NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

BSc THESIS

Slides: The CERN web-based slides' maker

Aristofanis I. Chionis Koufakos

Supervisors: **Maria Dimou,** CERN IT & Academic Training
Panagiotis Stamatopoulos, Assistant Professor NKUA

ATHENS

JULY 2020

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Slides: Εφαρμογή λογισμικού στο CERN για τη δημιουργία και προβολή διαφανειών μέσω φυλλομετρητή

Αριστοφάνης Ι. Χιόνης Κουφάκος

Επιβλέποντες: **Μαρία Δήμου,** CERN IT & Academic Training
Παναγιώτης Σταματόπουλος, Επίκουρος Καθηγητής ΕΚΠΑ

ΑΘΗΝΑ

ΙΟΥΛΙΟΣ 2020

BSc THESIS

Slides: The CERN web-based slides' maker

Aristofanis I. Chionis Koufakos
S.N.: 1115201500177

SUPERVISORS: **Maria Dimou,** CERN IT & Academic Training
Panagiotis Stamatopoulos, Assistant Professor NKUA

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Slides: Εφαρμογή λογισμικού στο CERN για τη δημιουργία και προβολή διαφανειών μέσω φυλλομετρητή

Αριστοφάνης Ι. Χιόνης Κουφάκος

Α.Μ.: 1115201500177

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Μαρία Δήμου,** CERN IT & Academic Training
Παναγιώτης Σταματόπουλος, Επίκουρος Καθηγητής ΕΚΠΑ

ABSTRACT

In this thesis, I present a web-based software application for users to create presentation slides. This project is built from scratch, using only open source libraries. Using this slide-maker, users can:

- Add, edit, format and place text anywhere in a slide.
- Add, resize and position images in a slide.
- Select a CERN theme for their presentation, which can be changed at any time.
- Save their presentation as a ".slides" file.
- Upload their existing presentation (".slides" file) to continue editing.
- Present their presentation from the web browser.
- Export a ".slides" presentation, to a PDF file.

More ideas and improvements are planned to come in the future, as the project remains under active development. The app is already available for trial use from the CERN IT department. The advantage of this app is that it is open source, vendor independent, free to use and has unlimited prospects for further development.

SUBJECT AREA: software development

KEYWORDS: open source, presentations, slides, web application

ΠΕΡΙΛΗΨΗ

Στο παρόν έγγραφο, παρουσιάζω μια διαδικτυακή εφαρμογή με σκοπό τη χρήση της για τη δημιουργία παρουσιάσεων. Αυτή η εφαρμογή ξεκίνησε από το μηδέν, χρησιμοποιώντας μόνο λογισμικό ανοιχτού κώδικα. Χρησιμοποιώντας αυτή την εφαρμογή, οι χρήστες μπορούν να:

- Προσθέσουν, επεξεργαστούν, μορφοποιήσουν και τοποθετήσουν κείμενο οπουδήποτε σε μια διαφάνεια.
- Προσθέσουν, αλλάξουν μέγεθος και τοποθετήσουν εικόνες σε μια διαφάνεια.
- Επιλέξουν ένα θέμα από τα παρεχόμενα από το CERN για την παρουσίασή τους, το οποίο μπορεί να αλλάξει ανά πάσα στιγμή.
- Αποθηκεύσουν την παρουσίασή τους ως αρχείο ".slides".
- Ανεβάσουν μια υπάρχουσα παρουσίαση (“.slides” αρχείο) για να συνεχίσουν την επεξεργασία.
- Παρουσιάσουν τις διαφάνειές τους μέσω ενός φυλλομετρητή.
- Εξάγουν μια “.slides” παρουσίαση, σε αρχείο PDF.

Περισσότερες ιδέες και βελτιώσεις προγραμματίζονται να έρθουν στο μέλλον, καθώς το έργο παραμένει υπό ενεργή ανάπτυξη. Η εφαρμογή είναι ήδη διαθέσιμη για δοκιμαστική χρήση από το τμήμα πληροφορικής του CERN. Τα πλεονεκτήματα αυτής της εφαρμογής είναι ότι χρησιμοποιεί μόνο ανοικτό κώδικα, είναι ανεξάρτητη από κάποια εμπορική εταιρεία, είναι δωρεάν και έχει απεριόριστες προοπτικές για περαιτέρω ανάπτυξη.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: ανάπτυξη εφαρμογής

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: ανοιχτός κώδικας, παρουσιάσεις, διαφάνειες, διαδικτυακή εφαρμογή

Αφιερωμένη στην υπέροχη οικογένειά μου.

ACKNOWLEDGMENTS

I would like to thank my supervisor at CERN, Maria Dimou, for all of the support and continuous motivation she was providing me, throughout my time at CERN and especially at the times when the project's difficulty was overwhelming. I would also like to thank several CDA and ST group members at CERN, especially Cristian Schuszter and Michal Kolodziejski for their technical help and advice. Last but not least, I would like to thank professor, Panagiotis Stamatopoulos for accepting to be my thesis supervisor, helping me obtain my bachelor's degree. Their contribution was crucial for the completion of this work.

CONTENTS

LIST OF FIGURES

PREFACE

This BSc Thesis Project is part of my job at CERN, from October 2019 until September 2020, in Geneva, Switzerland. I had the opportunity to work, as part of the Microsoft Alternatives (MAlt) team, which aims to migrate from commercial software products (Microsoft and others) to open source solutions, so as to minimize CERN's exposure to the risks of unsustainable commercial conditions. In this project, the goal was to create an open source web-based slides' maker. A software application that enables users to create beautiful presentations, using an intuitive UI. The project started from scratch and the result is a working prototype that offers the basic functionalities of a slides' maker software product. The project is under active development and I hope in the future it will be perfected and used throughout CERN.

1. BACKGROUND

The aim of this project is to have a flexible, vendor independent and cost-effective solution for users to work with, focusing on open software. After some initial research, four alternative products were presented:

- slides.com
- Google Slides
- LibreOffice Impress
- Kreator.js
- spectacle-editor
- CERN Slides' App

1.1 Product Evaluation

1.1.1 Slides.com

Slides.com is the commercial product, created by the developers of Revealjs. Revealjs, is an HTML presentation framework, which can convert HTML code into a presentation, that can be accessed from the browser. So slides.com, uses this framework in the background, and gives the users a web UI, making it easy to create presentations. Selecting this product would be a good idea, but the disadvantages were that CERN would still be vendor dependent, to the company behind slides.com and also would have to pay for the licenses, in order to use this application.

