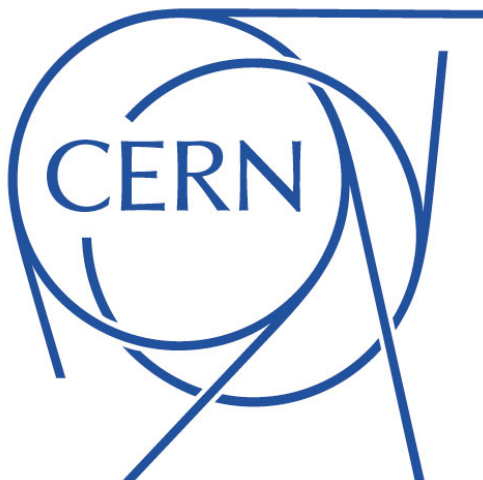




Summer Student project report

AUTHORS

Daniel Timon Viola
`daniel.timon.viola@cern.ch`

SUPERVISOR

Maria Dimou
`maria.dimou@cern.ch`

August 23, 2019

Abstract

This report outlines the main points of my work at CERN that was carried out within the boundaries of the 2019 Summer Student program. During my eight weeks stay at CERN I was given the chance to join the day-to-day life of one of the largest physics research centres IT Department. My project consisted of a prelude mini project connected to IP telephony and a main project related to front-end web development.

• • •

Contents

1	Introduction	1
2	IP Telephony	1
3	Services web page	2
3.1	Design	2
3.2	Implementation	4
3.3	IT department member synchronization script	6
4	Conclusion	7
	List of Figures	I
	References	II

1 Introduction

During my stay at CERN as a Summer Student I was lucky to participate and contribute to the work of the IT Department. Furthermore, as a Summer Student I participated in several lectures, site visits and company visits both offered by the Summer Student program and by the Openlab program.

In the followings I describe my work done at CERN during my eight weeks long stay. I will focus on my main projects and not detail smaller, ancillary tasks that are loosely related to my projects at the IT Department.

My work consisted of a prelude, mini project in relation with IP telephony services and the main project, to design and develop a new landing page for the IT Department to list all the various services provided by the department. [1].

2 IP Telephony

During the first 2 weeks of my stay at CERN I worked on the *end user and admin* documentation of the IP telephony services's new software client (Figure 1).

