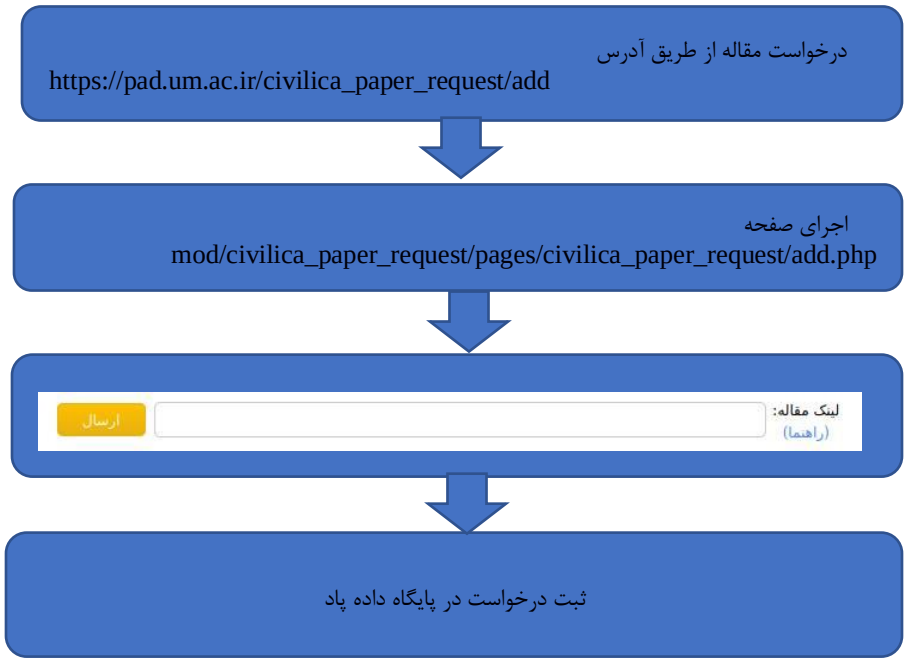
به نام خدا

درخواست و تامین مقاله از سیویلیکا در سیستم پاد

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

مراحل تامین مقاله به صورت زیر است:



ثبت درخواست در پایگاه داده پاد:

بعد از کلیک بر روی گزینه ارسال، درخواست تامین مقاله به صورت Ajax در پایگاه داده با وضعیت -۲ ثبت می شود.

```
//mod/civilica_paper_request/pages/civilica_paper_request/add.php
onclick: Ajax_func: send_request)()
{
  var link = document.getElementById("link").value;  //get link of civilica paper in request form
  xmlhttp.open("POST","' . elgg_get_site_url() .'civilica_paper_request/register_request",true);
  xmlhttp.send("link=" + encodeURIComponent(link));
}
---------------------------------

 civilica_paper_request/register_request:
// implemented in mod/civilica_paper_request/pages/civilica_paper_request/ajax/register_request.php
calee_func: register_civilica_paper_request($user_guid, $link, $status)
            {
            //implemented in : mod/civilica_paper_request/lib/civilica_paper_request.php:
              INSERT into civilica_paper_requests by status -2
            }
```

دانلود مقاله های درخواست داده شده از سیویلیکا:

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 با خواندن محتوای لینک درخواست داده شده، لینک pdf مقاله را پیدا میکند و آن را دانلود آپلود می کند. بخش های اصلی کد مربوطه به صورت زیر است:

```
// src/ssn/paperrequest/CivilicaRequestProcessor.java
processRequestsWaitingForDownload()
{
    calee_func: downloadRequestFromCivilica(request)
      {
        String url = request.getLink();
       CivilicaDownloader downloader = new CivilicaDownloader(request, this.ID);
      calee_func:  file = downloader.download(url)
                          {
                           //src/ssn/paperrequest/CivilicaDownloader.java
                           calee_func: loginResult = login()
                             {
                                 loginURL = BASE_URL + "login.html";
                                 Page page = webClient.getPage(loginURL);
                                 HtmlPage htmlPage = (HtmlPage) page;
                                 forms = htmlPage.getForms();
                                 String name = inputElement.getAttribute("name");
                                 if (name != null && name.equalsIgnoreCase("username")) {
                                         usernameSet = true;
                                         inputElement.setAttribute("value", Config.CIVILICA_USERNAME);
                                 } else if (name != null && name.equalsIgnoreCase("user_password")) {
                                         passwordSet = true;
                                         inputElement.setAttribute("value", Config.CIVILICA_PASSWORD);
                                 } else if (inputElement instanceof HtmlSubmitInput) {
                                         submitBtn = (HtmlSubmitInput) inputElement;
                                 }
                                 page = submitBtn.click();
                              }
                             ----------
                           Page page = webClient.getPage(pageURL);
                          HtmlPage htmlPage = (HtmlPage) page;
                         calee_func: pdfPath = downloadPdfFromPage(htmlPage)
                          {
                            downloadForm = htmlPage.getFormByName("dlForm");
                            HtmlElement downloadBtn = findDownloadBtn(htmlPage);
                           Page page = downloadBtn.click();
                           htmlPage = (HtmlPage) page;
                          href = anchor.getHrefAttribute();
                         if (href != null && href.contains("copied/RU/"))
                                 String link = href;
                                 Page page = webClient.getPage(link);
                                 seed = Util.getRandomString(50);
                                 File file = new File(PaperRequestProcessor.TEMP_PATH, seed + ".pdf");
                                 pdfPath = file.getAbsolutePath();
                                 Util.writeToFile(page.getWebResponse().getContentAsStream(), pdfPath);
                                 return pdfPath;
                      }
                }
                calee_func: updateAsDownloaded(request, request.getSeed())
                {
                 //status for WAITING_FOR_UPLOAD  = ·
                 //UPDATE paper_download_requests SET status = 0   WHERE id = ?
                }
`
```