1.1.2 Google Slides

Using Google Slides would again have the vendor dependency problem and also the license cost. Even though Google Slides are free of charge for personal use, using them at an enterprise level, imposes charges.

1.1.3 LibreOffice Impress

LibreOffice Impress is a product very close to Microsoft PowerPoint, which provides the users with the easiest transition to a new product. It is also able to open existing Microsoft PowerPoint presentations better than any other product in this list. And while for personal use, LibreOffice suite is free, in order to use it in an organization, long-term Service Level Agreements (SLA), personalized assistance, technical support, and custom new features are required, these are offered from some LibreOffice partners, who charge fees for these services. Despite all others, the main disadvantage of LibreOffice Impress is that it is a clone of Microsoft PowerPoint and thus will always be inferior. Furthermore, it requires users to download software to their computer, while CERN wanted a web-based alternative with a new approach in creating presentations, an approach like in slides.com.

1.1.4 Kreator.js

Kreator.js is a project with similar principles to slides.com, but it is not well designed, its UI is outdated and is full of programming bugs. It is difficult to use, and the result is not satisfying at all. It also lacks a lot of basic functionalities, like uploading images and having a decent text editor.

1.1.5 Spectacle-editor

Spectacle-editor is yet another project that was considered at the time. It is an open source desktop application. It has cross-platform compatibility issues, the only way to experiment with this application was to run a Windows executable provided by the company Formidable Labs. Downloading the project and starting the app, wouldn't work in Linux or MacOS. The main disadvantages of the spectacle-editor were that it is a desktop application which needs the users to download and install it in their computer and that it was full of programming bugs, not maintained and very difficult to build on top of it, extend its functionalities and customize it.

1.1.6 The CERN Slides' App

Developing a solution internally in CERN, would check all the boxes:

- Vendor independent
- Open source
- Flexible and customizable
- New approach in slides making
- Web-based (i.e. nothing for the user to install, nothing to maintain)
- Free of charge

The disadvantages were the huge difficulty of the project, the resources needed for the project to be completed and the development time for the application to be actually usable and able to replace existing alternatives.

After the evaluation of the different paths, the conclusion was to build CERN's solution from scratch. It was the only way to cover all the needs of the project. The goal was to build something similar to slides.com in-house, for free.

1.2 Beginning of the project

The first month was the investigation month, there were some things to be decided:

- Technologies to be used
- Presentation engine
- User requirements
- What is already done

1.2.1 Technologies' evaluation

The technologies had to be modern, open source and supported by the developers' community. It was decided that a ReactJS [1] frontend and NodeJS [2] backend application approach would be a good idea to start with and add more technologies later on, as the project required.

1.2.2 Presentation engine

Creating a slides' maker, needs a presentation engine to run in the background, so what the users do in the web browser will be getting translated to code that can be processed by a presentation framework and the presentation can be shown to the user. In the beginning, Revealjs was tried, because it seemed like the most popular solution. Other tools were evaluated as well, like: slides.com and CodiMD [3], both using Revealjs in the background.

1.2.2.1 Revealjs approach

Reveal.js is an open source HTML presentation framework. This means that a user can write HTML code, using special HTML tags, and the Revealjs framework will convert this code in a presentation, that can be accessed from the browser. The idea was to create a ReactJS application that gives a UI to the user, hides all the code complexity and transforms user's interactions with the react app, into meaningful Revealjs processable syntax. But there were a lot of obstacles while trying to implement this in code, so this engine was abandoned.

1.2.2.2 Spectacle approach

Spectacle is a ReactJS based library for creating presentations. This means that a user can write a ReactJS application and use ReactJS components from this library to create and render a deck of slides. Now the idea was much simpler to implement in code, because the application would be in ReactJS and the presentation library as well. What had to be done was to write a ReactJS application and render the spectacle components as part of the whole application. Example, the slides' maker offers an "Add a new Slide" button, when pressed, a new "<Slide />" Spectacle component would be created and added in the array of slides. The user sees a sidebar in the left, with all the buttons, like the "Add a new Slide", and next to it, the "<Deck />" spectacle component is rendered with all the "<Slide />" components inside it. This approach was chosen to continue with.

1.2.3 User Requirements

Next thing that had to happen, was to ask the user community for functionalities they would like to see in the CERN Slides' App. In the list of requirements, were included:

- Page number automatic assignment.
- Selection of CERN themes.
- A user-friendly URL.
- Conversion to PDF.
- Easy change of background color.
- Text formatting.
- Flexible image positioning and resizing.

2. ARCHITECTURE

One aspect of creating a software application from scratch, is the freedom of customization that is given to the project owners. Designing the CERN Slides' App architecture requested the cooperation of multiple people from different sections of the IT department at CERN. Designing the project's architecture, means deciding:

- The way, format, place that the presentations will be stored.
- The way users will be accessing the application.

2.1 Presentation Storage

After meetings with the team of storage specialists at CERN, it was decided that the file format to save Slides' presentations, would be named: ".slides". Essentially, it is a ".zip" file. It contains:

- an "assets" folder, with all of the presentation media (pictures and animated .gifs).
- and a "presentation.JSON" file, that describes the Redux state that has to be loaded, once the presentation is uploaded to the CERN Slides' App.

The part of the Redux state that is saved in a ".slides" file, contains data like:

- the presentation title.
- the theme.
- the background color.
- the number of slides.
- the content of each slide.
- and more, explained in the Redux section.

Once the user clicks on the "Save Presentation" button, a chain of actions takes place. Firstly, the request is sent from the client-side, to the server-side. There, the request is processed and the ".slides" file is created. Finally, this file becomes

available for download from the users. The users download it from the browser and save it locally in their hard drive.

2.2 How users access the Application

In the beginning, the idea was to integrate the CERN Slides' App, with CERNBox, the current CERN's cloud service, which will soon be upgraded and renamed to Phoenix [4]. This would be a huge plus for the application's future, because it offers uniform access to several CERN-supported applications from within the same cloud storage. The presentations' storage security, availability and backing up is handled by the CERNBox team and users can access their files from different devices that are synchronized with the cloud storage.

