UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Software Engineering Curriculum

Sergi Martínez Rodríguez

Developing a Web Augmentation Tool for Supporting End-User Ancillary Searches

Master's Thesis (30 ECTS)

Supervisor(s): Ishaya Peni Gambo, PhD; Alejandro Fernandez, PhD

Tartu 2022

Developing a Web Augmentation Tool for Supporting End-User Ancillary Searches

Abstract:
Searching the web is a task that people perform every day. A large amount of information is available to the user, which is positive. However, it can sometimes lead to confusion. Nowadays, the software can simplify users' tasks considerably and is part of many people's everyday life. That is why we believe that software could also help the user to change and improve the experience when searching the web. Therefore, this research aims to investigate if it is somehow possible to improve the user experience when performing searches through the use of the software. In addition, we evaluate the usability and efficiency of this software created by testing among users to see if the tool helps users and to what extent. This document shows how the research was done to determine to what extent software can improve user experience on searching. Additionally, it explains the process of making such software, in this case, a browser extension, and how it is tested on different users to validate its usability and effectiveness. In the thesis, it can be found that it is indeed possible to improve the user experience when searching the web by using the software. In addition, it explains the testing process that was carried out with different users and the conclusions that were reached. It is concluded that the software can help in the user's web searches, improving their experience and allowing them to perform searches in a faster way. The experiments conclude that they help users perform these searches, especially by decreasing the time of these but with room for improvement. This is why the project leaves room for future researchers to study the software, improve it, and rerun the experiment with more users to evaluate the improvement.

Keywords:
Software, primary searches, secondary searches, ancillary searches, browser extension, avaluation experiment.

CERCS: P170 - Computer science, numerical analysis, sytems, control.

Veebi täiendamise vahendi väljatöötamine lõppkasutajate abihangete toetamiseks

Lühikokkuvõte:
Veebis otsimine on ülesanne, mida inimesed täidavad iga päev. Kasutajale on kättesaadav suur hulk teavet, mis on positiivne. Siiski võib see mõnikord põhjustada segadust. Tänapäeval võib tarkvara kasutajate ülesandeid oluliselt lihtsustada ja see on osa paljude inimeste igapäevaelust. Seepärast usume, et tarkvara võiks aidata ka kasutajal muuta ja parandada veebis otsimise kogemust. Seetõttu on käesoleva uuringu eesmärk uurida, kas on kuidagi võimalik parandada kasutajakogemust otsingute tegemisel tarkvara abil. Lisaks on vaja hinnata selle tarkvara kasutatavust ja tõhusust, mis on loodud kasutajate seas testimise teel, et näha, kas ja mil määral aitab see vahend kasutajaid. See dokument näitab, kuidas uuriti, mil määral saab tarkvara parandada kasutajakogemust otsingul. Lisaks selgitatakse sellise tarkvara, antud juhul brauseripikenduse, valmistamise protsessi ning seda, kuidas seda testitakse erinevate kasutajate peal, et kontrollida selle kasutatavust ja tõhusust. Sellest dokumendist selgub, et tarkvara abil on tõepoolest võimalik parandada kasutajakogemust veebiotsingute tegemisel. Lisaks selgitatakse erinevate kasutajatega läbiviidud testimisprotsessi ja tehtud järeldusi. Leitakse, et tarkvara võib aidata kasutajatel veebiotsingutel, parandades nende kogemust ja võimaldades neil otsinguid kiiremini sooritada. Eksperimendid viivad järeldusele, et nad tõepoolest aitavad kasutajatel neid otsinguid teha, eelkõige vähendades nende aega, kuid on veel arenguruumi. Seetõttu jätab projekt tulevastele teadlastele ruumi tarkvara uurimiseks, selle täiustamiseks ja eksperimendi uuesti läbiviimiseks rohkemate kasutajatega, et hinnata parandusi.

Võtmesõnad:
Tarkvara, esmane otsing, sekundaarne otsing, kõrvalotsingud, brauseri laiendus, hindamiskatse.

CERCS: P170 - Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

List of Figures

List of Tables

Contents

1 Introduction

Searching is a significant task in web applications that necessitate developing new tools to support the user and provide a better experience [17]. As Aghaee Pautasso [1] observed, end-user development (EUD) offers promising techniques and strategies to build such supporting tools, especially in a context where manipulating textual information is still dominant and visual metaphors and new interaction paradigms could improve the search experience. Usually, in EUD research, the problem comes from end-users [2]; thus, it would be interesting to know the users' opinions about the tool's usefulness and customize the visualization of results in the tool description.

Remarkably, the users are meant to perform the extra task of creating a search service. Therefore, the focus of this thesis is threefold. Firstly, we focus on developing a tool to help users conduct web search tasks to support their primary tasks. Secondly, we describe the developed web browser add-on as a tool to support end-users in creating personalized ancillary searches. Ancillary search, in this context, is the search that allows a user to look for complementary information in a web browser to achieve another main task [7, 6]. Thirdly, we focus on performing further experiments to evaluate the tool's usability for creating search services through qualitative and experimental means.

Let's take an example. A user wants to go on vacation to a foreign country, but unfortunately he/she has no knowledge of this country, so he/she sets out to learn some facts. After a while performing several queries in his/her favorite search engine, he/she ends up with several tabs open, with information related to the country but without any necessary connection between them. The context right now, has several web pages, all of them with something in common, the country. From this context, the user has found interesting facts, names of places, names of important people, and including images of all kinds, whether monuments, celebrities or arts. It is at this point that the user wants to know what is the connection of all these elements, so an ancillary search will come in handy. Unfortunately, he/she has no means to do so, since the images and texts are scattered all over the pages. Therefore, this work is focused on performing ancillary searches of a more global context, able to contain both text and images, and from different web sites.

Specifically, the thesis aims to develop a browser extension able to help users in the process of doing searches on the web. A lot of information can be found during this process, so a browser extension could help the user in several aspects. Moreover, it's necessary also to prove that this extension indeed works, so a statistical study will be held among many users to prove if the extension did help them in this process or didn't.

1.1 Research goal

The main goal of this work is to develop an extension that may help the user perform searches and then perform a statistical study to prove whether the extension works or not.

It will be necessary to perform a test on each one of the individuals participating in this study and record the results on a table which will be used to determine if the searches of these users were improved or not.

The project will focus on the following research questions (RQ):

- RQ1: How can we improve the search experience in web applications to support end-users in creating personalized ancillary searches?
- RQ2: How to evaluate the usability of the suggested approach in RQ1 for creating and executing the search services?

1.2 Research Problem

Therefore, this project aims to study in which fields within the world of extensions based on primary and secondary searches there is a gap for us, investigate how to develop a browser extension, and validate the results by doing experiments among different users. These experiments allow us to know how successful the project has been and, above all, in which areas it can be improved.

1.3 Research Contribution

The contribution of this thesis is, based on the research questions, an alternative to the conventional way of searching the Internet. We present a possible implementation of the proposed alternative as a cross-platform open-source browser extension developed using software engineering principles. Additionally, the thesis shows how to evaluate usability, effectiveness, and user satisfaction when using a software tool based on existing frameworks. The tool evaluated during this evaluation is the one developed for RQ1.

1.4 Thesis Outline

State of the art (section 2) explains current knowledge about the studied matter. The section shows various techniques and existing tools related to primary and ancillary searches, some of which influenced the design and implementation of our tool. Next, methodology (section 3) presents the research questions and the tool developed. The section also explains how the tool was designed, how all its components were developed, and the steps to deploy the tool in the cloud. Finally, a scenario is demonstrated to solve a common problem using the extension that a user might encounter while surfing the web. Evaluation and discussion (section 4) present the results of the experiment conducted on different users based on usability and satisfaction. Analysis and results (section 5) show the results based on the Research Questions, threats to validity (section 6) exposes what challenges were encountered during the development and how they were overcome.

Finally, the conclusion and future work (section 7) present a conclusion on the whole project, as well as ideas for further development of the research.

2 State of the art

This work focuses on collecting information obtained using ancillary searches by exploring new forms of interaction and following an end-user approach. Therefore, in this section, we show the current situation regarding techniques, approaches, and existing software related to web searches and ancillary searches.

To begin with, it is important to explain a concept to which our work is closely related: *Focus+Context* [15]. This technique allows the user to see the object of interest while still having the context visible. In other words, he/she can see two types of information needed simultaneously: the overview or context and the focus or detail. This idea can be associated with ancillary searches since the user wants to execute a search without losing the context.

The paper *Usability Aspects of the Inside-in Approach for Ancillary Search Tasks on the Web* [35] provides a set of tools that can extend the set of elementary tasks users can do while browsing the internet. The tools allow the user to search quickly by right-clicking any word on any web page, which is an excellent example of the *Focus+Context* technique since the user does not lose context at any time. These tools, however, require computer skills to be able to define the queries on the server, making the user dependent on the developers.

