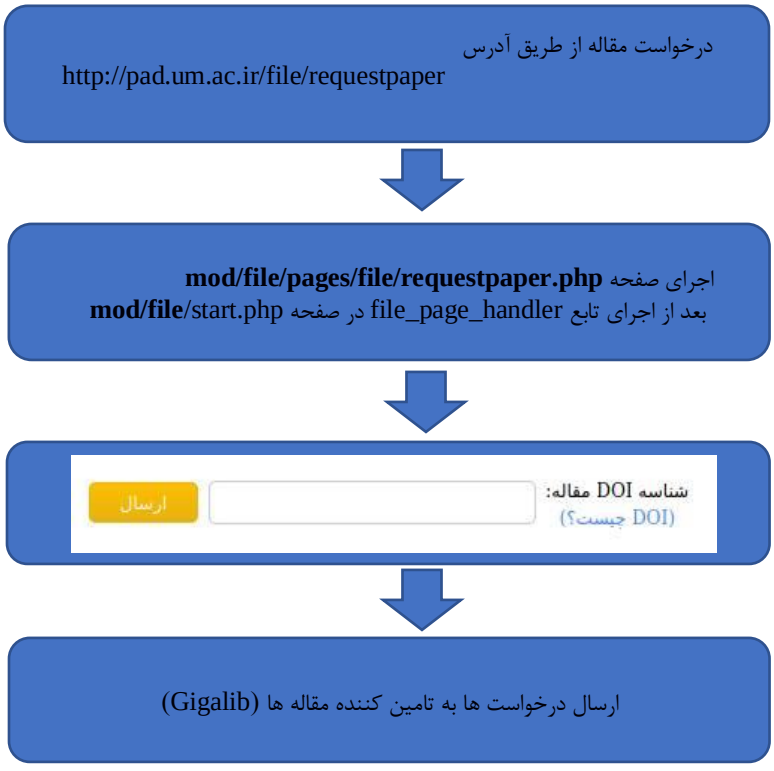
به نام خدا

درخواست و تامین مقاله در سیستم پاد

برای درخواست مقاله مشابه تصویر زیر به صفحه درخواست مقاله مراجعه می شود. سپس بعد از ورود Doi و زدن دکمه ارسال درخواست تامین مقاله داده می شود.

مراحل تامین مقاله به صورت زیر است:



ارسال درخواست ها به تامین کننده مقاله ها (GigaLib):

برای این کار، سیستم پاد باید یک URL که توسط GigaLib به ما معرفی شده است را فراخوانی کند و شناسه DOI رابرای آن ارسال کند. نمونه URL به صورت زیر است:

http://ws.gigalib.org/dl-article.aspx?doi=10.1038/nrc3419

۱. doi مقدار شناسه DOI مورد درخواست می باشد.

سپس سیستم GigaLib در پاسخ به این فراخوانی، پاسخی می دهد که یک رشته با قالب زیر است:

REQID:ID#PUBLISHER

۱. ID یک شناسه منحصربفرد (50 کاراکتری شامل حروف الفبا، ارقام انگلیسی و خط تیره) برای این درخواست می باشد. این شناسه یک رشته است مثلا:

f2383782-ffbd-4ecd-9df8-e0b88e91da83

۲. PUBLISHER نام پایگاهی است که مقاله درخواستی متعلق به آن است.

سیستم پاد شناسه ID را ذخیره می کند تا زمانی که پاسخی از سمت سرور GigaLib دریافت کرد بتواند تشخیص دهد که این پاسخ مربوط به کدام درخواستش است. بخش های اصلی کد PHP (مربوط به ثبت درخواست در پایگاه داده) به صورت زیر است:

توجه: فراخوانی URL مذکور تنها از IP سرور پاد در دانشگاه فردوسی امکان‌پذیر است و مجوز دسترسی تنها برای همین آدرس تعریف می شود.

```
onclick: ajax_func:send_request
calee_func: do_send_request
calee_ajax_page: register_request_from_paper_request_form
calee_func: process_request_by_doi
{
   calee_fun: is_doi_for_paper && is_valid_doi
                                        بررسی می کنند که doi درست باشد و خروجی داشته باشد

   calee_fun: get_file_by_doi  &&  is_duplicate_paper_request
   {
                                        بررسی میکند آیا این مقاله قبلا درخواست داده شده است
     calee_func: register_paper_request()
     {
        INSERT into paper_download_requests status=-2 ….
     }
   }
}
```