2.2.1 Phoenix

The CERN's storage team has been working on a new cloud service for CERN. Its name is Phoenix and will completely replace CERNBox. They are both based on an Owncloud server [5], but Phoenix is the new revised frontend.

Owncloud is a suite of client-server software for creating and using file hosting services. The Owncloud server is free and open source, thereby allowing anyone to install and operate it without charge, on their own private server. That is what CERNBox is, a hosted instance of Owncloud in CERN computers.

The storage team encouraged the integration effort of the CERN Slides' App with Phoenix, even though Phoenix remains under development and the exact date that it will be production ready, is still unknown.

The CERN Slides' App would be the first application to be integrated and work with and from Phoenix. An integration with Phoenix, would mean that the users would login to access their personal space in Phoenix, and then be able to create a new ".slides" presentation directly from their Phoenix space. This means that their presentation would be saved automatically and backed up in the CERN's cloud service without any user action.

The diagram below visualizes how the integration works internally.

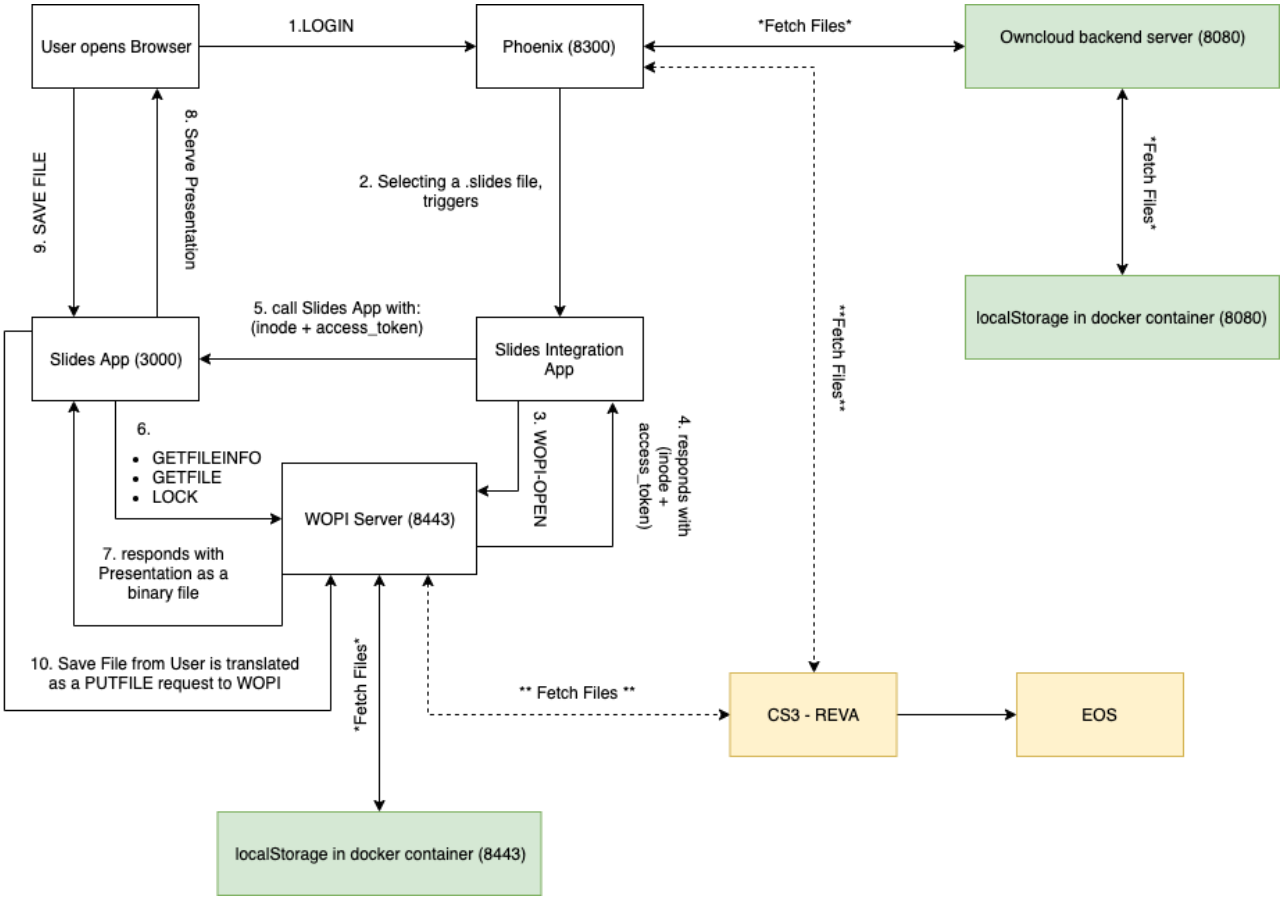

Figure 1: "Access CERN Slides' App from Phoenix"

With numbers, the sequence in which the actions happen, is indicated. The green color indicates how it currently works, using local storage in the test environment. With orange color and dotted lines, the future is indicated, once the Phoenix will be in production. The numbers in parenthesis, 8300, 8443, 3000 are just the local ports that the applications were running in the test environment.

The WOPI server [6] is a vendor-neutral Web-application Open Platform Interface (WOPI) gateway for Enterprise file synchronization and sharing (EFSS) systems. It is used in order to give create, read, update, delete (CRUD) access to the files stored in the cloud service, using a dedicated set of REST APIs.

CS3-REVA [7] is an interoperability platform. It connects storage, synchronizes and shares platforms and application providers, and it does it in a vendor and platform neutral way by using the CS3 APIS [8]. This way, moving from a storage solution to another becomes easy, there is no need to change anything due to the storage agnostic approach of REVA.

EOS [9] is a software solution that aims to provide fast and reliable multi-PB disk-only storage technology for both LHC and non-LHC use-cases at CERN. It is the storage system used in the background to store users' files in the "cloud".

The integration application that would connect Phoenix and the CERN Slides' App started right away. The whole environment was set up locally, a WOPI server (using a Dockerfile), an Owncloud instance (using another Dockerfile) and a Phoenix instance running locally in the development environment. The Phoenix integration application was created successfully and users connecting to their Phoenix space could click the "Create a new Presentation" button and this would open a new tab with the CERN Slides' App. The code for this integration application lives in: "https://github.com/aristofanischionis/phoenix/tree/slides-integration-app", a private user project space, because it is still in development mode, yet publicly available.

