Development of a Technology for Automation of Work with Sources of Information on the Internet

YEVHEN HRABOVSKYI
Department of Computer Systems and Technologies
Simon Kuznets Kharkiv National University of Economics
Nauky Avenue 9a, Kharkiv
UKRAINE

VITALINA BABENKO
International E-commerce and Hotel&Restaurant Business Department
V. N. Karazin Kharkiv National University
Svobody sq. 4, Kharkiv, 61022
UKRAINE

OLEKSANDR AL'BOSCHIY
Department of Technical and Logistics
National Academy of the National Guard of Ukraine
Zahysnykiv Ukrainy sq., 3,
UKRAINE

VALENTINA GERASIMENKO
Department of Management and the military economy
National Academy of the National Guard of Ukraine
Zahysnykiv Ukrainy sq., 3,
UKRAINE

Abstract: - The article describes the technology for automation of work with sources of information on the Internet. The created technology includes a development of a web service for storing and processing data; creation of a client part for viewing, editing and deleting records from the service; development of a browser extension for instant addition of articles to a web service database. Addition of records to the service is implemented by the development of an extension to Google Chrome browser. Creation of an extension involves getting information about the author of the article, the title of the article, addresses, key expressions, and highlighted text. Regular expressions and recursive search are used to obtain information about the author of the text. The article offers algorithms for finding the name of a literary source and its author. In order to allow the web service to interact with other services or programs, a REST service was developed in the article. A graphical user interface to display and edit records was developed for the web service using the AngularJS JavaScript framework. For the client part, a mechanism of an on request citations search and a mechanism of reference list formation for the selected citations is proposed. For the on request citations search, the article suggests the algorithm of citations ranking. Appropriate plugins for Google Chrome have been developed, which allow to connect content levels, background, and pop-up with each other. Functions for authorization, receiving a token, information search launch, search for an author, and a bibliographic record creation have been created.

Key-Words: - Web service, Framework, Citation ranking, Information search, Bibliographic record, Information sources

Received: September 28, 2019. Revised: April 3, 2020. Accepted: April 16, 2020. Published: April 24, 2020.

1 Introduction

In the course of scientific research work on literary sources is carried out at all its stages. At the preparatory stage, existing publications help to specify the topic, identify the objects of study, as well as to develop theoretical background for future work, its methodological support. The study of literary sources helps to realize the importance of the chosen topic of the study and equips with the fundamental theoretical and methodological principles of its implementation.

Working on literary sources requires the researcher to be able to read quickly, to understand and analyse the read, to focus on the main, essential for the disclosure of the research topic.

The use of different methods of working with literary sources allows to separate the known from the unknown, to use the accumulated experience, to clearly formulate the hypothesis of the study. After the initial review of the literature selected from the library catalogues, familiarization with the abstract, introduction, content, conclusion and a cursory review of the content, the method of processing the source is selected, which includes it`s careful study, summarizing, selective study, followed by extracts and annotated cards writing. No matter what medium (punch card, paper, electronic media) recorded information from a literary source is recorded, it should be grouped on a homogeneous basis to be used in the research process. Typically, this basis is the questions the research topic contains. Therefore, separate folders need to accumulate material by the sections of the study plan. Punched cards are more convenient to group in special filing cabinets, but they must have cardboard dividers with indicators by sections of the study plan. Similar file cabinets can be created on slot punched cards and punched cards with edge perforations.

Changes are made to the grouping of the abstracted material, since in the course of scientific research its original plan is corrected and specified. This is especially important when doing collective research when sections of the work plan are assigned to different performers.

Excerpts, quotations, digital data should be referenced: author, title, publisher, year and place of publication, pages. Before proceeding to work on the source, it is necessary to make a bibliographic description at the top of the sheet, specify a section of the plan on the topic of the study to which the statement relates, and then summarize the literary source.

References to the literary source should include a complete bibliographic description necessary to compile a list of used literature on the topic of study. Otherwise, a need to revert to it occurs.

An important role in the process of compiling a bibliography is the Internet, as a great source of information and a way of finding information.

Searching the Internet is a continuous process that requires attention and concentration on the task. Processing information sources online takes a long time because you need to save data for bibliographic lists and citations. This processing feature also brings the search process to a discrete view and puts the searcher's attention to the routine of storing data instead of concentrating on the search task.

When working with sources of information on the Internet, there is a problem of storing information because it is not always possible to record the page address, and finding the page you need is difficult again, because there is a great deal of information available today, both useful and distracting. That's why you need to store the right sources of information right away so you can access them quickly.

The second problem when working with literature is the formation of a bibliographic list when writing a scientific article. This process usually takes a lot of time and effort, as the author needs to know GOST and DSTU about compiling bibliographic records.

2 Problem Formulation

2.1 Literature review

General features of the analysis and role of web-based information catalogues of bibliographic information sources are given in a scientific article [12]. Research [2], [8], [10] examines methodological approaches to information visualization and the creation of information directories. Scientific papers [1], [4], [15] propose methods for assessing the quality of information bibliographic resources formation. In the study [13], methodological foundations were formed regarding the creation of web-based tools for the formation of a list of sources of information. Web-based tools for creating information directories are proposed in [8]. Empirical studies on the creation of information support systems for the formation of bibliographic catalogues are given in [14], [16], [19], [20]. Various variants of technologies of designing interfaces of information systems of data cataloguing are given in scientific articles [3], [6], [7]. An analysis of decision making support for the development and use of information directories is given in [9], [18]. The identification of probable risks and limitations of using web-based information catalogues of bibliographic sources of information has been reflected in studies [11], [17], [21].

However, at present there is no scientifically based technology for automation of work with sources of information on the Internet in the specialized literature.

These aspects determine the relevance of the claimed research.

The purpose of the research is to create technology for automation of work with sources of information on the Internet.

2.2. Materials and methods

At present, there are the following software tools for automating bibliographic records creation.