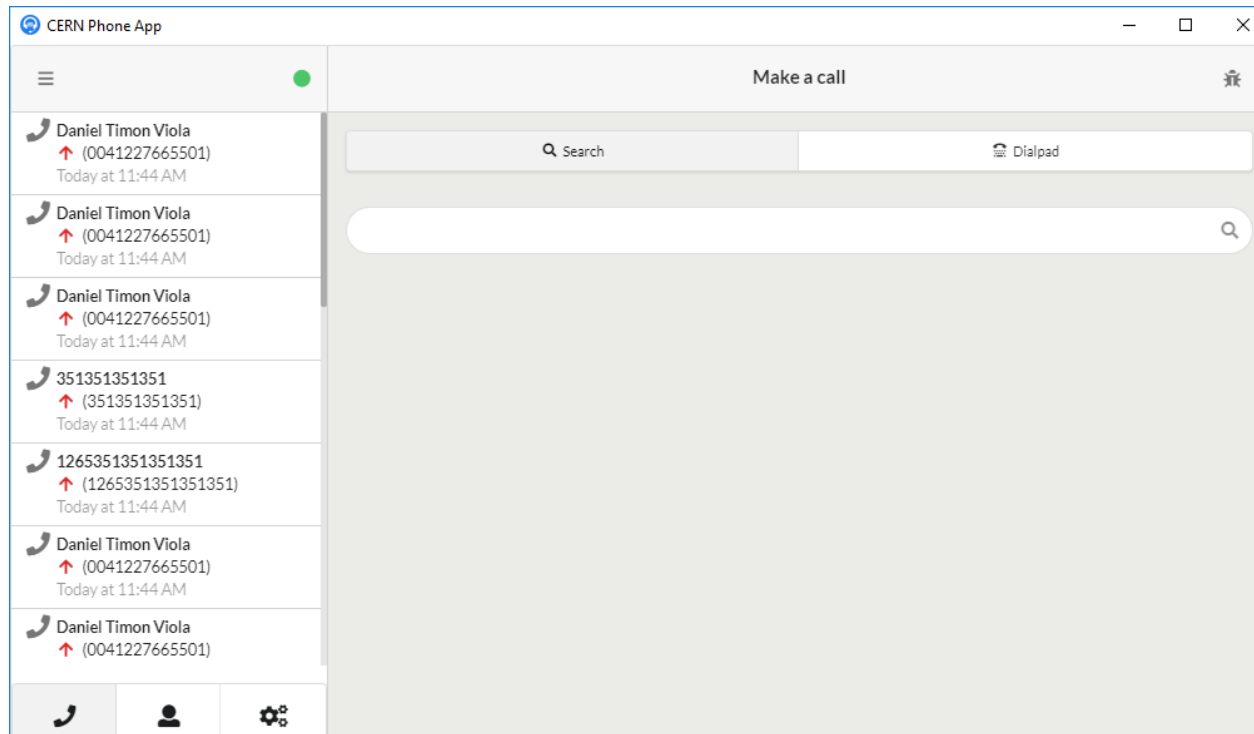


Figure 1: The new IP telephony software client

The new software client is an in-house CERN development that aims to replace other 3rd party products. My task was to test the latest version of the client and write a user's guide for the later coming production version of the software.

The documentation was written in markdown using CodiMD [2]. The documentation details the different functionalities of the software client. Especially, detailing the most relevant end user features — such as making a call, adding new contacts — with many screen shots to help the reader. The documentation can be reached on [this link](#).

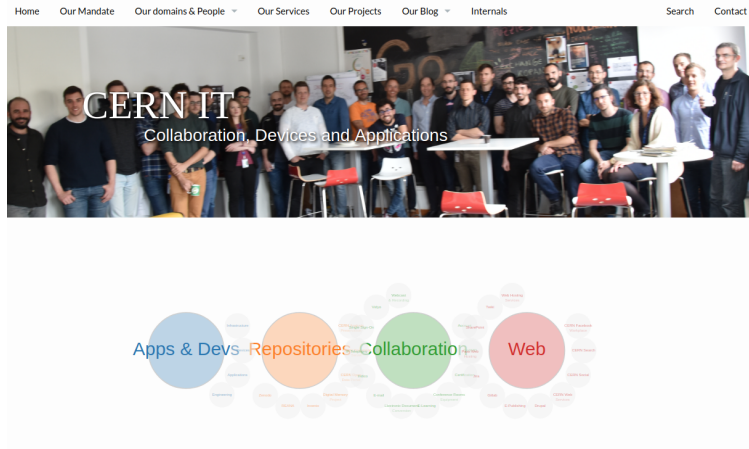
I was working closely with Thomas Baron, who gave me guidance regarding the documentation and main aspects of the project and Rene Fernandez Sanchez, who helped me to understand the technical details of the new client.

3 Services web page

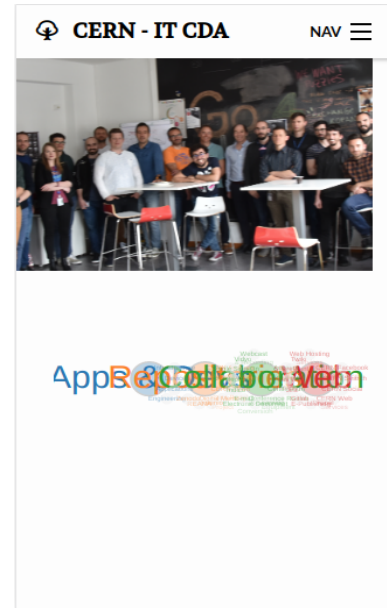
My main project during my stay at CERN was to design and develop a new web page that lists the many various services provided by the IT Department. In the followings the whole process of design, development and implementation is detailed, including discussions and early user feedback.