Next step, was to connect the CERN Slides' App with the WOPI server, so when one clicks the "Save Presentation" button in the app's UI, the request would go to the Slides' server, the ".slides" presentation would be created and then sent to the WOPI for cloud storage, using the WOPI set of APIs. There were some difficulties regarding user authentication when trying to establish this communication and the development was temporarily put on hold.

Phoenix was not production ready yet and the connection between WOPI and REVA was under development by the storage team. An alternative way of accessing the application and storing the presentations had to be decided.

2.2.2 Web Browser

The CERN Slides' App, being a web application, already foresaw access from the browser via https://slides.web.cern.ch, which is equivalent to https://cern.ch/slides. This required user authentication to be added to the application, using the new SSO (central Single Sign On) service at CERN, and also handling the presentation saving and uploading, interacting with the user's local disk before getting released to the production environment.

This is the only way to use the Slides App until CERNBox is replaced by Phoenix, the Phoenix+WOPI are connected to REVA and data storage to EOS is available.

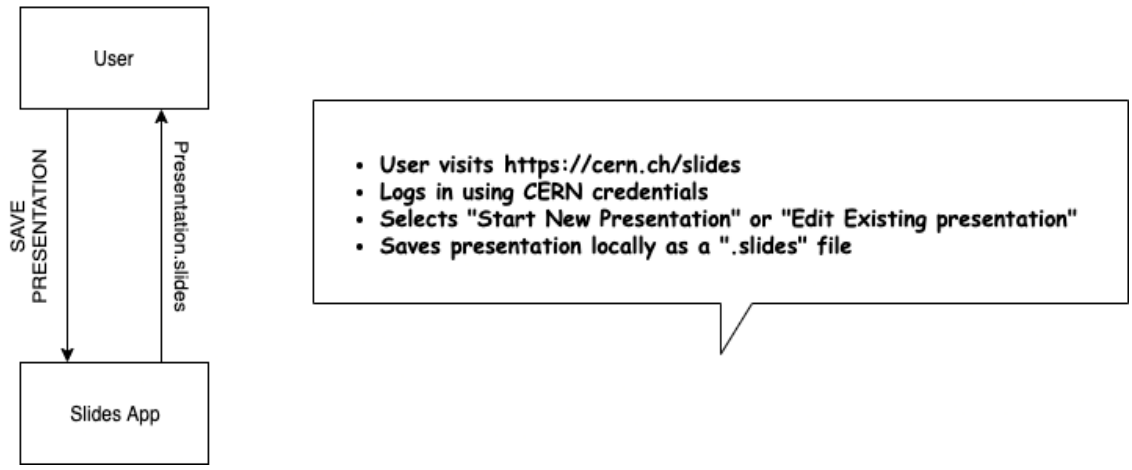

Figure 2: "Access CERN Slides' App from the browser"

3. FUNCTIONALITY REQUIREMENTS

The slides that CERN users make can be very dense and complex in content, containing histograms, diagrams, tables, graphs and equations. Even though the conversion of existing presentations was not a functional requirement, an attractive tool was still needed, in order for the users to adopt it and decide to make it the tool of preference over other commercial alternatives. Given the limited amount of time this is the short list of essential features that had to be included in the app, in order to be considered.

1. Page number automatic assignment.
2. Selection of CERN themes.
3. A user-friendly URL.
4. Conversion to PDF.
5. Easy change of background color.
6. Text formatting.
7. Flexible image positioning and resizing.
8. Slideshow functionality.
9. Cross browser compatibility.
10. Set an image as background of a slide.

Mainstream and corner-case presentations were collected in order to identify the requisites for the available time. Some examples are:

Figure 3: "Slide with formatted text, hyperlinks, image, CERN theme"

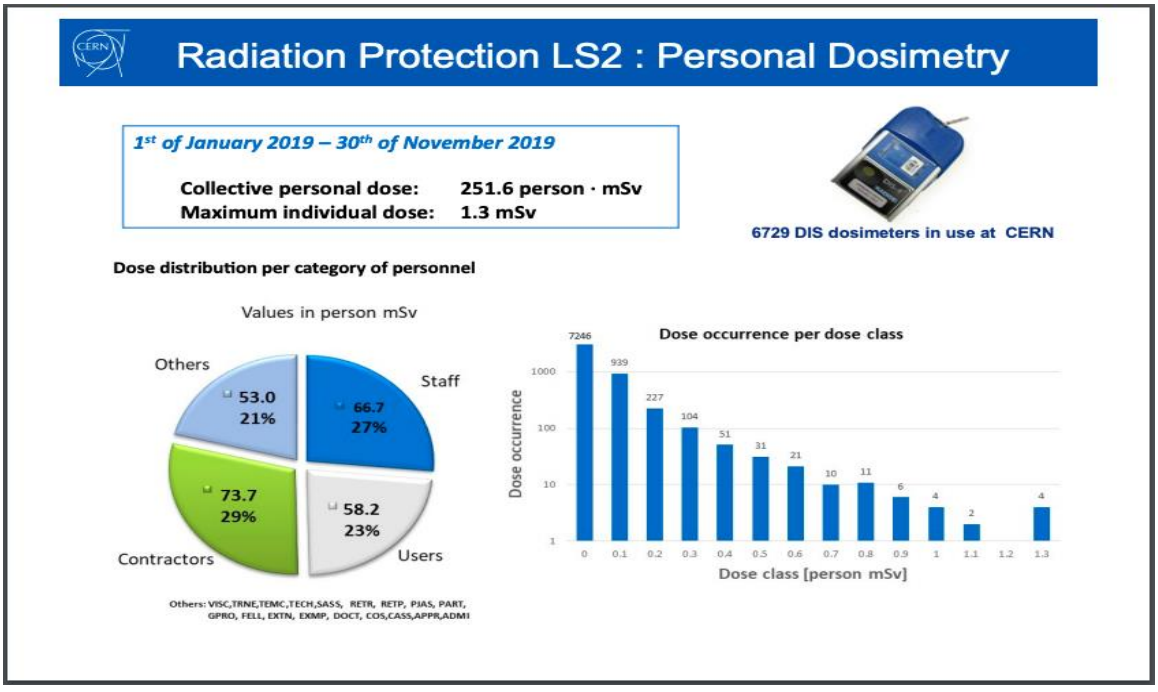

Figure 4: "Slide with title, formatted text, images, diagrams"

