Exordium Documentation

Release 1.4.3

CJ Kucera

Mar 21, 2023

Contents

1 Assumptions **3**

2 Limitations **5**

3 Requirements **7**

4 Installation **9**

5 Administration **11**
- 5.1 Library Configuration . . . 11
- 5.2 Library Upkeep . . . 13
- 5.3 Django Admin . . . 13

6 Screenshots **15**
- 6.1 Main Page . . . 15
- 6.2 Browsing Artists . . . 16
- 6.3 Artist Listing . . . 16
- 6.4 Browsing Albums . . . 18
- 6.5 Album View . . . 18
- 6.6 Classical Tags . . . 18
- 6.7 Album Zipfile Downloads . . . 21
- 6.8 Searching . . . 21
- 6.9 Live Albums . . . 21
- 6.10 Streaming . . . 22
- 6.11 Administration . . . 22

7 Apache/WSGI Deployment Issues **25**
- 7.1 Locale Issues . . . 25
- 7.2 Process Count . . . 26

8 Migration from Other Libraries **27**

9 Rocky/Alma/RHEL/OEL/Centos-Stream 8 Apache/WSGI Deployment HOWTO **29**
- 9.1 Requirements . . . 29
- 9.2 System Preparation . . . 29
- 9.3 Virtenv Creation / Django Installation . . . 30
- 9.4 Django Configuration / settings.py . . . 30
- 9.5 selinux . . . 31

9.6 WSGI Configuration in Apache . . . 32
9.7 Cross Origin Opener Policy Headers . . . 33
9.8 Apache Configuration: mp3/zipfile access . . . 33
9.9 Other Minor Tweaks . . . 33

10 Changelog **35**
10.1 1.4.3 (2023-03-21) . . . 35
10.2 1.4.2 (2022-03-15) . . . 35
10.3 1.4.1 (2022-03-04) . . . 35
10.4 1.4.0 (2022-03-03) . . . 35
10.5 1.3.4 (2021-03-25) . . . 36
10.6 1.3.3 (2020-05-28) . . . 36
10.7 1.3.2 (2018-09-20) . . . 36
10.8 1.3.1 (2018-09-20) . . . 37
10.9 1.3.0 (2018-01-02) . . . 37
10.10 1.2.1 (2017-11-28) . . . 37
10.11 1.2.0 (2017-11-28) . . . 37
10.12 1.1.1 (2016-12-30) . . . 37
10.13 1.1.0 (2016-12-30) . . . 37
10.14 1.0.3 (2016-11-22) . . . 38
10.15 1.0.2 (2016-10-21) . . . 38
10.16 1.0.1 (2016-10-21) . . . 38
10.17 1.0.0 (2016-10-18) . . . 38

11 TODO **39**

12 Licenses **41**
12.1 Exordium . . . 41
12.2 jPlayer . . . 42
12.3 jQuery . . . 42

13 Exordium Music Library **43**
13.1 Introduction . . . 43
13.2 Download . . . 44
13.3 Detailed Documentation . . . 44
13.4 Other Information . . . 44

As mentioned on the main page, Exordium does not really attempt to be a general-purpose web music library suitable for widespread use. The only configuration options currently available are those necessary for basic operation. Exordium was born out of my persistent dissatisfaction with existing web music library applications. I've been using various library applications over the years but have always ended up maintaining my own patchsets to alter their behavior to suit what I like, and in the end I figured it would be more rewarding to just write my own.

So, Exordium represents essentially my own personal ideal of a web music library application, and its design decisions and operational goals reflect a very specific set of requirements: my own. If your ideal music library differs from my own in even moderate ways, other music library apps are much more likely to be to your liking.

I would, of course, be happy to accept patches which add, extend, or modify Exordium behavior, so long as the current functionality remains the default. I certainly don't actually *expect* any patches, of course, given that Exordium's target market is exactly one individual.

CHAPTER 1

Assumptions

- Except for the Javascript necessary to hook into jPlayer, and jPlayer itself, there is no client-side Javascript or AJAX-style dynamic page content. All HTML is generated server-side. The application is quite usable from text-based browsers.
- Music files must be accessible via the local filesystem on which Django is running, and stored as either mp3, ogg vorbis, ogg opus, or m4a (mp4a).
- The entire music library must be available from a single directory prefix. If subdirs of this root dir span multiple volumes (or network mounts), that's fine, but there is NO support for multiple libraries in Exordium.
- Exordium itself will never attempt to write to your library directory for any reason - all music files (and album art) are managed outside of this app. Write access to a directory on the filesystem is required for zipfile downloads, but that directory need not be in your music library.
- Music files should be, in general, arranged scrupulously: All files within a single directory belong to the same album, and an album should never span multiple directories. There's actually plenty of wiggle room here, and Exordium should correctly deal with directories full of "miscellaneous" files, etc, but in general the library should be well-ordered and have albums contained in their own directories. This is less important during the initial library import, but becomes much more important when updating tags or rearranging your filesystem layout, as Exordium uses the directory structure to help determine what kind of changes have been made.
  - Directory containment is the primary method through which Various Artists albums are collated. A group of files in the same directory with different artists but the same album name will be sorted into a single "Various" album containing all those tracks. Conversely, if an album name is shared by tracks from different directories (each dir's files with a different artist name), multiple albums will be created.
  - Tracks without an album tag will be sorted into a "virtual" album entitled "Non-Album Tracks: Band Name" - this is the one case where it's expected that this virtual "album" might span multiple directories.
- The artwork for albums should be contained in gif/jpg/png files stored alongside the mp3s/oggs/opus/m4as, or in the immediate parent folder (in the case of multi-disc albums, for instance). Filenames which start with "cover" will be preferred over other graphics in the directory. PNGs will be preferred over JPGs, and JPGs will be preferred over GIFs.
  - Artwork thumbnails will be stored directly in Django's database, in blatant disregard for Django best practices. IMO the benefits far outweigh the performance concerns, given the scale of data involved.

- Music files should be available directly via HTTP/HTTPS, using the same directory structure as the library. This does not have to be on the same port or even server as Django, but the direct download and streaming functionality rely on having a direct URL to the files.
- Album zipfile downloads, similarly, require that the zipfile directory be accessible directly over the web. As with the music files, this does not need to be on the same port or even server as Django, but Django will not serve the zipfile itself. The reason for this is that I want to be able to pass the zipfile URL to other apps for downloading, and for downloads to be easily resumable in the event they're accidentally cancelled before they're finished. The text on the download page mentions that zipfiles are kept for around 48 hours, but that cleanup is actually not a function of Exordium itself. Instead, I just have a cronjob set on the box like so:

```
0 2 * * * /usr/bin/find /var/audio/exordiumzips -type f -name "*.zip" -mtime +2 -
↪print -exec unzip -v {} \; -exec rm {} \;
```

- Tags for information commonly associated with classical music are supported, namely: Group/Ensemble, Conductor, and Composer. *(For ID3 tags: TPE2, TPE3, and TCOM, respectively. In Ogg Vorbis, the more sensible ENSEMBLE, CONDUCTOR, and COMPOSER. M4A files only support a flag for Composer.)* Albums will still be defined by their main Artist/Album association, and Artist is always a required field, whereas Group/Conductor/Composer are all optional. Internally, these are all stored as "artists," so when browsing by artist, Exordium should do the right thing and show you all albums containing an artist, whether they showed up as artist, composer, conductor, or ensemble.
- There are many live concert recordings in my personal library, which I've uniformly tagged with an album name starting with "YYYY.MM.DD - Live". Given the volume of these albums, Exordium will automatically consider any album matching that name as a "live" album. *(Dashes and underscores are also acceptable inbetween the date components.)* By default, Exordium will hide those live albums from its display, since they otherwise often get in the way. A checkbox is available in the lefthand column to turn on live album display, though, and it can be toggled at any time.
- The "addition date" of albums into the library is an important data point; Exordium's main view is the twenty most recently-added albums. To that point, updates of the music files will allow the album records to be updated while keeping the addition time intact. Some specific cases in which this is ensured:
  - Updating album/artist names in the file's tags
  - Moving music files from one directory to another, or renaming the files

  Combining the two may, however, result in the album being deleted from the library and then re-added. If the tags on a collection of files are updated (so that the file's checksum changes), **and** the files are moved into a separate directory, the album will end up being re-added, since there's no reasonable way to associate the updated files with the old ones.

  The most common case of that would be if there was a typo in the album or artist name for an album, and that typo was replicated in the directory structure containing the files. Fixing the typo would involve changing both the tags and the directory names. In order to keep the addition time intact in this case, you would have to perform both steps separately, running an update after each one.

CHAPTER 2

Limitations

There are some inherent limitations of Exordium, based on the assumptions that have been made during its development (and in my own music library).

- The artist name "Various" is reserved. Tracks with an artist tag of "Various" will not be added to the library.
- Artist and Trackname tags are required. Tracks will not be added to the library if either of those tags are missing.
- If two Various Artists albums with the same album name exist in the library, they'll end up stored as one single album in the DB.
- If two directories contain files which seem to be in the same album (by the same artist), you'll end up with an album which spans directories. Behavior may not be well-defined in that case.
- Exordium completely ignores genre tags. I've personally always been lousy at putting reasonable values in there on my media, and so that's been very unimportant to me. It'd probably be good to support them anyway, though.
- Exordium only supports mp3, ogg vorbis, ogg opus, and m4a currently, though other support should be reasonably simple to add in, so long as Mutagen supports the format.
- m4a tags don't seem to allow for Ensemble or Conductor, so that data will never be present for m4a files. (If support for those tags is in there somewhere, I'd like to hear about it.)

