CIDS Repository Programmer Manual

9 June 2021

# Technology Platform

* Python
* Flask web framework with Jinja web templates
* OWL ontology language
* OWLReady2 ontology programming framework (<https://owlready2.readthedocs.io/en/v0.31/>)
* Database: OWLReady2 embedded database – SQLITE3

# Repository Directory Structure

* CA-Repository-init.py: initializes the repository database from scratch
* CA-Repository-server.py: runtime server for repository
* /codelist-library – directory of external organizations codelists
  + IRIS Taxonomy 5.1\_June2020.xlsx: Global Impact Investment Network’s standard social impact indicators (https://iris.thegiin.org/)
  + UNSDG.xlsx: outcomes and indicators for UNSDGs
* /db – database directory
  + /Backup: directory containing backup of database – created when init is performed
  + cidsrepository.sqlite3: server persistent database
* /Documents – document directory
  + CIDS Repository Programmer Manual
* /jsonUploads – contains files created for json uploads
* /lib – directory of CIDS Rep python modules
  + Analysis.py
  + config.py
  + ImpactModel.py
  + ImpactReport.py
  + Indicator.py
  + IndicatorReport.py
  + Load.py
  + Login.py
  + Organization.py
  + Outcome.py
  + Reasoning.py
  + Stakeholder.py
  + StakeholderCharacteristic.py
  + User.py
  + Util.py
* /logs – directory of log files generated by server – contains all changes to database
* /ontology – directory of directory specific ontologies
  + cadr.owl – extension of cids.owl for the repository
* /static – flask static directory
* /templates – flask html template directory
  + userSelect.html
  + base.html
  + displayDistance.html
  + impactModelEdit.html
  + impactModelSelect.html
  + impactReportEdit.html
  + impactReportSelect.html
  + index.html
  + indicatorEdit.html
  + indicatorReportEdit.html
  + indicatorReportSelect.html
  + indicatorSelect.html
  + loadJsonld.html
  + main.html
  + organizationEdit.html
  + organizationSelect.html
  + outcomeEdit.html
  + outcomeSelect.html
  + stakeholderCharacteristicEdit.html
  + stakeholderCharacteristicSelect.html
  + stakeholderEdit.html
  + stakeholderSelect.html
  + user.html
  + userEdit.html

# Initializing Database

python3 CA-Repository-init.py

CA-Repository-init.py backs up the current database into the db/Backup directory with the db file name appended with the current date/time. It then creates a new database (cidsrepository.sqlite3) and initializes the content with the ontologies and a variety of instances.

* Will load all ontologies from the ontology.eil.utoronto.ca server
* Will load organizations that have published outcome and/or indicator codelists from the codelist-library directory. Currently:
  + IRIS Taxonomy 5.1\_June2020.xlsx: Global Impact Investment Network’s standard social impact indicators (https://iris.thegiin.org/)
  + UNSDG.xlsx: outcomes and indicators for UNSDGs

Separate load functions are defined in Load.py.

* Will initialize a super user for development: [msf@eil.utoronto.ca](mailto:msf@eil.utoronto.ca) p
* Will initialize a regular user for development: [user@test.org](mailto:user@test.org) p

# Running the Repository

python3 CA-Repository-Server.py

The repository-server code attaches to the database (cidsrepository.sqlite3) created by CA-Repository-init.py and goes into suspension waiting for a request from the browser interface.

To access the server, enter the following address into a browser: localhost:5000

# User Types

There are five types of users, as defined in the http://ontology.eil.utoronto.ca/cids/cidsrep.owl ontology. Each successive level has greater editing rights. All are restricted to the Organization to which their user account is assigned, except for a superuser who can edit anything.

1. **reporter**: can generate reports for a single organization
2. **researcher**: can read information for a single organization
3. **editor**: can edit content, e.g., impact model, for a single organization
4. **admin**: same as editor, but can add users
5. **superuser**: can add organizations and users, and edit anything

# Code Structure

The code is organized around processing http requests for most classes in the Common Impact Data Standard. (Some html templates instantiate more than one class.) Generally, there are 5 requests for each class. The following lists the method for each request:

1. <class module>.add(): called to add a new instance of the class for user’s organization.

* returns the <class>Edit.html template (empty version).

1. <class module>.select(): called to select an existing instance of the class, defined for the user’s organization.

* returns the <class>Select.html template.

1. <class module>.edit(): called to edit the selected instance.

* returns the <class>Edit.html template filled in with instance values.

1. <class module>.update(): called to update the database with the results of the edit.

* returns the <class>Edit.html template (display version showing result of update).

1. <class module>.delete(): called to delete the instance.

* returns the main.html template.

**config.py** contains global variables used across all modules.