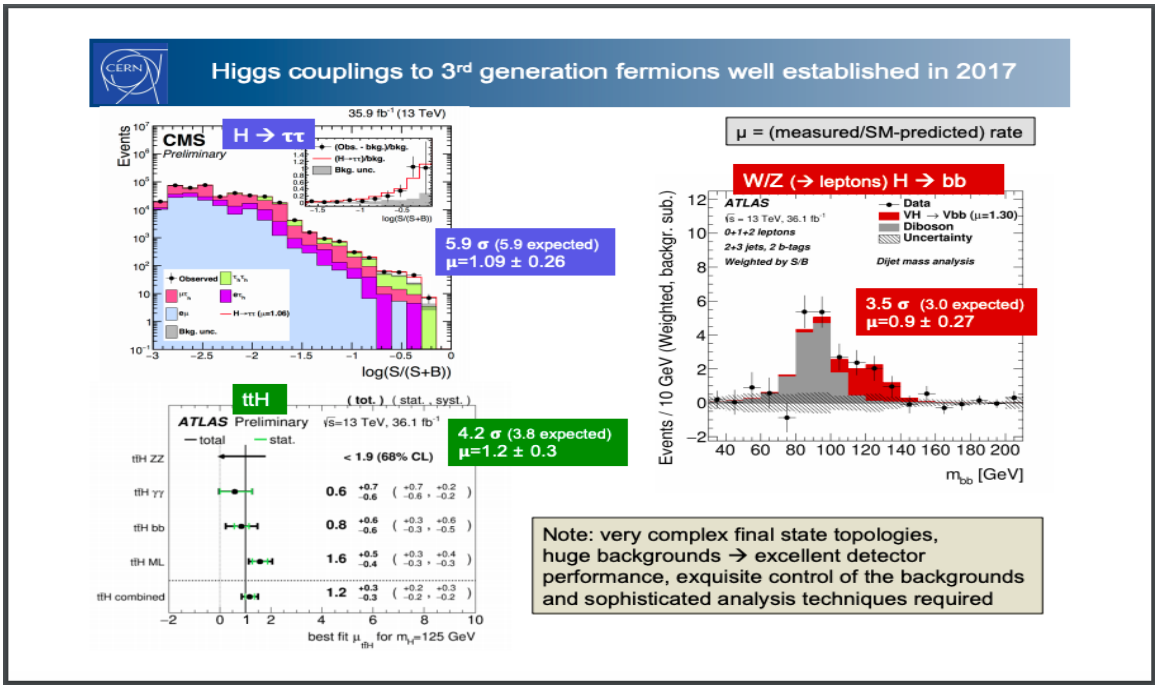

Figure 5: "Slide with title, formatted text, images, different types of diagrams"

4. USER FACING VIEWS

Users of the CERN Slides' App will see the following screens.

4.1 Authenticate

The users navigate to https://slides.web.cern.ch and login using their CERN credentials. The authentication is handled by the CERN's SSO service.

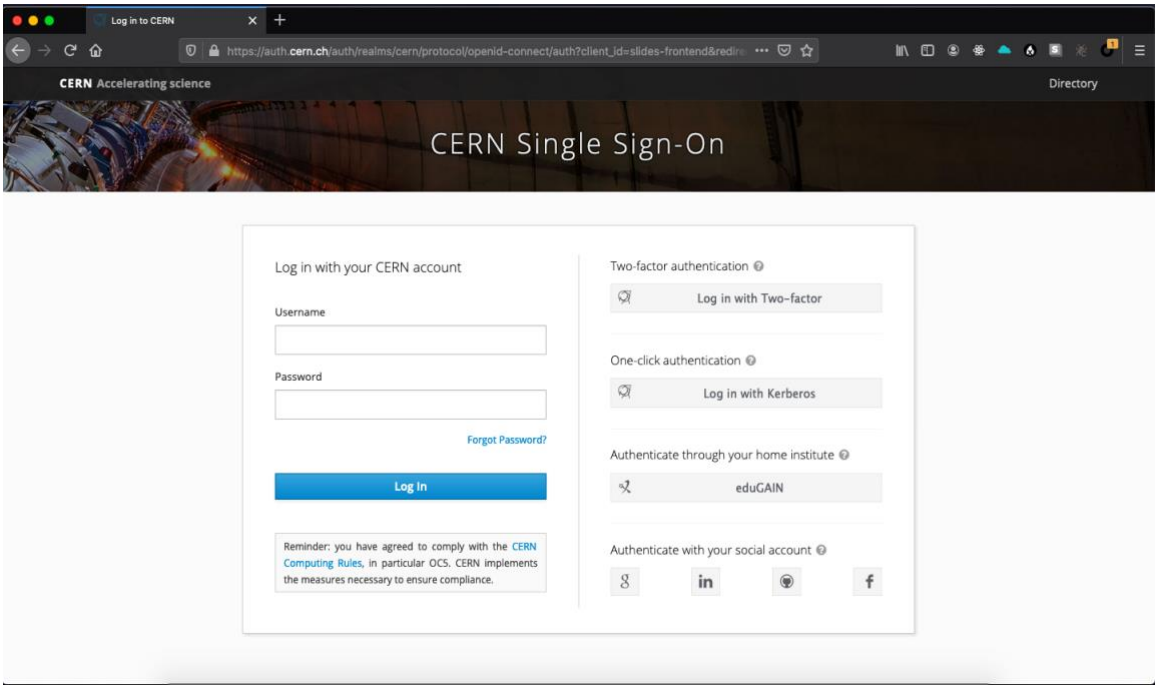

Figure 6: "User Authentication Screen"

After the users are authenticated, their CERN username is captured and saved in the presentation's Redux state.

The users see the next screen.