CHAPTER 3

Requirements

Exordium requires at least Python 3.8 *(tested on 3.9)*, and Django 4.0.

Exordium makes use of Django's session handling and user backend mechanisms, both of which are enabled by default. This shouldn't be a problem unless they've been purposefully disabled.

Exordium requires the following additional third-party modules:

- mutagen (built on 1.45)
- Pillow (built on 9.0)
- django-tables2 (built on 2.4)
- django-dynamic-preferences (built on 1.11), which in turn requires:
  - six (built on 1.16.0)
  - persisting-theory (built on 0.2.1)

One unit test module additionally requires django-test-migrations (tested with 1.2.0), but that's not required to run it.

These requirements may be installed with `pip`, if Exordium itself hasn't been installed via `pip` or some other method which automatically installs dependencies:

```
pip install -r requirements.txt
```

CHAPTER 4

Installation

These instructions assume that you already have a Django project up and running. For instructions on setting up Django for the first time, if installing a brand new application server just for a web music library doesn't deter you, djangoproject.com has some good documentation:

- https://docs.djangoproject.com/en/4.0/intro/install/
- https://docs.djangoproject.com/en/4.0/intro/tutorial01/

Once Django is installed and running:

1. Install Exordium via `pip install django-exordium`
    - If Exordium hasn't been installed via `pip` or some other method which automatically installs dependencies, install its dependencies:

    ```
    pip install -r requirements.txt
    ```

2. Add exordium, django_tables2, and dynamic_preferences to your `INSTALLED_APPS` setting like this:

    ```
    INSTALLED_APPS = [
        ...
        'exordium',
        'django_tables2',
        'dynamic_preferences',
        'dynamic_preferences.users.apps.UserPreferencesConfig',
    ]
    ```

3. *(Optional)* For jPlayer streaming to work properly on a "live" install, the Cross-Origin-Opener-Policy HTTP header has to be set properly. (This will generally not be an issue when running Django in "test" mode via `runserver`.) Django defaults to using `same-origin`, but unless your static content delivery also uses the same header, launching the streaming window will fail. You can set the header to `same-origin-allow-popups` inside `settings.py` to make this work, or ensure that your static files set the proper header. (Making sure static files use `Cross-Origin-Opener-Policy: same-origin` just like Django will do the trick.) Setting the Django default can be done with:

```
SECURE_CROSS_ORIGIN_OPENER_POLICY = 'same-origin-allow-popups'
```

If your static content isn't served from the same protocol/hostname/port as Django itself, you will likely have to set either Django or your static files' value to `unsafe-none` instead.

4. Include the exordium URLconf in your project `urls.py` like this:

```
path('exordium/', include('exordium.urls')),
```

5. Run `python manage.py migrate dynamic_preferences` to create the Dynamic Preferences models, if this wasn't already configured on your Django install.
6. Run `python manage.py migrate exordium` to create the Exordium models.
7. Run `python manage.py loaddata --app exordium initial_data` to load some initial data into the database. *(This is not actually strictly speaking necessary - the app will create the necessary data automatically if it's not found.)*
8. If running this from a "real" webserver, ensure that it's configured to serve Django static files. Then run `python manage.py collectstatic` to get Exordium's static files in place. If you didn't want to set `same-origin-allow-popups` for Django's COOP header, make sure that your server sends a `Cross-Origin-Opener-Policy: same-origin` header along with these static files, or possibly `unsafe-none` if the static files protocol/hostname/port doesn't match Django's.
9. Either start the development server with `python manage.py runserver` or bring up your existing server. Also ensure that you have a webserver configured to allow access directly to your music library files, and optionally to the zipfile downloads Exordium will create.
10. Visit the administrative area in *Dynamic Preferences* > *Global preferences* and set the values for the following:
    - **Exordium Library Base Path**: This is what defines where your music library can be found on disk.
    - **Exordium Media URL for HTML5**: This is the base URL which provides direct access to the files in your library, used by the HTML streaming player. Omit the trailing slash, though things will probably work fine even if it's in there. Without this set properly, the streaming player will not work properly. Note that if your base URL for Exordium is https, this will have to be https as well, to avoid browser errors.
    - **Exordium Media URL for m3u**: This is the base URL which provides direct access to the files in your library, used by the m3u Playlist functionality, and also the direct song download links when enumerating tracks. This can be the same as the HTML5 URL. Omit the slash, though things will probably work fine even if it's in there. Without this set properly, m3u playlists and direct track downloads will not work properly. This URL can be http even if the main site is https.
    - **Exordium Zip File Generation Path**: Path on the filesystem to store zipfile album downloads. This is the one location in which the user running Django needs write access.
    - **Exordium Zip File Retrieval URL**: This is the base URL providing web access to that zipfile directory. Note that if your base URL for Exordium is https, this will have to be https as well, to avoid browser errors.

    Without the last two options, Exordium will still function fine, but the album-download button will not be rendered. Exordium will also function without the "*Exordium Media URL*" options being set properly, though with the caveats mentioned above.
11. If Zipfile downloads are configured, a process should be put into place to delete the zipfiles after a period of time. I personally use a cronjob to do this:

```
0 2 * * * /usr/bin/find /var/audio/exordiumzips -type f -name "*.zip" -mtime +2 -
↪print -exec unzip -v {} \; -exec rm {} \;
```

12. Visit the **Library Upkeep** link from the Exordium main page and click on "Start Process" to begin the initial import into Exordium!

CHAPTER 5

Administration

When logged in to Django as a staff member, the lefthand sidebar will include three links at the bottom, for Django administration:

5.1 Library Configuration

Library Configuration links to a Django administrative backend page provided by `django-dynamic-preferences`, which provides access to the only real configuration options available in Exordium. There are five variables which can be configured:

Exordium Library Base Path This is the directory on the server where Exordium can find all your music files. There can be various mounts underneath this directory (network or otherwise), but all music files must be underneath here somewhere. Note that even though Exordium will never attempt to do any write operations on the library, it's best if the user running Django does not have write access into this directory, anyway.

Exordium Media URL for HTML5 This is the URL which provides direct web access to the files contained in the library base path, above, and is used for the HTML5 streaming player. This will most likely be a static directory configured in Apache or whatever other frontend web server is in use. Technically this option does not have to be specified for Exordium to work, but music streaming won't work unless it is. Note that if your base URL for Exordium is https, this will have to be https as well, to avoid browser errors.

Exordium Media URL for m3u This is the URL which provides direct web access to the files contained in the library base path, above, and is used for the m3u playlists and direct song downloads. Technically this option does not have to be specified for Exordium to work, but m3u playlists and direct song downloads won't work unless it is. This URL can be http even if the main site is https.

Exordium Zip File Generation Path For full-album downloads, Exordium will create a zipfile on the filesystem and then give the user a link to that zipfile. This option specifies the directory in which the zipfile will be written. This is the only location on the filesystem where Exordium requires any write access. If this option is not specified, the button for album zipfile downloads will be hidden.

Exordium Zip File Retrieval URL Similar to "Exordium Media URL" above, this is the URL to the zipfile generation path, typically configured via Apache or whatever the frontend webserver is. Without this option, the button

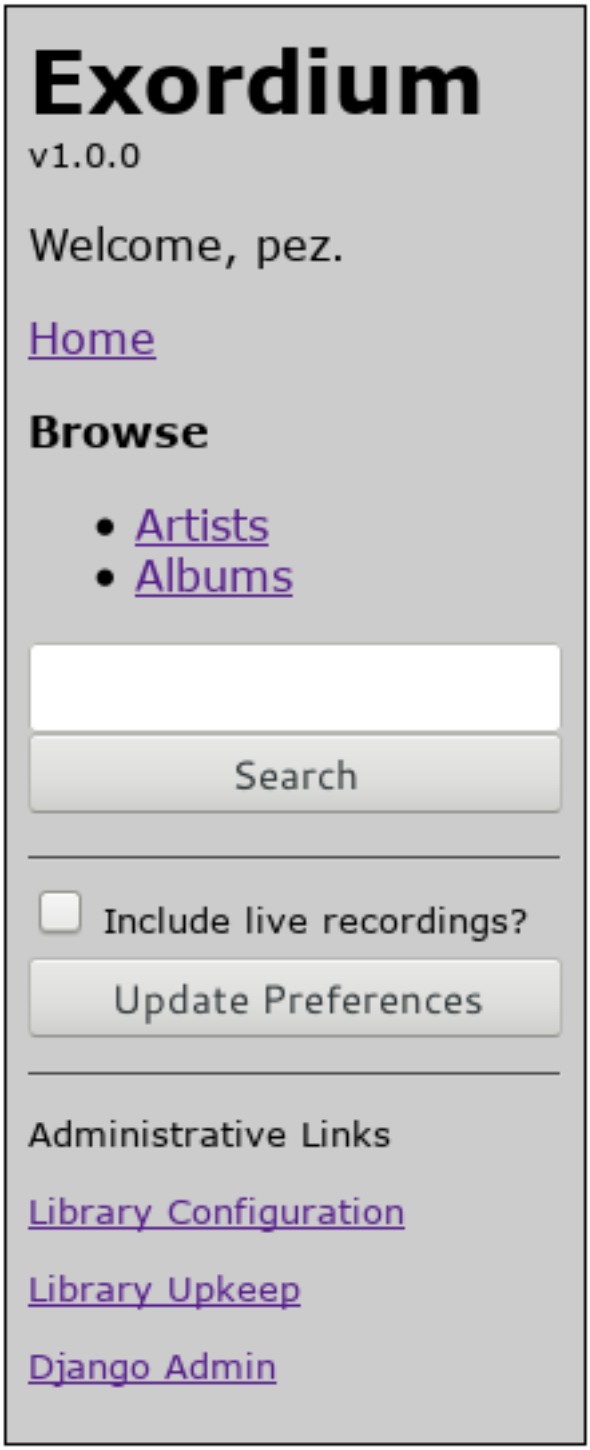

Fig. 1: Administrative Sidebar

for album zipfile downloads will be hidden. Note that if your base URL for Exordium is https, this will have to be https as well, to avoid browser errors.

