Pacifica Policy Documentation

David Brown

Jan 29, 2019

Contents:

1 Installation **3**
1.1 Installation in Virtual Environment 3

2 Configuration **5**
2.1 CherryPy Configuration File 5
2.2 Service Configuration File 5
2.3 Starting the Service 6

3 Example Usage **9**
3.1 The API 9
3.2 Admin Command Line 11

4 Policy Python Module **13**
4.1 Events Python Module 13
4.2 Ingest Python Module 13
4.3 Reporting Python Module 14
4.4 Status Python Module 15
4.5 Uploader Python Module 18
4.6 Admin Python Module 18
4.7 Admin Command Python Module 19
4.8 Config Python Module 19
4.9 Data Release Python Module 19
4.10 Globals Python Module 20
4.11 Root Rest Python Module 20
4.12 Search Render Python Module 20
4.13 Search Sync Python Module 22
4.14 Validation Python Module 22
4.15 WSGI Python Module 23

5 Indices and tables **25**

Python Module Index **27**

The Pacifica Policy service provides endpoints that define policy questions for institutions. This is separate from other services as certain operations required by other Pacifica Core services are more Policy base.

Practially speaking, when the question a Pacifica service wants to ask the Metadata service is sufficiently complex it should really be a Policy question. For example, when uploading data the ingest service needs to validate the metadata requesting to be added. This new metadata needs to be verified by some institutional requirements. So there is a Policy endpoint (several actually) that help ensure those requirements are met.

CHAPTER 1

Installation

The Pacifica software is available through PyPi so creating a virtual environment to install is what is shown below. Please keep in mind compatibility with the Pacifica Core services.

1.1 Installation in Virtual Environment

These installation instructions are intended to work on both Windows, Linux, and Mac platforms. Please keep that in mind when following the instructions.

Please install the appropriate tested version of Python for maximum chance of success.

1.1.1 Linux and Mac Installation

```
mkdir ~/.virtualenvs
python -m virtualenv ~/.virtualenvs/pacifica
. ~/.virtualenvs/pacifica/bin/activate
pip install pacifica-policy
```

1.1.2 Windows Installation

This is done using PowerShell. Please do not use Batch Command.

```
mkdir "$Env:LOCALAPPDATA\virtualenvs"
python.exe -m virtualenv "$Env:LOCALAPPDATA\virtualenvs\pacifica"
& "$Env:LOCALAPPDATA\virtualenvs\pacifica\Scripts\activate.ps1"
pip install pacifica-policy
```

CHAPTER 2

Configuration

The Pacifica Core services require two configuration files. The REST API utilizes CherryPy and review of their configuration documentation is recommended. The service configuration file is a INI formatted file containing configuration for database connections.

2.1 CherryPy Configuration File

An example of Policy server CherryPy configuration:

```
[global]
log.screen: True
log.access_file: 'access.log'
log.error_file: 'error.log'
server.socket_host: '0.0.0.0'
server.socket_port: 8181

[/]
request.dispatch: cherrypy.dispatch.MethodDispatcher()
tools.response_headers.on: True
tools.response_headers.headers: [('Content-Type', 'application/json')]
```

2.2 Service Configuration File

The service configuration is an INI file and an example is as follows:

```
[policy]
; This section has policy service specific config options

; This sets the admin group name
admin_group = admin
```

(continues on next page)

(continued from previous page)

```
; This sets the admin group id (should match group name in metadata)
admin_group_id = 0

[metadata]
; This section contains configuration for metadata service

; The global metadata url
endpoint_url = http://localhost:8121

; The endpoint to check for status of metadata service
status_url = http://localhost:8121/groups
```

2.3 Starting the Service

Starting the Policy service can be done by two methods. However, understanding the requirements and how they apply to REST services is important to address as well. Using the internal CherryPy server to start the service is recommended for Windows platforms. For Linux/Mac platforms it is recommended to deploy the service with uWSGI.

2.3.1 Deployment Considerations

The Policy server can have the same memory consumption issues as the Metadata service. Please consider those recommendations here similarly for the Policy service.

2.3.2 CherryPy Server

To make running the Policy service using the CherryPy's builtin server easier we have a command line entry point.

```
$ pacifica-policy --help
usage: pacifica-policy [-h] [-c CONFIG] [-p PORT] [-a ADDRESS]

Run the policy server.

optional arguments:
  -h, --help            show this help message and exit
  -c CONFIG, --config CONFIG
                        cherrypy config file
  -p PORT, --port PORT  port to listen on
  -a ADDRESS, --address ADDRESS
                        address to listen on
$ pacifica-policy
[09/Jan/2019:09:17:26] ENGINE Listening for SIGTERM.
[09/Jan/2019:09:17:26] ENGINE Bus STARTING
[09/Jan/2019:09:17:26] ENGINE Set handler for console events.
[09/Jan/2019:09:17:26] ENGINE Started monitor thread 'Autoreloader'.
[09/Jan/2019:09:17:26] ENGINE Serving on http://0.0.0.0:8181
[09/Jan/2019:09:17:26] ENGINE Bus STARTED
```

2.3.3 uWSGI Server

To make running the Policy service using uWSGI easier we have a module to be included as part of the uWSGI configuration. uWSGI is very configurable and can use this module many different ways. Please consult the uWSGI Configuration documentation for more complicated deployments.

```
$ pip install uwsgi
$ uwsgi --http-socket :8181 --master --module pacifica.policy.wsgi
```

CHAPTER 3

Example Usage

The usage of the Policy API is strictly a read-only interface. The command line usage of the system provides tools to update data in the metadata API and are mostly cronjob like processes.