First of all, such software is BiblioWeb – a research group solution that allows you to share a common bibliographic database on the Internet. All the basic features of this bibliographic program are provided through a web browser. The user can be assigned one of the properties: "read" or "write". Users in the "read" category can search, display, and export bibliographic material. Users in the "write" category can add, edit, delete, import and format documents using a web browser. The program also supports the creation of a discussion forum.

HotReference is a bibliographic database for sharing. It allows you to: organize your bibliography into categories, attach files in postscript, PDF, doc formats, create annotated bibliographies with your personal comments on each article. The program supports all of the most common formats for importing and exporting bibliographic data. The user can also use HotReference to create a list of posts on a personal site on the Internet. A personal directory is designed to keep your bibliography in order. The program is available for free.

Refbase is a platform-independent, multi-user interface for working with scientific literature and references. Users can receive automatic e-mail notifications of newly added literary sources and dynamic RSS feeds of recently added / revised literary sources, as well as requests from any of the admitted users.

RefWorks is a list and database manager that allows users to create their own database by importing links from text files or databases available on the Internet. When importing bibliographic descriptions, RefWorks automatically creates pointers for authors, descriptors, and periodicals, so you only need to select ("click") the word you want. You can use the Quick Search command to search all the fields for the most complete search, or narrow your search within special terms and fields using the Advanced Search command. Free of charge for 30 days.

WIKINDX is a bibliography and citations / notes program, as well as a system for author's work on articles designed for both a single user (working on different operating systems) and a group of users collaborating on the Internet. In the case of multiple users, you create a bibliography and work with your bibliographic sources from the main WIKINDX bibliography and view the bibliographies of other users. The program allows you to add an unlimited number of attachment files to each resource and create a directory of resources by category and keywords. The program also provides numerous options for exporting data (to other bibliographic programs).

All of these services allow you to work with bibliographic tools, share them with colleagues, or even sell your records, but they do not allow you to add an entry immediately when you are on another site when you need to remember a useful article.

To somehow compare the completely different programs: paid and free, online services and simple programs, we will set the task: we need a simple (non-professional) program (or service), with a convenient modern interface, which has at least the minimum required functionality:

1) rapid entry of the source in the "filing cabinet";
2) import and export of links;
3) easy installation.

For the review, we select the most popular programs that meet the requirements:

1) Google Keep;
2) EvernoteEndNote;
3) Wunderlist;
4) CintaNotesSpringpad;
5) Simplenote.

This review is limited to services that allow you to quickly store links, web content, and other content; it does not cover the generation of references from selected sources.

The new Google Keep notes storage service is presented as a web version and an Android version. It allows you to save online text notes, to-do lists, and accompany them with images. In addition, you can save audio to Google Keep.

One feature of the service is the ability to synchronize between all devices running Google Keep, including data stored using the web version of the service.

Users can search the contents of notes, select the mode of display of all created notes, select different colours for displaying notes.

Many experts believe that Google, along with the launch of Google Keep, is trying to compete with Evernote, the most popular note-taking and web browsing service to date. However, it is obvious today that Google Keep is only a copy of Evernote.

The main difference between these two tools is, first of all, a more convenient note categorization system – in Evernote you can tag them, specify names, and the built-in editor allows you to work with the note text in the same way as in a full text editor.

In addition, Evernote offers special browser extensions that allow you to save a page of a site you are browsing in a repository - a sort of tool to save links "for later", or simply a service to store useful links. So, if Evernote is an online notepad plus web content service, Google Keep is just a functional note-taking service with synchronization features.

However, in the long run Google Keep will be able to improve its position and become a real competitor to Evernote if Google integrates its note service with its other products - first of all, the social network Google+, as well as with Gmail, Google Calendar, and ideally - with the preparation to start in Bose Google Reader.

In addition, integrating Google Keep into Google Chrome means even more opportunities for Google's new project.

The Wunderlist online task manager recently introduced its web content storage service – a novelty from Wunderlist works just like Evetnote tools for storing web content: after installing a browser extension, users can save the content of web browsing by simply clicking a button on the toolbar.

Despite the fact that Google is abandoning Google Reader, the company is clearly aware that the modern Internet user is faced with a lot of web content.

This is why user-friendly, interesting and popular web content management and storage services are important. So far, Google Keep is more like an ordinary online notebook, analogous to the Google Notebook closed in 2011.

Only device sync features may be of interest to the user. However, Evernote successfully handles this task, and it is much more functional than a web content storage service.

The most functional analogue of Evernote is the Springpad multifunctional online notebook, which also has a web version, iOS and Android applications. A special feature of this service is the presence of the Apps module, which allows you to connect to it various applications, which are more than enough.

The service can be synchronized with a number of popular web tools, such as Gmail, and create a contact list based on the address book, Google Calendar, Twitter, and Facebook.

The Simplenote service lets you work with notes online, also offering synchronization features for different devices. Simplenote also has a large number of applications, including browser extensions that allow you to save links to popular web pages in Simplenote.

Evernote – a web service and software suite for creating and storing notes. A piece of text, a full web page, a photo, an audio file, or a handwritten recording can serve as a note. Notes may also contain attachments with files of another type. Notes can be sorted by notepad, labelled, edited, and exported.

Evernote supports several computer and mobile platforms, including OS X, iOS, Chrome OS, Android, Microsoft Windows, Windows Phone, BlackBerry, and webOS, and offers online synchronization and backup.

The analysed services meet the requirements for quick access and editing, but they do not allow to make a list of references and do not provide API (application programming interface).

There are also a number of programs that create a list of literature, such as Zotero (a very powerful tool for creating a list of references), however all of them do not provide access to their APIs, which makes it impossible for them to be embedded in other services required by the user. It is therefore advisable to build a service that integrates these features and has an open API.

Having analysed the work with information sources on the Internet, it becomes clear that this process takes a long time, and also makes the search process discrete, which in turn is a continuous process and requires concentration and attention.

Let's formulate a hypothesis: "The efficiency of work with information sources on the Internet will increase if, as a result of automation of this process, the time to store citations and information for bibliographic records will strive to zero, all things being equal."