5.2 Library Upkeep

Library Upkeep is where music will be added to the library, and changes/deletions will be processed. Clicking on the link will bring up the following, in addition to showing the library configuration values configured in **Library Configuration**:

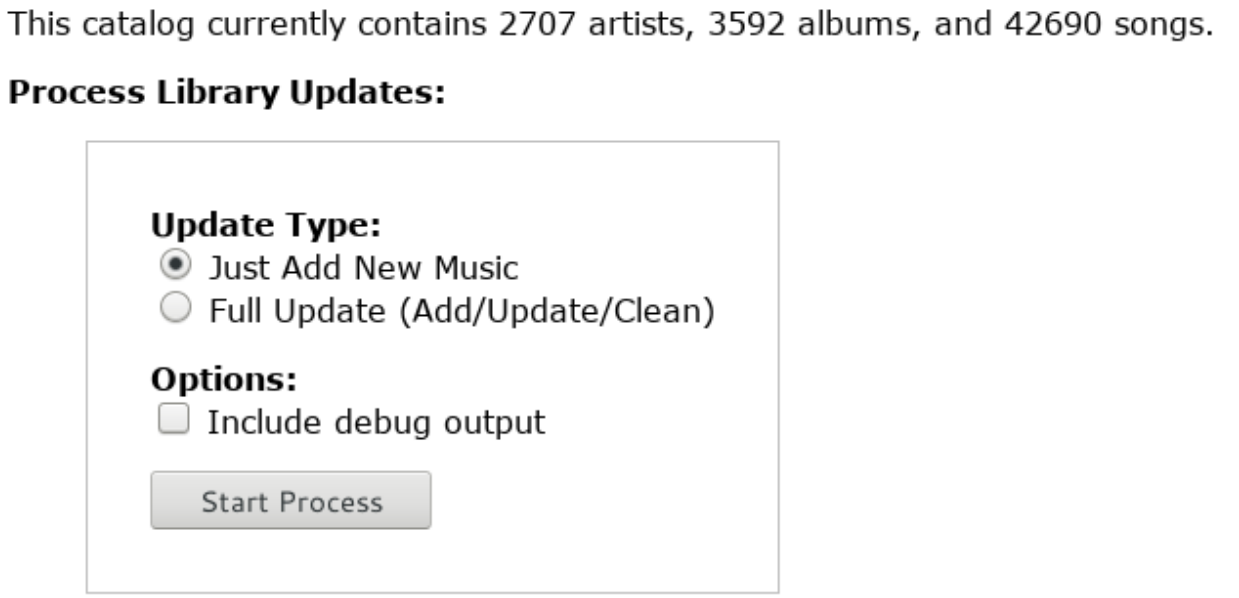

Fig. 2: Library Management

Most of the time you will just be interested in adding new music to the database, so keeping the defaults and hitting "Start Process" is all you need. If there have been changes to existing files on disk (or if files have been deleted, etc), then you can use the "Full Update" option to do a full comparison of the database to the on-disk state. In practice there isn't much difference between the two options, speedwise, even though the update option technically does more work. The "Full Update" will also re-scan for album art, for albums which do not have album art already.

The checkbox to "Include debug output" can be used to include more information about the update process as it proceeds, though in general there isn't a need to do so. If you encounter problems during the update, it would probably be nice to have that option turned on while investigating/reporting the bug.

Note that the initial load of a largeish music library into Exordium can take quite awhile. On my system, a library of 42,000 tracks takes about an hour to do the initial add, the majority of that time spent looping through the filesystem computing checksums of all the tracks. Exordium uses SHA256 for its checksums, and while SHA1 or MD5 are faster, and would probably be sufficient for our use, on my system this process is primarily I/O bound, and using the faster algorithms don't actually provide any significant speed increase.

5.3 Django Admin

This is just a convenience link to the main Django administration area. In general, there is unlikely to be much need to edit Exordium objects from inside the administration area, but it might be useful in some circumstances to tweak

values manually in there.

CHAPTER 6

Screenshots

This page contains various screenshots showing off Exordium's functionalty from a user's perspective.

6.1 Main Page

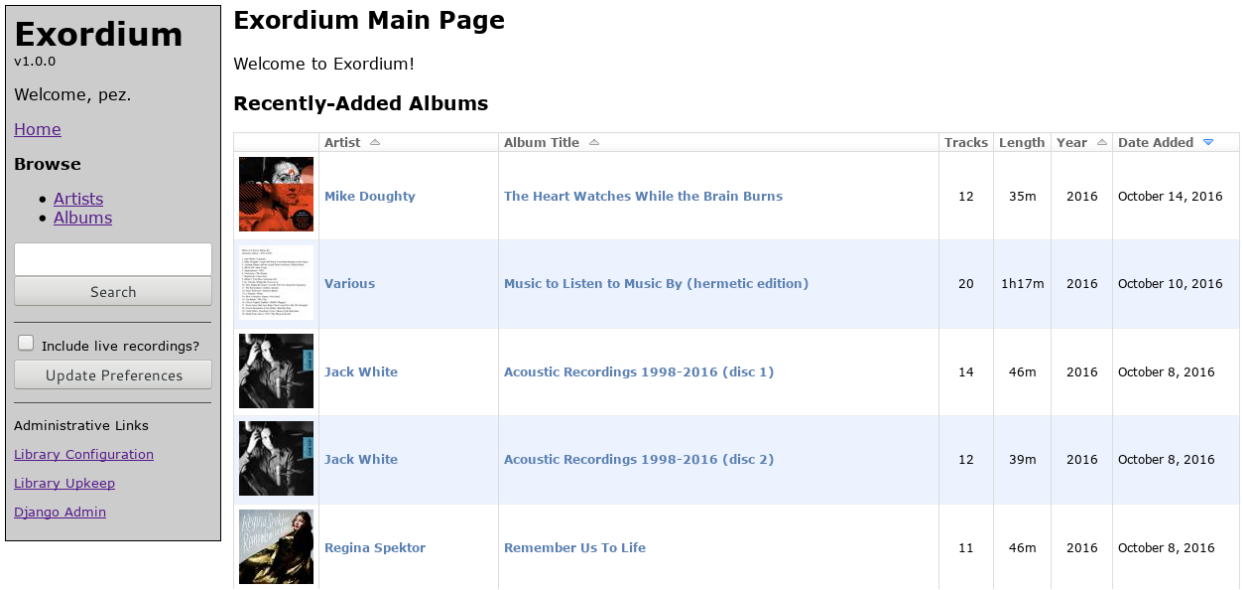

Fig. 1: Exordium Main Screen, showing recently-added albums.

All lists of albums will show the number of tracks, total album length, the released year of the album, and the date added to the library. The sidebar will be present on all pages. The "administrative links" at the bottom of the sidebar will only be shown to logged-in users who are set to staff.

The sidebar contains a textbox to search through the library, and a checkbox to either include or exclude live recordings

while browsing the library. Live recordings are albums whose title is of the format "*YYYY.MM.DD - Live**". Logged-in users will have their live-album preference saved between sessions.

6.2 Browsing Artists

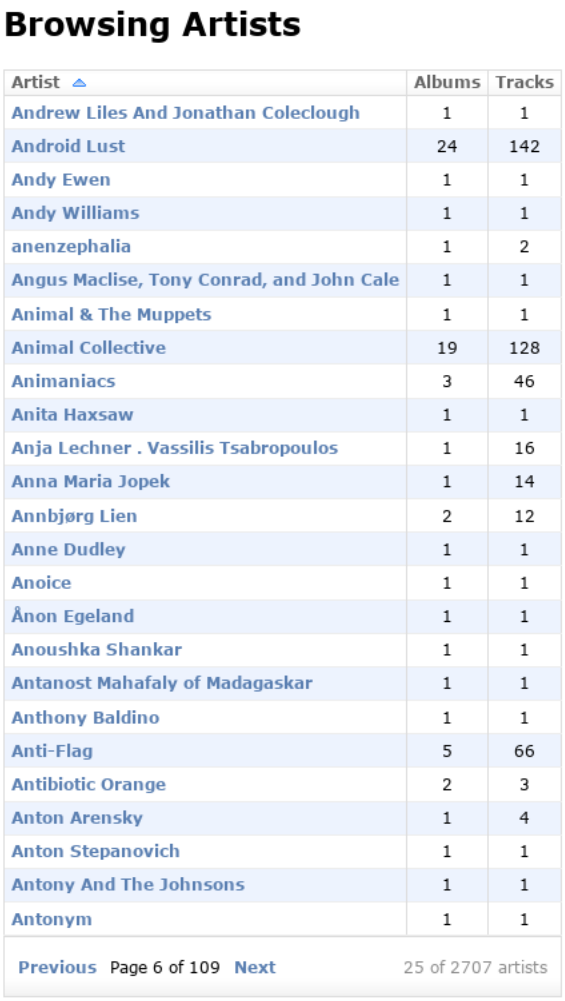

Fig. 2: Browsing artists.