3.1 The API

The policy server is split up into endpoints named for their Pacifica project that utilizes them. So the path `/uploader` is used by the Pacifica Uploader (http://github.com/pacifica/pacifica-uploader) to control its behavior. The idea is that workflow implemented by the various Pacifica projects has some element of site or instance specific policy that can be applied to the running service. The policy is driven by the metadata and thus this project should talk to the metadata service.

3.1.1 Events API

The Events API is used by the Notifications service. The role of this query is to verify the event recieved by the Notifications services is allowed to be sent to the user on the URL path.

Request Example:

```
POST /events/dmlb2001
Content-Type: application/json
{
  "data": [
    ...
  ]
}
```

Good Response Example:

```
Http-Code: 200
{
```

(continues on next page)

(continued from previous page)

```
  "status": "success"
}
```

Failed Response Example:

```
Http-Code: 401
{
  "error": "..."
}
```

The underlying logic for this implementation is the same as the ingest endpoint discussed next.

3.1.2 Ingest API

The Ingest API is used by the Ingest service. This endpoint verifies the relationships between user, proposal and instrument before allowing an upload. The content of the body document is defined by the uploader.

Request Example:

```
POST /ingest
Content-Type: application/json
[
  ...
]
```

Good Response Example:

```
Http-Code: 200
{
  "status": "success"
}
```

Failed Response Example:

```
Http-Code: 401
{
  "error": "..."
}
```

3.1.3 Reporting and Status API

This document is not going into details about these APIs currently. These endpoints are supposed to be used by tools that provide status of current uploads to users of Pacifica as well as institutional reporting tools that aggregate metrics about uploads in Pacifica. Eventually, Pacifica should have a basic set of these websites to allow users to use these endpoints but not currently.

3.1.4 Uploader API

The Uploader API is a simple query interface to get complex metadata interactively while users are using the Uploader. This API has a JSON document that looks very SQL like but is not complete.

Request Example:

```
POST /uploader
Content-Type: application/json
{
  "user": 100,
  "from": "instruments",
  "columns": [
    "_id",
    "name"
  ],
  "where": {
    "_id": 54
  }
}
```

Good Response Example:

```
Http-Code: 200
[
  {
    "_id": 54,
    "name": "NMR PROBES: Nittany Liquid"
  }
]
```

Failed Response Example:

```
Http-Code: 500
```

3.2 Admin Command Line

There is a single admin command line tool (`pacifica-policy-cmd`) with two subcommands, `data_release` and `searchsync`. The `data_release` subcommand handles setting the `data_release` attributes of the Proposals and Transactions. The `searchsync` subcommand handles formatting and synchonizing metadata to ElasticSearch.

```
$ pacifica-policy-cmd --help
usage: pacifica-policy-cmd [-h] [--verbose] {data_release,searchsync} ...

positional arguments:
  {data_release,searchsync}
                        sub-command help
    data_release        data_release help
    searchsync          searchsync help

optional arguments:
  -h, --help            show this help message and exit
  --verbose             enable verbose debug output
```

3.2.1 Data Release

The data release process involves two phases, updating the suspense date and setting data release. The suspense date is a date that the metadata and data associated with that object in metadata will be released in the future. The data release phase checks the suspense date with now to determine if the object needs to have it released.

```
$ pacifica-policy-cmd data_release --help
usage: pacifica-policy-cmd data_release [-h]
                                        [--exclude [EXCLUDE [EXCLUDE ...]]]
                                        [--keyword KEYWORD]
                                        [--time-after TIME_AFTER]
                                        [--time-ago TIME_AGO]

data release by policy

optional arguments:
  -h, --help            show this help message and exit
  --exclude [EXCLUDE [EXCLUDE ...]]
                        id of keyword prefix to exclude.
  --keyword KEYWORD     keyword one of proposals.actual_end_date,
                        proposals.actual_start_date, proposals.submitted_date,
                        proposals.accepted_date, proposals.closed_date,
                        transactions.created, transactions.updated.
  --time-after TIME_AFTER
                        set suspense date on data to X days after keyword.
  --time-ago TIME_AGO   only objects updated after X days ago.
```

3.2.2 Search Sync

The search synchronization to Elasticsearch is driven by the Policy service. The metadata in Elasticsearch is meant to be consumed by client applications and in order to be performant those clients should communicate directly with Elasticsearch. This does mean that the metadata in Elasticsearch is not as current as the Metadata API.

```
$ pacifica-policy-cmd searchsync
usage: pacifica-policy-cmd searchsync [-h] [--objects-per-page ITEMS_PER_PAGE]
                                      [--threads THREADS]
                                      [--time-ago TIME_AGO]

sync sql data to elastic for search

optional arguments:
  -h, --help            show this help message and exit
  --objects-per-page ITEMS_PER_PAGE
                        objects per bulk upload.
  --threads THREADS     number of threads to sync data
  --time-ago TIME_AGO   only objects newer than X days ago.
```

CHAPTER 4

Policy Python Module

4.1 Events Python Module

Events module to drive policy for who can see events.

Events rest module for the cherrypy endpoint.

class pacifica.policy.events.rest.**EventsPolicy**

CherryPy Events Policy.

This exposes whether a user can see an event from.