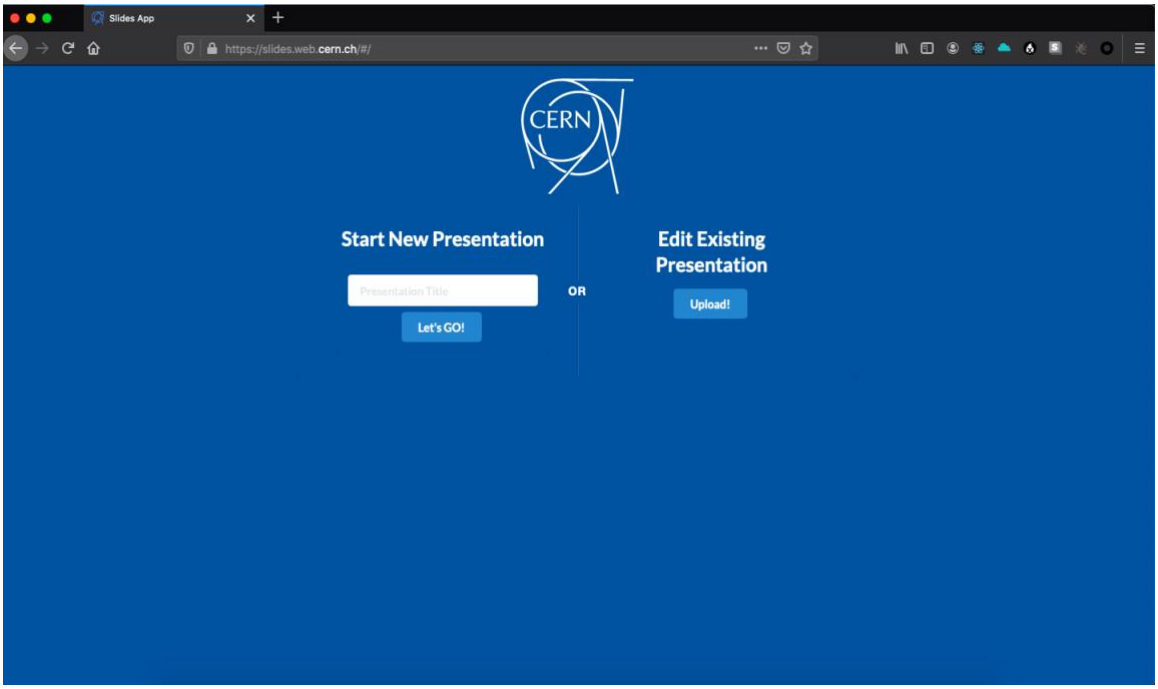

Figure 7: "Homepage of CERN Slides' App"

Select to either start a new presentation or upload and edit an existing one.

4.2 Edit

If they start a new presentation, they need to provide a presentation title, and click the "Let's GO!" button. The URL changes to: "https://slides.web.cern.ch/edit/achionis/title/", which signifies that it is in "edit" mode, the username is "achionis", this is the username provided by the authentication service, after user's login and "title" is the given presentation title. The view when the user is inside the application environment is the following:

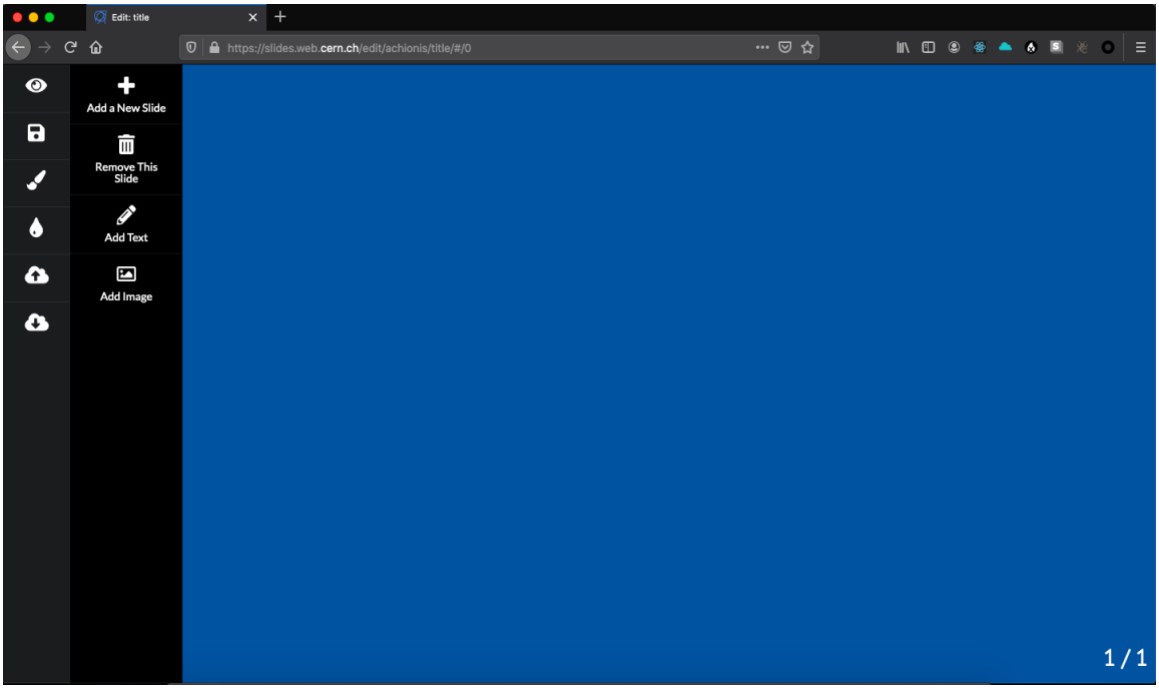

Figure 8: "Edit mode of the CERN Slides' App"

In the leftmost sidebar, there are the buttons concerning "presentation-wide" functionalities. Specifically:

- Slideshow.
- Save presentation.
- Change background color.
- Change theme.
- Upload existing presentation.
- Export to PDF.

In the next bar, the user finds "slide-wide" functionalities. Which are:

- Add a new slide.
- Remove current slide.
- Add a text box.
- Add image.