This view will show the number of albums and number of tracks. Clicking on an artist name will bring up a list of albums and songs by that artist.

6.3 Artist Listing

Tracks by an artist which don't have an Album tag will get sorted into a special "Non-Album Tracks" album, as can be seen here. After all the albums explicitly belonging to the artist have been shown, any "Various Artists" album they appear in will be listed. In this case, Plaid can be found on two compilation albums.

Below the album list will be a song list. Each song will have two icons on the right-hand side of the table. The first, the arrow pointing down, provides a direct link to the track. The second will open up a popup window with the HTML5

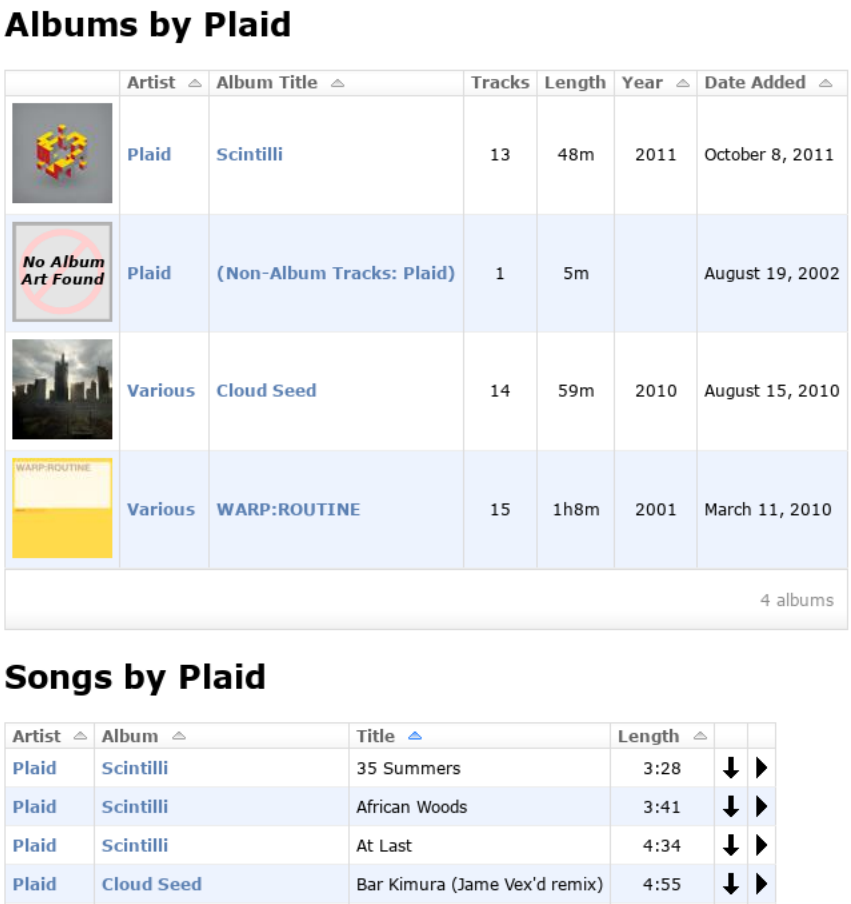

Fig. 3: Browsing all albums/songs by an artist.

media player jPlayer, which will then stream the track. Clicking on more than one track will add the track to jPlayer's playlist.

Artists with more than 500 songs will not have their song lists shown here, for performance reasons.

6.4 Browsing Albums

Fig. 4: Browsing albums.

The "Browse Albums" view will sort by Album Title by default.

6.5 Album View

After clicking on an album link, a full page will be shown containing all the album details. If zipfile downloading is configured, a "Download as Zipfile" button will be shown at the top. The two streaming buttons will be shown in any case - the first will open the HTML5 media player jPlayer in a popup window, and the second will generate an `.m3u` playlist which other media player applications should be able to use. The "Force Album Art Regen" button will only be visible to logged-in staff members, and will tell Exordium to look for new/updated album art in the album's directory.

A list of all tracks in the album is shown after the summary information. Like in the Artist view above, each track will have a download and stream button.

6.6 Classical Tags

Exordium supports tags for Ensemble/Group, Conductor, and Composer. These are most commonly seen with classical music, though of course they can be used on any track. Any tags common to all tracks in the album will only be reported up at the top section of the album view screen. Those which may differ from track-to-track will be inserted into the Artist column in the song list, as seen above.

These extra tags will also be shown in album lists where appropriate. For instance, the above album will look like the following in an album list:

Fig. 5: Album View

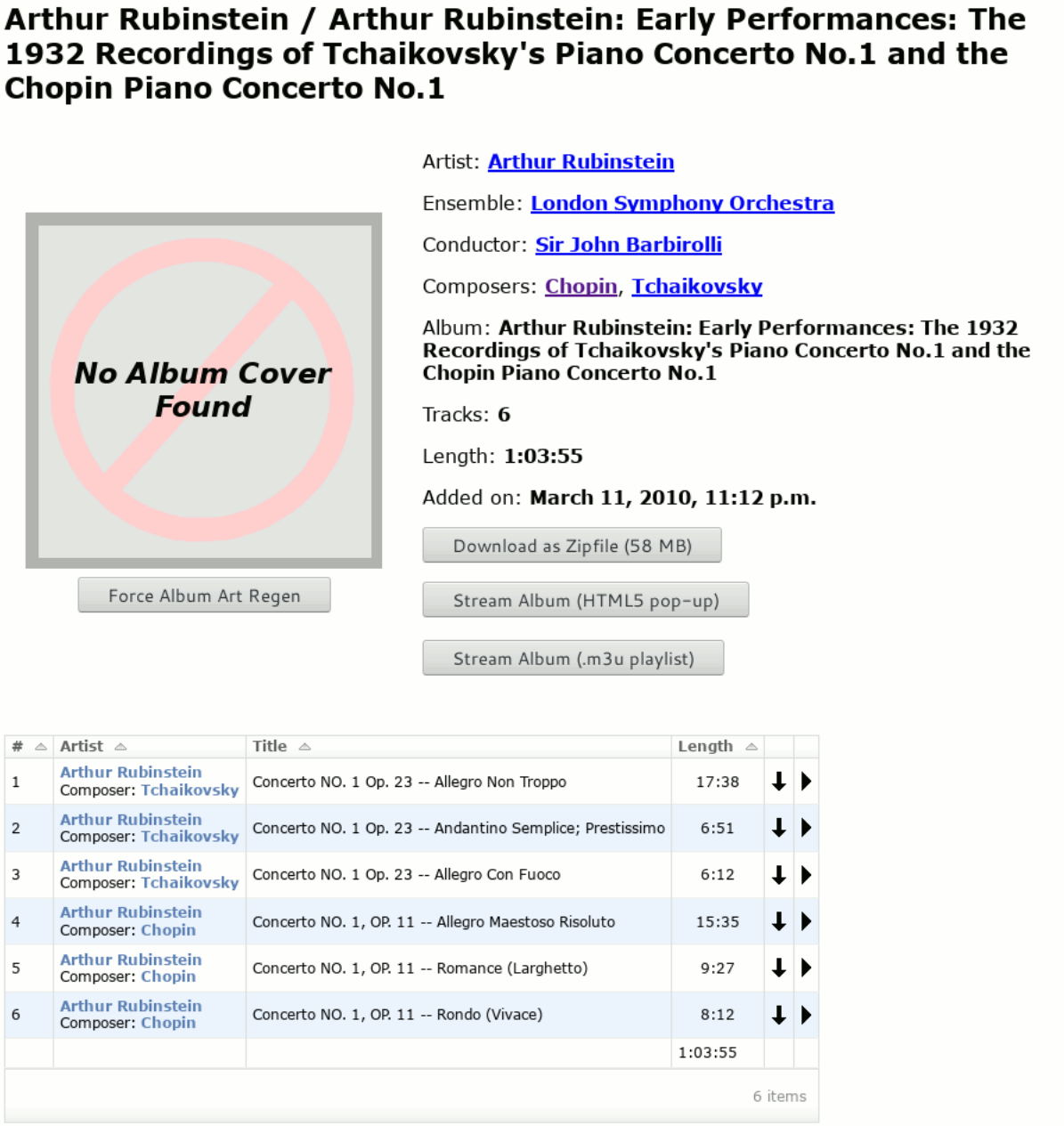

Fig. 6: Classical Tags

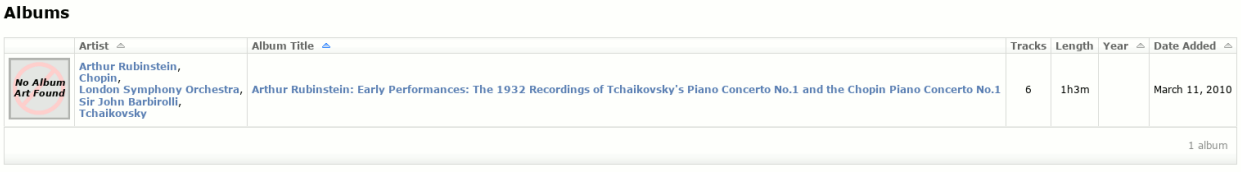

Fig. 7: Album List with Classical Tags

6.7 Album Zipfile Downloads

Fig. 8: Album Zipfile Downloads

Clicking the "Download as Zipfile" button will result in a page showing you the exact zipfile contents, and a link directly to the zipfile (using your configured Zipfile URL as a prefix). If the download link is clicked again while the zipfile is still present, Exordium will just provide a URL to the existing file, rather than regenerate.