POST(*username*)

Pull the json content and validate the user can see the event.

4.2 Ingest Python Module

Ingest valication module.

The CherryPy rest object for the structure.

Below is an example post body:

```
[
    {"destinationTable": "Transactions._id", "value": 1234},
    {"destinationTable": "Transactions.submitter", "value": 34002},
    {"destinationTable": "Transactions.proposal", "value": "34002"},
    {"destinationTable": "Transactions.instrument", "value": 34002},
    {"destinationTable": "TransactionKeyValue", "key": "Tag", "value": "Blah"},
    {"destinationTable": "TransactionKeyValue", "key": "Taggy", "value": "Blah"},
    {"destinationTable": "TransactionKeyValue", "key": "Taggier", "value": "Blah"}
    {
        "destinationTable": "Files",
        "_id": 34, "name": "foo.txt", "subdir": "a/b/",
```

(continues on next page)

(continued from previous page)

```
        "ctime": "Tue Nov 29 14:09:05 PST 2016",
        "mtime": "Tue Nov 29 14:09:05 PST 2016",
        "size": 128, "mimetype": "text/plain"
    },
    {
        "destinationTable": "Files",
        "_id": 35, "name": "bar.txt", "subdir": "a/b/",
        "ctime": "Tue Nov 29 14:09:05 PST 2016",
        "mtime": "Tue Nov 29 14:09:05 PST 2016",
        "size": 47, "mimetype": "text/plain"
    },
]
```

class pacifica.policy.ingest.rest.**IngestPolicy**

CherryPy Ingest Policy Class.

POST()

Read in the json query and return results.

static _pull_data_by_rec(*query*, *table*)

Pull the value for the table.

_valid_query(*query*)

Validate the metadata format.

4.3 Reporting Python Module

CherryPy Uploader Policy object class.

CherryPy Uploader Policy object class.

CherryPy Status Metadata proposalinfo base class.

class pacifica.policy.reporting.transaction.query_base.**QueryBase**

Formats summary data for other classes down the tree.

__weakref__

list of weak references to the object (if defined)

static _merge_two_dicts(*dict_a*, *dict_b*)

Given two dicts, merge them into a new dict as a shallow copy.

static get_full_user_info(*user_id*)

Return user information for the given user_id.

CherryPy Status Metadata object class.

class pacifica.policy.reporting.transaction.transaction_details.**TransactionDetails**

Retrieves a list of all transactions matching the search criteria.

static POST(*user_id=None*)

CherryPy GET method.

CherryPy Status Metadata object class.

class pacifica.policy.reporting.transaction.transaction_summary.**TransactionSummary**

Retrieves a summary of all transactions matching the search criteria.

static POST(*time_basis=None*, *object_type=None*, *start_date=None*, *end_date=None*, ***kwargs*)
CherryPy GET method.

The CherryPy rest object for the structure.

class `pacifica.policy.reporting.rest.`**ReportingPolicy**
CherryPy root object class.

not exposed by default the base objects are exposed.

__init__()
Create local objects to allow for import to work.

__weakref__
list of weak references to the object (if defined)

4.4 Status Python Module

CherryPy Uploader Policy object class.

Base class module for standard queries for the upload status tool.

class `pacifica.policy.status.base.`**QueryBase**
This pulls the common bits of instrument and proposal query into a single class.

CherryPy Status Policy object class.

class `pacifica.policy.status.instrument_query.`**InstrumentQuery**
CherryPy root object class.

__init__()
Create local objects for sub tree items.

__weakref__
list of weak references to the object (if defined)

CherryPy Status Policy object class.

class `pacifica.policy.status.proposal_query.`**ProposalQuery**
CherryPy root object class.

__init__()
Create local objects for sub tree items.

__weakref__
list of weak references to the object (if defined)

The CherryPy rest object for the structure.

class `pacifica.policy.status.rest.`**StatusPolicy**
CherryPy root object class.

not exposed by default the base objects are exposed.

__init__()
Create local objects to allow for import to work.

__weakref__
list of weak references to the object (if defined)

CherryPy Status Policy object class.

class pacifica.policy.status.transaction_query.**TransactionQuery**
CherryPy root object class.

__init__()
Create local objects for sub tree items.

__weakref__
list of weak references to the object (if defined)

CherryPy Status Policy object class.

class pacifica.policy.status.user_query.**UserQuery**
CherryPy root object class.

__init__()
Create local objects for sub tree items.

__weakref__
list of weak references to the object (if defined)

CherryPy Proposal Policy object classes.

CherryPy Status Policy object class.

class pacifica.policy.status.instrument.by_proposal_id.**InstrumentsByProposal**
Retrieves instrument list for a given proposal.

static GET(*proposal_id=None*)
CherryPy GET method.

static _get_instruments_for_proposal(*proposal_id*)
Return a list with all the instruments belonging to this proposal.

CherryPy Status Policy object class.

class pacifica.policy.status.instrument.search.**InstrumentKeywordSearch**
Retrieves a set of proposals for a given keyword set.

GET(*search_terms=''*, ***kwargs*)
CherryPy GET method.

_clean_up_instrument_list(*inst_response*, *user_id*)
Clear out entries that done belong to this user.

_get_instruments_for_keywords(*user_id*, *search_terms=''*)
Return a list with all the instruments having this term.

static _squash_output_list(*inst_for_user_list*, *full_inst_list*)
Filter entries in the full instrument list.

