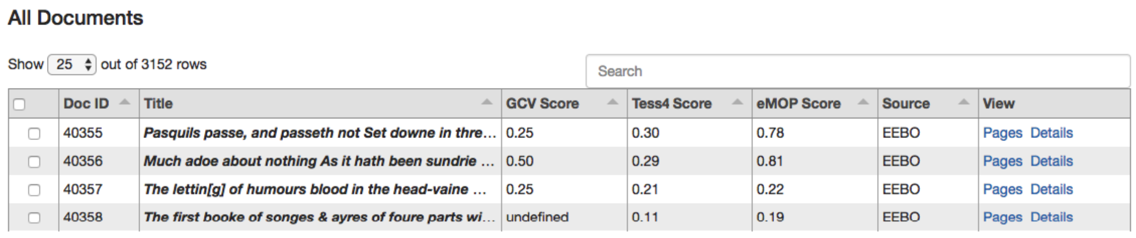
DH Dashboard Description

The "DH Dashboard" is a term that collectively refers to an infrastructure designed to build, manage, and perform tasks on large scale corpora of documents. The component parts of this infrastructure are outlined below.

The DH Dashboard Django web application is written in Python 3.5, and currently consists of two major sub-components: the Corpus Manager and the Job Manager.

The Corpus Manager allows for users to view and organize their corpus at both the document and page level of stratification. Users can create subsets of these items within their corpus (Document Sets and Page Sets), and can associate arbitrary key/value pairs with any document or page in the corpus (key/value pairs can also be associated with the corpus itself). The way data is viewed in the dashboard is fully configurable, in that a JSON configuration file exists for Pages, Page Sets, Documents, and Document Sets, allowing for the user to specify which columns (whose values can be determined via key/value pairs) are visible as they page through their data. A corpus oriented toward OCR, for instance, might allow a user to see for each document three separate columns, each showing the aggregate OCR accuracy score for a different OCR engine.