In recent years, numerous browser extensions have been released, helping to run increasingly complex searches with better results. An excellent example to consider is *SearchPanel* [33], which allows users to gather information by augmenting results in search engines such as Google, Bing, or Yahoo. However, this extension does not allow related image search nor a context of more than one page. Another good example is *SearchPad* [4], which enhances primary searches, but does not support ancillary searches of text or images.

Web augmentation allows the end-user to manipulate websites without needing computer skills. This allows users to collect information or manipulate the content of websites with a few mouse interactions. This is valuable since the goal of our extension is to allow all types of users to be able to perform ancillary searches. Several browser extensions use this technique, like *Marmite* [36], which allows users without programming skills to create Web components directly from existing Web pages. This is an excellent example, as the user can interact with the elements by simply selecting them with the mouse.

The extension proposed in *ANDES: an approach to embedding search services on the web browser* [5] allows end-users to create customizable search services that meet the user's needs and to perform ancillary searches without the need to open a new tab, using

techniques and approaches mentioned in this section. The user can trigger an ancillary web search from any web page by simply using the mouse and the context menu. All the information is displayed on one side of the page, so the user does not lose the context. However, the extension does not allow to perform a search on images or various elements, and it does not allow a context based on several web pages. In other words, it cannot perform an ancillary search based on several texts and images taken from different sites.

An exciting feature that our extension offers is the possibility to perform ancillary web searches based on images. A good example is *TinEye* [32], which allows you to reverse search on an image by right-clicking on it and selecting the option in the context menu, which allows the user to collect information about its origin, license, and other attributes. However, the extension allows only a single image search, and the search result is displayed in another tab, thus losing the context. In addition, the extension does not allow ancillary searches, only information about the image itself.

3 Methodology

3.0.1 RQ1

To answer the first RQ, it was first necessary to determine which areas browser extensions based on primary and ancillary searches had room for improvement. Doing some research made it possible to realize that there are many extensions focused on text and keyword search, and others more focused on image search. From the address https://chrome.google.com/webstore/category/ext/38-search-tools?hl=en it is possible to find different browser extensions focused on the user's searches. Just browsing a little (Figure 1) makes it possible to find numerous browser extensions focused on primary searches.

There are numerous articles and videos on the internet on how to make browser extensions, so it was not a big problem to get familiar with them. Everything is written in Javascript, and the browsers offer an extension developer environment that makes installation, debugging, testing, and uninstalling very easy.

In order to develop a piece of software, it is advisable first to determine in which scenarios it can help. Therefore, before starting to code, it needed to be established in which scenarios the user might be interested in using the application. *The Evaluation Strategies for HCI Toolkit Research* [19] document shows good examples of scenarios for different applications and software, like *Retrofab* [26]. and *Pineal* [20].

3.0.2 RQ2

Once the software was successfully developed, it was time to proceed with RQ2. To answer this RQ satisfactorily, it was necessary first to study how to test software on different users to know how useful and efficient it is. The paper *ANDES: an approach to*

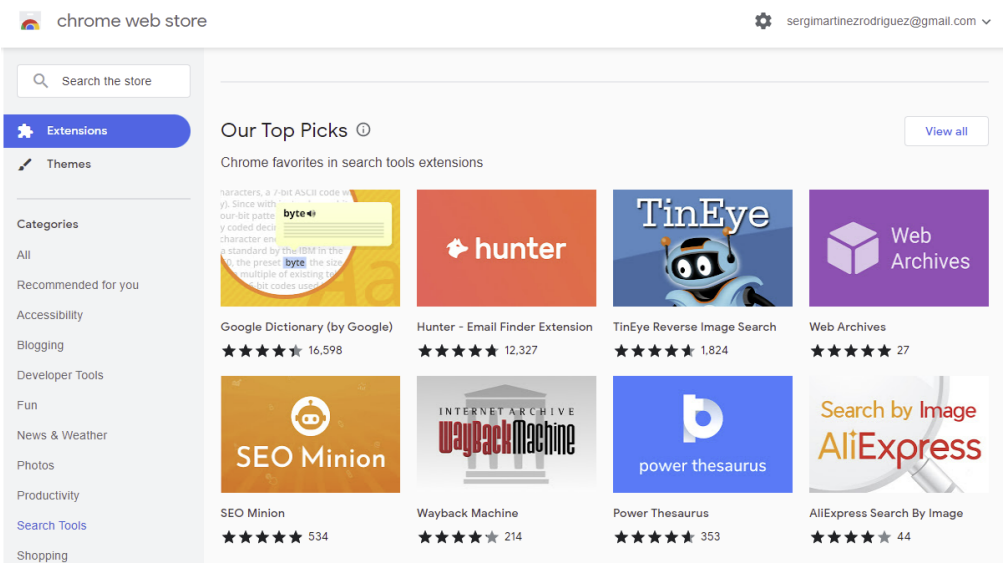


Figure 1. Chrome Extension Marketplace

embed search services on the Web browser [5] shows how to test a browser extension on different users to get statistics on usability, efficiency, and satisfaction. This paper also gives good examples of questions to ask the user beforehand and uses a vital framework to measure the user's satisfaction. The System Usability Scale [28] is an excellent framework to test such. Therefore, it was necessary to study this tool and understand its use. It is a widely known tool, so it was not difficult to find enough documentation. Figure 13 shows the set of questions of this framework.

3.1 The tool

After studying where browser extensions based on primary and ancillary searches could be improved, we concluded that it would be interesting to make an extension capable of performing ancillary searches on a group of keywords and images. This extension would allow the user to find and add different texts and images to the extension, click the search button, and receive an ancillary search result based on all the added elements. If the user wanted to modify the search, he would go back, change the keywords or images, and perform the search again. In addition to web search, the extension should also allow image search. This means that if the user wanted to, he could get images related to the search instead of web pages.

Today, there are numerous web browsers available. They are free and can be easily installed on any user's personal computer. Each browser has its own engine, so an extension that works in X browser will not work in Y browser if it is based on a different engine. This is why, for the sake of this project, we decided to base this project on

Table 1. Browser market share

Browser name	Share
Chrome	69.28%
Edge	7.75%
Firefox	7.48%
Internet Explorer	5.21%
Safari	3.73%
QQ	1.96%
Sogou Explorer	1.73%
Opera	1.12%
Yandex	0.90%

the Chromium engine. Chromium is a free and open-source engine used by many web browsers, such as Google Chrome, Microsoft Edge, Samsung Internet, and Opera. Almost 78% of users use these browsers. Table 1 shows the market share of web browsers [8].

3.1.1 Design of the interface

Since the user should not want to spend too much time using the extension, we designed a straightforward interface. A small popup accessible from the toolbar can give access to all the software's functionalities. We used a small rectangle for the base as the popup size, which contained all the necessary buttons and inputs. The first part of the design consisted of creating the context menu. A pretty easy part since the context menu only allows the user to add photos and images. Therefore, only two menus had to be designed, as shown in Figure 2.

Much work had to be done regarding the popup to achieve a clean interface allowing the user to move through all the functionalities quickly.

The design allows the user to switch between several outputs after the execution of the search. The options are webs, images, and hard and soft connections. Hard connections are found using the AND operand, and soft connections using the OR operand. If he wants to change the elements, either the images or the keywords, he has to go back, modify the elements, and perform the search again. The modification of the elements is simple, the chips containing the keywords are removable, and the images are as well. In case there is more than one image, the set of images is displayed in a convenient carousel, allowing the deletion of the images by clicking on the corresponding cross. The design of the popup is shown in Figure 3.

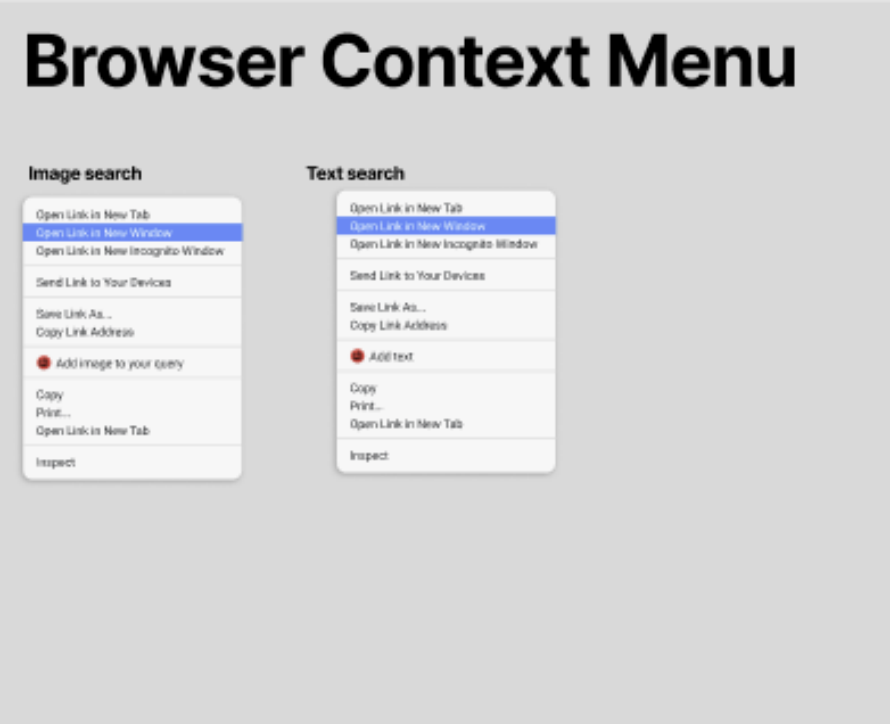


Figure 2. Context Menu Design

Finally, for the results, all the elements of the search result were placed in the popup. Additionally, it was decided to allow the user to switch between web and image search with simple tabs and AND operand searches with a checkbox, as seen in Figure 4.