CherryPy Proposal Policy object classes.

CherryPy Status Policy object class.

class pacifica.policy.status.proposal.by_user.**ProposalUserSearch**
Retrieves proposal list for a given user.

static GET(*user_id=None*)
CherryPy GET method.

static _get_proposals_for_user(*user_id=None*)
Return a list with all the proposals involving this user.

CherryPy Status Policy object class.

class `pacifica.policy.status.proposal.lookup.`**`ProposalLookup`**

Retrieves details of a given proposal.

static GET(*proposal_id=None*)

CherryPy GET method.

`__weakref__`

list of weak references to the object (if defined)

static _get_proposals_details(*proposal_id=None*)

Return a details about this proposal.

CherryPy Status Policy object class.

class `pacifica.policy.status.proposal.search.`**`ProposalKeywordSearch`**

Retrieves a set of proposals for a given keyword set.

GET(*search_terms=None*, ***kwargs*)

CherryPy GET method.

_get_proposals_for_keywords(*user_id*, *search_terms=None*)

Return a list with all the proposals involving this user.

CherryPy Uploader Policy object class.

CherryPy Status Policy object class.

class `pacifica.policy.status.transaction.files.`**`FileLookup`**

Retrieves files for a given transaction_id.

static GET(*transaction_id=None*)

CherryPy GET method.

`__weakref__`

list of weak references to the object (if defined)

static _get_file_list(*transaction_id=None*)

Return files for the specified transaction entry.

CherryPy Status Policy object class.

class `pacifica.policy.status.transaction.lookup.`**`TransactionLookup`**

Retrieves details of a given proposal.

static GET(*transaction_id=None*)

CherryPy GET method.

`__weakref__`

list of weak references to the object (if defined)

static _get_transaction_details(*transaction_id=None*)

Return details for the specified transaction entry.

CherryPy Status Policy object class.

class `pacifica.policy.status.transaction.search.`**`TransactionSearch`**

Retrieves a set of transactions for a given keyword set.

static GET(*option=None*, ***kwargs*)

CherryPy GET method.

`__weakref__`

list of weak references to the object (if defined)

static _get_transactions_for_keywords(*kwargs*, *option=None*)
Return a list with all the proposals involving this user.

CherryPy Uploader Policy object class.

CherryPy Status Policy object class.

class `pacifica.policy.status.user.lookup.`**UserLookup**
Retrieves info for the specified user.

static GET(*user_id=None*)
CherryPy GET method.

__weakref__
list of weak references to the object (if defined)

static _get_user_info(*user_id*)
Return detailed info about a given user.

CherryPy Status Policy object class.

class `pacifica.policy.status.user.search.`**UserSearch**
Retrieves a set of proposals for a given keyword set.

GET(*search_terms=None*, *option=None*, ***kwargs*)
CherryPy GET method.

static _get_users_for_keywords(*kwargs*, *search_terms=None*, *option=None*)
Return a list with all the proposals involving this user.

4.5 Uploader Python Module

CherryPy Uploader Policy object class.

The CherryPy rest object for the structure.

class `pacifica.policy.uploader.rest.`**UploaderPolicy**
CherryPy root object class.

not exposed by default the base objects are exposed

POST()
Read in the json query and return results.

static _clean_user_query_id(*query*)
determine the user_id for whatever is in the query.

4.6 Admin Python Module

The Admin module has logic about checking for admin group info.

class `pacifica.policy.admin.`**AdminPolicy**
Enforces the admin policy.

Base class for checking for admin group membership or not.

__init__()
Constructor for Uploader Policy.

__weakref__
list of weak references to the object (if defined)

_format_url (*url*, ***get_args*)
Append the recursion_depth parameter to the url.

4.7 Admin Command Python Module

This is the admin main method.

`pacifica.policy.admin_cmd.`**`create_subcommands`**(*subparsers*)
Create the subcommands from the subparsers.

`pacifica.policy.admin_cmd.`**`datarel_options`**(*datarel_parser*)
Add data release options to the parser.

`pacifica.policy.admin_cmd.`**`main`**(**argv*)
Main method for admin command line tool.

`pacifica.policy.admin_cmd.`**`objstr_to_keyword`**(*obj_str*)
Verify the obj_str is a valid keyword.

`pacifica.policy.admin_cmd.`**`objstr_to_timedelta`**(*obj_str*)
Turn an object string of the format X unit ago into timedelta.

`pacifica.policy.admin_cmd.`**`searchsync_options`**(*searchsync_parser*)
Add the searchsync command line options.

4.8 Config Python Module

Configuration reading and validation module.

`pacifica.policy.config.`**`get_config`**()
Return the ConfigParser object with defaults set.

Currently metadata API doesn't work with SQLite the queries are too complex and it only is supported with MySQL and PostgreSQL.

4.9 Data Release Python Module

Data release policy for command line tools.

`pacifica.policy.data_release.`**`collate_objs_from_key`**(*resp*, *objs*, *date_key*)
Deduplicate objs and make sure they have dates.

`pacifica.policy.data_release.`**`data_release`**(*args*)
Data release main subcommand.

The logic is to query updated objects between now and args.time_ago. If the objects args.keyword is set to something calculate the suspense date as args.time_after the keyword date. Then save the object back to the metadata server.

The follow on task is to use orm_obj to calculate the released data based on the set suspense dates and add that released data to the transaction_release table.

