagram



Data Paths

- Input Paths:

- /var/www/html/applications/Workflows/php/workflow_joblist.jobs
- /export/Workflows/{PROJECT}/ror/
- /var/www/html/applications/Workflows/php/ror

- Output Paths:

- /export/Workflows/{PROJECT}/ror/datajob_{JOB_NUMBER}_{IMAGE_NAME}_*
- /export/Workflows/{PROJECT}/ror/datajob_{JOB_NUMBER}_{IMAGE_NAME}_*_output/
- /tmp/{DIRECTORY_NAME}run.log
- /home/processing/logs/Workflows_RunOneJob.log

Script docstring starts here —>>> ->> ->>

Run a single job from the workflow_joblist.jobs file. The file contains json code per line.

Need to run as user “processing” with flock.

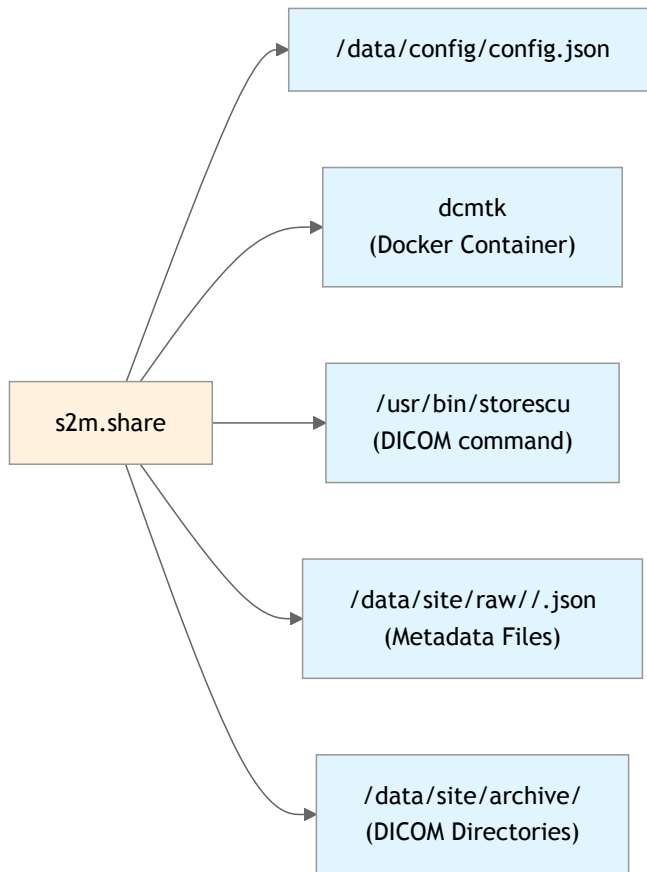
Example cron job:

```
/usr/bin/flock -n /home/processing/.pids/Workflows_RunOneJob.pid
/var/www/html/applications/Workflows/php/runOneJob.sh >> /home/processing/logs/Workflows_RunOneJob.log
2>&1
```