3.1.2 Frontend

The frontend consists of two parts: a popup and a background file. The popup creates a bubble accessible from the toolbar, which the user can access by clicking on the extension icon. Once the popup has appeared, the user can interact with all the functionalities of the software. As soon as a new tab is opened, a bookmark is removed, or a bookmark is closed, the background file monitors these events and reacts accordingly. Among other options, the background file allows adding new options to the context menu. For the sake of this project, the context menu was modified, adding an option to add keywords and images to the query.

Chromium browser extensions are written in Javascript, so for the development of this software, it was decided to use Typescript. Typescript is a language that has been gaining a lot of popularity [29] in the last few years, and it allows the use of static types, classes, and optional interfaces. One of the great benefits is that it allows IDEs to provide a more prosperous environment for detecting common errors as the user writes code.

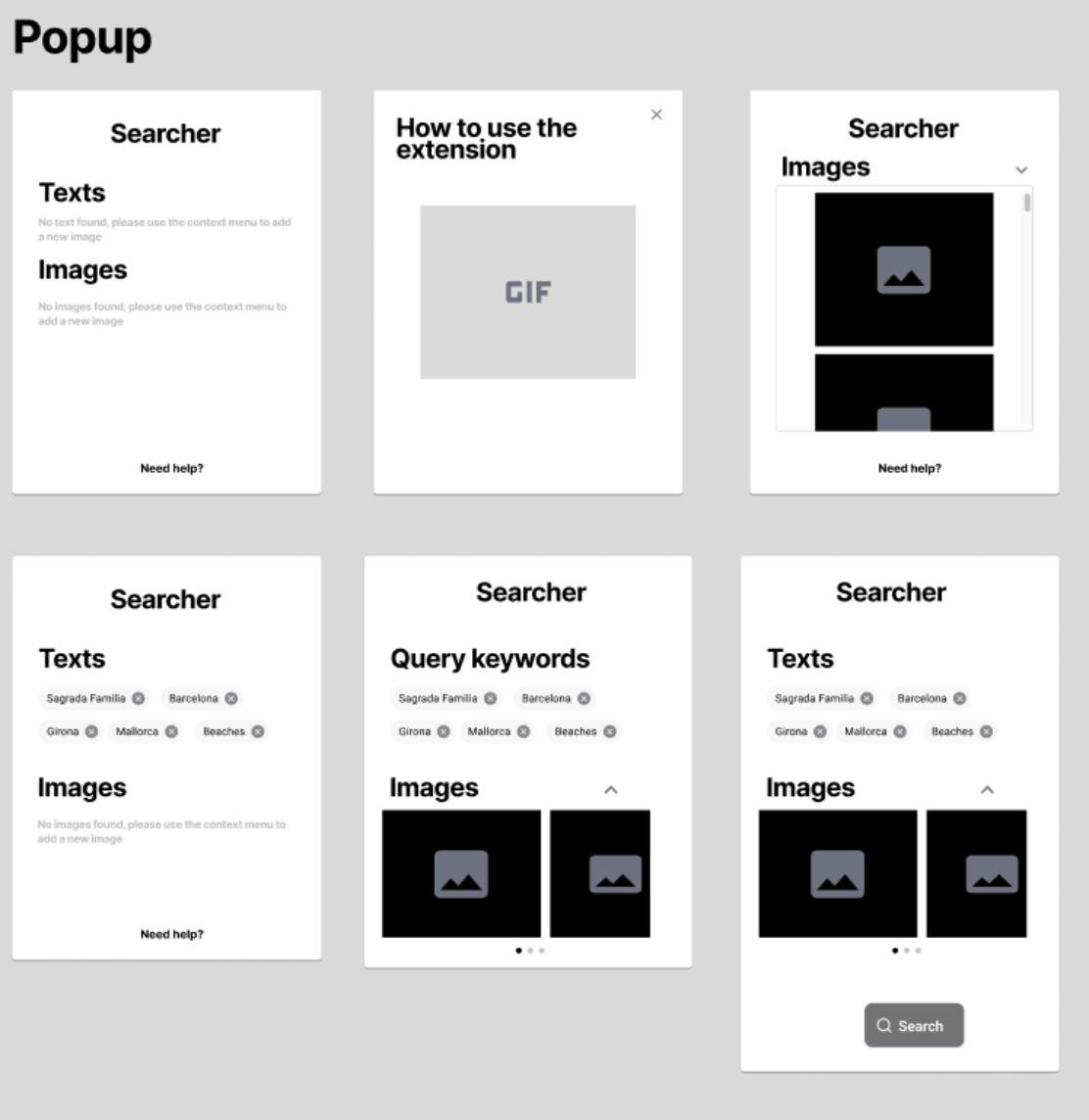


Figure 3. Popup Design

In summarizing, it makes coding and debugging much more straightforward. Browser extensions contain a lot of boilerplate that is quite cumbersome to write, so by doing an extensive search, a boilerplate code template [9] for open source browser extensions were found and used.

Additionally, the Material UI component library [21] was used. This library contains dozens of components that work out of the box and are easily customizable. Figure 5 displays a few examples of components in Material UI.

The Material UI is straightforward to install in any application and is extensively

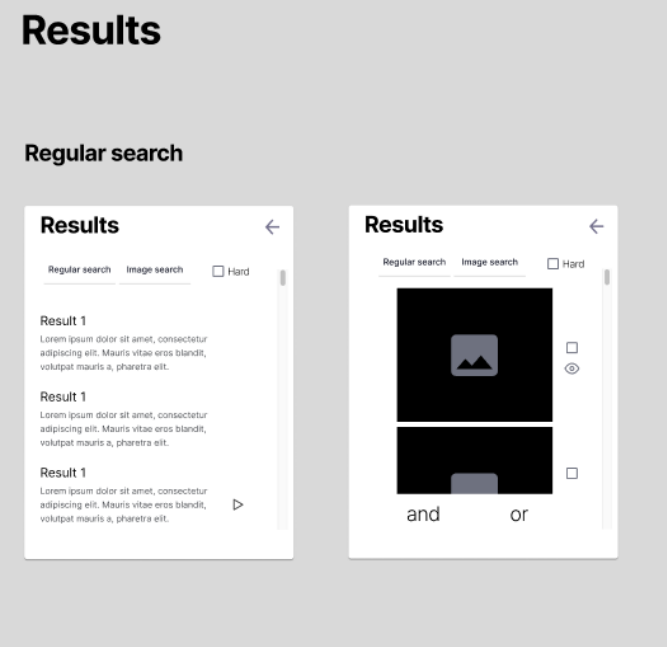


Figure 4. Results Screen Design

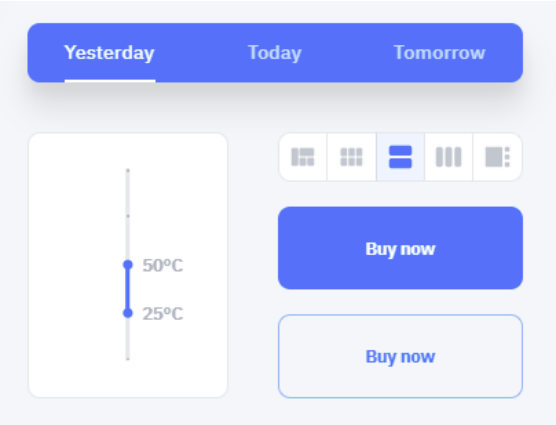


Figure 5. Material UI Components examples

documented on the internet.

3.1.3 Backend

The backend is a service deployed in the cloud, responsible for receiving requests from the client and making the relevant calls to external APIs. For this part of the project, the framework NestJs [16] was used. NestJs is open-source, very versatile, and compatible

with Typescript.