`pacifica.policy.data_release.`**`relavent_data_release_objs`**(*time_ago*, *orm_obj*, *exclude_list*)
Query proposals or transactions that has gone past their suspense date.

`pacifica.policy.data_release.`**`relavent_suspense_date_objs`**(*time_ago*, *orm_obj*, *date_key*)
generate a list of relavent orm_objs saving date_key.

`pacifica.policy.data_release.`**`update_data_release`**(*objs*)
Add objs transactions to the released transactions table.

`pacifica.policy.data_release.`**`update_suspense_date_objs`**(*objs*, *time_after*, *orm_obj*)
update the list of objs given date_key adding time_after.

4.10 Globals Python Module

Global static variables.

4.11 Root Rest Python Module

CherryPy root object class.

`class` `pacifica.policy.root.`**`Root`**
CherryPy root object class.

not exposed by default the base objects are exposed

`__init__`()
Create the local objects we need.

`__weakref__`
list of weak references to the object (if defined)

`classmethod try_meta_connect`(*attempts=0*)
Try to connect to the metadata service see if its there.

`pacifica.policy.root.`**`error_page_default`**(***kwargs*)
The default error page should always enforce json.

4.12 Search Render Python Module

This is the render object for the search interface.

`class` `pacifica.policy.search_render.`**`LimitedSizeDict`**(**args*, ***kwds*)
Limited caching dictionary.

`__getitem__`(*key*)
Get the item and put it back so it's on top.

`__init__`(**args*, ***kwds*)
Constructor for caching dictionary.

`__setitem__`(*key*, *value*)
Set item foo[key] = value.

_check_size_limit()
Function to set the item and remove old ones.

class pacifica.policy.search_render.**SearchRender**
Search render class to contain methods.

__weakref__
list of weak references to the object (if defined)

classmethod generate(*obj_cls*, *objs*, *trans_ids=False*, *render_release=False*)
generate the institution object.

classmethod get_groups_from_instrument(*inst_id*)
Get the list of groups from an instrument.

classmethod get_institutions_from_user(*user_id*)
Get an institution list based on user id.

classmethod get_obj_by_id(*obj*, *obj_id*)
Get the user from metadata and put it in cache.

classmethod get_prop_release(*prop_id*)
Get the proposal release from transactions on that prop.

classmethod get_trans_release(*trans_id*)
Get the transaction release and return true/false.

classmethod get_transactions_from_groups(*group_id*)
Get a list of instruments for a group.

classmethod get_transactions_from_institutions(*inst_id*)
Get a list of transactions from an institution.

classmethod get_transactions_from_instruments(*inst_id*)
Get a list of transactions for a instrument.

classmethod get_transactions_from_proposals(*prop_id*)
Get a list of transactions for a proposal.

classmethod get_transactions_from_science_theme(*science_theme*)
Get a list of transactions for a science theme.

classmethod get_transactions_from_users(*user_id*)
Get a list of transactions for a user.

classmethod get_user_release(*user_id*)
Get the user release from transactions on that prop.

classmethod merge_get_args(*get_args*)
Change a hash of get args and global get args into string for url.

classmethod render(*obj_cls*, *obj*, *trans_ids=False*, *render_release=False*)
Render the instrument object hash.

classmethod render_science_theme(*obj*, *trans_ids=False*)
Render the science theme as an object…

pacifica.policy.search_render.**prop_release**(*obj*)
Render the transaction release attribute.

pacifica.policy.search_render.**trans_inst_groups**(*trans_obj*)
Render the instrument groups for a transaction.

`pacifica.policy.search_render.`**`trans_institutions`**`(`*trans_obj*`)`
Render the institutions for a transaction.

`pacifica.policy.search_render.`**`trans_instruments`**`(`*trans_obj*`)`
Render the instruments for a transaction.

`pacifica.policy.search_render.`**`trans_proposals`**`(`*trans_obj*`)`
Render the proposals for a transaction.

`pacifica.policy.search_render.`**`trans_release`**`(`*obj*`)`
Render the transaction release attribute.

`pacifica.policy.search_render.`**`trans_science_themes`**`(`*trans_obj*`)`
Render the science theme from a proposal.

`pacifica.policy.search_render.`**`trans_users`**`(`*trans_obj*`)`
Render the users list for transactions.

`pacifica.policy.search_render.`**`transsip_transsap_render`**`(`*trans_obj*, *render_func*, *obj_cls*, *trans_key*`)`
Wrapper method to handle transaction relationships.

`pacifica.policy.search_render.`**`user_release`**`(`*obj*`)`
Render the transaction release attribute.

4.13 Search Sync Python Module

Sync the database to elasticsearch index for use by Searching tools.

`pacifica.policy.search_sync.`**`create_worker_threads`**`(`*threads*, *work_queue*`)`
Create the worker threads and return the list.

`pacifica.policy.search_sync.`**`es_client`**`()`
Get the elasticsearch client object.

`pacifica.policy.search_sync.`**`generate_work`**`(`*items_per_page*, *work_queue*, *time_ago*`)`
Generate the work from the db and send it to the work queue.

`pacifica.policy.search_sync.`**`search_sync`**`(`*args*`)`
Main search sync subcommand.

`pacifica.policy.search_sync.`**`start_work`**`(`*work_queue*`)`
The main thread for the work.

`pacifica.policy.search_sync.`**`try_doing_work`**`(`*cli*, *job*`)`
Try doing some work even if you fail.