6.8 Searching

The search box will match on artist names, album names, and song titles, and will show all relevant hits of each type. The screenshot above matched on an album name and a bunch of track names.

6.9 Live Albums

Exordium has the ability to hide or show "live" albums as-requested. If Exordium sees an album whose name looks like "YYYY.MM.DD - Live*", it will consider it a "live" album and hide it by default. This was put into place because

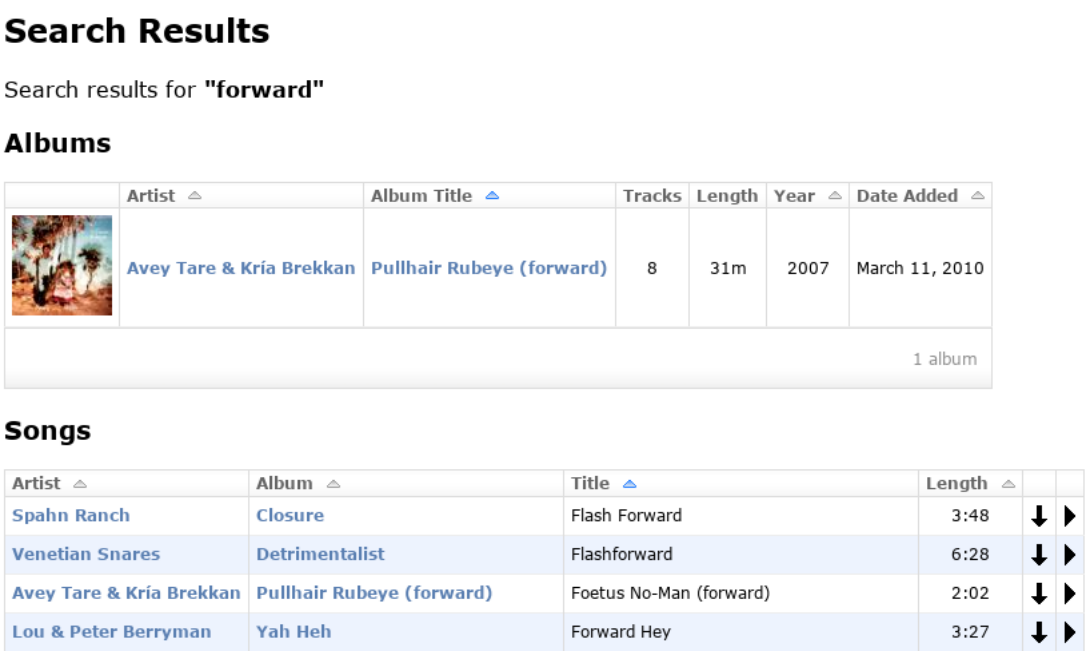

Fig. 9: Search Results

I have many live recordings in my library and they often overwhelm the list of albums that I'm more commonly interested in, otherwise. Here is a search for the band "23 Skidoo" without, and then with live albums turned on:

6.10 Streaming

The HTML5 media player jPlayer is used to handle in-browser streaming, via a popup. It's not fancy, but it gets the job done. It turns out that this player will even work on Android phones (and possibly iPhone, though I don't have one of those to test).

6.11 Administration

Screenshots of the administration sections can be found in *Administration*.

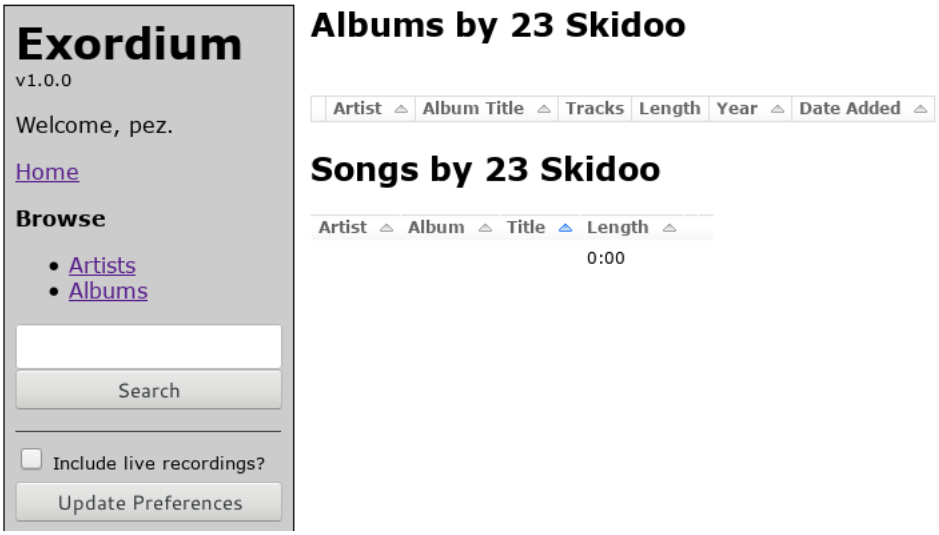

Fig. 10: Live albums turned off (the default)

Fig. 11: Live albums turned on

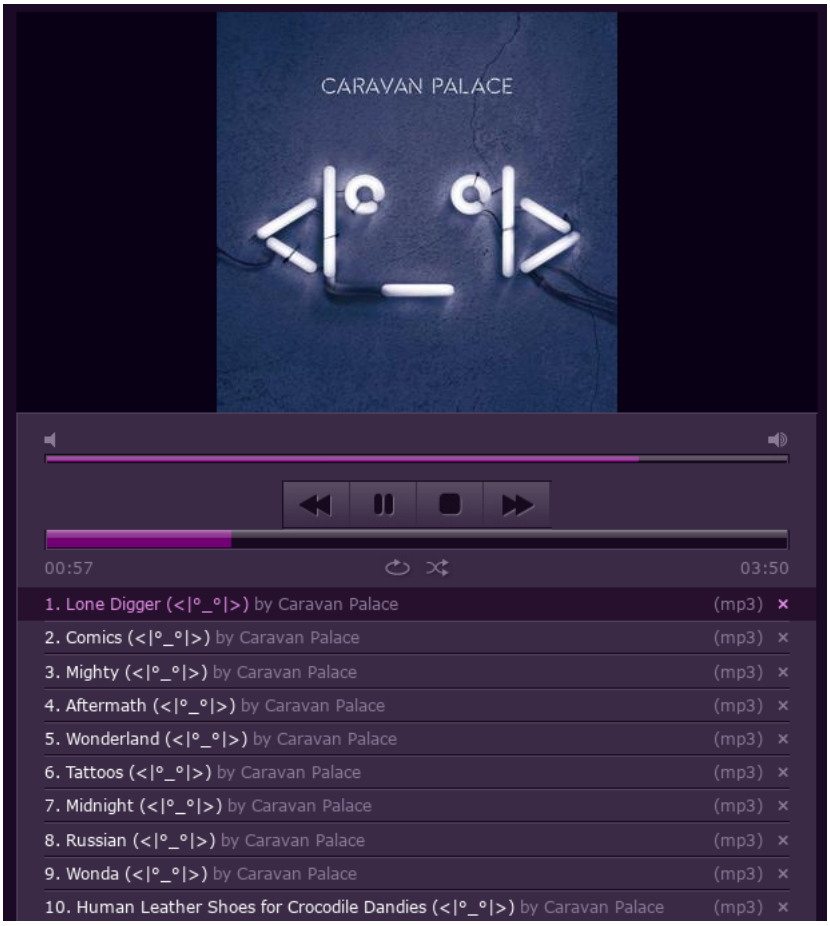

Fig. 12: Streaming an album

CHAPTER 7

Apache/WSGI Deployment Issues

7.1 Locale Issues

If deploying via Apache/WSGI, in EL7 (CentOS 7, RHEL7) there's a serious problem which can occur if any non-ASCII characters are found in your filenames. Basically, by default the WSGI process will be launched with a `$LANG` of `C`, making ascii the default encoding for various things, including the filesystem encoding as reported by `sys.getfilesystemencoding()`. If you try and import any files with non-ASCII characters in the filename, you can end up with absurd errors like this in your logs:

```
UnicodeEncodeError: 'utf-8' codec can't encode character '\\udcc3' in position 7160:␣
↪surrogates not allowed
```

This behavior is especially difficult to track down since it will NOT be repeatable in any unit tests, nor will it be repeatable when running the development test server - it'll only ever show up in the WSGI deployment.

Currently Exordium doesn't have a check for this - I'll hope to eventually add that in - but for now just make sure that you're specifying the following after your `WSGIDaemonProcess` line:

```
lang='en_US.UTF-8' locale='en_US.UTF-8'
```

Of course, replacing the encoding with the proper one for the data stored on your filesystem.

There may be some similar problems if more than one encoding is found in your system's filenames - that's another thing I have yet to investigate.