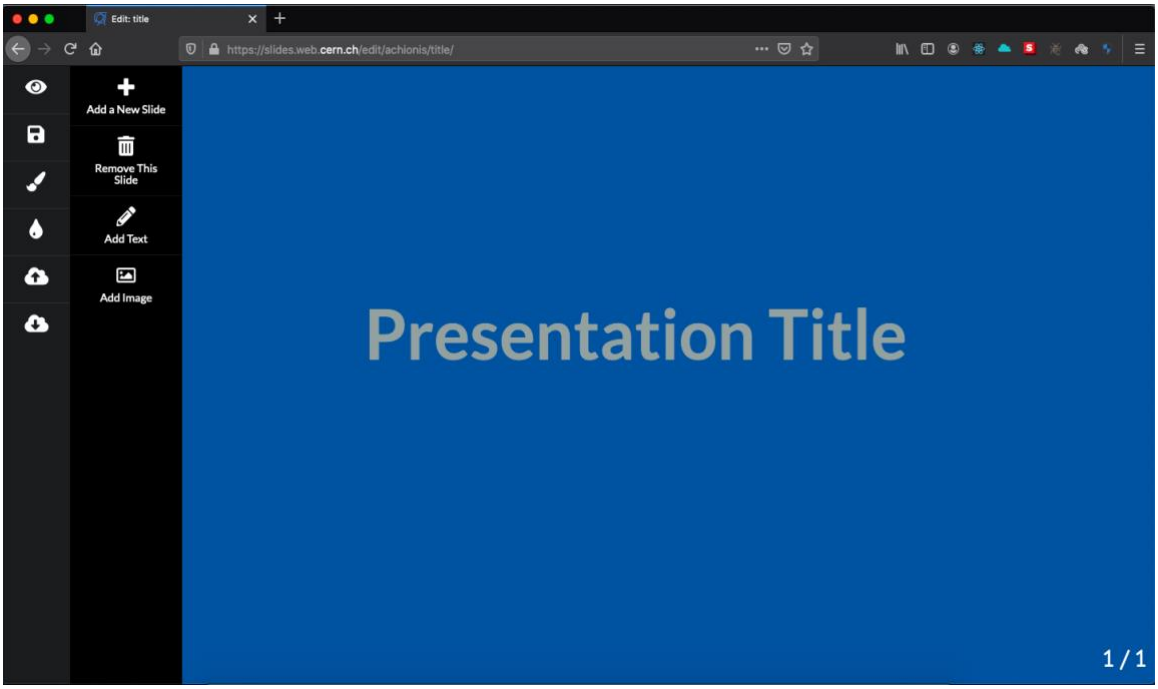

Figure 9: "Adding a title"

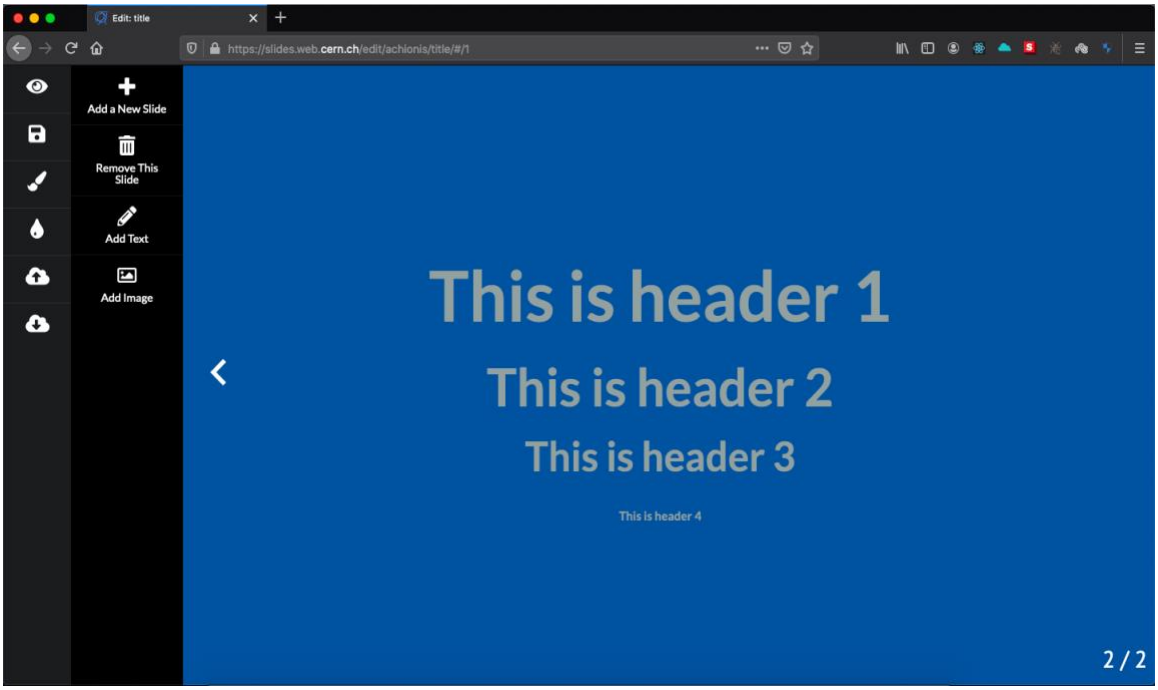

Figure 10: "Adding a new slide and experiment with different header types"

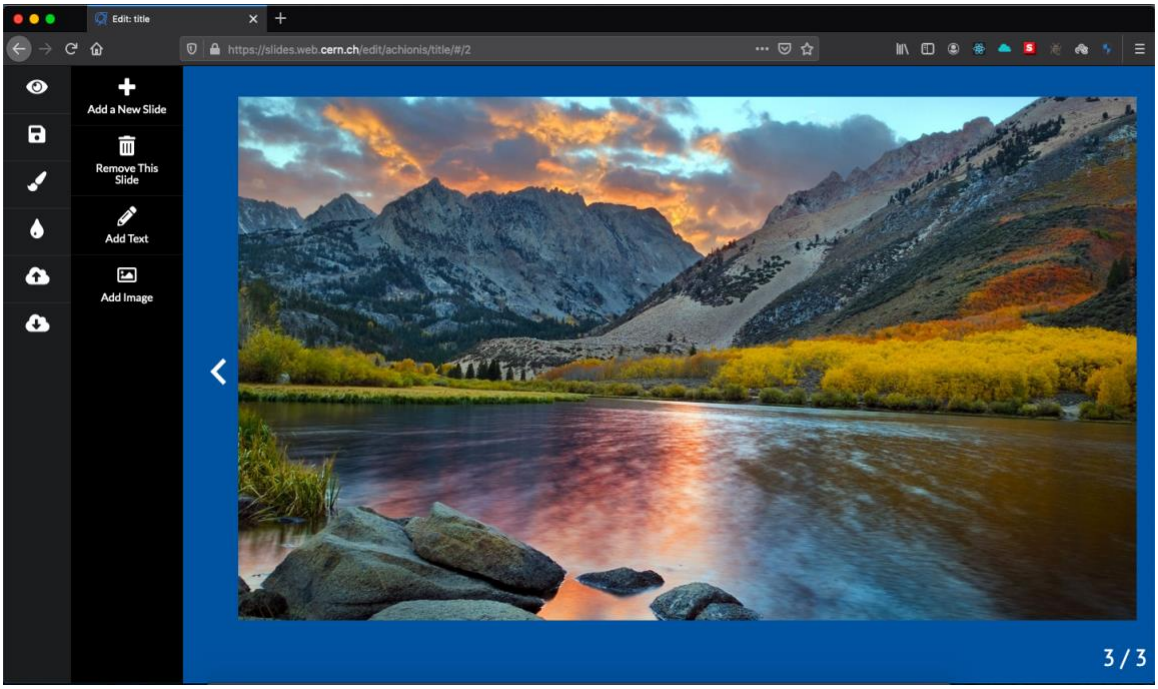

Figure 11: "Adding an image"