3.1 Design

The original design of the services page can be seen at Figure 2. The page used so called bubbles to give a logical and aesthetic overview of the various services. Meanwhile on the browser view the page rendered properly (Figure 2a), on mobile devices the implementation of the site could not preserve its aesthetic aspects (Figure 2b). Having said that, it is clear that one of the main goals of the new page was to ensure that the new page will be usable from mobile and smaller screen devices as well.



(a) Browser view



(b) Mobile view

Figure 2: The [old services page](#) (using bubbles).

To explore possible design options a number of already existing projects were considered. One of the technical aspects of the design, besides mobile view compatibility, was the ability to integrate the new page into the already existing Jekyll [3] ecosystem.

Keeping in mind that the site has to be Jekyll compatible meant some technical limitations. Thus, mainly already existing Jekyll sites were considered, a collection of Jekyll compatible websites can be found at [this site gallery](#). After considering many options and keeping usability in mind as a priority, the [HTML Reference](#) website was chosen as a main source of inspiration.

The design of the new site was done using MockFlow [4], an online UI prototyping tool. This tool proved to be useful, as it helped to clearly communicate the design choices of the new page towards colleagues. The design tool can be seen at Figure 3.

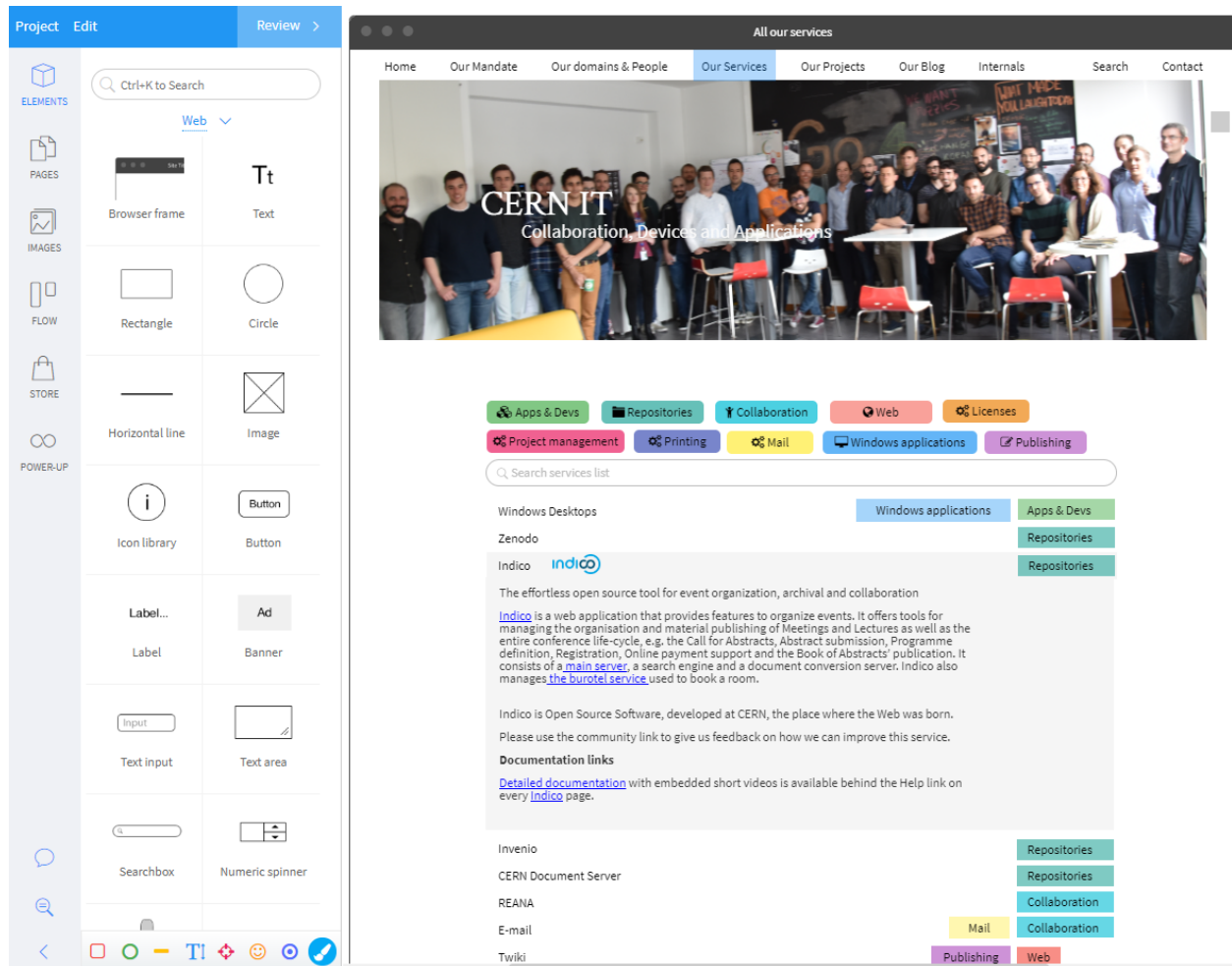


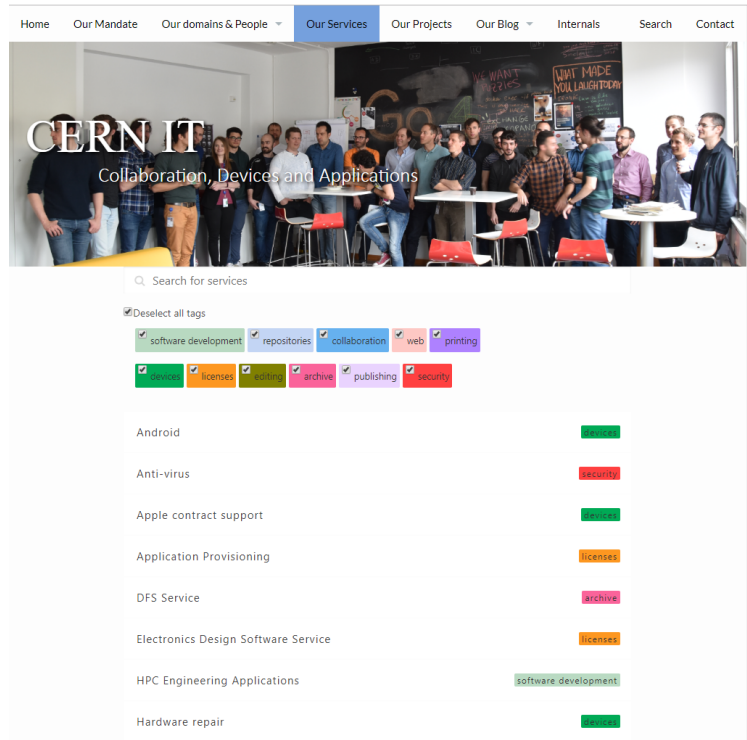
Figure 3: Mock-up design of the new services page

The design showed at Figure 3 was satisfactory and convincing, thus it was further developed as a prototype.

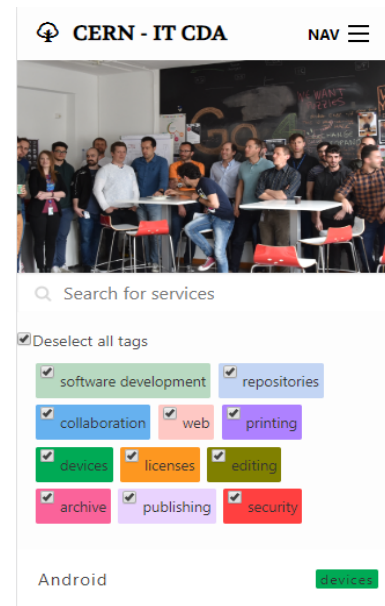
3.2 Implementation

After the successful design of the new services page the implementation and development could be started. As a first step, the page was prototyped using pure HTML, CSS and JavaScript (the main building blocks of any modern website).