You can read a bit more on this problem here, FWIW: http://blog.dscpl.com.au/2014/09/setting-lang-and-lcall-when-using.html

It's worth noting that this problem was discovered on CentOS 7. When deploying on Rocky 8, I'd kept those `lang` and `locale` lines in my WSGI config without really doublechecking that this was still necessary, so it's possible that this might not be actually needed anymore.

7.2 Process Count

The `WSGIDaemonProcess` parameter in Apache lets you specify an arbitrary number of `processes` (in addition to `threads`). If `processes` is set to more than `1`, problems can be encountered when setting preferences (such as library path, download URLs, live album display, etc). Namely, the preference change will often only be seen by the process in which it was changed, which can lead to some vexing behavior.

I believe the root of this problem is that the dynamic_preferences module probably uses a cache (presumably a builtin Django cache), and that cache must be configured properly so that multiple processes can share it. I have not actually investigated this, though. Given that my personal activity needs with Exordium are quite light, I've just made do with a single process.

CHAPTER 8

Migration from Other Libraries

Practically no support is included for converting an existing music library database in some other app to Exordium. There IS one administrative subcommand provided to import album addition times from an Ampache MySQL database, though, which can be accessed by running:

```
python manage.py importmysqlampachedates --dbhost <host> --dbname <name> --dbuser
↪<user>
```

The subcommand will prompt you for the database password via STDIN. Note that this has only been tested with Ampache 3.7.0.

CHAPTER 9

Rocky/Alma/RHEL/OEL/Centos-Stream 8 Apache/WSGI Deployment HOWTO

Exordium is my first application written in Django, and served as my introduction to Django in general. This page is more for my own reference than anyone else's, though perhaps it will come in useful for someone else with similar requirements who's unfamiliar with Django.

9.1 Requirements

I have a Rocky 8 server which runs Apache and MariaDB which serves a variety of web-based applications (mostly PHP-based), primarily for my own personal use. Apache is already set up to handle user authentication itself, via Apache's native `Auth*` configuration directives, and all my webapps share that common authentication mechanism.

I have one vhost on SSL which is where the actual webapps live, but I also have another vhost which uses plain HTTP (and no authentication), and a subdirectory of that had already been set up in the past to provide direct access to my music library. I've always enjoyed having that in place, because URLs to songs can be constructed which don't require authentication, can be plugged into .m3u playlists for remote music listening, and are generally just easier to deal with. The directory doesn't have directory indexing enabled, so there's a bit of obscurity there, though given a link to a single track it wouldn't be hard to guess my naming conventions and figure out links to other media. C'est la vie!

Regardless, there's a couple of differences to a "stock" Django deployment here, namely that I don't want to use Django's default user authentication methods, and I'd like to continue to use MariaDB instead of Django's recommended PostgreSQL. Fortunately, both are quite easy to configure in Django.

9.2 System Preparation

EL8 systems offer a variety of Python versions supported out-of-the-box using Modules. At time of writing, EL8 systems provide 2.7, 3.6, 3.8, and 3.9. Those versions each have their own support life cycle which is good to be aware of. I wanted to be on the latest Django (4.0, at time of writing), which supports 3.8 at a minium, so choosing 3.9 was the obvious choice.

The packages that I've got installed are:

- python39
- python39-pip
- python39-mod_wsgi
- python39-devel
- mariadb-connector-c-devel

The last two packages are, I believe, required for pip to build mysql client library that I'm using.

9.3 Virtenv Creation / Django Installation

The next step was to create a virtual environment to hold all the necessary Django code, and Exordium dependencies. I chose to put that under a `/var/www/django` directory (which is of course not actually inside my Apache web root). My initial steps for this were just:

```
$ cd /var/www/django
$ python3.9 -m venv virtenv
$ source virtenv/bin/activate
(virtenv) $ pip install django
(virtenv) $ pip install mysqlclient
```

That last step, I believe, is what required the `python39-devel` and `mariadb-connector-c-devel` packages above, since it probably does some actual compilation.

I decided to name my Django project "hex", and created it like so:

```
(virtenv) $ pwd
/var/www/django
(virtenv) $ django-admin startproject hex
```

At that point, inside `/var/www/django` I had a `virtenv` directory containing a Python virtual environment, and a `hex` directory containing the Django project.

9.4 Django Configuration / settings.py

Here are the relevant values in `settings.py` which I'd changed/modified (I'd also updated `TIME_ZONE`, `DEBUG`, etc, but that's irrelevant):

```
ALLOWED_HOSTS = ['servername']

MIDDLEWARE = [
    ...
    # This line must be *underneath* AuthenticationMiddleware
    'django.contrib.auth.middleware.RemoteUserMiddleware',
    ...
]

AUTHENTICATION_BACKENDS = [
    'django.contrib.auth.backends.RemoteUserBackend',
]
```

(continues on next page)

(continued from previous page)

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'hex',
        'USER': 'hex',
        'PASSWORD': 'password',
        'HOST': '127.0.0.1',
        'PORT': '3306',
        'OPTIONS': {
            'init_command': "SET sql_mode='STRICT_TRANS_TABLES'",
        }
    }
}

STATIC_URL = '/hex/static/'
STATIC_ROOT = '/var/www/django/hex/static'
```

ALLOWED_HOSTS I believe I had to set this, rather than leave it blank, to get Django to respond properly via Apache, though I don't actually recall.

MIDDLEWARE Adding in the `RemoteUserMiddleware` line is necessary for me to make use of Apache's already-configured authentication mechanisms. As noted above, it must be underneath the `AuthenticationMiddleware` line which is already present.

AUTHENTICATION_BACKENDS This is the second component of using Apache's already-configured auth mechanisms.

DATABASES Simple MySQL configuration. The `OPTIONS` line lets you avoid some warnings which will otherwise pop up while using MySQL in Django.

STATIC_URL and STATIC_ROOT Static file configuration for Django.

You could, also, set `SECURE_CROSS_ORIGIN_OPENER_POLICY = 'same-origin-allow-popups'` in here, to ensure that the jPlayer streaming popup works properly, but I prefer to make sure that my static file delivery sets its headers properly instead.

Once these have been set up, and the necessary database created in MySQL, Django's basic database models can be created, and we can make sure that Django recognizes an administrative user. Apache is handling authentication in my case, but I still needed to tell Django that "my" user was an administrator:

```
(virtenv) $ cd /var/www/django/hex
(virtenv) $ python manage.py migrate
(virtenv) $ python manage.py createsuperuser
```

Any password given to `createsuperuser` won't actually be used in my case, since `RemoteUserBackend` just accepts the information given to it by Apache about authentication.

At this point, Django functionality can be tested with their test server:

```
(virtenv) $ python manage.py runserver 0.0.0.0:8080
```

9.5 selinux

Shared objects inside Django's virtual env need to be of type `httpd_sys_script_exec_t` in order to be executed via WSGI. If you don't set that properly, you'll end up getting some reasonably crazy errors in your logs.

Setting this is pretty easy. I decided to just set that context for the entire `lib/python3.9` dir, rather than trying to cherry pick:

```
# semanage fcontext -a -t httpd_sys_script_exec_t '/var/www/django/virtenv/lib/
↪python[0-9\.]+(/.*)?'
# restorecon -rv /var/www/django/virtenv/lib
```

9.6 WSGI Configuration in Apache

Next up was configuring WSGI/Django inside Apache, so it's accessible via my existing SSL vhost. The full config section that I used in the relevant virtual host, including Django static file configuration, was:

```
WSGIDaemonProcess servername socket-timeout=480 processes=1 threads=15 display-
↪name=django python-path=/var/www/django/hex:/var/www/django/virtenv/lib/python3.9/
↪site-packages lang='en_US.UTF-8' locale='en_US.UTF-8'
WSGIProcessGroup servername
WSGIScriptAlias /hex /var/www/django/hex/hex/wsgi.py

Alias /music /var/audio
<Location /music>
    Require all granted
    Options -Indexes
</Location>

Alias /hex/static /var/www/django/hex/static
<Location /hex/static>
    Require all granted
    Header set Cross-Origin-Opener-Policy same-origin
</Location>
```

A few notes on some of those options:

socket-timeout This is actually just a holdover from before I started using `HttpStreamingResponse` for the library add/update functions, which was causing those pages to take a long time to respond. Leaving it out of the line should be fine since Exordium is pretty responsive now.

processes I'd originally had this set to `2`, but as mentioned elsewhere in these docs, if you set `processes` to a value greater than `1`, changing Exordium's preferences (library paths, zipfile paths, etc) will only change the preference effectively in the process it was actually set on, which can lead to inconsistency. I'd like to figure that out eventually, but for now I've been happy enough with `1`.

threads Number of threads to use. Not sure where I got `15` from.

python-path These are important for ensuring that WSGI is using our virtenv properly.

lang and locale By default, WSGI will operate using a `$LANG` value of `C`, which causes problems for Exordium if it encounters music files with non-ASCII characters in their filenames. See *Apache/WSGI Deployment Issues* for a bit more information, but regardless: just set these to appropriate values for your system.

COOP Header The `Header` line in the static file delivery stanza is what I use to ensure that the jPlayer streaming popup works properly. You'll either have to do something like this (or even set the header more globally on your site), or edit `settings.py` to use a different Django default COOP header (as described above). Note that despite the Apache documentation implying that `<Location>` isn't a valid place to put the `Header` directive, it seems to work just fine for me.

9.7 Cross Origin Opener Policy Headers

One further note about the COOP headers: if your static content isn't served from the same protocol/hostname/port as Django itself, you will likely have to set either Django or your static files' value to `unsafe-none`, instead. I'm not sure which exactly would be required, in that case.

9.8 Apache Configuration: mp3/zipfile access

Exordium requires that the files in the music library be accessible directly via a webserver, which I had configured already on a non-SSL Apache vhost. It also needs a URL for zipfile downloads, if you want album zipfile downloads. A vhost similar to the following would do the trick:

```
<VirtualHost servername:80>
    ServerName servername
    # other common Apache config directives here

    Alias /music /var/audio
    <Directory /var/audio>
        Require all granted
        Options -Indexes
    </Directory>

    Alias /zipfiles /var/www/django/zipfiles
    <Directory /var/www/django/zipfiles>
        Require all granted
        Options -Indexes
    </Directory>

</VirtualHost>
```

With that configuration, you'd end up setting the following in Django's settings:

- **Exordium Library Base Path:** `/var/audio`
- **Exordium Media URL (for HTML5):** `https://servername/music`
- **Exordium Media URL (for m3u):** `http://servername/music`
- **Exordium Zip File Generation Path:** `/var/www/django/zipfiles`
- **Exordium Zip File Retrieval URL:** `http://servername/zipfiles`

9.9 Other Minor Tweaks

At this point, after an `apachectl graceful` Django itself should be working properly inside the SSL vhost. Other apps (such as Exordium itself) can be installed with the virtenv active with simple `pip install django-exordium` commands, and following the other instructions from *Installation*.

One more thing I've done which required some Googling to figure out is that I wanted Django's base project URL to redirect to Exordium, since Exordium is currently my only Django app. My project's `urls.py` looks like this, now, to support that:

```
from django.contrib import admin
from django.urls import path, re_path, include
from django.views.generic.base import RedirectView

urlpatterns = [
    re_path(r'^$', RedirectView.as_view(pattern_name='exordium:index')),
    path('admin/', admin.site.urls),
    path('exordium/', include('exordium.urls')),
]
```

CHAPTER 10

Changelog

10.1 1.4.3 (2023-03-21)

Bugfixes/Tweaks

- Added an extra normalization char (ō / Ō)

10.2 1.4.2 (2022-03-15)

Bugfixes/Tweaks

- Fixed some more edge cases with file updates (specifically cases where all files in an album get updated, but a subset are also renamed – only the renamed files were actually getting updated in the database properly).
- Expanded the internal "normalization" character set to handle some more cases, namely super-and-subscript numerals.

10.3 1.4.1 (2022-03-04)

Bugfixes/Tweaks

- Added some custom config for readthedocs.org so that our RTD docs get generated on that end again.

10.4 1.4.0 (2022-03-03)

Configuration Updates Needed

- To continue using jPlayer HTML5 streaming properly on prod installations where static content is served via an external webserver, some configuration related to Cross Origin Opener Policy needs to be tweaked, either in your Django site's `settings.py`, or in the webserver config. So, one of the following must be done:

  - Set `SECURE_CROSS_ORIGIN_OPENER_POLICY = 'same-origin-allow-popups'` in `settings.py`, so that the jPlayer popup doesn't need to supply its own COOP headers. Or:
  - Configure your webserver to add `Cross-Origin-Opener-Policy: same-origin` to static file delivery.
  - If your static content isn't served from the same protocol/hostname/port as Django itself, you will likely have to set either Django or your static files' value to `unsafe-none` instead.
- The "Exordium Media URL" preference has been split into two separate preferences: one for HTML5 streaming, and one for m3u playlists and direct track downloads.
  - If migrations are run prior to accessing the updated site (which is always a good idea anyway), the previous "Media URL" value will be copied over to both new preferences, which should allow the install to continue to work without problems.
  - If Exordium is visited *prior* to running migrations, the two new prefs will instead inherit their defaults, as if Django was installed for the first time, so they would have to be manually set via the web UI in that case.

Bugfixes/Tweaks

- Updated for Django 4.x compat, along with more recent versions of our other dependencies.
- Split unit tests into multiple files
- Added HTML <label> entries for radio/checkbox text on Library Upkeep page, so the labels themselves are clickable instead of just the buttons.
- Link directly to Exordium preferences in Global Preferences, rather than just to the main Global Preferences screen.
- Fixed some edge cases with library updates
- Prevent jPlayer popup from opening too many popups when Cross Origin Opener Policy problems are encountered.
- Added License text to the generated docs. Not sure why I never included those originally!

10.5 1.3.4 (2021-03-25)

Bugfixes/Tweaks

- Expanded the internal "normalization" character set to handle a bunch more Greek characters specifically, and probably a few others as well. This is used for normalizing filenames in zip downloads, and normalizing search strings.

10.6 1.3.3 (2020-05-28)

New Features

- Added support for Ogg Opus files (`.opus`)

10.7 1.3.2 (2018-09-20)

Bugfixes/Tweaks

- Stupid little formatting fix in README and docs requirements RST

10.8 1.3.1 (2018-09-20)

Bugfixes/Tweaks

- Use a label for our "live album" checkbox so the text can be clicked in addition to the checkbox itself
- Disallow django-tables2 >= 2.0 until an issue with that has been either fixed or denied: https://github.com/jieter/django-tables2/issues/621 (we can work around, if they opt not to merge that fix, but I'd prefer to use the fix on that Issue)

10.9 1.3.0 (2018-01-02)

Bugfixes/Tweaks

- Updates to work properly with Django 2.0
- Use time-added as secondary sorting for albums when sorting by Year

10.10 1.2.1 (2017-11-28)

Bugfixes/Tweaks

- Updates to documentation, to have `django-dynamic-preferences` properly configured.

10.11 1.2.0 (2017-11-28)

Bugfixes/Tweaks

- Updates to work properly with Django 1.11 and `django-dynamic-preferences` >= 1.0
- Fixed live recording checkbox when not logged in to Django

10.12 1.1.1 (2016-12-30)

Bugfixes/Tweaks

- Fixed the release date in the Changelog. Bah.

10.13 1.1.0 (2016-12-30)

New Features

- Added support for M4A audio files

Bugfixes/Tweaks

- Added a few more "normalization" characters, for easy searching from the web UI and correct association across possibly- inconsistent tags. Specifically: İ, ğ, and ş. Also fixed normalizing filenames (for zipfile downloads) for capital Ç.

- Fixed album summary information when some tracks have classical music tags (ensemble, composer, conductor) but other tracks don't. (Explicitly say that not all tracks have the tags.)
- Change the display order of a few elements on the album download page, and use an HTML `<meta>` tag to automatically queue up the download, rather than only having the direct link.
- Override table footers to always include item counts, as was present in `django-tables2` 1.2.6 but patched out in 1.2.7.
- Use newlines when reporting multiple artists in tables, to keep the table width down as much as possible.

10.14 1.0.3 (2016-11-22)

Bugfixes/Tweaks

- Fixed admin area to allow blank album art, song, and artist fields, where the fields shouldn't be required

10.15 1.0.2 (2016-10-21)

Bugfixes/Tweaks

- Fixed packaging manifest to include changelog, and exclude rendered HTML documentation (the latter was causing the source archive to be twice as large as it should be)

10.16 1.0.1 (2016-10-21)

Bugfixes/Tweaks

- Added a "login" link in the sidebar for not-logged-in users
- Fixes for tests which were failing when run against databases other than MySQL/MariaDB. Actual app functionality appears to be fine, just a problem with the test suite.
  - Case-related tests
  - Album Art tests
- Tweaked/reworked some documentation
- Set `setup.py` development classifier to Production
- Reordered a few fields on the admin screens

10.17 1.0.0 (2016-10-18)

- Initial Release

CHAPTER 11

TODO

Things I want to get done before inevitably letting the project bitrot:

- When we added Opus support, we did various checks to make sure that we don't try to stream Opus files w/ the HTML5 jPlayer, since that doesn't support Opus. We should probably have an "is_streamable" method on the Song class, though, rather than just checking for Opus all over the place. Something to think about...
- I think I'd like to include "slug"-like text in album URLs. Would put 'em after the ID, and they'd be optional (and completely ignored by the album view). Would make sharing links a bit friendlier, though. Make sure to include a maximum length, if I do it...
- Direct links to pages with django-tables2 (instead of just next/prev) – the v2.4.1 that I'm using now while testing el8 hosting has a nice solution for that built-in. Will probably keep my own *X of Y* modifications in place, but pull those in.

Things which may or may not make that cut:

- Might be nice to actually spend some time to make it look better. The sidebar's pretty ugly, etc.
- Should implement a max-number-of-results in search. Probably only really an issue on Songs, maybe just use 500 like we do on the artist view page.
- Might be nice to have a mobile-optimized CSS
- Ability to specify subdirectories when doing add/update actions, to only process a subset of the library.
- http://reinout.vanrees.org/weblog/2014/05/19/context.html Basically, overriding get_context_data() in our views is wordy and not needed most of the time. Use 'view.varname' in the template and it'll pull from the named var/func. No fussy context fiddling required!
- Use annotate() for some of our aggregate information on artist browse, for instance.
- Convert our three forms to actual django Forms.
- Might be nice to include an admin subcommand to clean the zip directory, which could be plugged into a cron rather than having to do your own.

Things which, if I'm being honest here, are likely to never actually happen:

- Exordium can be slow, and it'd be nice to figure out if there are reasonable ways to speed it up, though it appears that in my case I'm primarily I/O bound. The initial import of a 42k-track library on my box takes about an hour, the majority of which is spent doing checksums of the files. I've tested out using faster, less-secure checksum method (sha1sum, md5sum) and while those technically save us some time in CPU, it turns out that on my machine at least I'm primarily I/O bound while doing such large imports, and using the faster methods don't actually save any time. Beyond the initial checksumming in the add process, there's still a good 15 minutes after that for database imports. Relatedly, I'd put in some logic so that if an artist has more than 500 tracks, the track list isn't shown on the artist page, because performance was lousy there, too. It'd be nice to see if there's a good way to get that faster as well. The good news is that outside of cases like that, it does seem to perform fine. Adds/ updates are speedy enough for me, and in general the app is pretty responsive.
- More precise summary text for classical * "Composer, tracks 1-5, 8: foo", etc.
- Extra details on album song list (format, bitrate, etc) (something else for user prefs. Will have to wait until I figure out a *reasonable* way to have dynamic column definitions in django-tables2. I don't want to define a different class for each possibility.)
- Might be nice to support some artist groupings. ("Amanda Palmer and the Grand Theft Orchestra" albums should probably show up alongside "Amanda Palmer" albums, etc. I assume this would be something manual-only from the admin area. The "correct" thing to do might be to find a tag to use, instead, but I'm guessing that'd overly-complicate the backend, which I think I'd like to keep cleaner. But it might be a monster anyway.)
- "Random albums" page?
- If deployed with WSGI configured for multiple processes, both user+global preferences only get applied properly on the process for which they were set. Subsequent page loads may or may not come from the same process, so results can be inconsistent. I'm not sure if there's a way around that or not - for now I've just dropped my number of processes down to 1.
- Maybe make "Various" a non-reserved artist name? We could put in a big ol' random mess of characters for the name, and just rely on Artist.various to display it properly on the UI and all that. That way if we ever have a band whose name really is "Various" we should be able to support it. Would have to figure out some UI differences to differentiate, if so. Maybe italics or something for the special artist?
- Theme support might be nice, and should be pretty trivial, though I doubt I'll take the time to figure it out.
- Could maybe apply some sorting while adding/updating so that updates appear in some kind of order rather than effectively randomly (would be especially nice for the initial bulk add(), so you'd have at least SOME indication of how much more time will be needed)
- Management subcommands to add/update the library (actually, this is more complicated than I'd considered initially, because the user running the add/update won't be the same as the user Django is running as, so permission issues come into play.)
- Relatedly, the add/update stuff DOES sort of belong in something like celery instead, but I don't actually have any interest in implementing that on my host at the moment.
- Perhaps we should read genre from tags?

CHAPTER 12

Licenses

12.1 Exordium