Having analysed the tools for creating notes and bibliographic records, we understand that none of the programs considered meets the necessary requirements, which is why it is advisable to automate the process of working with sources of information on the Internet, if this will make the process of search and processing continuous.

3 Problem Solution

The system of automation of work with sources of information on the Internet should allow to store information about the source immediately. The information should be variable – it can be deleted from the service. Each source must contain the site address, article title, some article text, key phrases for sorting and searching, and the date the record was created. The simplified structure is shown in fig. 1.

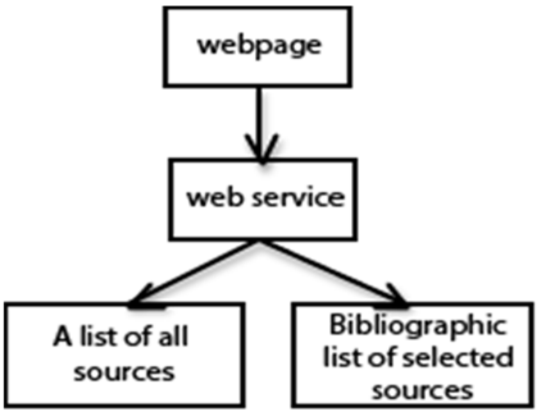


Fig. 1 Structure of the automation system

Therefore, the technology of automation of work with sources of information on the Internet should include:

1) development of a web service for storage and processing of data, which will be delivered by the API (application programming interface) to enable its use on other sites and services;

2) creating a client part for viewing, editing and deleting records from the service;

3) developing a browser extension to instantly add articles to a web service database.

You need to implement Google Chrome browser extensions to add service entries.

Google Chrome browser extensions are written using JavaScript and a custom API. Extensions have access to both page content and browser capabilities (opening, pages, etc.).

The extension should receive information about the author of the article, the title of the article, the addresses, key expressions and highlighted text.

The problem is getting information about the author of the text, because about 50% of users ignore the meta tag "author", so you need to search the page for data in which the author may be specified. To do this, we use regular expressions and recursive search.

To search for the title of the article, we search through the tags "H1", "H2", and all tags with class or ID "title", as well as tag "title". This algorithm can be written as a formula:

$$Name = \begin{cases} name, if\ h1 - h2 \\ name, if\ title\ \neq \emptyset \\ name, if\ id\ or \\ class =' title' \end{cases} \tag{1}$$

where Name is the final variable, name is the local variable that is being processed.

To find the author of the article we use an algorithm similar to the search for the article title:

$$Author = \begin{cases} author, if [ref = author] \subseteq tag \\ author, ifclass = author \\ author, if\ id = author \end{cases} \tag{2}$$

where Author is the final variable, author is the local variable undergoing validation and processing.

When designing a web service, keep in mind that the service must be able to interact with other services or programs. For this purpose, we develop a REST-service.

A graphical user interface for the web service was developed to display and edit records using the AngularJS JavaScript framework. This is a relatively new tool developed by Google, which is guided by the MVW principle, meaning model, view and whatever. "Whatever" means a controller, a template, directives, and more. This approach is taken into account by almost all modern development tools. It improves the readability of the code and improves its support. Consider a comparison of the most popular JavaScript programming frameworks in Table 1.

Experts from multimedia publishing companies "TessLab", "ABC-Code" and "WebProject" evaluated each of the parameters of the framework, which is significant for the developer.

According to table. 2, AngularJS framework is the best for now, it should be added that it is the only framework that works without additional dependencies, that is, we only need to download its code and we can start our development immediately.

For the client part, a search engine for quotations on request and a mechanism for forming a list of references for selected citations should be developed.

To search for quotes on request, you should use the algorithm of ranking quotes:

$$p = \sum_{i=1}^{n} kw_{rsi} \tag{3}$$

where p is the weighting factor for which sorting is performed, k is the weighting factor of each query, and w is the percentage match between query and tag.

Table 1 Comparative characteristics of frameworks

Parameters	Angular	Backbone	CanJS	Ember
Functions	5	2	4	5
Flexibility	3	5	4	3
Documentation	4	4	5	3
Productivity	4	2	4	5
Community	4	5	3	4
Ecosystem	4	5	2	4
Size	4	5	5	2
Productivity	5	4	5	4
Maturity	4	5	4	3
Memory loss protection	5	3	5	5
Total	42	40	41	38

Python3 and Django's web framework were chosen to create the web service.

The simplified scheme of the product is shown in fig. 2.

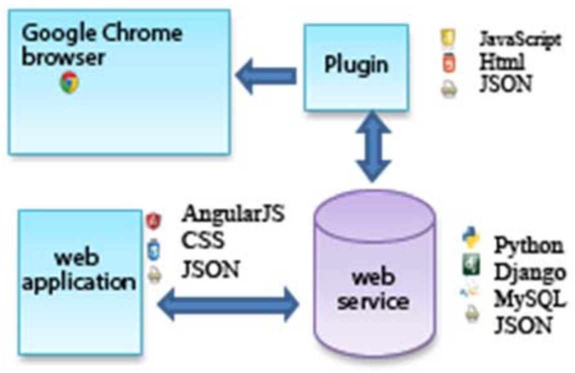
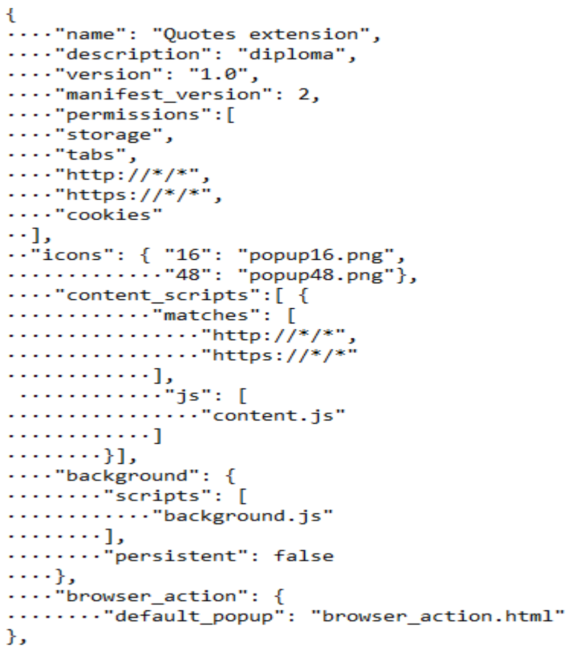