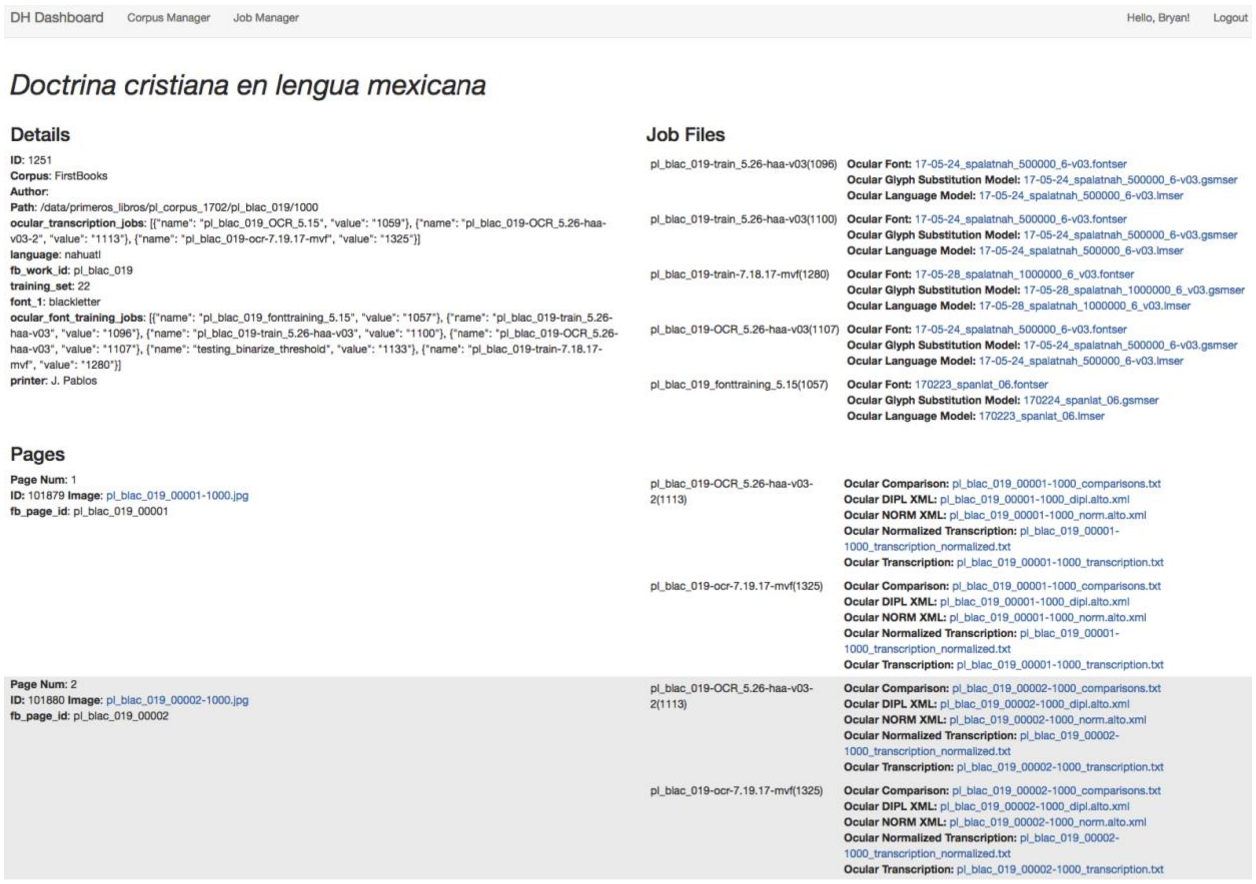
All Documents

Show 25 out of 3152 rows

Search

	Doc ID	Title	GCV Score	Tess4 Score	eMOP Score	Source	View
☐	40355	*Pasquils passe, and passeth not Set downe in thre...*	0.25	0.30	0.78	EEBO	Pages Details
☐	40356	*Much adoe about nothing As it hath been sundrie ...*	0.50	0.29	0.81	EEBO	Pages Details
☐	40357	*The lettin[g] of humours blood in the head-vaine ...*	0.25	0.21	0.22	EEBO	Pages Details
☐	40358	*The first booke of songes & ayres of foure parts wi...*	undefined	0.11	0.19	EEBO	Pages Details

From within the Corpus Manager, documents, document sets, pages, or page sets can each be queued up as the target for a "task" such as "OCR this with Tesseract 3." Each task is also configurable via a JSON specification which allows the user to specify which parameters need to be set at run-time, which files need to be transferred over to the job site in order for the task to run, and which files need to be collected and associated as results for the task. For an example of a JSON task configuration, see the end of this document. Once a job has finished and its results have been processed, the Corpus Manager also allows the user to view the results, which are associated with the job's target (a particular document or page), and accounted for separately according to each individual job.



Doctrina cristiana en lengua mexicana

Details

ID: 1251
Corpus: FirstBooks
Author:
Path: /data/primeros_libros/pl_corpus_1702/pl_blac_019/1000
ocular_transcription_jobs: [{"name": "pl_blac_019_OCR_5.15", "value": "1059"}, {"name": "pl_blac_019-OCR_5.26-haa-v03-2", "value": "1113"}, {"name": "pl_blac_019-ocr-7.19.17-mvf", "value": "1325"}]
language: nahuatl
fb_work_id: pl_blac_019
training_set: 22
font_1: blackletter
ocular_font_training_jobs: [{"name": "pl_blac_019_fonttraining_5.15", "value": "1057"}, {"name": "pl_blac_019-train_5.26-haa-v03", "value": "1096"}, {"name": "pl_blac_019-train_5.26-haa-v03", "value": "1100"}, {"name": "pl_blac_019-OCR_5.26-haa-v03", "value": "1107"}, {"name": "testing_binarize_threshold", "value": "1133"}, {"name": "pl_blac_019-train-7.18.17-mvf", "value": "1280"}]
printer: J. Pablos