آپلود مقاله های دانلود شده از سیویلیکا در پاد:

کد جاوا درخواست های دانلود شده را از پایگاه داده پاد استخراج می کند و فایل ها در سیستم پاد آپلود می کند.
بخش های اصلی کد مربوطه به صورت زیر است:

```
processRequestsWaitingForUpload()
{
   Calee_func: getCivilicaPaperRequestWaitingForUpload()
   {
    //SELECT id, link, requester_guid, seed FROM civilica_paper_requests WHERE status = 0
   };
    ----------------
   Calee_func: uploadRequestedFile(request)
   {
       filepath = TEMP_PATH + File.separatorChar + request.getSeed() + ".pdf";
       File file = new File(filepath);
       Calee_func: File movedFile = moveDownloadedFile(file, request, start)
          {
          path = "/san/pad-data/" + yearStr + File.sep + monthStr + File.sep + dayStr +File.sepr+ () + File.sep + "file";
          File directory = new File(path);
          FileUtils.copyFileToDirectory(file, directory, true);
          };
   };
    -------------
   Calee_func: fileGuid = uploadFileToPAD(request, movedFile) //get & set metadata
    -------------
    Calee_func:  updateAsFinishedSuccessfully(request, fileGuid) {
              //    UPDATE civilica_paper_requests SET status = 1
     }
     -------------
    Calee_func:   file.delete();
}
```

جزئیات تابع uploadFileToPAD در ادامه بیشتر توضیح داده شده است:

uploadFileToPAD(){

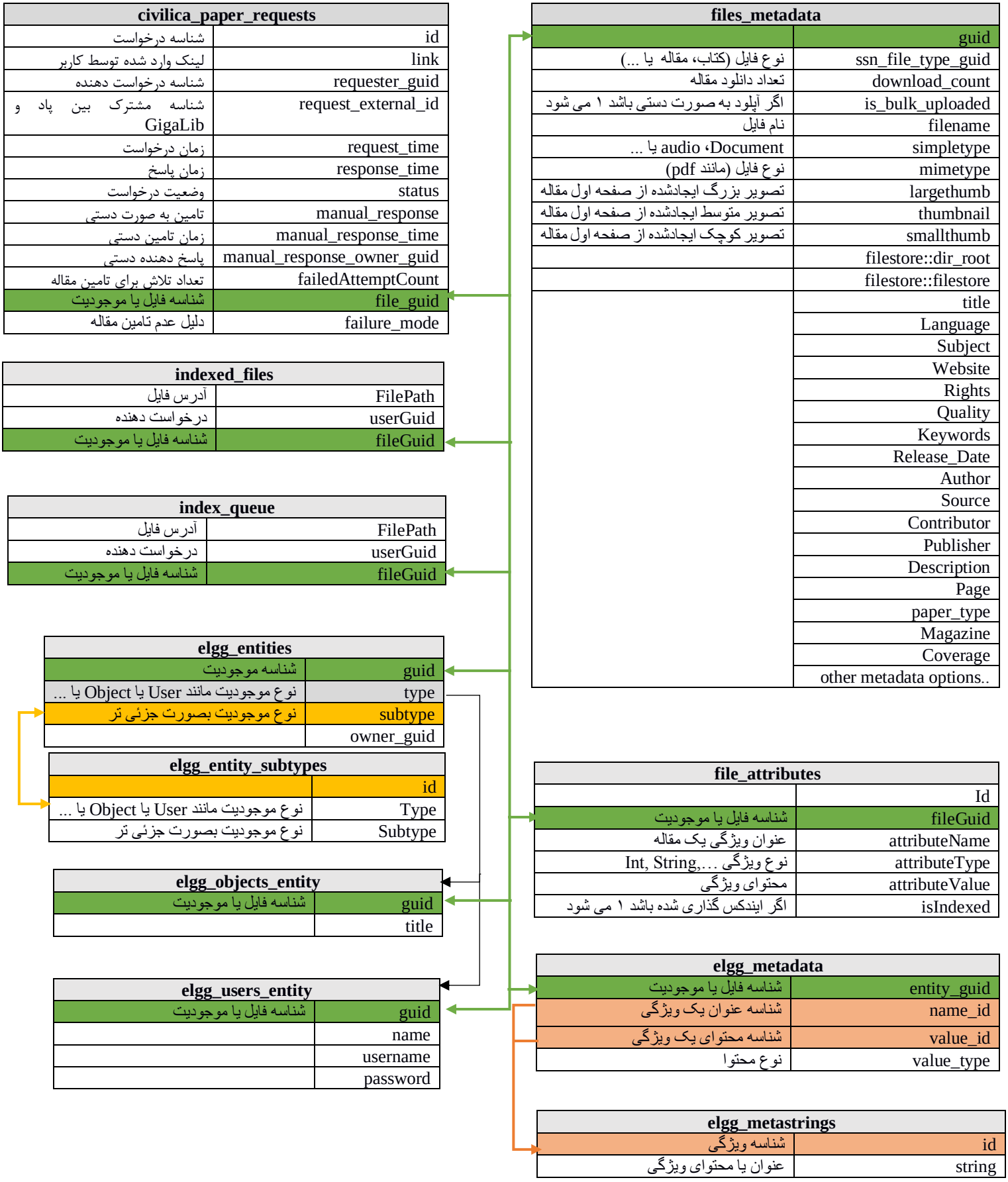
```
  String[] attributeNames = {"title", "Title", "Author(s)","Publisher", "Page(s)","Magazine/Journal", "Coverage", ….};
Calee_func: metadata = getMetadataFromCivilica(link) {
        xPathExpression = "//*[name() = 'meta']";
        resultList = htmlPage.getByXPath(xPathExpression);
        iterator = resultList.iterator();
        while (iterator.hasNext()) {
        HtmlElement element = (HtmlElement) iterator.next();
        name = element.getAttribute("name");
        value = element.getAttribute("content");
        property = element.getAttribute("property");
        if (name != null && name.equalsIgnoreCase("citation_title")) {
                metadata.add("atitle");
                metadata.add(value);
        }
        else if (name != null && name.equalsIgnoreCase("citation_conference_title")) {
                metadata.add("conference_title");
                metadata.add(value);
        }else if ….
      }
   }
   -------------------------
Calee_func: papreAttributesFromMetadata(request, attributeValues, metadata){
attributeValues[0] = title;
attributeValues[1] = getMetadata(metadata, "atitle");
attributeValues[2] = getMetadata(metadata, "au");
attributeValues[3] = null;//paper.getPublisher();
….}
-----------------------------
Calee_func: fileGuid = uploader.uploadFile(){ //src/ssn/gui/BulkUploader.java
    Calee_func:  douploadFile(){
        Calee_func: addToIndexQueue(userId, file, null, scheduleTimestamp, Long.toString(fileGuid))
           {  //INSERT INTO index_queue(filePath, fileSize, }
      }
    Calee_func: addAttributes(fileGuid, fileAttributes, userId){
            Calee_func:  ElggInterface.addAttributes(fileGuid, fileAttributes, userId) {  //src/ssn/index/ElggInterface.java
                    name = fileAttributes.get(i).getAttributeName();
                   value = fileAttributes.get(i).getAttributeValue();
                   type = fileAttributes.get(i).getAttributeType();
                  isAnalyzed = fileAttributes.get(i).isAnalyzed();
                  isIndexed = fileAttributes.get(i).isIndexed();
                Calee_func:   addAttribute(fileGuid, name, value, type, isAnalyzed, isIndexed, userId) {
                    Calee_func:   DBManager.addToElggMetadata(){
                          //NSERT INTO elgg_metadata(entity_guid, name_id, value_id, value_type
                     }
                    Calee_func:   DBManager.addToFileAttributes(){
                        //INSERT INTO file_attributes(fileGuid, attributeName, attributeValue, attributeType,…
                    }
                }
            }
      }
}
----------------------
Calee_func: PaperRequestProcessor.createCoverForFile(file);
}
```

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

وضعیت مقاله	کد وضعیت مقاله
درخواست منتظر دانلود	-۲
درخواست منتظر جهت آپلود فایل دانلود شده در سرور پاد	۰
درخواست هایی که قابل دانلود نبودند	-۱
درخواست هایی 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. در زمان آپلود فایل در سیستم پاد، گرفتن متادیتا انجام می شود. برای دریافت متادیتا لینک صفحه مقاله سیویلیکا خوانده می‌شود. سپس بر اساس ساختار HTML صفحه متادیتای مقاله از جمله عنوان و نویسندگان بازیابی می شود و در پایگاه داده ذخیره می شود. شبه کد مربوط به این قسمت در قسمت آپلود فایل، تابع **uploadFileToPAD** آورده ششده است. در ادامه جداول مرتبط با فرآیند درخواست مقاله سیویلیکا و تامین آن توضیح داده شده است.


civilica_paper_requests
id
شناسه درخواست
link
لینک وارد شده توسط کاربر
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
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
failure_mode
دلیل عدم تامین مقاله
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

civilica_paper_requests:

به ازای هر درخواست مقاله یک رکورد در این جدول اضافه می شود. اطلاعاتی مانند لینک مقاله و درخواست دهنده مقاله 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 type می‌باشد 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:

در زمان آپلود مقاله، متادیتای مقاله جهت جستجوی بهتر index می شود. 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