The most relevant files in the backend are a controller file and a service file. The controller is in charge of receiving incoming requests and returning responses to the client. On the other hand, the service file contains all the logic that the program executes once an endpoint has been called. All the necessary documentation to understand how the service files [25] and controllers [10] work can be found in the NestJs documentation. The only endpoint in the application is */search*. This endpoint is in charge of communicating to the backend all the keywords and images to be used for the ancillary search. Listing 1 shows an example of a call made to the backend, and Figure 6 is an architectural diagram of the full software.

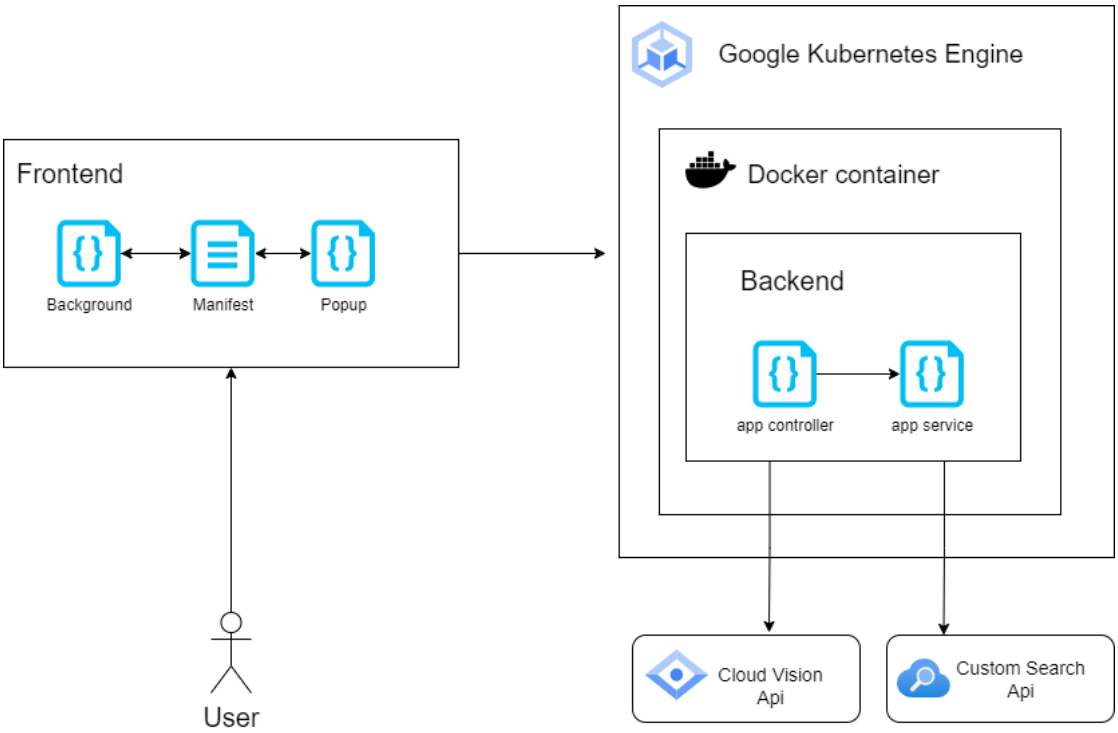


Figure 6. Architectural diagram

This packet contains all the keywords and images the user has added to the query. Once the backend has received them, it will communicate with the services to obtain the resolution of the query, which will be displayed in the popup.

3.1.4 Services

Our software uses two Google APIs in order to work. This section will explain these APIs, how they have been configured, and the function they perform.

```
{
    "keywordList": ["Barcelona", "Sagrada Familia"]
    "imageList" : ["www.barcelona.cat/imatge1.jpg"]
}
```

Listing 1. JSON example for post request to the Backend

```
{
    "cx": "8df08b19b90eaa60d",
    "q": ["Barcelona", "Sagrada Familia", "Casa Batlló"]
}
```

Listing 2. JSON example for post request to the Custom Search API

Custom Search API The Custom Search API [11] allows us to perform a search in the network using JSON, which means that we can make a POST containing the parameters of our search, and we will receive a JSON file with all the results. An example of a POST request would be the following: 2, where *cx* is the Google Custom Search Engine (previously created using the Google Cloud Interface), and *q* the search itself. To make these requests from our backend, we used Google's open source library: Google APIs Node.js Client [13]. This library allows us to make calls to the different Google APIs simply. We only have to link our file with the Google keys, and it is ready.

This request will return the following result: Listing 3.

The original file returned by the API is huge. It contains much information that is not important to us. The only elements we need to ensure the operation of our software are those shown in the figure: The title of the site, the link, and the description.

Cloud Vision API This API [12] allows us to extract tags and landmarks from any image. Therefore, the user can add any image in the query, and the service will take care of solving for us what exactly is in that image. If the image corresponds to a landmark, the service will return precisely what landmark it is. If the image is not a landmark, the service will return a list of tags corresponding to the objects shown. A POST request to the Cloud Vision API would look like Listing 4, and the response would look like Listing 5.

```
{
   "items":[
      {
         "title":"Mercat de Santa Caterina
         | Barcelona, Spain | Sights - Lonely Planet",
         "link":"https://www.lonelyplanet.com
         /spain/barcelona
         /la-ribera/attractions
         /mercat-de-santa-caterina/a/poi-sig
         /1204433/1320668",
         "snippet":"Mercat de Santa Caterina ... Santa
         Caterina
         market in the streets of Barcelona.
         ... resemblance to that of the
         Escoles de Gaudí at La Sagrada Família."
      },
      {
         "title":"Santa Caterina Market (Mercat de
         Santa Caterina)
         - What To Know ...",
         "htmlTitle":"\u003cb\u003eSanta Caterina
         \u003c/b\u003e Market
         (\u003cb\u003eMercat de Santa
         Caterina\u003c/b\u003e)
         - What To Know ...",
         "snippet":"How to Choose a Sagrada Familia Tour.
         Find the best way to visit one of
         Barcelona's top landmarks."
      },
      {
         "title":"List of markets in Barcelona -
         Wikipedia",
         "link":"https://en.wikipedia.org/wiki/
         List_of_markets_in_Barcelona",
         "snippet":"1 La Boqueria ·
         2 Mercat de Sant Antoni · 3 Mercat de
         Santa Caterina · 4 Mercat
         de la Barceloneta ·
         5 Mercat de la Concepció · 6 Mercat
         del Ninot · 7 Mercat d' ..."
      }
   ]
}
```


Listing 3. JSON example for post request response from the Custom Search API

```
{
    "imageList" : ["www.barcelona.cat/imatge1.jpg"]
}
```

Listing 4. JSON example for post request to Cloud Vision API

```
{
    "result": ["Mercat de Santa Catarina"]
}
```

Listing 5. JSON example result from Cloud Vision API

3.1.5 Deployment

To deploy the application correctly, two aspects had to be taken into account: the frontend and the backend. For the frontend, it was simple since it was each one of the users who installed it manually on their respective browsers. Installing a browser extension is very simple since today's browsers make the whole process straightforward.

The backend deployment was a bit more complicated since it has to be in the cloud so that it is always accessible from the client. Since the backend makes calls to paid APIs, it is essential to have full server access and shut it down when there is no experiment going on. A container was created using Docker. This container contains all the necessary code to run the application and the necessary dependencies. To host this container, we used Google Kubernetes Engine [14], a Google platform that allows easy hosting and management of containers. To get the NestJs application working in a Docker container, we followed the instructions of an article written by Tom Ray [34].

3.2 Demonstration of the tool

The objective of this section is to show the functionalities of the software employing a scenario. According to the paper *Evaluation Strategies for HCI Toolkit Research*, [19], Scenarios break down tasks into individual steps that demonstrate the workflow, showing the results of each step. The example also shows a complete flow, from finding the problem to solving it using the software.

3.2.1 Scenario

Let us pretend the user is using the browser, trying to find out his next traveling destination. At some point, he/she comes across several keywords which seem interesting to him/her. Not only that, there are few images which contain exciting monuments. Having encountered a few interesting keywords and images, the user would like now to perform an ancillary search containing all of them. Of course, he/she could manually enter the keywords in the Omnivar and perform a search, but what about the images? The only way for the user to enter the images into the Omnivar is first to know what the images are about and then be able to enter them into the search bar. Additionally, the user would lose part of the context by switching from all the tabs and copy-pasting elements into the Omnivar.

This is where our software comes in. The user can execute an ancillary search, including all the selected keywords and images. If the search contains images, the software itself will extract as much information as possible from the images in order to perform the most accurate search possible. In addition, the user can select between a loose search (using OR operators) or a strong search (using AND operators).

3.2.2 Resolution using the software

The software allows the selection of text and images to be put together in a single ancillary search. It takes care of putting all the terms together, performing the search, and displaying the results, so that the user can select which of them are relevant to him or her.

Once the user has come across an exciting keyword or image, he/she can easily select it using the Context Menu, as shown in Figure 7.