Job Files

pl_blac_019-train_5.26-haa-v03(1096)	**Ocular Font:** 17-05-24_spalatnah_500000_6-v03.fontser **Ocular Glyph Substitution Model:** 17-05-24_spalatnah_500000_6-v03.gsmser **Ocular Language Model:** 17-05-24_spalatnah_500000_6-v03.lmser
pl_blac_019-train_5.26-haa-v03(1100)	**Ocular Font:** 17-05-24_spalatnah_500000_6-v03.fontser **Ocular Glyph Substitution Model:** 17-05-24_spalatnah_500000_6-v03.gsmser **Ocular Language Model:** 17-05-24_spalatnah_500000_6-v03.lmser
pl_blac_019-train-7.18.17-mvf(1280)	**Ocular Font:** 17-05-28_spalatnah_1000000_6_v03.fontser **Ocular Glyph Substitution Model:** 17-05-28_spalatnah_1000000_6_v03.gsmser **Ocular Language Model:** 17-05-28_spalatnah_1000000_6_v03.lmser
pl_blac_019-OCR_5.26-haa-v03(1107)	**Ocular Font:** 17-05-24_spalatnah_500000_6-v03.fontser **Ocular Glyph Substitution Model:** 17-05-24_spalatnah_500000_6-v03.gsmser **Ocular Language Model:** 17-05-24_spalatnah_500000_6-v03.lmser
pl_blac_019_fonttraining_5.15(1057)	**Ocular Font:** 170223_spanlat_06.fontser **Ocular Glyph Substitution Model:** 170224_spanlat_06.gsmser **Ocular Language Model:** 170223_spanlat_06.lmser