`pacifica.policy.search_sync.`**`try_es_connect`**`(`*attempts=0*`)`
Recursively try to connect to elasticsearch.

`pacifica.policy.search_sync.`**`yield_data`**`(`*obj*, *time_field*, *page*, *items_per_page*, *time_delta*`)`
yield objects from obj for bulk ingest.

4.14 Validation Python Module

Validation methods for various objects.

`pacifica.policy.validation.`**`_get_check_id`**`(`*index*, **args*, ***kwargs*`)`
Return the check ID in args or kwargs.

`pacifica.policy.validation.`**`validate_proposal`**`(`*index=0*`)`
Validate the proposal id.

`pacifica.policy.validation.`**`validate_transaction`**`(`*index=0*`)`
Validate the transaction id.

`pacifica.policy.validation.`**`validate_universal`**`(`*index*, *regex*`)`
Decorator generator to validate proposal field.

`pacifica.policy.validation.`**`validate_user`**`(`*index=0*`)`
Validate the user id.

4.15 WSGI Python Module

This is the main policy server script. This is the policy module.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

pacifica.policy, 23
pacifica.policy.admin, 18
pacifica.policy.admin_cmd, 19
pacifica.policy.config, 19
pacifica.policy.data_release, 19
pacifica.policy.events, 13
pacifica.policy.events.rest, 13
pacifica.policy.globals, 20
pacifica.policy.ingest, 13
pacifica.policy.ingest.rest, 13
pacifica.policy.reporting, 14
pacifica.policy.reporting.rest, 15
pacifica.policy.reporting.transaction, 14
pacifica.policy.reporting.transaction.query_base, 14
pacifica.policy.reporting.transaction.transaction_details, 14
pacifica.policy.reporting.transaction.transaction_summary, 14
pacifica.policy.root, 20
pacifica.policy.search_render, 20
pacifica.policy.search_sync, 22
pacifica.policy.status, 15
pacifica.policy.status.base, 15
pacifica.policy.status.instrument, 16
pacifica.policy.status.instrument.by_proposal_id, 16
pacifica.policy.status.instrument.search, 16
pacifica.policy.status.instrument_query, 15
pacifica.policy.status.proposal, 16
pacifica.policy.status.proposal.by_user, 16
pacifica.policy.status.proposal.lookup, 16
pacifica.policy.status.proposal.search, 17
pacifica.policy.status.proposal_query, 15
pacifica.policy.status.rest, 15
pacifica.policy.status.transaction, 17
pacifica.policy.status.transaction.files, 17
pacifica.policy.status.transaction.lookup, 17
pacifica.policy.status.transaction.search, 17
pacifica.policy.status.transaction_query, 15
pacifica.policy.status.user, 18
pacifica.policy.status.user.lookup, 18
pacifica.policy.status.user.search, 18
pacifica.policy.status.user_query, 16
pacifica.policy.uploader, 18
pacifica.policy.uploader.rest, 18
pacifica.policy.validation, 22
pacifica.policy.wsgi, 23

Index

Symbols

__getitem__() (pacifica.policy.search_render.LimitedSizeDict method), 20
__init__() (pacifica.policy.admin.AdminPolicy method), 18
__init__() (pacifica.policy.reporting.rest.ReportingPolicy method), 15
__init__() (pacifica.policy.root.Root method), 20
__init__() (pacifica.policy.search_render.LimitedSizeDict method), 20
__init__() (pacifica.policy.status.instrument_query.InstrumentQuery method), 15
__init__() (pacifica.policy.status.proposal_query.ProposalQuery method), 15
__init__() (pacifica.policy.status.rest.StatusPolicy method), 15
__init__() (pacifica.policy.status.transaction_query.TransactionQuery method), 16
__init__() (pacifica.policy.status.user_query.UserQuery method), 16
__setitem__() (pacifica.policy.search_render.LimitedSizeDict method), 20
__weakref__ (pacifica.policy.admin.AdminPolicy attribute), 18
__weakref__ (pacifica.policy.reporting.rest.ReportingPolicy attribute), 15
__weakref__ (pacifica.policy.reporting.transaction.query_base.QueryBase attribute), 14
__weakref__ (pacifica.policy.root.Root attribute), 20
__weakref__ (pacifica.policy.search_render.SearchRender attribute), 21
__weakref__ (pacifica.policy.status.instrument_query.InstrumentQuery attribute), 15
__weakref__ (pacifica.policy.status.proposal.lookup.ProposalLookup attribute), 17
__weakref__ (pacifica.policy.status.proposal_query.ProposalQuery attribute), 15
__weakref__ (pacifica.policy.status.rest.StatusPolicy attribute), 15
__weakref__ (pacifica.policy.status.transaction.files.FileLookup attribute), 17
__weakref__ (pacifica.policy.status.transaction.lookup.TransactionLookup attribute), 17
__weakref__ (pacifica.policy.status.transaction.search.TransactionSearch attribute), 17
__weakref__ (pacifica.policy.status.transaction_query.TransactionQuery attribute), 16
__weakref__ (pacifica.policy.status.user.lookup.UserLookup attribute), 18
__weakref__ (pacifica.policy.status.user_query.UserQuery attribute), 16
_check_size_limit() (pacifica.policy.search_render.LimitedSizeDict method), 20
_clean_up_instrument_list() (pacifica.policy.status.instrument.search.InstrumentKeywordSearch method), 16
_clean_user_query_id() (pacifica.policy.uploader.rest.UploaderPolicy static method), 18
_format_url() (pacifica.policy.admin.AdminPolicy method), 19
_get_check_id() (in module pacifica.policy.validation), 22
_get_file_list() (pacifica.policy.status.transaction.files.FileLookup static method), 17
_get_instruments_for_keywords() (pacifica.policy.status.instrument.search.InstrumentKeywordSearch method), 16
_get_instruments_for_proposal() (pacifica.policy.status.instrument.by_proposal_id.InstrumentsByPropo static method), 16
_get_proposals_details() (pacifica.policy.status.proposal.lookup.ProposalLookup static method), 17
_get_proposals_for_keywords() (pacifica.policy.status.proposal.search.ProposalKeywordSearch method), 17
_get_proposals_for_user() (pacifica.policy.status.proposal.by_user.ProposalUserSearch