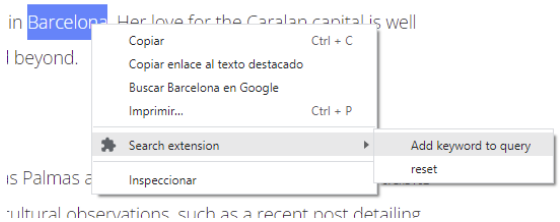


Figure 7. Using the context menu to add a keyword)

Furthermore, the user can add an image to the query using the same procedure, as shown in Figure 8.

Once the user has added the keywords and images he/she is interested in, he/she can access the popup menu. This menu is easily accessible from the toolbar 9 in the extensions section. The user may need to enable this extension menu from the browser options.

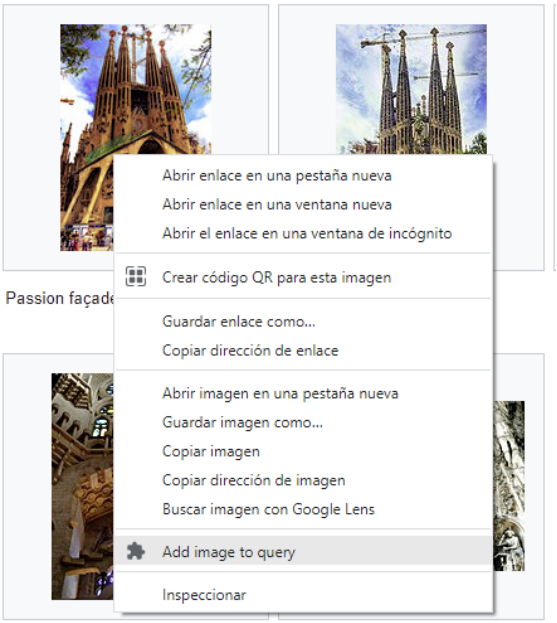


Figure 8. Using the context menu to add an image

Figure 9. The extensions menu containing the Searcher extension in Google Chrome

The Popup (Figure 10 and Figure 11) shows the user all the keywords and images he has selected, with the option to delete those that are no longer of interest and to perform the ancillary search once the user is happy with the selection of keywords and images.

A request will be sent to the backend containing all the keywords and images just by clicking the Search button. This backend will communicate with different services to obtain a search result that will be displayed to the user, with the possibility of navigating around the websites he finds most attractive.

3.2.3 Results

Once the server manages to perform the search, the results will be displayed in the popup 12. The interface shows in a list the different results (the user can access them by clicking on them), both from web pages and images. Users can now navigate the different options by clicking on the respective tabs and buttons. These options are webs and images, and each one of these uses AND and OR connections.

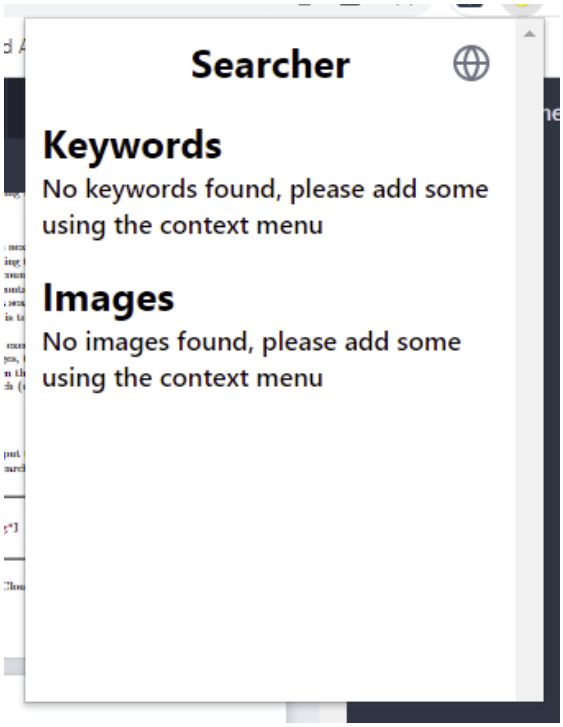


Figure 10. The extension interface

4 Evaluation and Discussion

In order to measure how successful the software is, an experiment was performed on different users. This experiment allowed us to measure the effectiveness and the efficiency, along with the user's satisfaction after using the tool. This section will explain how the experiment was designed and performed on several users. The experiments were conducted via a Google Meets call, with the interviewee showing their screen at all times. The first part of the interview consisted of asking a set of simple questions, as well as collecting some necessary personal data, such as occupation and studies.

The evaluation involved 12 participants, ranging in age from 20 to 27, of Spanish nationality and different industries and specializations. All users use the browser at different times of the day, some a few hours a day, and others for more than 8 hours, as it is a significant part of their work. Although not all use browser extensions daily, they all say they have used a browser extension at some time. Regarding the context menu, most say they use it, usually for copying and pasting pieces of text. Two of them say they use only Windows shortcuts. Omnivar is quite popular among the users since only one person says he does not use it. This person prefers to search by clicking on a new tab and performing the search from the Google page that opens automatically. In general, all users are familiar with the browser and browser extensions.

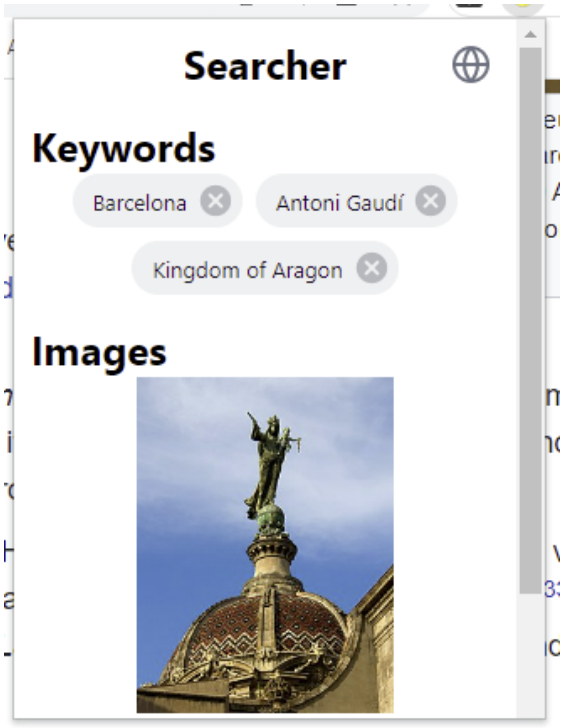


Figure 11. The extension interface

Before starting the experiment, we asked the users a few questions about the use of web browsers and browser extensions. The questions were the following:

1. How often you use the browser a day?
2. For how long you been using a browser?
3. Do you use the Omnivar?
4. Do you user search plugins?
5. Do you use the browser context menu?

4.1 The experiment

Once the questions were answered, it was time for the experiment to start. First, we taught the users how to use the extension through a simple demo. In this demo, we presented a simple scenario where a person tried to perform an ancillary search of several words and images. The demo showed how to add the words and images in the extension, perform the search and browse through the results. In the end, it showed how to perform

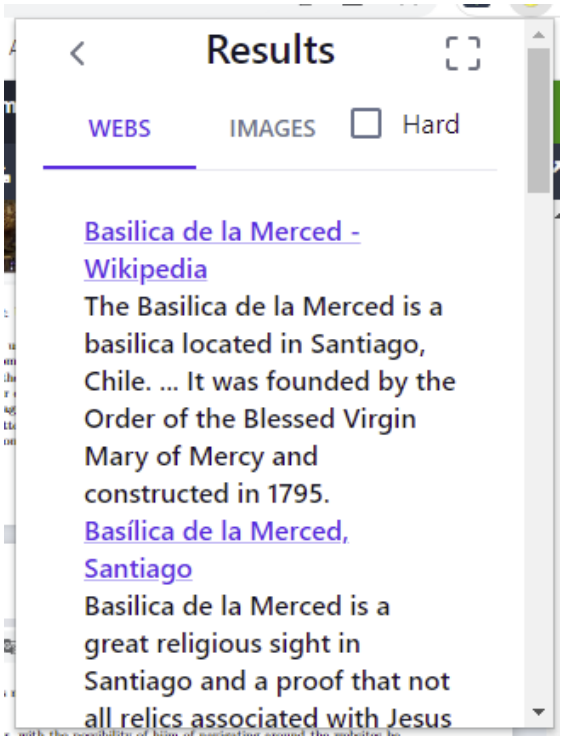


Figure 12. The results of the search

a search again if the user was unsatisfied with the results. Once the demo was visualized and the user understood how to use the tool, it was time to experiment.