Pages

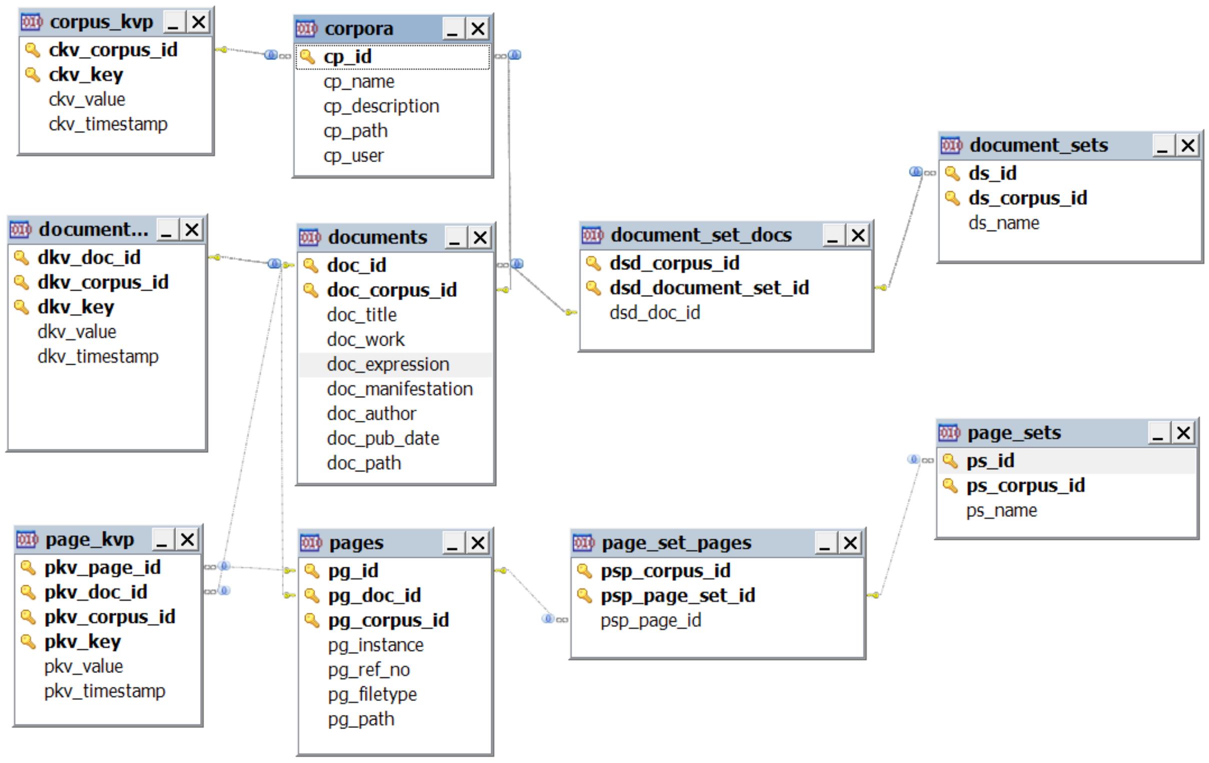
Page Num: 1 **ID:** 101879 **Image**: pl_blac_019_00001-1000.jpg **fb_page_id**: pl_blac_019_00001	pl_blac_019-OCR_5.26-haa-v03-2(1113)	**Ocular Comparison:** pl_blac_019_00001-1000_comparisons.txt **Ocular DIPL XML:** pl_blac_019_00001-1000_dipl.alto.xml **Ocular NORM XML:** pl_blac_019_00001-1000_norm.alto.xml **Ocular Normalized Transcription:** pl_blac_019_00001-1000_transcription_normalized.txt **Ocular Transcription:** pl_blac_019_00001-1000_transcription.txt
	pl_blac_019-ocr-7.19.17-mvf(1325)	**Ocular Comparison:** pl_blac_019_00001-1000_comparisons.txt **Ocular DIPL XML:** pl_blac_019_00001-1000_dipl.alto.xml **Ocular NORM XML:** pl_blac_019_00001-1000_norm.alto.xml **Ocular Normalized Transcription:** pl_blac_019_00001-1000_transcription_normalized.txt **Ocular Transcription:** pl_blac_019_00001-1000_transcription.txt
Page Num: 2 **ID:** 101880 **Image**: pl_blac_019_00002-1000.jpg **fb_page_id**: pl_blac_019_00002	pl_blac_019-OCR_5.26-haa-v03-2(1113)	**Ocular Comparison:** pl_blac_019_00002-1000_comparisons.txt **Ocular DIPL XML:** pl_blac_019_00002-1000_dipl.alto.xml **Ocular NORM XML:** pl_blac_019_00002-1000_norm.alto.xml **Ocular Normalized Transcription:** pl_blac_019_00002-1000_transcription_normalized.txt **Ocular Transcription:** pl_blac_019_00002-1000_transcription.txt
	pl_blac_019-ocr-7.19.17-mvf(1325)	**Ocular Comparison:** pl_blac_019_00002-1000_comparisons.txt **Ocular DIPL XML:** pl_blac_019_00002-1000_dipl.alto.xml **Ocular NORM XML:** pl_blac_019_00002-1000_norm.alto.xml **Ocular Normalized Transcription:** pl_blac_019_00002-1000_transcription_normalized.txt **Ocular Transcription:** pl_blac_019_00002-1000_transcription.txt