This prototype page was necessary to showcase the basic usability of the new page. Furthermore, it was also used to ask first round user feedback from CERN Services Managers. The prototype page also proved to be a satisfactory usability test regarding the final implementation.



(a) Browser view



(b) Mobile view

Figure 4: The new services page .

After the prototype implementation of the page, it had to be integrated into the already existing Jekyll ecosystem. As a first step a test environment was setup by forking the original GitLab repository of the web page and creating a private project. Using this method, the production instance of the services page could be operated at the same time as the new page was implemented and tested continuously.

A number of separate approaches have been tried before finally deciding with the final structural choices of the page. The considerations were mainly made based on the fact that the Jekyll site should be kept simple in a manner that non-technical, or people with limited technical knowledge should be able to maintain the page. One of the keywords here is maintainability. Since every service page has its own separate page, this landing page had to be developed in a way that it does not require tedious copy-pasting when a new service is introduced or an already existing page is changed/removed.

Based on the above given reasons the services page was implemented using Jekyll Collections. This meant a few structural changes that had to be made in order to ensure maintainability of the site. For detailed technical description of the changes introduced see the [documentation](#) of the new page and [the implementation of the new page](#) (Unfortunately

some of the links have restricted accessibility).

Using Jekyll collections enabled the display of the service descriptions inside the accordion and made possible the seamless integration of the new services page into the Jekyll ecosystem.

Since the implementation and deployment of the page, a number of user requests and suggestions have arrived. Most of these suggestions have been further discussed and some of them were implemented on the page such as the *Select all* button (see [GitLab implementation](#)) or *changing the search mechanism of the tags* (see [the implementation of the changes on GitLab](#)).