Fig. 2 Simplified product scheme

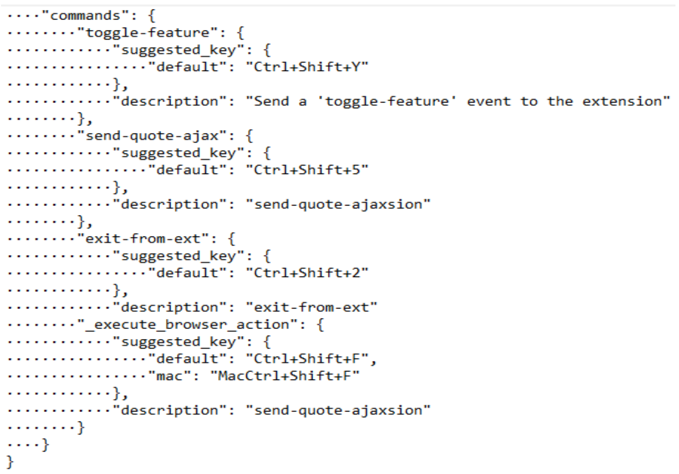
The Google Chrome browser plugin is created using JavaScript and HTML. It consists of several levels: 1) content; 2) background; 3) pop-ups.

Each of these levels can be linked through queries and messages. The entire structure should be described in the manifest.json file. In fig. fig. 3 shows the general settings, fig. 4. depicts expansion actions in different situations.



```
{
····"name": "Quotes extension",
····"description": "diploma",
····"version": "1.0",
····"manifest_version": 2,
····"permissions":[
····"storage",
····"tabs",
····"http://*/*",
····"https://*/*",
····"cookies"
··],
··"icons": { "16": "popup16.png",
············"48": "popup48.png"},
····"content_scripts":[ {
············"matches": [
················"http://*/*",
················"https://*/*"
············],
············"js": [
················"content.js"
············]
········}],
····"background": {
········"scripts": [
············"background.js"
········],
········"persistent": false
····},
····"browser_action": {
········"default_popup": "browser_action.html"
},
```

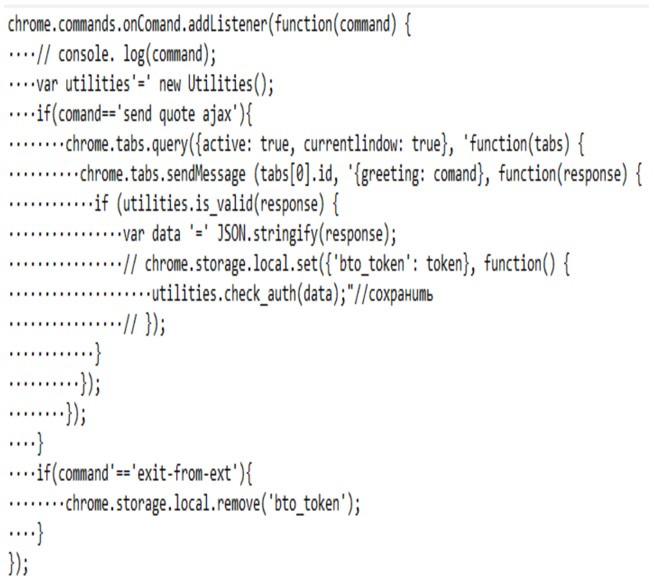
Fig. 3 General extension settings



```
····"commands": {
········"toggle-feature": {
············"suggested_key": {
················"default": "Ctrl+Shift+Y"
············},
············"description": "Send a 'toggle-feature' event to the extension"
········},
········"send-quote-ajax": {
············"suggested_key": {
················"default": "Ctrl+Shift+5"
············},
············"description": "send-quote-ajaxsion"
········},
········"exit-from-ext": {
············"suggested_key": {
················"default": "Ctrl+Shift+2"
············},
············"description": "exit-from-ext"
········"_execute_browser_action": {
············"suggested_key": {
················"default": "Ctrl+Shift+F",
················"mac": "MacCtrl+Shift+F"
············},
············"description": "send-quote-ajaxsion"
········}
····}
}
```

Fig. 4 Extension actions

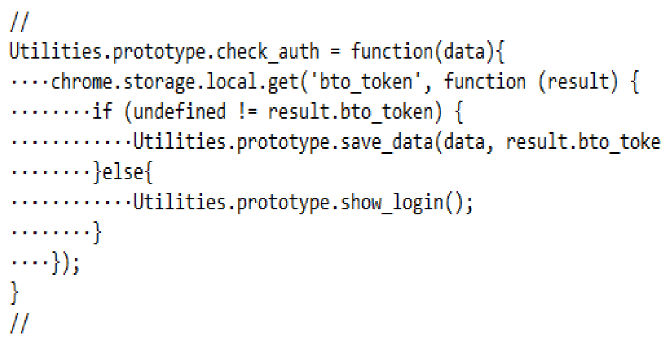
The manifest.json file also specifies keyboard shortcuts, all files that you want to download, and access levels for the extension. To save the quote, you must wait for the shortcut specified in manifest.json. To do this, use the following code, which is shown in Fig. 5.



```
chrome.commands.onComand.addListener(function(command) {
····// console. log(command);
····var utilities'=' new Utilities();
····if(comand=='send quote ajax'){
········chrome.tabs.query({active: true, currentWindow: true}, 'function(tabs) {
··········chrome.tabs.sendMessage (tabs[0].id, '{greeting: comand}, function(response) {
············if (utilities.is_valid(response) {
················var data '=' JSON.stringify(response);
················// chrome.storage.local.set({'bto_token': token}, function() {
····················utilities.check_auth(data);"//сохранить
················// });
············}
··········});
········});
····}
····if(command'=='exit-from-ext'){
········chrome.storage.local.remove('bto_token');
····}
});
```