بخش های مهمی از سیستم پاد با زبان جاوا پیاده سازی شده است. کد جاوا درخواست های ثبت شده در پایگاه داده را استخراج می کند و به GigaLib ارسال می کند. بخش های اصلی کد مربوطه به صورت زیر است:

```
packgae: paperRequest
file: PaperRequestProccess.php
func: run
{
  Based on type:
     calee_fun: processRequestsWaitingForUpload();
     calee_fun:  processRequestsWaitingForDownloadFromJDB();
     calee_fun: handleRequestsFailedOrWaitingTooMuchForResponseFromGigalib();
     calee_fun: RequestsWaitingForUpload();
     calee_fun: processRequestsWaitingToBeSentToTargetServerprocess()
                {
                  calee_fun: DoSendRequestToServer(){

                        <entry key="GIGALIB_URL" value="http://ws.gigalib.org/dl-article.aspx"/>

                   }
                   calee_fun: UpdateAsSentToTargetServer(){

                         Update  paper_download_requests set status=0 …

                   }
                }
}
```

دریافت پاسخ از GigaLib:

هر گاه GigaLib پردازش درخواست مورد نظر را به اتمام رساند (موفق یا ناموفق). با فراخوانی یک URL از سیستم پاد (که ما به ایشان معرفی کرده ایم)، نتیجه درخواست را به آن اطلاع می دهد. اگر درخواست با شکست مواجه شد URL ای که GigaLib فراخوانی می کند دارای این قالب است:

http://pad.um.ac.ir/file/notifyDownload4?id=ID&code=FAILURE&link

که مقدار FAILURE مشخص می کند که این پاسخ، پاسخ قطعی و بیان کننده ناموفق بودن تأمین مقاله است.

تا زمانی که پاد، پاسخی از GigaLib، مبنی بر موفقیت یا ناموفق بودن درخواست دریافت نکند، درخواست در سمت پاد در حالت انتظار باقی می ماند. اگر نتیجه پردازش موفق باشد، URL ای که GigaLib فراخوانی میکند دارای این قالب است:

http://pad.um.ac.ir/file/notifyDownload4?id=ID&code=SUCCESS&link=pdf_download_link&doi=DOI

1. ID همان شناسه درخواست است که در زمان ثبت درخواست ایجاد شده است.
2. مقدار پارامتر code نشان دهنده موفقیت یا عدم موفقیت درخواست است.
3. مقدار پارامتر link هم برابر آدرسی دانلود فایل (از روی سیستم GigaLib) می باشد.

توجه: این آدرس تنها توسط سرور گیگالیب قابل دسترس است

دانلود فایل با استفاده از لینک دریافتی از GigaLib:

بعد از تامین لینک فایل مقاله ها توسط GigaLib، کد جاوا (paperrequest/PaperRequestProcessor.java) به خواندن درخواست ها با وضعیت ۰ (یعنی آماده دانلود یا DOWNLOADED_BUT_WAITING_TO_UPLOAD) می پردازد.

```
Func: processRequestsWaitingForDownloadFromJDB(){

    calee_fun: getPaperRequestForDownloadFromTargetServer();
    calee_fun:  downloadRequestFromJDBOrURL(request)
    {
     String link = request.getJdbDownloadLink();
     String filepath = PaperRequestProcessor.TEMP_PATH + File.separatorChar + seed + ".pdf";
     URL url = new URL(link);
     WebClient webClient2 = new WebClient(BrowserVersion.FIREFOX_24);
        webClient2.getOptions().setJavaScriptEnabled(false);
        webClient2.getOptions().setThrowExceptionOnFailingStatusCode(false);
        webClient2.getOptions().setThrowExceptionOnScriptError(false);
        webClient2.getOptions().setPrintContentOnFailingStatusCode(false);
        webClient2.getOptions().setCssEnabled(false);
        Page page = webClient2.getPage(url);
        Util.writeToFile(page.getWebResponse().getContentAsStream(), file.getAbsolutePath());
        if (file != null && file.exists() && file.isFile() && file.canRead())
        {
           return true;
        }
    }
    calee_fun: updateAsDownloaded()
            {
            //  UPDATE paper_download_requests SET status = -3 ,  download_time = ? WHERE id = ?
            }
```

آپلود فایل دانلود شده

بعد از دانلود فایل مقاله، کد جاوا (paperrequest/PaperRequestProcessor.java) به خواندن درخواست ها با وضعیت -۳ (یعنی آماده آپلود یا DOWNLOADED_BUT_WAITING_TO_UPLOAD) می پردازد. فایل های دانلود شده در پوشه شخصی فرد درخواست دهنده آپلود می شود و وضعیت مقاله به -۵ تبدیل می شود.