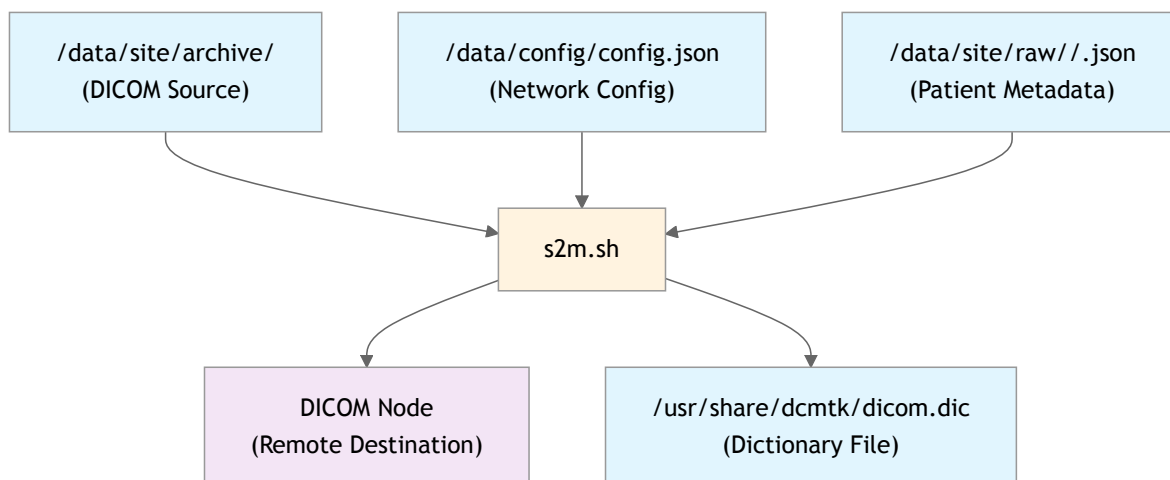
3.3.20 s2m.sh

The s2m.sh script is a DICOM file transmission utility that sends DICOM directories to a local DICOM node using the dcmtool docker container. It supports sending individual directories, all studies for a specific PatientID, or all studies from the last N days. The script reads DICOM network configuration from /data/config/config.json and can handle project-specific routing. It includes fallback mechanisms using direct storescu commands if Docker-based transmission fails.

Related Files



Data Flow Diagram



Data paths

- Input Paths:

- /data/config/config.json
 - /data/site/archive/
 - /data/site/raw/*/*.json
 - /usr/share/dcmtd/dicom.dic
 - Output Paths:
 - Remote destination for DICOM files, no local output files.
-

- send to me (s2m)
- Sends a DICOM directory using the dcmtd docker container from the local machine to the local DICOM node. This script can be used to re-classify DICOM files (creates /data/site/raw and /data/site/participant information).

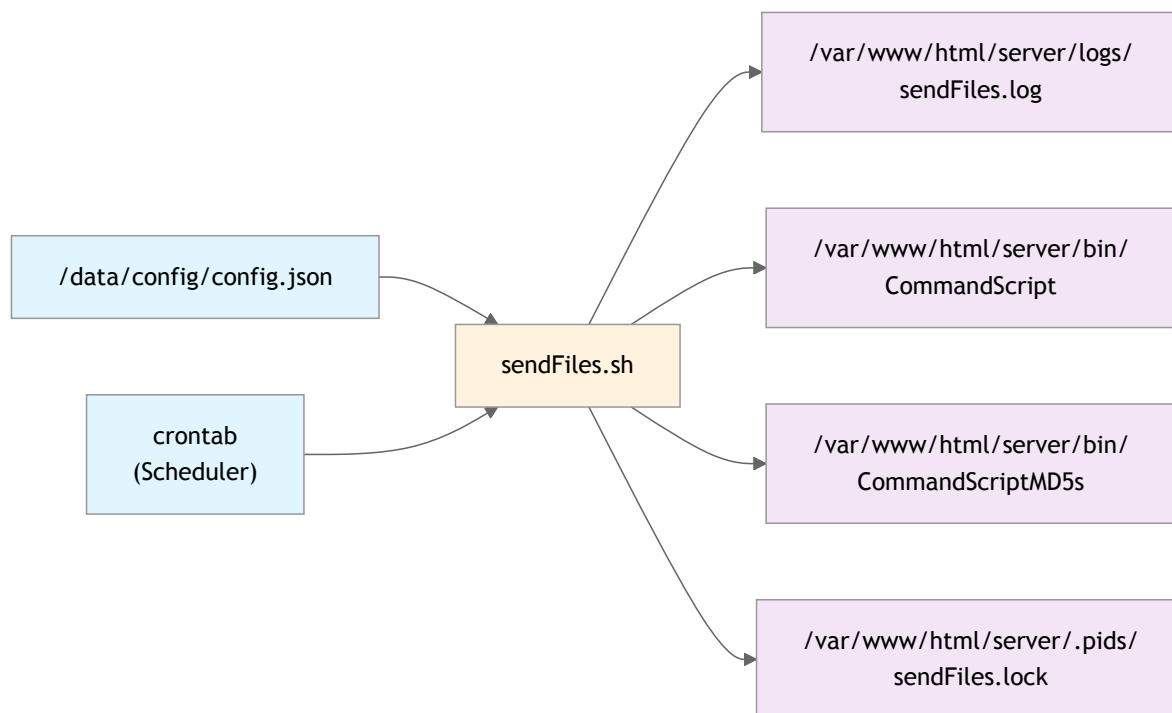
Usage:

- # Send a single directory with DICOM files s2m.sh <DICOM directory to send> [project]
- # Send all studies of a single PatientID s2m.sh <PatientID> [project]
- # send all studies of the last 7 days s2m.sh last 7 [project]

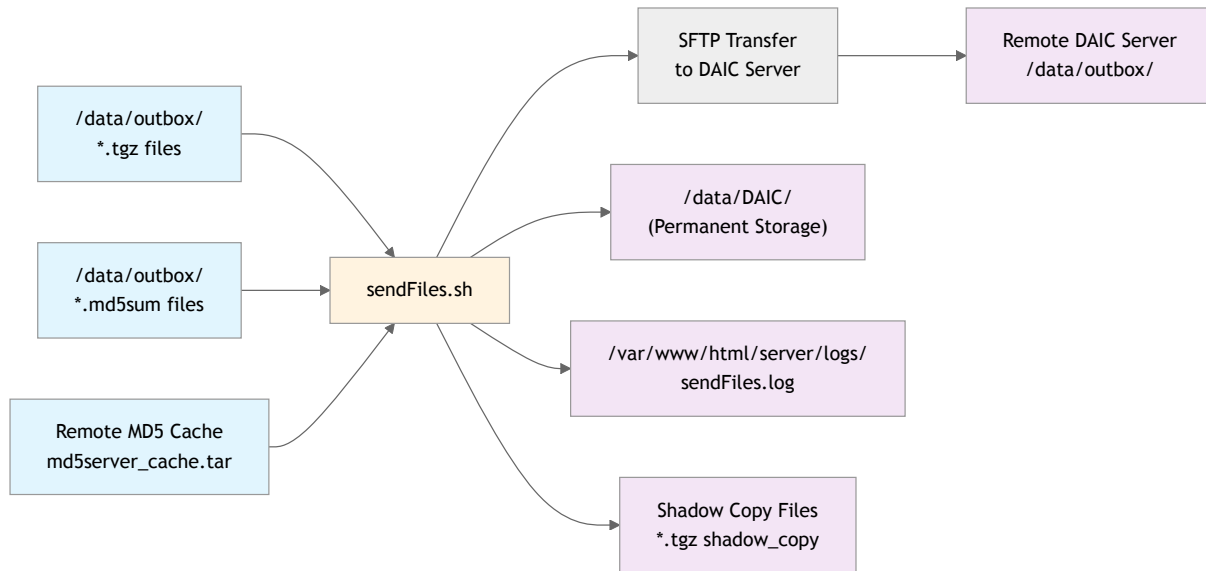
3.3.21 sendFiles.sh

This bash script automates the secure transfer of compressed DICOM and k-space data files from a local outbox directory to a remote DAIC (Data Analysis and Informatics Center) server using SFTP. The script compares local and remote MD5 checksums to avoid redundant transfers and implements file locking to prevent concurrent executions. It includes error handling for corrupt checksums, symbolic link conflicts, and disk space issues, with comprehensive logging of all operations.

Related Files



Data Flow Diagram



Data Paths

- Input Directories:

- /data/outbox/
- /data/config/config.json
- Output Directories:
 - /data/DAIC/
 - /var/www/html/server/logs/
 - /var/www/html/server/.pids/
 - Remote DAIC server endpoint (specified in config.json)
- Temporary Directories:
 - /tmp/md5sums_server_XXXX

Script docstring starts here —>>> ->> ->>

Example crontab entry that starts this script every 30 minutes

```
*/30 * * * * /usr/bin/nice -n 3 /var/www/html/server/bin/sendFiles.sh
```

Add the above line to your machine using:

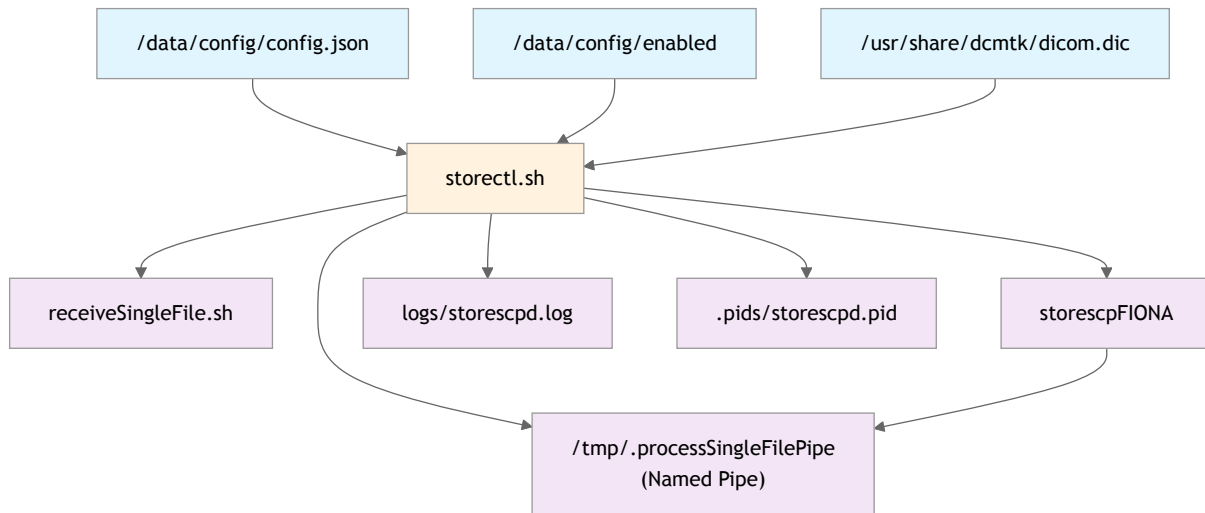
```
> crontab -e
```

This script is supposed to send compressed data files for DICOM and k-space to the DAIC endpoint using sftp. All data in the /data/outbox directory will be send using local and DAIC md5sum files.