Fig. 5 Action Tracking

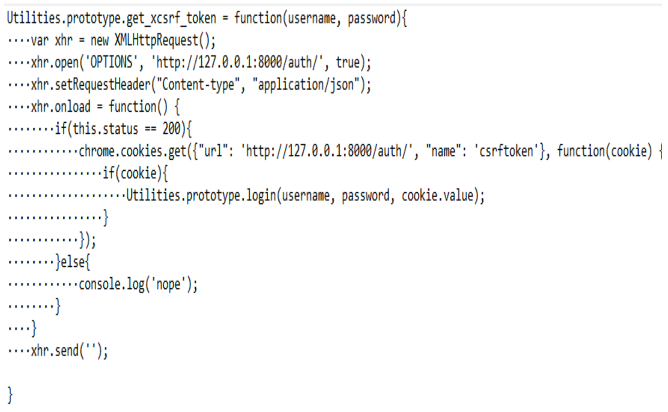
Then, check that the user has authenticated in the check_auth (data) method shown in fig. 6.



```
//
Utilities.prototype.check_auth = function(data){
····chrome.storage.local.get('bto_token', function (result) {
········if (undefined != result.bto_token) {
············Utilities.prototype.save_data(data, result.bto_toke
········}else{
············Utilities.prototype.show_login();
········}
····});
}
//
```

Fig. 6 Authentication verification

Next, verify that the token used for authentication is installed by making a request using the save_data () method and open the user authorization form. When authorizing a user, you must first request the csrf token to be installed using the get_xcsrf_token () function, which is shown below in fig. 7.

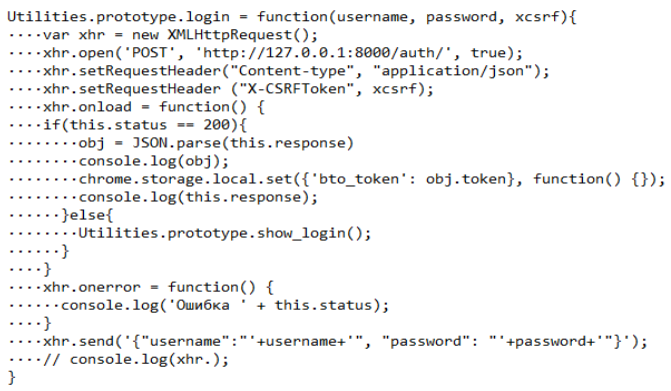


```
Utilities.prototype.get_xcsrf_token = function(username, password){
····var xhr = new XMLHttpRequest();
····xhr.open('OPTIONS', 'http://127.0.0.1:8000/auth/', true);
····xhr.setRequestHeader("Content-type", "application/json");
····xhr.onload = function() {
········if(this.status == 200){
············chrome.cookies.get({"url": 'http://127.0.0.1:8000/auth/', "name": 'csrftoken'}, function(cookie) {
················if(cookie){
····················Utilities.prototype.login(username, password, cookie.value);
················}
············});
········}else{
············console.log('nope');
········}
····}
····xhr.send('');

}
```

Fig. 7 Function for receiving a token

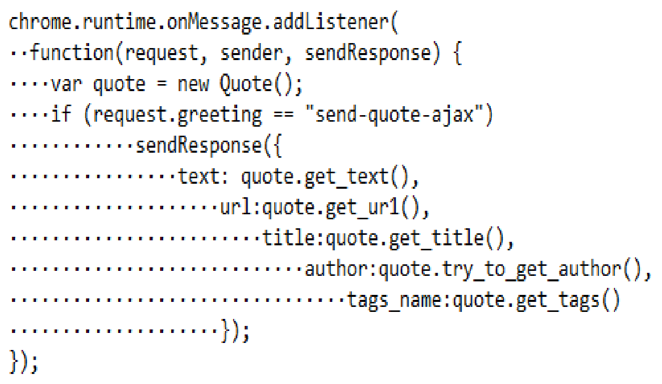
This is necessary to send dangerous requests from another domain, and even after installing the csrf token in the browser cookies, it is possible to send a POST request in order to receive the token for authentication on the web service using the login () function, which is shown below in fig. 8.



```
Utilities.prototype.login = function(username, password, xcsrf){
····var xhr = new XMLHttpRequest();
····xhr.open('POST', 'http://127.0.0.1:8000/auth/', true);
····xhr.setRequestHeader("Content-type", "application/json");
····xhr.setRequestHeader ("X-CSRFToken", xcsrf);
····xhr.onload = function() {
····if(this.status == 200){
········obj = JSON.parse(this.response)
········console.log(obj);
········chrome.storage.local.set({'bto_token': obj.token}, function() {});
········console.log(this.response);
······}else{
········Utilities.prototype.show_login();
······}
····}
····xhr.onerror = function() {
······console.log('Ошибка ' + this.status);
····}
····xhr.send('{"username":"'+username+'", "password": "'+password+'"}');
····// console.log(xhr.);
}
```

Fig. 8 Authorization function

If the correct answer is received, the authentication token is written to the browser cookie, otherwise we reopen the authorization form. Searching for the necessary information on the page is performed using the function shown in fig. 2.

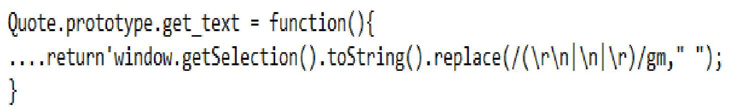


```
chrome.runtime.onMessage.addListener(
··function(request, sender, sendResponse) {
····var quote = new Quote();
····if (request.greeting == "send-quote-ajax")
···········sendResponse({
···············text: quote.get_text(),
···················url:quote.get_url(),
·······················title:quote.get_title(),
···························author:quote.try_to_get_author(),
·······························tags_name:quote.get_tags()
···················});
});
```

Fig. 9 Function for starting information search

In which Quote () is the class responsible for processing the data on the page. It receives selected text and bibliographic data.