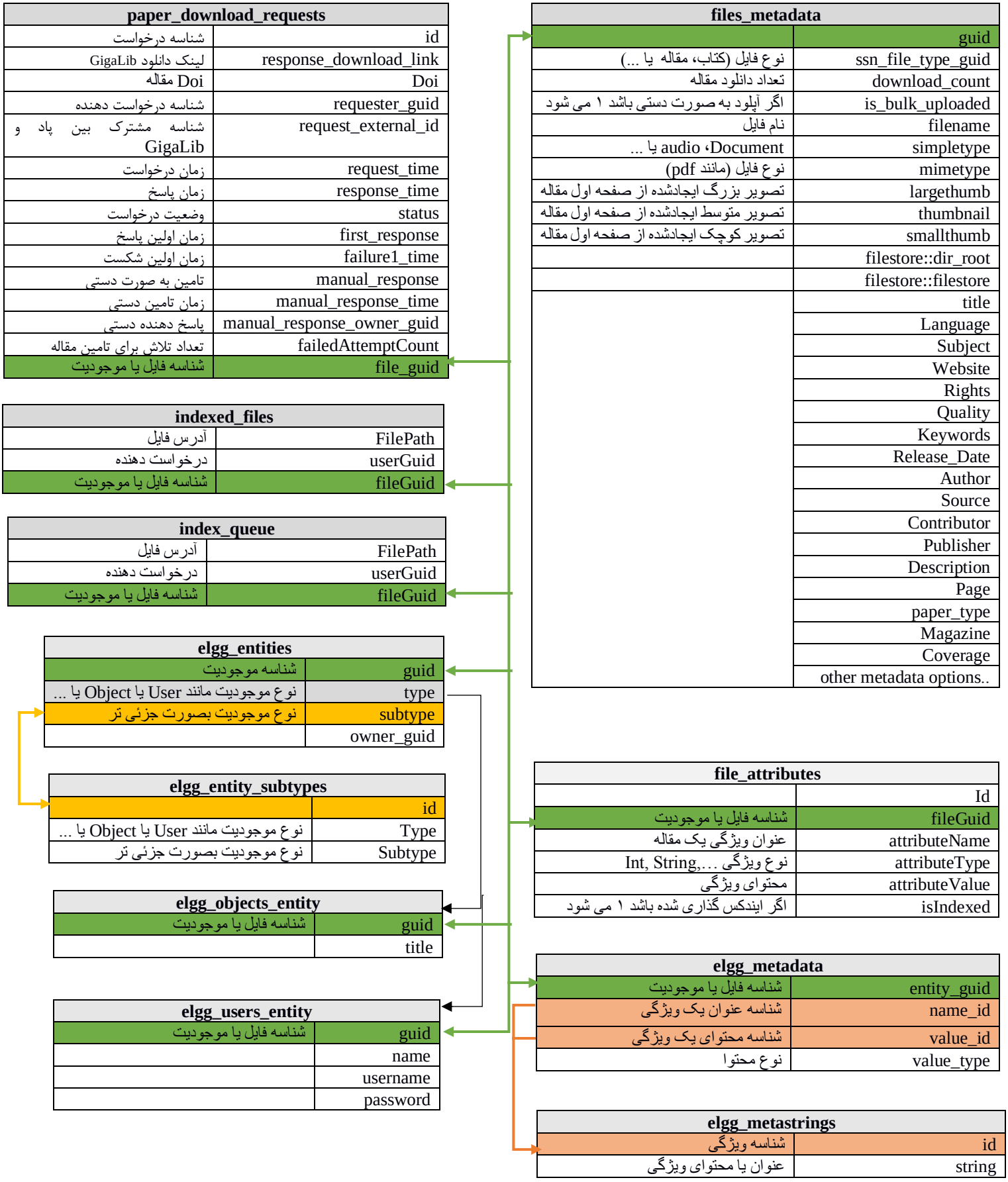
```
processRequestsWaitingForUpload()
{
  calee_fun: getPaperRequestWaitingForUpload()
  calee_fun: requestedFileExists(request);
  calee_fun: uploadRequestedFile(request)
  {
      String filepath = TEMP_PATH + File.separatorChar + request.getSeed() + ".pdf";
      File file = new File(filepath);
      calee_fun: File movedFile = moveDownloadedFile(file, request, start);
      calee_fun:  long fileGuid = uploadFileToPAD(request, movedFile); //set metadata
         if (fileGuid != -1) {
            calee_fun: updateAsFinishedSuccessfully(request, fileGuid)
            {
             //UPDATE paper_download_requests SET  status = ? response_time = ? WHERE id = ?
            }
            calee_fun: sendPaperRequestNotification(request, fileGuid)
            {
            //INSERT INTO pad_notifications(owner_guid, issuer_guid, …
            }
            calee_fun: file.delete();
         }
  }
}
```