static method), 16
_get_transaction_details() (pacifica.policy.status.transaction.lookup.TransactionLookup static method), 17
_get_transactions_for_keywords() (pacifica.policy.status.transaction.search.TransactionSearch static method), 17
_get_user_info() (pacifica.policy.status.user.lookup.UserLookup static method), 18
_get_users_for_keywords() (pacifica.policy.status.user.search.UserSearch static method), 18
_merge_two_dicts() (pacifica.policy.reporting.transaction.query_base.QueryBase static method), 14
_pull_data_by_rec() (pacifica.policy.ingest.rest.IngestPolicy static method), 14
_squash_output_list() (pacifica.policy.status.instrument.search.InstrumentKeywordSearch static method), 16
_valid_query() (pacifica.policy.ingest.rest.IngestPolicy method), 14

A

AdminPolicy (class in pacifica.policy.admin), 18

C

collate_objs_from_key() (in module pacifica.policy.data_release), 19
create_subcommands() (in module pacifica.policy.admin_cmd), 19
create_worker_threads() (in module pacifica.policy.search_sync), 22

D

data_release() (in module pacifica.policy.data_release), 19
datarel_options() (in module pacifica.policy.admin_cmd), 19

E

error_page_default() (in module pacifica.policy.root), 20
es_client() (in module pacifica.policy.search_sync), 22
EventsPolicy (class in pacifica.policy.events.rest), 13

F

FileLookup (class in pacifica.policy.status.transaction.files), 17

G

generate() (pacifica.policy.search_render.SearchRender class method), 21
generate_work() (in module pacifica.policy.search_sync), 22
GET() (pacifica.policy.status.instrument.by_proposal_id.InstrumentsByProp static method), 16
GET() (pacifica.policy.status.instrument.search.InstrumentKeywordSearch method), 16
GET() (pacifica.policy.status.proposal.by_user.ProposalUserSearch static method), 16
GET() (pacifica.policy.status.proposal.lookup.ProposalLookup static method), 17
GET() (pacifica.policy.status.proposal.search.ProposalKeywordSearch method), 17
GET() (pacifica.policy.status.transaction.files.FileLookup static method), 17
GET() (pacifica.policy.status.transaction.lookup.TransactionLookup static method), 17
GET() (pacifica.policy.status.transaction.search.TransactionSearch static method), 17
GET() (pacifica.policy.status.user.lookup.UserLookup static method), 18
GET() (pacifica.policy.status.user.search.UserSearch method), 18
get_config() (in module pacifica.policy.config), 19
get_full_user_info() (pacifica.policy.reporting.transaction.query_base.QueryBase static method), 14
get_groups_from_instrument() (pacifica.policy.search_render.SearchRender class method), 21
get_institutions_from_user() (pacifica.policy.search_render.SearchRender class method), 21
get_obj_by_id() (pacifica.policy.search_render.SearchRender class method), 21
get_prop_release() (pacifica.policy.search_render.SearchRender class method), 21
get_trans_release() (pacifica.policy.search_render.SearchRender class method), 21
get_transactions_from_groups() (pacifica.policy.search_render.SearchRender class method), 21
get_transactions_from_institutions() (pacifica.policy.search_render.SearchRender class method), 21
get_transactions_from_instruments() (pacifica.policy.search_render.SearchRender class method), 21
get_transactions_from_proposals() (pacifica.policy.search_render.SearchRender class method), 21
get_transactions_from_science_theme() (pacifica.policy.search_render.SearchRender class

method), 21
get_transactions_from_users() (pacifica.policy.search_render.SearchRender class method), 21
get_user_release() (pacifica.policy.search_render.SearchRender class method), 21

I

IngestPolicy (class in pacifica.policy.ingest.rest), 14
InstrumentKeywordSearch (class in pacifica.policy.status.instrument.search), 16
InstrumentQuery (class in pacifica.policy.status.instrument_query), 15
InstrumentsByProposal (class in pacifica.policy.status.instrument.by_proposal_id), 16

L

LimitedSizeDict (class in pacifica.policy.search_render), 20

M

main() (in module pacifica.policy.admin_cmd), 19
merge_get_args() (pacifica.policy.search_render.SearchRender class method), 21

O

objstr_to_keyword() (in module pacifica.policy.admin_cmd), 19
objstr_to_timedelta() (in module pacifica.policy.admin_cmd), 19