The experiment performed an ancillary search of 4 elements: two keywords and two images. Those keywords were the names of the capital cities of Riga and Warsaw, and the images were the Latvian Academy of Sciences observation platform in Riga and the Palace of Culture and Science in Warsaw. The objective of this experiment was to navigate around several websites containing these elements (all provided to the user), find them, and add them to the query. The elements the users were expected to find are found on these websites:

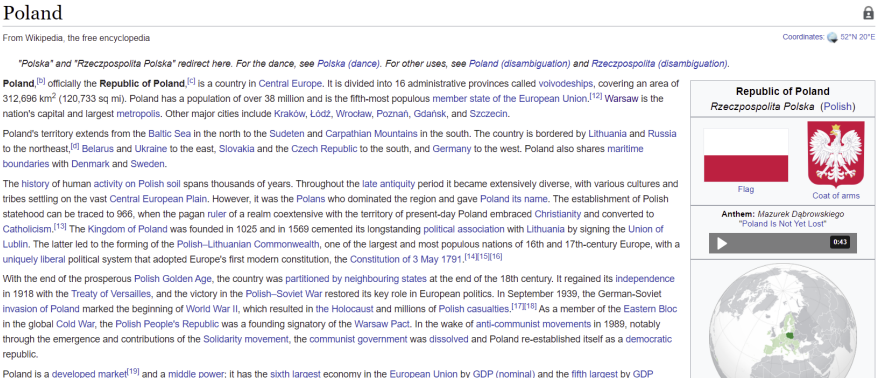
1. https://en.wikipedia.org/wiki/Poland [24]
2. https://en.wikipedia.org/wiki/Latvia [18]
3. https://www.traveller.ee/blog/tallinn/things-to-do-in-riga-the-essential-need-to-know-guide/ [31]
4. https://theculturetrip.com/europe/poland/articles/the-top-10-things-to-do-and-see-in-warsaw/ [30]

System Usability Scale Questionnaire	Strongly Disagree				Strongly Agree
1. I think that I would like to use this product frequently.	1	2	3	4	5
2. I found the product unnecessarily complex.	1	2	3	4	5
3. I thought the product was easy to use.	1	2	3	4	5
4. I think that I would need the support of a technical person to be able to use this product.	1	2	3	4	5
5. I found the various functions in the product were well integrated.	1	2	3	4	5
6. I thought there was too much inconsistency in this product.	1	2	3	4	5
7. I imagine that most people would learn to use this product very quickly.	1	2	3	4	5
8. I found the product very awkward to use.	1	2	3	4	5
9. I felt very confident using the product.	1	2	3	4	5
10. I needed to learn a lot of things before I could get going with this product.	1	2	3	4	5

Figure 13. System Usability Survey

Figure 14. Website containing the keyword Riga

When all the elements are in the query, the user can execute the search and analyze whether the results are correct or not. The experiment ended if the search results correctly showed the connection between the elements. If not, the user could keep adding or removing elements from the query until the search was satisfied. For this experiment,



Poland

From Wikipedia, the free encyclopedia

Coordinates: 52°N 20°E

"Polska" and "Rzeczpospolita Polska" redirect here. For the dance, see Polska (dance). For other uses, see Poland (disambiguation) and Rzeczpospolita (disambiguation).

Poland,[b] officially the **Republic of Poland**,[c] is a country in Central Europe. It is divided into 16 administrative provinces called voivodeships, covering an area of 312,696 km^2 (120,733 sq mi). Poland has a population of over 38 million and is the fifth-most populous member state of the European Union.[12] Warsaw is the nation's capital and largest metropolis. Other major cities include Kraków, Łódź, Wrocław, Poznań, Gdańsk, and Szczecin.

Poland's territory extends from the Baltic Sea in the north to the Sudeten and Carpathian Mountains in the south. The country is bordered by Lithuania and Russia to the northeast,[d] Belarus and Ukraine to the east, Slovakia and the Czech Republic to the south, and Germany to the west. Poland also shares maritime boundaries with Denmark and Sweden.

The history of human activity on Polish soil spans thousands of years. Throughout the late antiquity period it became extensively diverse, with various cultures and tribes settling on the vast Central European Plain. However, it was the Polans who dominated the region and gave Poland its name. The establishment of Polish statehood can be traced to 966, when the pagan ruler of a realm coextensive with the territory of present-day Poland embraced Christianity and converted to Catholicism.[13] The Kingdom of Poland was founded in 1025 and in 1569 cemented its longstanding political association with Lithuania by signing the Union of Lublin. The latter led to the forming of the Polish–Lithuanian Commonwealth, one of the largest and most populous nations of 16th and 17th-century Europe, with a uniquely liberal political system that adopted Europe's first modern constitution, the Constitution of 3 May 1791.[14][15][16]

With the end of the prosperous Polish Golden Age, the country was partitioned by neighbouring states at the end of the 18th century. It regained its independence in 1918 with the Treaty of Versailles, and the victory in the Polish–Soviet War restored its key role in European politics. In September 1939, the German-Soviet invasion of Poland marked the beginning of World War II, which resulted in the Holocaust and millions of Polish casualties.[17][18] As a member of the Eastern Bloc in the global Cold War, the Polish People's Republic was a founding signatory of the Warsaw Pact. In the wake of anti-communist movements in 1989, notably through the emergence and contributions of the Solidarity movement, the communist government was dissolved and Poland re-established itself as a democratic republic.

Poland is a developed market[19] and a middle power; it has the sixth largest economy in the European Union by GDP (nominal) and the fifth largest by GDP

Republic of Poland
Rzeczpospolita Polska (Polish)

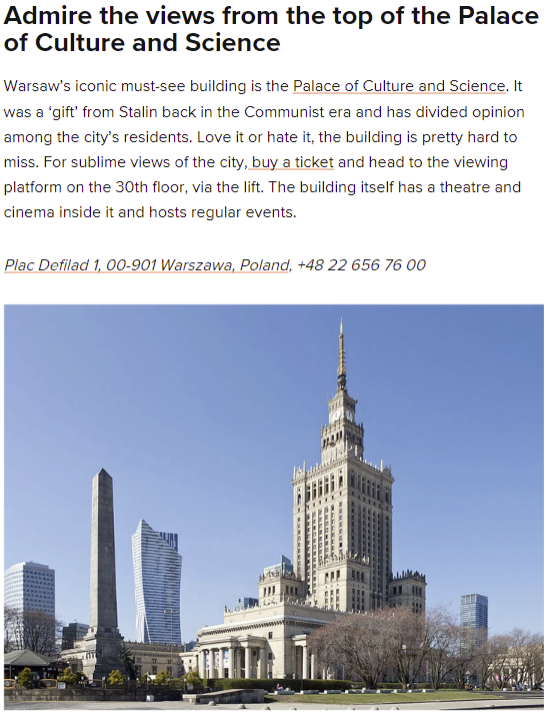
Flag

Coat of arms

Anthem: *Mazurek Dąbrowskiego*
"Poland Is Not Yet Lost"

0:43

Figure 15. Website containing the keyword Warsaw



Admire the views from the top of the Palace of Culture and Science

Warsaw's iconic must-see building is the Palace of Culture and Science. It was a 'gift' from Stalin back in the Communist era and has divided opinion among the city's residents. Love it or hate it, the building is pretty hard to miss. For sublime views of the city, buy a ticket and head to the viewing platform on the 30th floor, via the lift. The building itself has a theatre and cinema inside it and hosts regular events.

Plac Defilad 1, 00-901 Warszawa, Poland, +48 22 656 76 00

Figure 16. Website containing and image of Palace of Culture and Science

we say that the search has been successful when the first three search results already show the relationship between the elements. Figure 18 shows an excellent example of good results for this experiment. It is possible to see how these web pages connect these towers and their collective history.

The time the user takes to achieve the results is timed, and any exciting behavior the user may have is noted. The Table 2 displays the answers to the questions asked at the

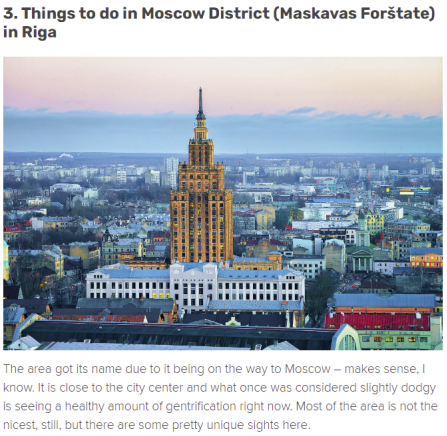


Figure 17. Website containing an image of The Academy of Sciences

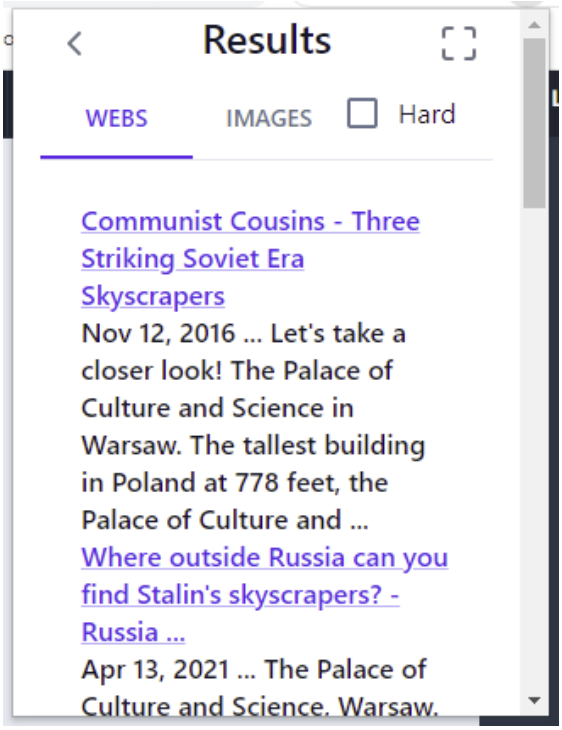


Figure 18. Popup showing a valid result for the sake of the experiment

beginning, plus the total times the user took to complete the experiment with and without the tool.