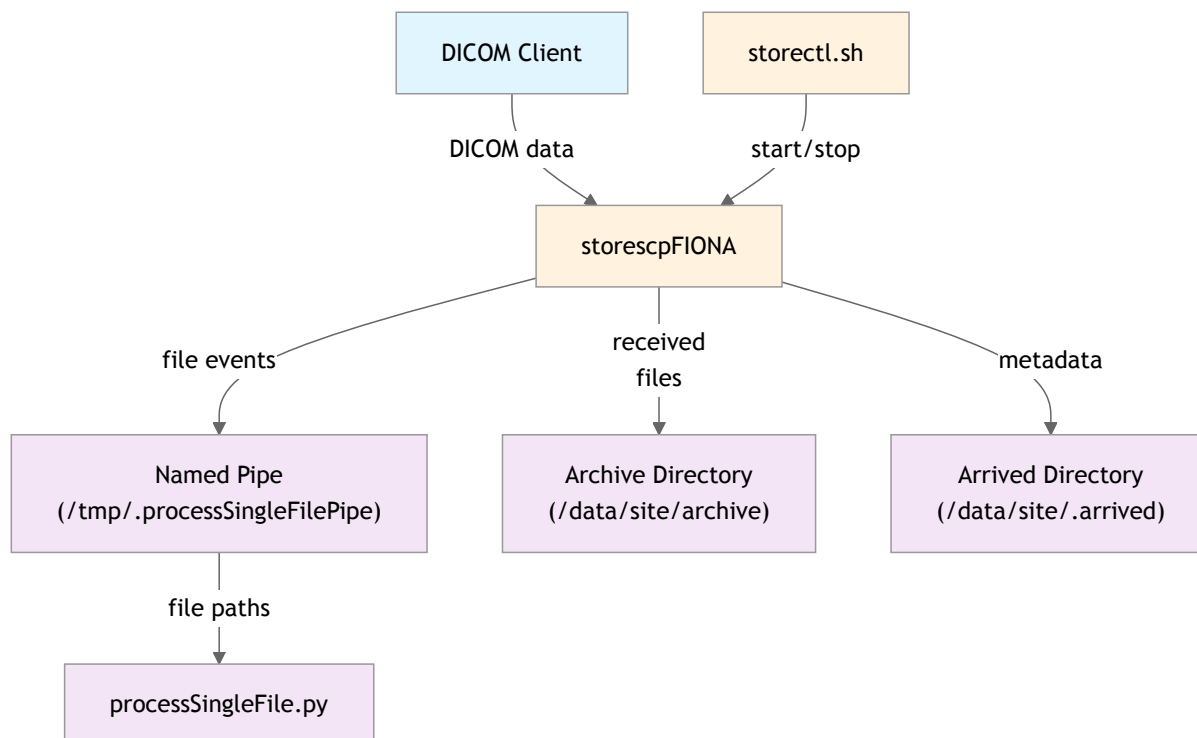
3.3.22 storectl.sh

This script manages a DICOM storage service (storescp daemon) that receives medical imaging data on a specified port. It starts or stops the storescpFIONA service for the processing user, which listens for incoming DICOM files and moves them to project-specific directories. The service can be controlled via an enabled/disabled flag file and supports multiple projects with different configurations (ABCD as default).

Related Files



Data Flow Diagram



Data Paths

- Input paths:

- /data/config/
 - /usr/share/dcm2tk/
 - /var/www/html/server/
 - Output paths (data saved to):
 - /data/site/archive/
 - /data/site/.arrived/
 - /var/www/html/server/logs/
 - /var/www/html/server/.pids/
 - /tmp/
-

Script docstring starts here —>>> ->> ->>

filename: storescpd

purpose: start storescp server for processing user at boot time to receive data | Move files to project specific file system

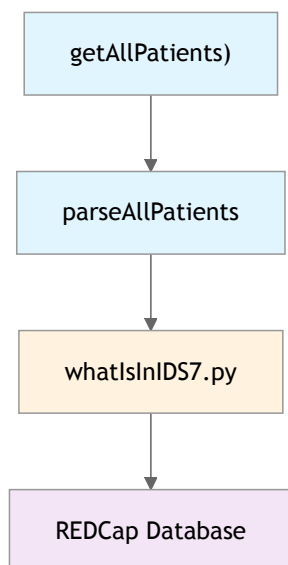
This system service will fail if a control file /data/enabled exists and its first character is a “0”.

(Hauke Bartsch)

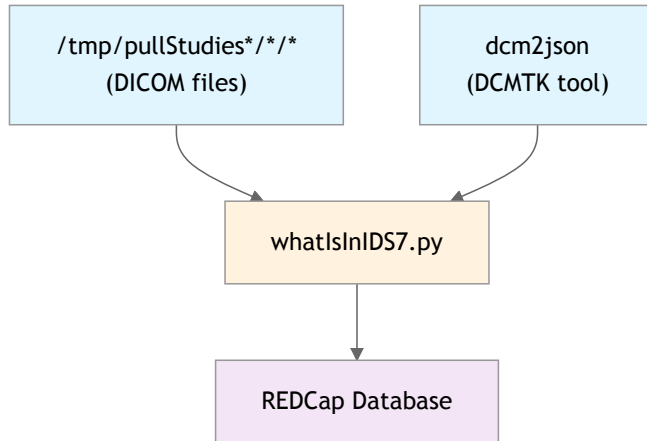
3.3.23 whatIsInIDS7.py

The `whatIsInIDS7.py` script processes DICOM medical imaging files from the IDS7 research PACS system and extracts metadata to populate a REDCap database. It parses DICOM files using `dcm2json`, extracts key study information (patient data, study details, series counts), handles duplicate studies by merging data, and uploads the processed information to REDCap via API calls. The script can process all studies or filter by institution name when provided as a command-line argument.