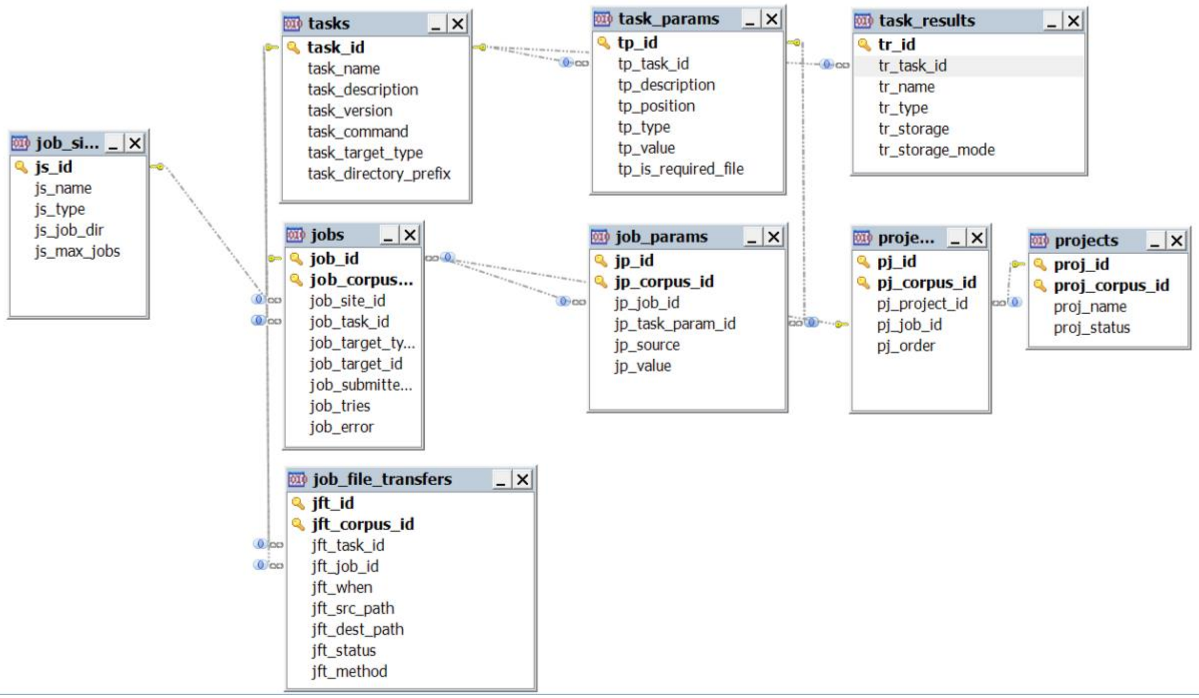
The Job Manager allows for users to edit the JSON configurations for any tasks available to the corpus, and also provides a way to track the progress of running jobs (jobs are tasks applied to targets, such as documents or pages). Those jobs can also be reconfigured or retried from within the Job Manager.

It's important to note a couple of features of the web application that make it particularly suitable for the managing of large corpora. The first is that all requests for corpus data are made using a REST API that enforces data paging. At no point is the transmission of huge, unmanageable chunks of data entrusted to an ephemeral, short-lived web request. The web application also handles certain tasks *asynchronously*, such as the preparation of submitted jobs which includes monitoring files that are being transferred over via Globus, or the processing of job results once they have been transferred over by the DH Agent (see below). In order to accomplish this, the web application relies on a Redis message broker and a Celery job queue in order to process tasks in the background.

While there is a git repo available for the web application, much work remains to be done, especially in terms of documentation: https://gitlab.dh.tamu.edu/bptarpley/dh_dashboard

The DH Dashboard MySQL database uses a special schema that allows for two unique features. The first is a set of tables that support the association of arbitrary key/value pairs to objects at the corpus, document, and page level, allowing for the kinds of data being stored about a corpus to remain, in effect, "schemaless" and therefore flexible enough to cater to any kind of corpus-based DH project. The second feature is the implementation of table partitioning according to each corpus' unique identifier. In other words, while the key/value pairs for every page across every corpus lives in one table, that table is physically partitioned across multiple files—one file per corpus. In this way, should a corpus that attempts to keep track of all of EEBO, for instance, require millions of rows in this table, the presence of those rows in no way affects the performance of key/value pair retrieval for other corpora, given that data lookups only happen on a per-corpus basis. In this way, the DH Dashboard MySQL schema is particularly suited to multiple large-scale corpora. Two simple ERD diagrams appear below—one for the tables used to keep track of corpora, documents, pages, etc., and another for the tables used to keep track of tasks, jobs, etc. While it seems like bad table design to include the corpus_id in so many of these tables, we require it to enable the partitioning mentioned above.