Table 2. Results table

nº	How often do you search	Use of Search extensions	Use of Context menu	Use of Omnivar	First or later	Time without the extension (seconds)	Time with the extension (seconds)
1	10h/day	No	Yes	Yes	Later	312	68
2	12h/day	No	Yes	Yes	Later	219	68
3	8h/day	No	Yes	Yes	First	318	218
4	2h/day	No	Yes	No	First	240	206
5	6h/day	No	No	No	Later	154	51
6	7h/day	No	Yes	No	First	109	73
7	4h/day	No	Yes	No	Later	440	92
8	3h/day	No	Yes	No	Later	258	107
9	1h/day	No	No	No	First	411	405
10	9h/day	Yes	Yes	Yes	First	201	69
11	3h/day	No	Yes	No	First	307	201
12	7h/day	Yes	No	Yes	Later	302	138

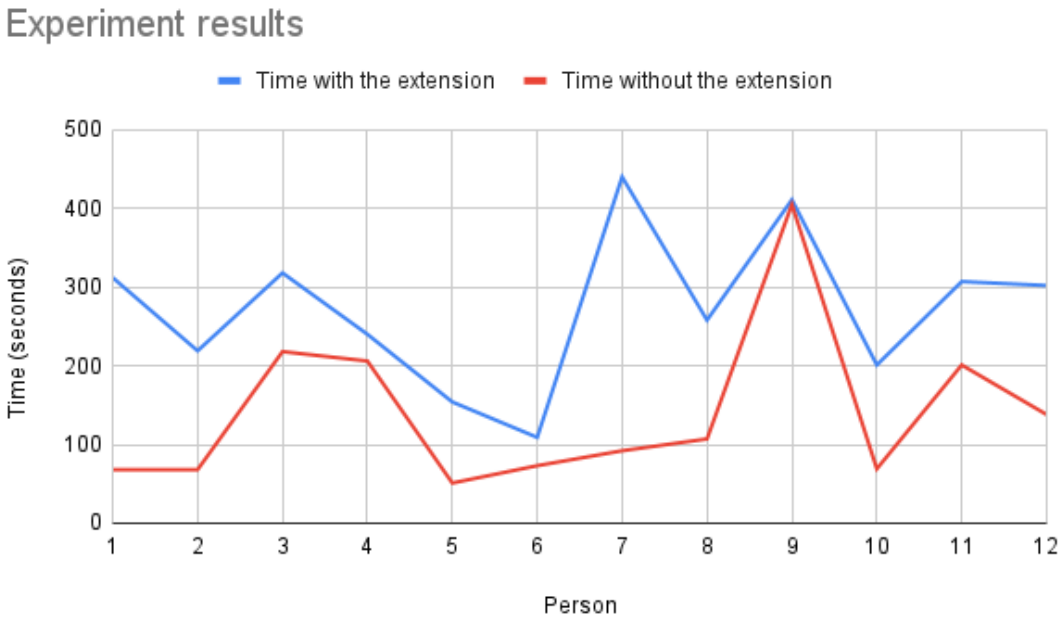


Figure 19. Line graph showing the results

nº	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
1	4	1	4	2	4	2	5	2	4	1
2	5	1	5	1	5	1	3	1	4	2
3	1	2	4	2	5	3	4	2	4	2
4	1	1	4	3	4	3	5	2	4	1
5	2	1	5	1	4	2	5	2	2	1
6	2	1	5	1	4	1	5	2	5	1
7	4	2	4	3	4	2	4	1	5	1
8	2	2	4	1	4	2	4	2	4	2
9	3	1	3	3	4	3	3	1	5	3
10	4	2	4	2	5	2	3	3	5	1
11	3	2	3	3	3	4	4	2	3	3
12	4	1	3	1	4	2	2	3	3	4

Table 3. SUS survey results

4.1.1 Efficiency

In order to measure the efficiency, the time users needed to conclude the experiment was measured using a chronometer and noted in Table 2. In the line graph from Figure 19, it is possible to observe the difference in time between using and not using the tool amongst all the users. As a result of the experiment, it is evident that using the extensions reduces the time spent performing ancillary searches. This suggests that the extension could reduce the time spent performing ancillary searches for all users since the sample is quite diverse regarding their studies and professions.

4.1.2 Satisfaction

In order to measure the satisfaction of the user after using the tool, we used the System Usability Survey tool. It consists of a questionnaire of ten questions, which each answer goes from *strongly agree* to *strongly disagree*. Each of the answers is rated from 1 to 5, so at the end of the survey, an actual score can be used to determine the user's satisfaction. Based on research, a SUS score above 68 would be average, and anything below would be below average. Table 3 shows the results of this survey, where one means strongly disagree, and five means strongly agree.

Once all the samples have been collected, it was possible to calculate the score of each user, and the average among all the users to obtain the final score. The table 4 collects the scores of all users and calculates the total average, which is 74.2. The paper *Determining what individual SUS scores mean: Adding an adjective rating scale* [3] makes a good study of how to represent SUS Survey scores. According to the author, a

nº	1	2	3	4	5	6	7	8	9	10	11	12	**Average**
Score	82.5	90	67.5	70	77.5	87.5	80	72.5	67.5	77.5	55	62.5	**74.2**

Table 4. SUS survey results after calculation

score between 70 and 85 is considered good.

5 Analysis and results

The following section explains the project results based on the research questions.

5.1 RQ1

Indeed, it can improve the user experience when doing ancillary searches. Browser extensions offer many possibilities, and although there are already numerous extensions on the market, there is still room to explore. Our extension is an example of this since, with relatively simple software, we could cut time in the execution of users' ancillary searches. Therefore, the result of this Research Question is software capable of improving the user experience when performing ancillary searches. The software is compatible with various browsers and operating systems, easy to install, and lightweight. Moreover, it is open source, so anyone who wants to participate can find the code at [27].

This extension is just an example of how to help the end-user in their ancillary searches, so it would be interesting to see with what other means this would be possible, perhaps with a web application, a desktop application, or even a native Android or IOs application to help the user in mobile searches.

5.2 RQ2

Once the experiments were concluded, it was possible to conclude that the methodology used to test the software was satisfactory so it would be interesting to consider it when carrying out other experiments with other users and software. It is common to use the SUS survey to check user satisfaction, but sometimes it is common to forget to measure efficiency, among others. The project has allowed measuring both satisfaction and effectiveness, so we can conclude that using this methodology is correct when testing a browser extension among several users.

6 Threats to Validity

This section presents the threats or challenges encountered in the research and how they were mitigated.

During the project, several conflicts arose, which were solved by investigating the problem in-depth, or looking for a workaround. The manifest file of the Chromium extensions has three different versions, each containing a different set of permissions and features, making it difficult to find a workaround on the internet while using a specific version. Currently, Chromium recommends using Manifest V3 [23], introduced in 2022, so most Stack Overflow posts about extensions based on a Manifest V2 [22] do not apply. This is a problem since there are many more posts about extensions in Manifest V2 than in Manifest V3. However, we could find solutions for all the problems between the Chromium documentation and Stack Overflow. Another conflict that was encountered during development was in the use of APIs. Before the project started, it was thought that the potential APIs to be used were free. This was not the case since these APIs are paid APIs. Both APIs contain a free plan for the first calls, which is sufficient for us, but there is a danger that someone with the extension installed will make calls outside of an experimental context, making the total cost very expensive. For this reason, we decided to set up a backend and deploy it in the cloud, which contains our API keys, and we can stop it whenever we want. This way, users could only run the extension during experiments, the only times when the backend was on. As is well known, APIs are sometimes not well documented. Fortunately, Google does a good job documenting theirs, although there is still room for improvement. In the Custom Search API, for example, we had to learn how to use it by setting random parameters and understand how it works through trial and error. This may be because it is a reasonably old API and lacks popularity. The Cloud Vision API, on the other hand, contains a well-documented article, so it was straightforward to learn to use.