P

pacifica.policy (module), 23
pacifica.policy.admin (module), 18
pacifica.policy.admin_cmd (module), 19
pacifica.policy.config (module), 19
pacifica.policy.data_release (module), 19
pacifica.policy.events (module), 13
pacifica.policy.events.rest (module), 13
pacifica.policy.globals (module), 20
pacifica.policy.ingest (module), 13
pacifica.policy.ingest.rest (module), 13
pacifica.policy.reporting (module), 14
pacifica.policy.reporting.rest (module), 15
pacifica.policy.reporting.transaction (module), 14
pacifica.policy.reporting.transaction.query_base (module), 14
pacifica.policy.reporting.transaction.transaction_details (module), 14
pacifica.policy.reporting.transaction.transaction_summary (module), 14
pacifica.policy.root (module), 20
pacifica.policy.search_render (module), 20
pacifica.policy.search_sync (module), 22
pacifica.policy.status (module), 15
pacifica.policy.status.base (module), 15
pacifica.policy.status.instrument (module), 16
pacifica.policy.status.instrument.by_proposal_id (module), 16
pacifica.policy.status.instrument.search (module), 16
pacifica.policy.status.instrument_query (module), 15
pacifica.policy.status.proposal (module), 16
pacifica.policy.status.proposal.by_user (module), 16
pacifica.policy.status.proposal.lookup (module), 16
pacifica.policy.status.proposal.search (module), 17
pacifica.policy.status.proposal_query (module), 15
pacifica.policy.status.rest (module), 15
pacifica.policy.status.transaction (module), 17
pacifica.policy.status.transaction.files (module), 17
pacifica.policy.status.transaction.lookup (module), 17
pacifica.policy.status.transaction.search (module), 17
pacifica.policy.status.transaction_query (module), 15
pacifica.policy.status.user (module), 18
pacifica.policy.status.user.lookup (module), 18
pacifica.policy.status.user.search (module), 18
pacifica.policy.status.user_query (module), 16
pacifica.policy.uploader (module), 18
pacifica.policy.uploader.rest (module), 18
pacifica.policy.validation (module), 22
pacifica.policy.wsgi (module), 23
POST() (pacifica.policy.events.rest.EventsPolicy method), 13
POST() (pacifica.policy.ingest.rest.IngestPolicy method), 14
POST() (pacifica.policy.reporting.transaction.transaction_details.Transactio
static method), 14
POST() (pacifica.policy.reporting.transaction.transaction_summary.Transac
static method), 14
POST() (pacifica.policy.uploader.rest.UploaderPolicy method), 18
prop_release() (in module pacifica.policy.search_render), 21
ProposalKeywordSearch (class in pacifica.policy.status.proposal.search), 17
ProposalLookup (class in pacifica.policy.status.proposal.lookup), 16
ProposalQuery (class in pacifica.policy.status.proposal_query), 15
ProposalUserSearch (class in pacifica.policy.status.proposal.by_user), 16

Q

QueryBase (class in pacifica.policy.reporting.transaction.query_base), 14

QueryBase (class in pacifica.policy.status.base), 15

R

relavent_data_release_objs() (in module pacifica.policy.data_release), 19
relavent_suspense_date_objs() (in module pacifica.policy.data_release), 20
render() (pacifica.policy.search_render.SearchRender class method), 21
render_science_theme() (pacifica.policy.search_render.SearchRender class method), 21
ReportingPolicy (class in pacifica.policy.reporting.rest), 15
Root (class in pacifica.policy.root), 20

S

search_sync() (in module pacifica.policy.search_sync), 22
SearchRender (class in pacifica.policy.search_render), 21
searchsync_options() (in module pacifica.policy.admin_cmd), 19
start_work() (in module pacifica.policy.search_sync), 22
StatusPolicy (class in pacifica.policy.status.rest), 15

T

trans_inst_groups() (in module pacifica.policy.search_render), 21
trans_institutions() (in module pacifica.policy.search_render), 21
trans_instruments() (in module pacifica.policy.search_render), 22
trans_proposals() (in module pacifica.policy.search_render), 22
trans_release() (in module pacifica.policy.search_render), 22
trans_science_themes() (in module pacifica.policy.search_render), 22
trans_users() (in module pacifica.policy.search_render), 22
TransactionDetails (class in pacifica.policy.reporting.transaction.transaction_details), 14
TransactionLookup (class in pacifica.policy.status.transaction.lookup), 17
TransactionQuery (class in pacifica.policy.status.transaction_query), 15
TransactionSearch (class in pacifica.policy.status.transaction.search), 17
TransactionSummary (class in pacifica.policy.reporting.transaction.transaction_summary), 14
transsip_transsap_render() (in module pacifica.policy.search_render), 22
try_doing_work() (in module pacifica.policy.search_sync), 22
try_es_connect() (in module pacifica.policy.search_sync), 22
try_meta_connect() (pacifica.policy.root.Root class method), 20

U

update_data_release() (in module pacifica.policy.data_release), 20
update_suspense_date_objs() (in module pacifica.policy.data_release), 20
UploaderPolicy (class in pacifica.policy.uploader.rest), 18
user_release() (in module pacifica.policy.search_render), 22
UserLookup (class in pacifica.policy.status.user.lookup), 18
UserQuery (class in pacifica.policy.status.user_query), 16
UserSearch (class in pacifica.policy.status.user.search), 18

V

validate_proposal() (in module pacifica.policy.validation), 22
validate_transaction() (in module pacifica.policy.validation), 23
validate_universal() (in module pacifica.policy.validation), 23
validate_user() (in module pacifica.policy.validation), 23

Y

yield_data() (in module pacifica.policy.search_sync), 22