Related Files



Data Flow Diagram



Data Paths:

- Input paths:

- /tmp/pullStudies{InstitutionName}/*/* - DICOM files directory structure Command line arguments: sys.argv[1] (optional institution name filter)
 - Output paths:
 - REDCap API endpoint
 - External dependencies:
 - dcm2json - DCMTK toolkit
 - /usr/share/dcm2k/dicom.dic
-

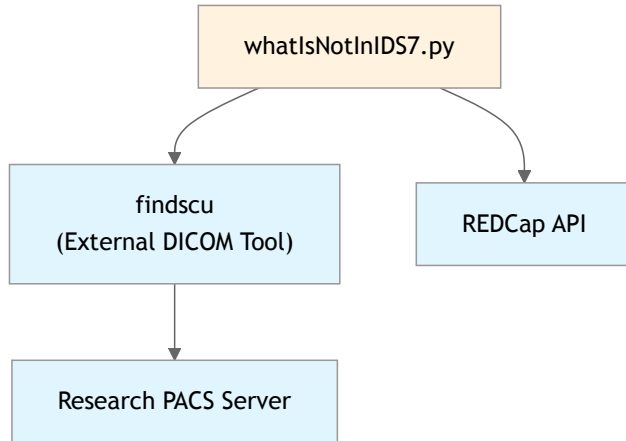
Script docstring starts here —>>> ->> ->>

TODO: check if the number of study related series is correct (looks too large in Export app)

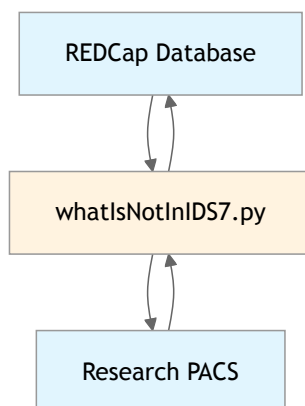
3.3.24 whatIsNotInIDS7.py

This script cleans up a REDCap database by removing records that no longer exist in the research PACS system. It queries the REDCap project “whatIsInIDS7” to retrieve all stored records, then validates each record’s existence in the research PACS using DICOM findscu commands. Records that are not found in the PACS (indicated by zero SeriesInstanceUID occurrences) are marked for deletion and optionally removed from REDCap in batches of 200.

Related Files



Data Flow Diagram



Data Paths

- Input Source:

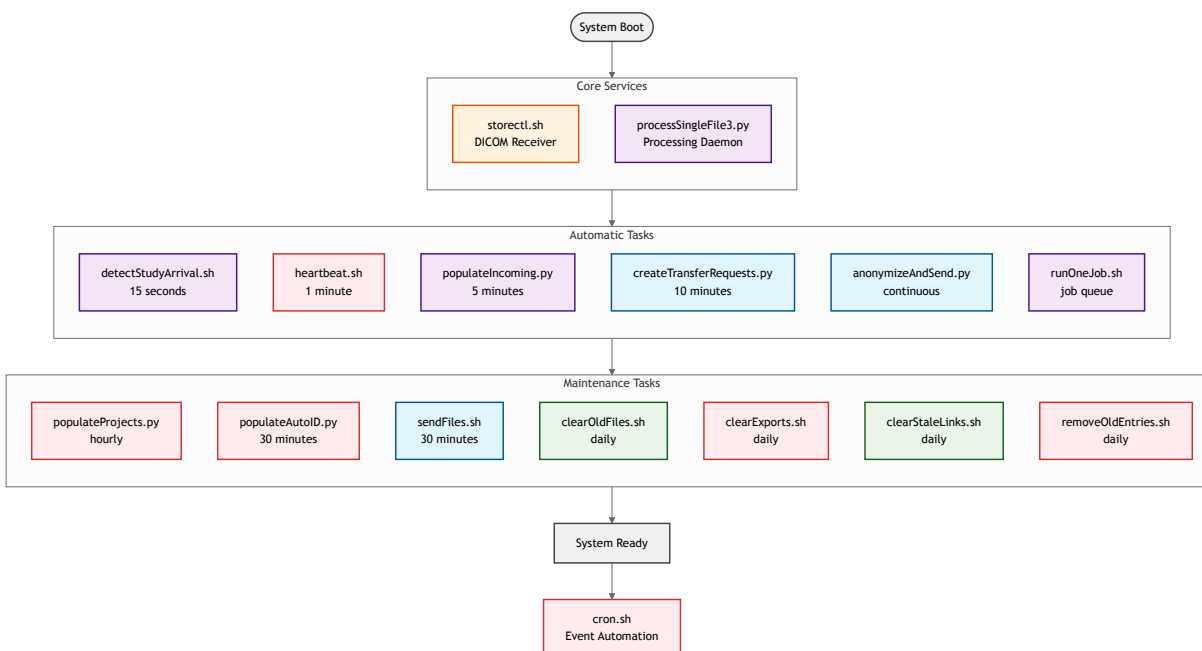
- REDCap API
 - Output Destination:
 - REDCap API
-