به طور کلی وضعیت یک درخواست مقاله در مراحل مختلف به صورت زیر است:

وضعیت مقاله	کد وضعیت مقاله
درخواست منتظر جهت ارسال به تامین کننده	-۲
درخواست منتظر جهت گرفتن پاسخ از تامین کننده	۰
درخواست منتظر جهت دانلود فایل بر اساس لینک دریافتی از تامین کننده	-۵
درخواست منتظر جهت آپلود فایل دانلود شده در سرور پاد	-۳
درخواست هایی که از سمت تامین کننده پاسخ داده نشده اند	-۱
درخواست هایی که از سمت گیرنده به درستی پاسخ داده شده اند و دانلود و آپلود شده است	۱
معنای هر کد در file/language/fa.php وجود دارد	

تامین متادیتای مقاله:

برای هر فایل، متادیتای فایل گرفته می شود و در پایگاه داده ذخیره می شود. برای گرفتن متادیتا از سایت‌هایی مانند http://api.crossref.org/v1/works/dx.doi.org/ استفاده می شود. Doi به این آدرس ارسال می‌شود و خروجی آن تحلیل می‌شود و اطلاعات لازم مانند عنوان و نویسندگان مقاله استخراج می شود و در پایگاه داده ذخیره می شود. کد مربوط به متادیتا در تابع uploadFileToPAD پیاده سازی شده است. متادیتا از چند سایت گرفته می شود که پیاده سازی آن ها در CrossRefAnalyzer2.java و CrossRefAnalyzer3.java و CrossRefAnalyzer4.java در src/ssn/paperrequest/ قرار گرفته است. ذخیره بسیاری از اطلاعات از جمله متادیتای مقالات در پایگاه داده داری ساختاری است که در ادامه توضیح داده شده است.


paper_download_requests
id
شناسه درخواست
response_download_link
لینک دانلود GigaLib
Doi
Doi مقاله
requester_guid
شناسه درخواست دهنده
request_external_id
شناسه مشترک بین پاد و GigaLib
request_time
زمان درخواست
response_time
زمان پاسخ
status
وضعیت درخواست
first_response
زمان اولین پاسخ
failure1_time
زمان اولین شکست
manual_response
تامین به صورت دستی
manual_response_time
زمان تامین دستی
manual_response_owner_guid
پاسخ دهنده دستی
failedAttemptCount
تعداد تلاش برای تامین مقاله
file_guid
شناسه فایل یا موجودیت
files_metadata
guid
ssn_file_type_guid
نوع فایل (کتاب، مقاله یا ...)
download_count
تعداد دانلود مقاله
is_bulk_uploaded
اگر آپلود به صورت دستی باشد ۱ می شود
filename
نام فایل
simpletype
Document، audio یا ...
mimetype
نوع فایل (مانند pdf)
largethumb
تصویر بزرگ ایجادشده از صفحه اول مقاله
thumbnail
تصویر متوسط ایجادشده از صفحه اول مقاله
smallthumb
تصویر کوچک ایجادشده از صفحه اول مقاله
filestore::dir_root
filestore::filestore
title
Language
Subject
Website
Rights
Quality
Keywords
Release_Date
Author
Source
Contributor
Publisher
Description
Page
paper_type
Magazine
Coverage
other metadata options..
indexed_files
FilePath
آدرس فایل
userGuid
درخواست دهنده
fileGuid
شناسه فایل یا موجودیت
index_queue
FilePath
آدرس فایل
userGuid
درخواست دهنده
fileGuid
شناسه فایل یا موجودیت
elgg_entities
guid
شناسه موجودیت
type
نوع موجودیت مانند User یا Object یا ...
subtype
نوع موجودیت بصورت جزئی تر
owner_guid
elgg_entity_subtypes
id
Type
نوع موجودیت مانند User یا Object یا ...
Subtype
نوع موجودیت بصورت جزئی تر
elgg_objects_entity
guid
شناسه فایل یا موجودیت
title
elgg_users_entity
guid
شناسه فایل یا موجودیت
name
username
password
file_attributes
Id
fileGuid
شناسه فایل یا موجودیت
attributeName
عنوان ویژگی یک مقاله
attributeType
نوع ویژگی Int, String,...
attributeValue
محتوای ویژگی
isIndexed
اگر ایندکس گذاری شده باشد ۱ می شود
elgg_metadata
entity_guid
شناسه فایل یا موجودیت
name_id
شناسه عنوان یک ویژگی
value_id
شناسه محتوای یک ویژگی
value_type
نوع محتوا
elgg_metastrings
id
شناسه ویژگی
string
عنوان یا محتوای ویژگی