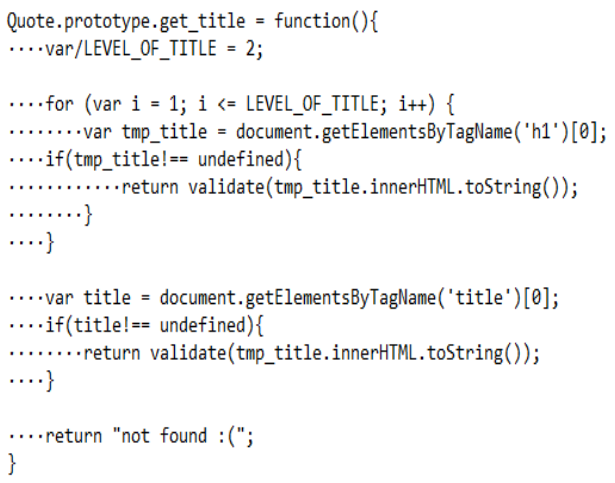
The get_text () method retrieves the selected text and clears it from unnecessary characters as shown in fig. 10.



```
Quote.prototype.get_text = function(){
....return'window.getSelection().toString().replace(/(\r\n|\n|\r)/gm," ");
}
```

Fig. 10 get_text function

The get_title () method retrieves the document name using formula (2). This method is included in the Quote class, which has a constant indicating the title levels that will be searched, and also uses the validate function, which recursively clears text from HTML tags and returns plain text (Fig. 12). The get_title function is indicated in fig. 11.



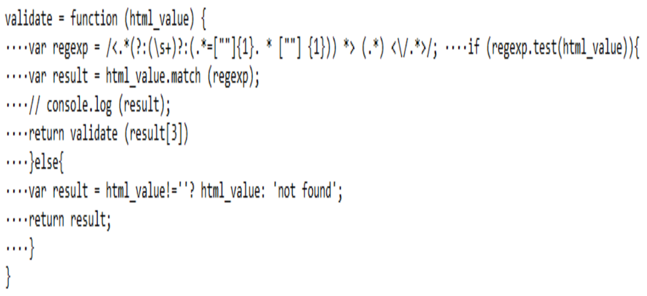
```
Quote.prototype.get_title = function(){
····var/LEVEL_OF_TITLE = 2;

····for (var i = 1; i <= LEVEL_OF_TITLE; i++) {
········var tmp_title = document.getElementsByTagName('h1')[0];
····if(tmp_title!== undefined){
············return validate(tmp_title.innerHTML.toString());
········}
····}

····var title = document.getElementsByTagName('title')[0];
····if(title!== undefined){
········return validate(tmp_title.innerHTML.toString());
····}

····return "not found :(";
}
```

Fig. 11 Title search function

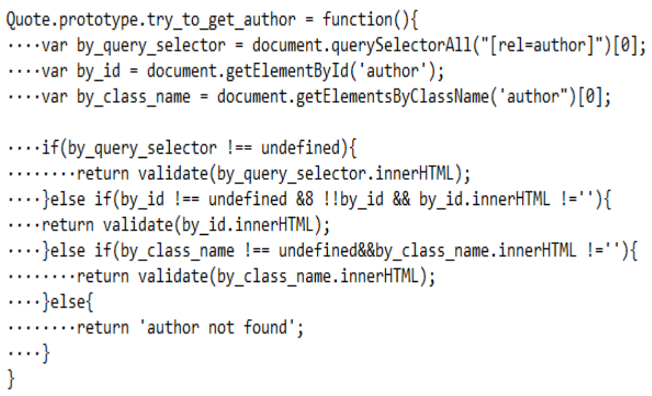


```
validate = function (html_value) {
····var regexp = /<.*(?:(\s+)?:(.*=[""]{1}. * [""] {1})) *> (.*) <\/.*>/; ····if (regexp.test(html_value)){
····var result = html_value.match (regexp);
····// console.log (result);
····return validate (result[3])
····}else{
····var result = html_value!=''? html_value: 'not found';
····return result;
····}
}
```

Fig. 12 Function for text clearing

The above function is used in almost all methods of the Quote class to obtain text without unnecessary characters. It uses recursion, that is, calls itself if needed.

Obtaining the author of the text is performed by the function try_to_get_author (), an algorithm is explained by formula 2.2 shown in fig. 13.

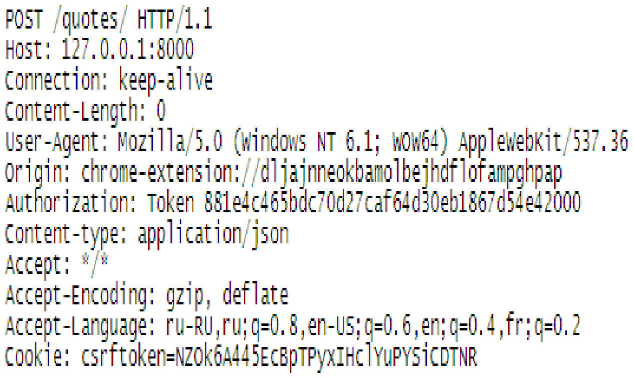


```
Quote.prototype.try_to_get_author = function(){
····var by_query_selector = document.querySelectorAll("[rel=author]")[0];
····var by_id = document.getElementById('author');
····var by_class_name = document.getElementsByClassName('author")[0];

····if(by_query_selector !== undefined){
········return validate(by_query_selector.innerHTML);
····}else if(by_id !== undefined &8 !!by_id && by_id.innerHTML !=''){
····return validate(by_id.innerHTML);
····}else if(by_class_name !== undefined&&by_class_name.innerHTML !=''){
········return validate(by_class_name.innerHTML);
····}else{
········return 'author not found';
····}
}
```

Fig. 13 Author search function

HTTP is used to communicate with the web service. When writing a quote, a post request is sent to http://127.0.0.1:8000/quotes/. The following is an example of a query for fig. 14



```
POST /quotes/ HTTP/1.1
Host: 127.0.0.1:8000
Connection: keep-alive
Content-Length: 0
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36
Origin: chrome-extension://dljajnneokbamolbejhdflofampghpap
Authorization: Token 881e4c465bdc70d27caf64d30eb1867d54e42000
Content-type: application/json
Accept: */*
Accept-Encoding: gzip, deflate
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.6,en;q=0.4,fr;q=0.2
Cookie: csrftoken=NZOk6A445EcBpTPyxIHclYuPYSiCDTNR
```

Fig. 14 Example HTTP request

Ctrl + Shift + 5 is used to store the quote, Ctrl + Shift + 2 – to exit the extension.