Script docstring starts here —>>> ->> ->>

There is no docstring.

1. *anonymizeAndSend.py* - Processes imaging studies, performs anonymization, and sends them to research PACS
2. *clearExports.sh* - Removes old export files when storage reaches capacity thresholds
3. *clearOldFiles.sh* - Removes old studies from /data/site/archive when disk usage exceeds 80%
4. *clearStaleLinks.sh* - Removes broken symbolic links and empty directories from data structures
5. *createTransferRequests.py* - Generates transfer requests for studies that need anonymization and forwarding to research projects
6. *createTransferRequestsForProcessed.py* - Handles transfer requests for processed/derived imaging data from workstations back to research PACS
7. *createZipFileCmd.php* - Creates anonymized ZIP archives for research data distribution
8. *cron.sh* - Processes trigger-action pairs from JSON configuration files for event-driven automation
9. *getAllPatients2.sh* - Retrieves patient and study information from research PACS using findscu
10. *heartbeat.sh* - Checks DICOM service responsiveness and restarts failed components
11. *parseAllPatients.sh* - Parses patient data retrieved by getAllPatients2.sh and extracts study-level metadata for REDCap import
12. *populateAutoID.py* - Generates automatic participant IDs for projects using pseudonymized identifiers
13. *populateIncoming.py* - Processes incoming DICOM studies and creates metadata records in REDCap
14. *populateProjects.py* - Populates individual research project databases with distributed data
15. *processSingleFile3.py* - Extracts metadata from DICOM files and creates directory structures
16. *process_tiff.sh* - Converts whole slide imaging (WSI) files to DICOM format for pathology processing
17. *removeOldEntries.sh* - Removes old entries from incoming data tracking files
18. *resendProject.py* - Handles re-transmission of studies when initial transfers fail or new data arrives
19. *runOneJob.sh* - Processes containerized analysis jobs from job queue
20. *s2m.sh* - Re-sends DICOM directories through the processing pipeline for re-classification
21. *sendFiles.sh* - Uploads anonymized data to external research repositories via secure file transfer
22. *storectl.sh* - Manages the main DICOM C-STORE receiver daemon
23. *whatIsInIDS7.py* - Catalogs all studies present in the research imaging database
24. *whatIsNotInIDS7.py* - Identifies and removes database entries for studies no longer in PACS

3.4 System setup pipeline



Download image as PDF

3.5 Data flow

Data flow scheme through the FIONA system, from initial DICOM reception to final transfer to research PACS, is presented below.

Legend:

3.6 Data Storage Structure

The FIONA system uses a hierarchical storage structure:

```
/data/
├── site/
│   ├── .arrived/      # Initial file reception
│   ├── archive/       # Raw DICOM storage
│   ├── raw/           # Processed DICOM files
│   └── output/        # Processing results
├── config/            # Configuration files
└── logs/              # System logs
```

Project-specific directories follow the pattern: /data{PROJECT}/site/...

CONTACT INFORMATION

- Website: <https://fiona.ihelse.net>
- Location: Haukeland University Hospital, Bergen, Norway

INDICES AND TABLES

- genindex
- modindex
- search