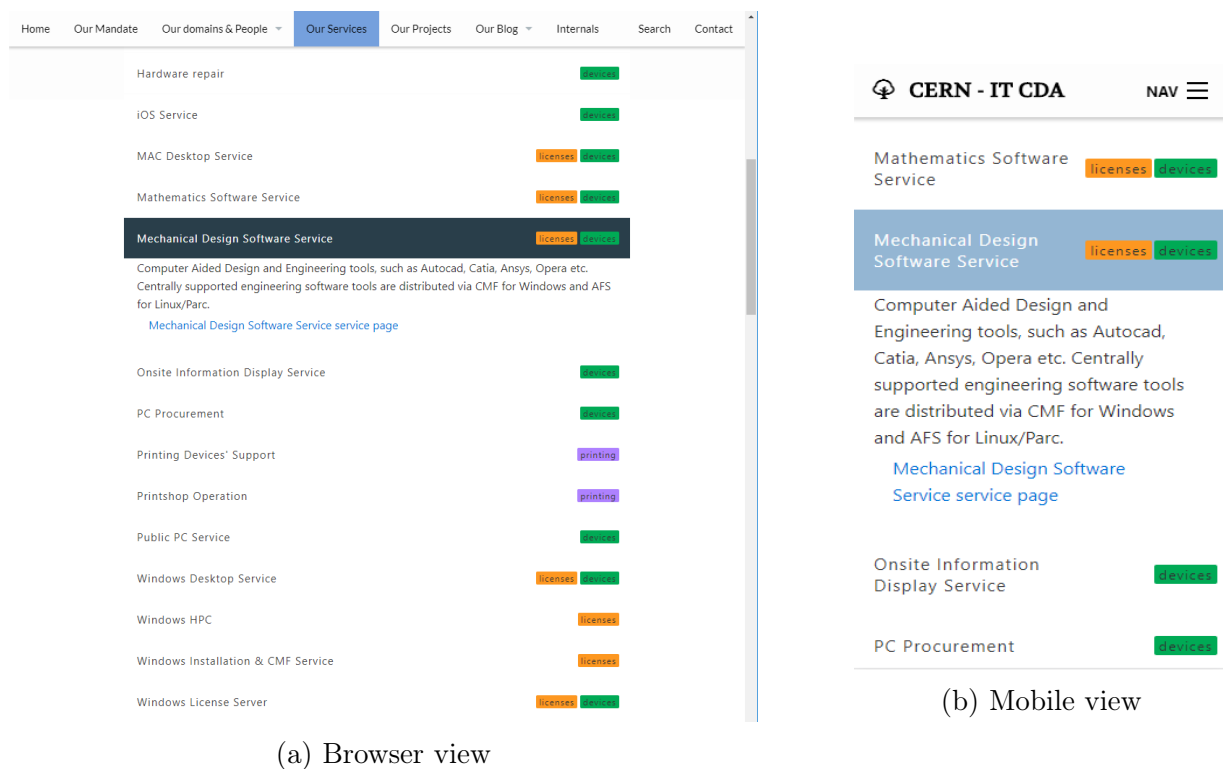


Figure 5: The new services page.

During this project I worked closely together with Maria Dimou, my supervisor. Maria helped throughout each phase of creating the new page, giving always support and meaningful insights.

3.3 IT department member synchronization script

I had the task to convert a PowerShell script to a non-Microsoft based scripting language that can be run from Linux machines. The PowerShell script retrieves data from a Microsoft

Active Directory and writes that into JSON format. The JSON file is used by the public website to show current members of the sections e.g.: [IC members](#).

Unfortunately, it was not possible to use PowerShell core on Linux [5], as the original script depends on RSAT tools and this package is not implemented to Linux based OS [6].

Two obvious implementation choices are BASH and Python script. After a short investigation BASH script was discarded as it wouldn't scale well and requires more development resources.

Python seems to be a valid choice as the latest implementation of the `ldap3` library gives a friendly API to retrieve ldap information. Also, Python is a high level scripting language, thus the script is easily maintainable according to future needs.

While implementing the original Microsoft PowerShell script to Python a number of compromises had to be made. As `xldap` does not store some of the private or sensitive attributes of users, such as *Title* and *thumbnailphoto*, these attributes could not be retrieved.

To tackle this challenge, the script connects directly to AD instead of connecting to `xldap`. A further downside of this approach is that before one can run the script, they have to authenticate themselves using `kinit` to activate their Kerberos token.

Turning the original PowerShell script into a Python script had surprisingly lot of pitfalls. Nevertheless, all these obstacles had been tackled and a Linux compatible synchronization script was developed.

I was working closely together with my supervisor, Maria Dimou on this project. Also, my colleague Mary Georgiou helped me a lot with her expertise regarding ldap.

4 Conclusion

During my 8 weeks stay at CERN I was lucky to participate in various tasks and work together with many experienced computer science and IT experts. I believe that this short introduction of my main tasks illustrates the diversity of problems that I was given. I believe that each of these exercises provided an exceptional opportunity to learn and to work together with other people.

I would like to say thank you and express my gratitude to my ever supporting supervisor Maria Dimou.

List of Figures

1	The new IP telephony software client	1
2	The old services page (bubbles).	3
3	Mock-up design of the new services page	4
4	New services page	5
5	New services page (accordion)	6

References

- [1] “Summer Student Projects description.” <http://it-student-projects.web.cern.ch/projects/malt-related-project-cda-jekyll-site-finalisation>. Accessed: 2019-08-15.
- [2] “CodiMD - Collaborative markdown notes.” <https://codimd.web.cern.ch/>. Accessed: 2019-08-19.
- [3] “Jekyll - Simple, blog-aware, static sites.” <https://jekyllrb.com/>. Accessed: 2019-08-19.
- [4] “MockFlow - Online Wireframe tools.” <https://www.mockflow.com/>. Accessed: 2019-08-19.
- [5] “Microsoft PowerShell core.” <https://docs.microsoft.com/en-us/powershell/scripting/install/installing-powershell-core-on-linux?view=powershell-6>. Accessed: 2019-08-20.
- [6] “Microsoft ActiveDirectory package.” <https://docs.microsoft.com/en-us/powershell/module/addsadministration/?view=win10-ps>. Accessed: 2019-08-20.