The web service for the automation system for working with literary sources on the Internet should be designed using the REST methodology.

The basic principle of web service architecture that uses REST methodology is that the necessary actions are transmitted using HTTP methods. The urls of a web service look like this:

1) GET / quotes - get a list of all quotes;
2) GET / quotes / 12 - get a specific quote (# 12);
3) POST / quotes - create a note;
4) PUT / quotes / 12 - update quote (# 12);
5) DELETE / quotes / 12 - delete quote (# 12);
6) POST / auth - authentication;
7) POST / user- registration.

Django's project development methodology involves designing individual modules and then grouping them together. We divide the project into two conditional parts:

1) work with quotes;
2) work with users;

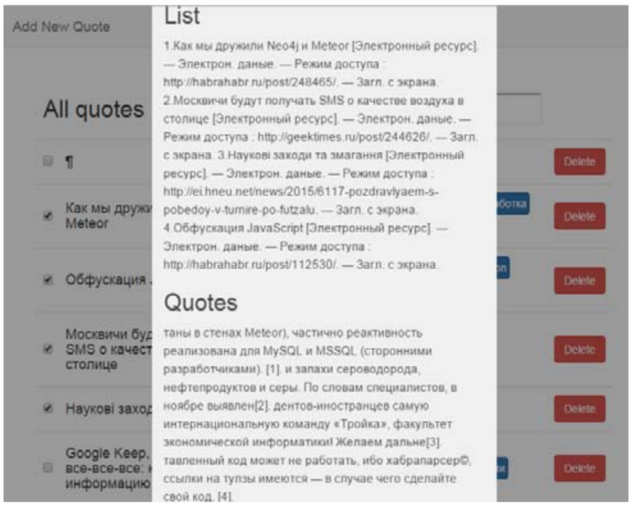
The controller.js file is a file that contains the basic logic of the application, such as a function for forming a list of references (fig. 15), the code of which is shown in fig. 16.

```
var makeList = function(quotes){
      var pattern = new RegExp('([а-яА-Я]+\s?)');
      var litteratureList = [];
      var quotesList = [];
      angular.forEach($scope.checkedQuotes, function(quote, counter){
         var cnt = counter + 1;
         if(pattern.test(quote.text)){

            result = cnt + "." + quote.title+" [Electronic resource]. - Electronic data. - Access mode :
         }
         quotesList += quote.text + "["+cnt+"]. ";
            // pattern.test(quote.text);

      return {'litteratureList': litteratureList, 'quotesList': quotesList};
   }
```

Fig. 15 Function for creating a bibliographic record



List

1.Как мы дружили Neo4j и Meteor [Электронный ресурс]. — Электрон. даные. — Режим доступа : http://habrahabr.ru/post/248465/. — Загл. с экрана. 2.Москвичи будут получать SMS о качестве воздуха в столице [Электронный ресурс]. — Электрон. даные. — Режим доступа : http://geektimes.ru/post/244626/. — Загл. с экрана. 3.Наукові заходи та змагання [Электронный ресурс]. — Электрон. даные. — Режим доступа : http://ei.hneu.net/news/2015/6117-pozdravlyaem-s-pobedoy-v-turnire-po-futzalu. — Загл. с экрана. 4.Обфускация JavaScript [Электронный ресурс]. — Электрон. даные. — Режим доступа : http://habrahabr.ru/post/112530/. — Загл. с экрана.

Quotes

таны в стенах Meteor), частично реактивность реализована для MySQL и MSSQL (сторонними разработчиками). [1]. и запахи сероводорода, нефтепродуктов и серы. По словам специалистов, в ноябре выявлен[2]. дентов-иностранцев самую интернациональную команду «Тройка», факультет экономической информатики! Желаем дальне[3]. тавленный код может не работать, ибо хабрапарсер©, ссылки на тулзы имеются — в случае чего сделайте свой код. [4].

Fig. 16 Forming a list of references

The search ranking described by formula (3) is ranked using the litteratureListfilter filter, the code snippet of which is shown below in fig. 17.

```
····var filtered =[];
····angular.forEach(items, function(item){
········if (typeof searhTextStr !='underfined' && searhTextStr!=''){
························var keepGoing =true;
············var searhtextArr = searhTextStr.split(',');
················angular.forEach(searhTextArr, function(searhText){
····················if (keepGoing) {
························angular.forEach(item.tags_name, function(tag){
····························if(keepGoing) {
································var name = tag.title.replace(/\s+/g, '');
································var needle = searhText.replace(/\s+/g, '');
································if(name.indexof (needle) >= 0){
····································keepboing = false;
····································return filtered.push(item);
································}
····························}
························});
····················};
················});
```

Fig. 17 The code snippet for ranking

Requests are handled by the $ http directive, which provides the following methods:

1) $ http.get;
2) $ http.head;
3) $ http.post;
4) $ http.put;
5) $ http.delete;
6) $ http.jsonp;
7) $ http.patch.

Each of the above methods is responsible for sending a request to the server using the appropriate HTTP protocol. With this method, we transmit and receive HTTP data from our service. By using such principles of web services construction and asynchronous communication with them, it is possible to accelerate the work of the program and improve the support and scaling of the project, as well as to provide your service to third-party applications for use.