```
Exordium is available under the New/Modified BSD License (below)

Exordium also packages jPlayer (http://jplayer.org), which is available under
the MIT license (see LICENSE-jPlayer.txt)

Exordium also packages jQuery (https://jquery.com/), which is available
under the MIT license (see LICENSE-jQuery.txt)

------------------------------------------------------------------------------
Exordium License:
------------------------------------------------------------------------------

Copyright (c) 2016-2023, Christopher J. Kucera
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:
    * Redistributions of source code must retain the above copyright
      notice, this list of conditions and the following disclaimer.
    * Redistributions in binary form must reproduce the above copyright
      notice, this list of conditions and the following disclaimer in the
      documentation and/or other materials provided with the distribution.
    * Neither the name of the Exordium team nor the names of its contributors
      may be used to endorse or promote products derived from this software
      without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL CJ KUCERA BE LIABLE FOR ANY
DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
```

(continues on next page)

(continued from previous page)

```
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

12.2 jPlayer

```
Copyright (c) 2009 - 2013 Happyworm Ltd
http://happyworm.com
http://jplayer.org

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in
all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
THE SOFTWARE.
```

12.3 jQuery

```
/*!
 * jQuery JavaScript Library v3.1.1
 * https://jquery.com/
 *
 * Includes Sizzle.js
 * https://sizzlejs.com/
 *
 * Copyright jQuery Foundation and other contributors
 * Released under the MIT license
 * https://jquery.org/license
 *
 * Date: 2016-09-22T22:30Z
 */
```

CHAPTER 13

Exordium Music Library

13.1 Introduction

Exordium is a read-only web-based music library system for Django. Exordium will read mp3, ogg vorbis, ogg opus, and m4a files from the host filesystem and provide an online interface to browse, download (as zipfiles or otherwise), and stream.

The HTML5 media player jPlayer is used to provide arbitrary streaming of music.

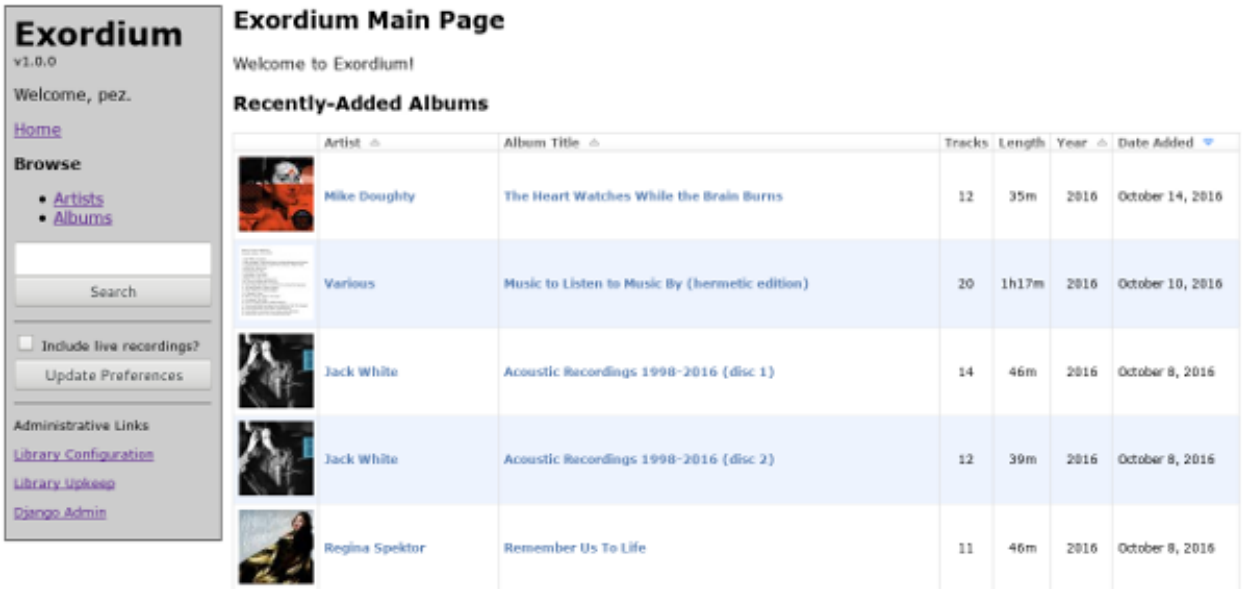

Fig. 1: Exordium Main Screen

Exordium was built with a very specific set of operational goals and does not attempt to be a generic library suitable for widespread use. There are, in fact, no configuration options beyond those to define the file paths/URLs necessary for basic usage. Patches to add/change functionality will be happily received so long as they don't interfere with or

disable the current functionality by default, but there is no internal development goal to make Exordium a generic solution.

13.2 Download

Exordium is available to install on PyPI via `pip install django-exordium`. PyPI also hosts Python packages for Exordium in both source and Wheel formats, at https://pypi.python.org/pypi/django-exordium/. Source and Wheel downloads of all released versions can also be found at Exordium's hompeage at https://apocalyptech.com/exordium/.

Exordium sourcecode is hosted at GitHub, and sourcecode archives of released versions can be found there at https://github.com/apocalyptech/exordium/releases

Documentation is included in the project's `docs/` directory, but is also uploaded to:

- https://apocalyptech.com/exordium/
- https://exordium.readthedocs.io/

13.3 Detailed Documentation

Assumptions and Limitations provides information about how Exordium does things, and would be a good place to determine if Exordium is a good operational fit for the kind of web library app you're looking for.

Screenshots contains screenshots of all of Exordium's main pages, and is probably the best place to look to get a feel for how Exordium operates from a user perspective.

See *Requirements* for Exordium's requirements, *Installation* for installation instructions onto an existing Django project, and *Administration* for information on administration and library upkeep.

If deploying via Apache, *Apache/WSGI Deployment Issues* contains some information that might be useful. *Rocky/Alma/RHEL/OEL/Centos-Stream 8 Apache/WSGI Deployment HOWTO* is a complete guide to how Django is deployed for my own use.

Changelog and *TODO* contain some information about version history and future plans.

Licenses contains information about Exordium's license (3-Clause BSD) and of a couple of its components (MIT).

13.4 Other Information

The name "Exordium" comes from the fictional technology of the same name in Alastair Reynolds' "Revelation Space" novels. It's not a perfect name for the app, given that the Revelation Space *Exordium* would make a pretty lousy music library, but at least there's some element of data storage and retrieval. Exordium the *web-based music library*, as opposed to its fictional counterpart, is only capable of retrieving music which has been imported to it in the past. I'll be sure to contact all the major news organizations if I figure out a way to get it to retrieve music stored in the future.