Finally, when running the experiments, we had trouble finding users. Time is valuable, and many people are busy during the day during working hours. Still, with patience and getting used to everyone's schedule, all the necessary experiments could be carried out.

7 Conclusion and Future Work

7.1 Conclusion

Searching the web is a task we all perform many times every day; that is why it has been interesting to study what aspects we could help users through software, specifically browser extensions. They are easy to use and install and work on almost all devices. It was interesting to realize that users, although familiar with browser extensions, do not use them daily. Doing this project has also helped us to realize the power of browser

extensions and how they can make some tasks easier for the user. With a few lines of code, we can modify a website's content, show hidden content, or hide annoying elements, all to improve the user experience. For this reason, it would be interesting to continue researching in the field of browser extensions.

7.2 Future Work

Although the software meets all the needs to develop the project, it has some limitations, and there are some areas where it could be improved. To begin with, the extension only works in browsers written in Chromium. Chrome and Edge browsers are written in Chromium, but Mozilla and Safari are not. Right now, in 2022, it is estimated that 63% of users use Chrome, 17% Safari, 6% Firefox, and 5% Edge. This means that an essential part of the users use browsers that are incompatible with the extension, so it would be interesting to make a version compatible with them. A quick look at the Firefox documentation, for example, shows that the development of extensions is not very different from Chromium extensions. They are also written in JavaScript and contain a Manifest file where the permissions and the different files that the extension uses are defined.

Aside from cross-browser compatibility, there are other areas where further development could be pursued.

As mentioned above, the software allows searching using AND operators if the results do not satisfy the user. Currently, the algorithm is simple, and a search using AND operators commonly does not return any results. That is why it would be interesting to redo this algorithm so that this does not happen. One option would be, for example, to allow different search iterations and let the user decide to stop the iterations once he is satisfied with the search.

The Google Custom Search API allows a vast amount of parameters. This means that the user experience could be improved by allowing more options since it currently only supports web and image search, using OR and AND operators. Therefore, another interesting work would be to study the API entirely and integrate more functionalities with our software to make the experience more customizable and overall better.

The other API being served, Google's Cloud Vision, is also highly complex. Currently, our software only allows the extraction of landmarks in the images, making the overall software quite limited. This means that the image recognition only works on landmarks, so if the user decided to search with an image of a famous person, for example, it would not work.

That is why improving our software to enhance image recognition would be interesting. So the software can detect, apart from landmarks, faces, monuments, paintings, art, and all kinds of objects in general. The experiments carried out with different users gathered valuable feedback. While the interface convinced most people, others would have preferred another type of interface. For example, one user commented that he would

have preferred to view the results in a new tab rather than in the popup. The popup allows to display information conveniently, but some users might find it too small and a bit overwhelming. That is why it would be interesting to study how to improve this interface or even allow a menu of options where the user could customize the interface to his liking.

The search engine that this software uses is Google. This means that all results, both images and web pages, are extracted from the Google database. 91% of all users use the Google search engine, but it could be interesting to study if more search engines could be added. These engines are Bing, Yahoo, Baidu, and Yandex, and although they only account for 9% of total users, it might be interesting to spend some time integrating them.

In addition to search engines, specific websites with their search engines could also be considered. We are talking about sites like Wikipedia, which has a search engine to search for articles within the database. For this reason, it could be interesting to add the option of being able to choose the website where the user wants to search, and even the possibility of allowing the user to add the websites he wants. The ANDES [5] browser extension, for instance, allows adding search engines for specific websites, using only the context menu and a simple interface.

References

[1] Saeed Aghaee and Cesare Pautasso. "End-user development of mashups with naturalmash". In: *Journal of Visual Languages & Computing* 25.4 (2014), pp. 414–432.

[2] Carmelo Ardito et al. "End users as co-designers of their own tools and products". In: *Journal of Visual Languages & Computing* 23.2 (2012), pp. 78–90.

[3] Aaron Bangor, Philip Kortum, and James Miller. "Determining what individual SUS scores mean: Adding an adjective rating scale". In: *Journal of usability studies* 4.3 (2009), pp. 114–123.

[4] Krishna Bharat. "SearchPad: Explicit capture of search context to support web search". In: *Computer Networks* 33.1-6 (2000), pp. 493–501.

[5] Gabriela Bosetti et al. "ANDES: an approach to embed search services on the Web browser". In: *Computer Standards & Interfaces* 82 (2022), p. 103633.

[6] Gabriela Bosetti et al. "From search engines to augmented search services: an end-user development approach". In: *International Conference on Web Engineering*. Springer. 2017, pp. 115–133.

[7] Gabriela Bosetti et al. "From Search Engines to Search Services: An End-User Driven Approach". In: *CoRR* abs/1905.10215 (2019). arXiv: 1905.10215. URL: http://arxiv.org/abs/1905.10215.

[8] *Browser Market Share Worldwide | Statcounter Global Stats*. https://gs.statcounter.com/browser-market-share.

[9] *Chrome Extension Boilerplate with React 17 and Webpack 5*. https://github.com/lxieyang/chrome-extension-boilerplate-react.

[10] *Controllers | NestJS - A progressive Node.js framework*. https://docs.nestjs.com/controllers.

[11] *Custom Search JSON API: Introduction*. https://developers.google.com/custom-search/v1/introduction.

[12] *Detect landmarks Landmark Detection detects popular natural and human-made structures within an image.* https://cloud.google.com/vision/docs/detecting-landmarks.

[13] *Google APIs Node.js Client*. https://github.com/googleapis/google-api-nodejs-client.

[14] *Google Kubernetes Engine (GKE)*. https://cloud.google.com/kubernetes-engine.

[15] Helwig Hauser. “Generalizing focus+ context visualization”. In: *Scientific visualization: The visual extraction of knowledge from data*. Springer, 2006, pp. 305–327.

[16] *Hello, nest! A progressive Node.js framework for building efficient, reliable and scalable server-side applications.* https://docs.nestjs.com/.

[17] Christoph Hölscher and Gerhard Strube. “Web search behavior of Internet experts and newbies”. In: *Computer networks* 33.1-6 (2000), pp. 337–346.

[18] *Latvia - Wikipedia*. https://en.wikipedia.org/wiki/Latvia.

[19] David Ledo et al. “Evaluation strategies for HCI toolkit research”. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 2018, pp. 1–17.

[20] David Ledo et al. “Pineal: Bringing passive objects to life with embedded mobile devices”. In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2017, pp. 2583–2593.

[21] *Material UI - Overview*. https://mui.com/material-ui/getting-started/overview/.

[22] *Overview of Manifest V2*. https://developer.chrome.com/docs/extensions/mv2/.

[23] *Overview of Manifest V3*. https://developer.chrome.com/docs/extensions/mv3/intro/mv3-overview/.

[24] *Poland - Wikipedia*. https://en.wikipedia.org/wiki/Poland.

[25] *Providers | NestJS - A progressive Node.js framework*. https://docs.nestjs.com/providers.

[26] Raf Ramakers et al. “Retrofab: A design tool for retrofitting physical interfaces using actuators, sensors and 3d printing”. In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 2016, pp. 409–419.

[27] *Searchr | A simple tool to help end-users on their ancillary searches*. https://github.com/lmeullibre/searchr.

[28] *System Usability Scale (SUS)*. https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html.

[29] *The relevance of TypeScript in 2022 and beyond*. https://css-tricks.com/the-relevance-of-typescript-in-2022/#:~:text=According%20to%20Stack%20Overflow's%202021,and%20most%2Dawaited%20developer%20surveys..

[30] *The Top 10 Things to Do and See in Warsaw*. https://theculturetrip.com/europe/poland/articles/the-top-10-things-to-do-and-see-in-warsaw/.

[31] *Things to Do in Riga in 2022: The Complete Guide*. https://www.traveller.ee/blog/tallinn/things-to-do-in-riga-the-essential-need-to-know-guide/.

[32] *TinEye Reverse Image Search*. https://tineye.com/.

[33] Simon Tretter, Gene Golovchinsky, and Pernilla Qvarfordt. "SearchPanel: A Browser Extension for Managing Search Activity." In: *EuroHCIR* 54 (2013).

[34] *Ultimate Guide: NestJS Dockerfile For Production [2022]*. https://www.tomray.dev/nestjs-docker-production.

[35] Marco Winckler et al. "Usability aspects of the inside-in approach for ancillary search tasks on the web". In: *IFIP Conference on Human-Computer Interaction*. Springer. 2015, pp. 211–230.

[36] Jeffrey Wong and Jason I Hong. "Making mashups with marmite: towards end-user programming for the web". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. 2007, pp. 1435–1444.

I. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Sergi Martínez Rodríguez**,
(author's name)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Developing a Web Augmentation Tool for Supporting End-User Ancillary Searches,
   (title of thesis)

   supervised by Ishaya Peni Gambo and Alejandro Fernandez.
   (supervisor's name)

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Sergi Martínez Rodríguez
09/08/2022