4 Conclusion

The proposed technology for automating of work with sources of information on the Internet allows to avoid the information retrieval, which is quite difficult, because finding the necessary source can sometimes involve going through links on sites for a long period of time, and sometimes it is almost impossible to repeat the sequence of the actions.

Designed automated system reduces the time to organize the list of literature and store interesting and necessary sources. In turn, such an ordering of the found sources of information creates the possibility of avoiding errors in the formation of bibliographic records.

In the course of the theoretical analysis, it was found that the search is a continuous process, and the storage of citations and data for the bibliographic list makes it differentiable and thus reduces the search performance. When trying out the automation system to work with sources of information on the Internet, it was found that it makes the search process almost continuous and reduces the time lost in creating a list of bibliographic sources. This confirms the hypothesis of increasing the efficiency of work with information sources on the Internet if, as a result of the automation of this process, the time to store citations and information for bibliographic records will strive to zero, all things being equal.

The research made it possible to structure the process of automation of work with sources of information on the Internet, to agree the main components of this process with the basic stages of creating a web service, client development and browser extension, and to rank the search for literary sources of information.

The further direction of development for this research is to develop the methodology of information support for publishing.

As a further improvement of the developed system of automation of work with sources of information on the Internet it is possible to develop a system for searching keywords by page text.

References:

[1] Babenko V., Kulczyk Z., Perevosova I., Syn iavska O. and Davydova O. Factors of the development of international e-commerce under the conditions of globalization. SHS Web of Conferences, 2019, pp. 10-16. doi: https://doi.org/10.1051/shsconf/20196504016

[2] Brambilla M. Large-scale Model-Driven Engineering of web user interaction: The WebML and WebRatio experience. Science of Computer Programming, № 89, 2014, pp. 71-87 doi: https://doi.org/10.1016/j.scico.2013.03.010

[3] Canessa E., Zennaro M. A Mobile Science Index for Development. International Journal of Interactive Mobile Technologies, № 6(1), 2012, pp. 4 – 6.

[4] Hood, N. Quality in MOOCs : Surveying the terrain. Burnaby: Commonwealth of Learning 40 p, 2016.

[5] Hrabovskyi Y., Brynza N, Vilkhivska O. Development of information visualization methods for use in multimedia applications. EUREKA: Physics and Engineering, № 1, 2020, pp. 3 – 17.

[6] Hrabovskyi Y., Fedorchenko V. Development of the optimization model of the interface of multimedia edition. EUREKA: Physics and Engineering, 3, 2019, pp. 3 – 12. doi: 10.21303/2461-4262.2019.00902

[7] Hryshchuk, R. Synergetic control of social networking services actors' interactions. Recent Advances in Systems, Control and Information Technology, № 543, 2017, pp. 34-42. doi: https://doi.org/10.1007/978-3-319-48923-0_5.

[8] Hu, M. AHP and CA Based Evaluation of Website Information Service Quality: An Empirical Study on High-Tech Industry Information Center Web Portals. Journal Service Science & Management, № 3, 2009, pp. 168-180. DOI: https://doi.org/10.4236/jssm.2009.23020

[9] Kock, A., Georg Gemünden, H. Antecedents to Decision-Making Quality and Agility in Innovation Portfolio Management. Journal of Product Innovation Management, № 33(6), 2016, pp. 670-686.

[10] Kuisma, J., Hyönä, J., Simola, J. (Eds.) Perception of visual advertising in different media: from attention to distraction, persuasion, preference and memory. Lausanne: Frontiers Media SA, 2015, 124 p. doi: https://doi.org/10.3389/978-2-88919-416-2

[11] Kuznetsov A., Kavun S., Smirnov O., Babenko V., Nakisko O., Kuznetsova K. Malware Correlation Monitoring in Computer Networks of Promising Smart Grids. 2019 IEEE 6th International Conference on Energy Smart Systems, ESS 2019 – Proceedings, №. 8764228, 2019, pp. 347-352. doi: 10.1109/ESS.2019.8764228

[12] Lazarenko, N. Symbiosis of Methodological Approaches to the Development of Education in the Information Society. Nauka i osvita, № 4, 2017, pp. 107-112. doi: https://doi.org/10.24195/2414-4665-2017-4-18

[13] Martins, P. A Web-based Tool for Business Process Improvement. International Journal of Web Portals, № 9, 2017, pp. 68 – 84 doi: https://doi.org/10.4018/IJWP.2017070104

[14] McShane, B., Chen, C., Anderson, E., Simester, D. Decision stages and asymmetries in regular retail price pass-through. Marketing Science, № 35(4), 2016, pp. 619-639.

[15] Meyer, J.P., Creusier, J., Biétry, F. Multiple-Group Analysis of Similarity in Latent Profile Solutions. Organizational Research Methods, № 19(2), 2016, pp. 231-254.

[16] Mohammadzadeh, Darrodi M. Models of Colour Semiotics. Leeds : University of Leeds, 209 p., 2012

[17] Norris, D. Content Machine: Use Content Marketing to Build a 7-Figure Business With Zero Advertising. Kindle Edition, 164p, 2017.

[18] Patru, M. Making sense of MOOCs: A guide of policy-makers in developing countries. Paris: United Nations Educational, Scientific and Cultural Organization (UNESCO) and Commonwealth of Learning (COL), 102 p., 2016.

[19] Pedraza-Martinez, A. J., Van Wassenhove, L. N. Empirically grounded research in humanitarian operations management: The way forward. Journal of Operations Management, № 45, 2016, pp. 1-10.

[20] Wan X., Dresner M. Losing the Loop: An Empirical Analysis of the Dynamic Decisions Affecting Product Variety. Decision Sciences Journal, № 46(6), 2015, pp 1141-1164.

[21] Zhou, L., Zhang, Y. Y., Rao, L. L., Wang, Z. J., Wang, W. A Scanpath Analysis of the Risky Decision-Making Process. Journal of Behavioral Decision Making, № 29(2-3), 2016, pp. 169-182.