paper_download_requests:

به ازای هر درخواست مقاله یک رکورد در این جدول اضافه می شود. اطلاعاتی مانند Doi مقاله، لینک دانلود دریافتی از GigaLib و شناسه مربوط به مقاله در جدول متادیتاهای مقاله ها ذخیره می شود.

elgg_entities:

به ازای هر موجودیتی در پایگاه پاد، یک رکورد در این جدول ذخیره می شود. برای مثال به ازای هر درخواست مقاله یک رکورد در این جدول اضافه می شود که شناسه این جدول در جداول مختلف مانند elgg_users_entity، elgg_objects_entity و files_metadata مورد ارجاع قرار میگیرد. این جدول شامل فیلدی به نام typeمی باشد که محتوای آن مشخص میکند موجودیت مربوطه جزء مقاله ها، کاربران یا موارد دیگر می باشد. برای مثال اگر type برابر user باشد مشخص می شود اطلاعات مرتبط با کاربر است و اطلاعات بیشتر در جدول elgg_users_entity می باشد. همچنین فیلدی با عنوان subtype دارد. هر subtype به طور جزئی تر مشخص میکند که موجودیت مربوط به چه چیزی است. برای مثال اگر type برابر object باشد و subtype آن برابر ۱ باشد مشخص می شود که رکورد مذکور مربوط به یک مقاله است. جزئیات بیشتر در مورد subtypeها در جدول elgg_entity_subtypes قرار گرفته است.

files_metadata:

متادیتای اصلی مربوط به هر مقاله در این جدول قرار میگیرد. شناسه این جدول همان شناسه جدول elgg_entities می باشد. از طریق این شناسه بین جداول مختلف مرتبط با درخواست مقاله، ارتباط برقرار می شود.

index_queue و indexed_files:

بعد از تامین هر مقاله، یک رکورد در جدول index_queue ذخیره می شود. به عبارتی مقاله در صف ایندکس شدن قرار میگیرد. کد جاوا مقاله های ثبت شده در صف index را میخواند و جهت جستجوی بهتر index می کند. هر مقاله ای ک index شد به لیست مقاله های index شده در جدول indexed_files اضافه می شود. در این جداول شناسه متادیتای مقاله (از جدول files_metadata) و آدرس فایل مقاله ذخیره می شود.

elgg_metadata:

متادیتای مربوط به مقاله ها که در جدول files_metadata قرار نگرفته است در این جدول قرار می گیرد. در این جدول مشخص می شود برای هر مقاله، چه ویژگی ای با چه محتوایی وجود دارد. برای مثال مشخص می شود

برای یک مقاله خاص، score برابر 229 است. Score و ۲۲۹ در جدول elgg_metastrings ذخیره می شود و شناسه مربوط به آن ها در جدول elgg_metadata ذخیره می شود.

file_attributes:

در این جدول ویژگی های یک مقاله (attributes) نگهداری می شود. یعنی مشخص می شود یک فایل چه ویژگی و چه محتوای دارد. بسیاری از این ویژگی ها با فیلدهای ثابت جدول elgg_metadata یکسان است. این جدول بیشتر توسط کد جاوا جهت ایندکس گذاری محتوای مقاله ها و جستجوی مقاله ها استفاده شده است.

شهبازی
z.shahbazi@um.ac.ir