The DH Agent is a Python package that can be installed on any Linux-based server allowing it to establish that server as a "job site" for the running of tasks queued up by the DH Dashboard web application. The Agent is designed to work like a daemon, in that it is run every minute via a Cron job. When run, the Agent checks to see whether it can run—if enough Agents have already been spawned by Cron jobs, it will immediately stop (the number of Agents capable of running at once is configurable in the DH Agent settings file). If it can run, it then checks the MySQL database for any jobs that are ready to run. It then passes the job's JSON configuration (including the parameters configured by the user at launch time) off to the Python script designated for running that job's task. These Python scripts (one for every available task) are then entrusted with the running of that task. At present, many of these scripts support the parallelization of tasks by submitting sub-tasks to a SLURM job queue, thus enabling the DH Agent to take advantage of "job sites" like a high-performance computing cluster. Once a task's script finishes executing, the DH Agent can then either run a post-processing script assigned to that task, or simply collect the results and initiate a Globus transfer so that the resulting files can be processed by the DH Dashboard web application. At present we have not made public the git repo for the DH Agent, as we have a few security concerns we want vetted by the Brazos Supercomputing Cluster.

The DH Corpus Python package is a module that is used by both the web application and the agent, and it's purpose is to serve as a wrapper for the MySQL database providing simple, Python classes for the manipulation of corpora, documents, pages, etc. Git repo: https://gitlab.dh.tamu.edu/bptarpley/dh_corpus

The DH Job Python package is another module used by both the web application and the agent, and it provides the same simple Python class interface for tasks and jobs that live in the MySQL database. Git repo: https://gitlab.dh.tamu.edu/bptarpley/dh_job

A sample task configuration in JSON:

```
{
    "parameters": {
        "document_pageset_key": {
            "type": "static",
            "value": "training_set"
        },
        "input_font_path": {
            "type": "choice_from_kvp_list",
            "label": "Font",
            "kvp_path": "target.corpus",
            "key": "ocular_fonts"
        },
        "input_language_model_path": {
            "type": "choice_from_kvp_list",
            "label": "Language Model",
            "kvp_path": "target.corpus",
            "key": "ocular_lms"
        },
        "input_gsm_path": {
            "type": "choice_from_kvp_list",
            "label": "GSM",
            "kvp_path": "target.corpus",
            "key": "ocular_gsms"
        },
        "continue_from_last_complete_iteration": {
            "type": "static",
            "value": "true"
        },
        "allow_glyph_substitution": {
            "type": "static",
            "value": "true"
        },
        "allow_overlapping_lines": {
            "type": "choice_from_kvp_list",
            "label": "Allow Overlapping Lines",
            "kvp_path": "target.corpus",
            "key": "allow_overlapping_lines"
        },
        "update_lm": {
            "type": "static",
            "value": "true"
        },
        "update_gsm": {
            "type": "static",
            "value": "true"
        },
        "emission_engine": {
            "type": "static",
            "value": "DEFAULT"
        },
        "binarizeThreshold": {
                                                "type": "user_string",
                                                "default": "0.18",
                                                "label": "Binarize Threshold"
                                },
                                "crop": {
                                                "value": "FALSE",
                                                "type": "static"
                                }
    },
    "file_transfers": [
        {
            "type": "list_of_objects",
            "objects_path": "PageSet(connection, id=target.kvp['training_set'], corpus_id=target.corpus_id).all_pages",
            "file_path_attribute": "path"
        },
        {
            "type": "file_path_from_parameter",
            "parameter": "input_font_path"
        },
        {
            "type": "file_path_from_parameter",
            "parameter": "input_language_model_path"
        },
        {
            "type": "file_path_from_parameter",
            "parameter": "input_language_model_path"
        },
        {
            "type": "file_path_from_parameter",
            "parameter": "input_gsm_path"
        }
    ],
    "results": [
        {
            "type": "file_paths",
            "targets": "[task_target]",
            "file_filter": "*.fontser",
            "target_match_condition": "True",
            "file_type": "Ocular Font"
        },
        {
            "type": "file_paths",
            "targets": "[task_target]",
            "file_filter": "*.lmser",
            "target_match_condition": "True",
            "file_type": "Ocular Language Model"
        },
        {
            "type": "file_paths",
            "targets": "[task_target]",
            "file_filter": "*.gsmser",
            "target_match_condition": "True",
            "file_type": "Ocular Glyph Substitution Model"
        },
        {
            "type": "append_choice",
            "target": "task_target",
            "kvp_key": "ocular_font_training_jobs",
            "name": "{config['job'].name}",
            "value": "{config['job'].id}"
        }
    ]
}
```