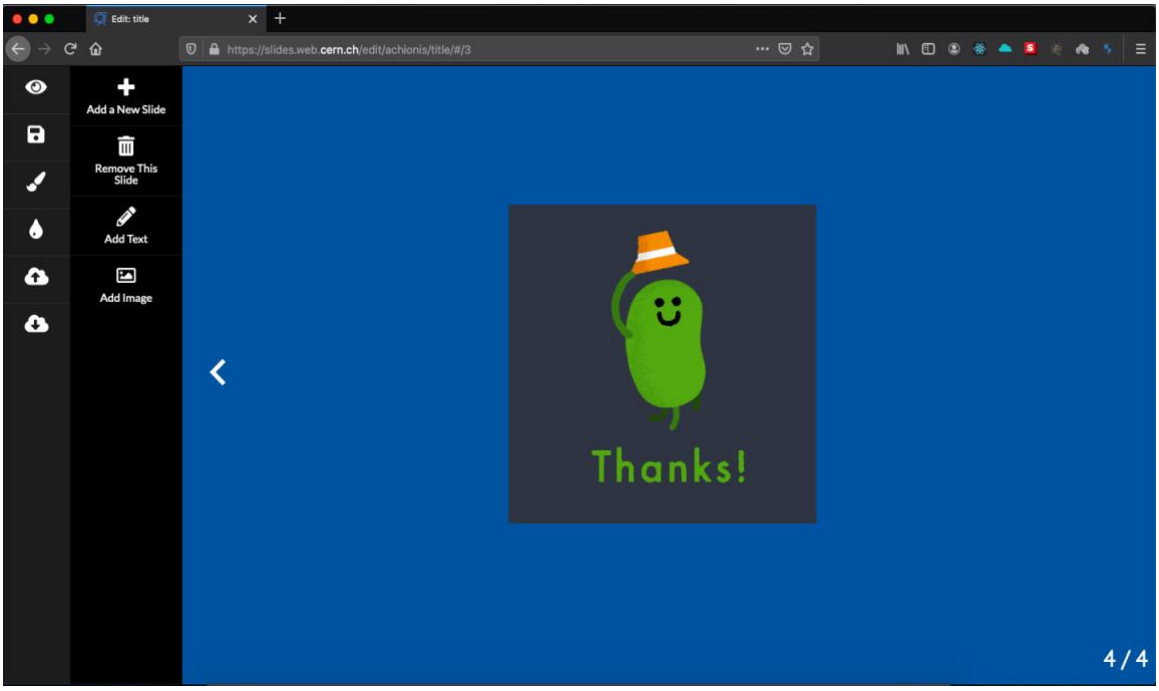

Figure 12: "Adding an animated .gif file"

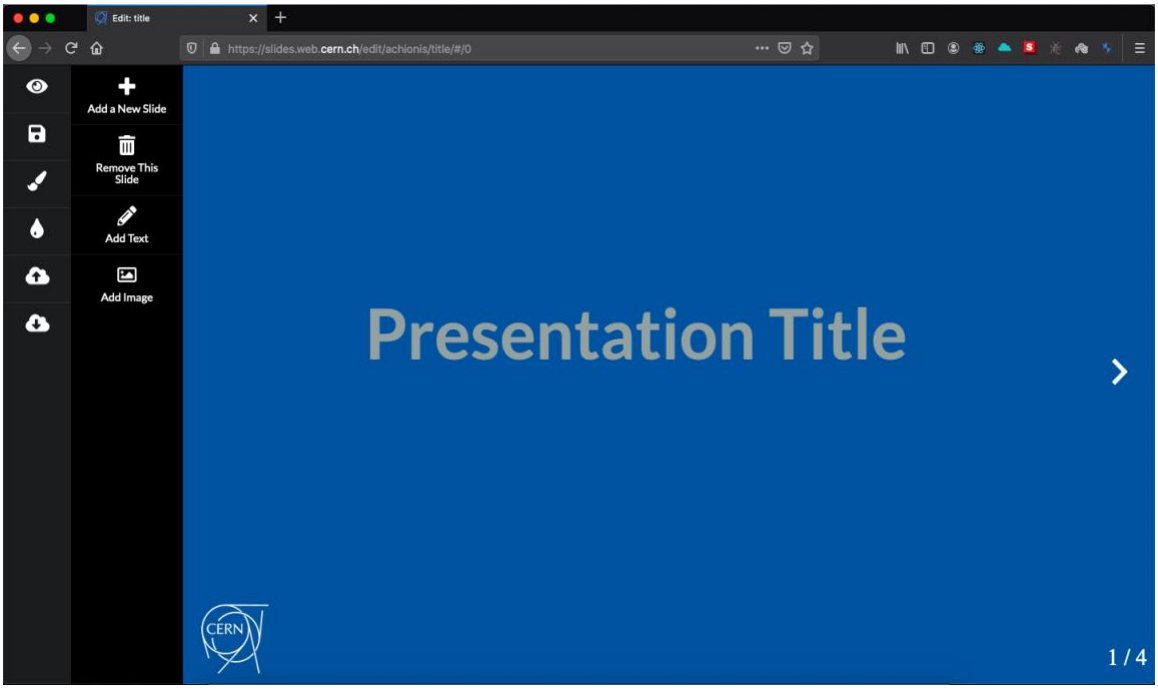

Figure 13: "Changing the theme"

Now the CERN logo is added in the bottom left corner.

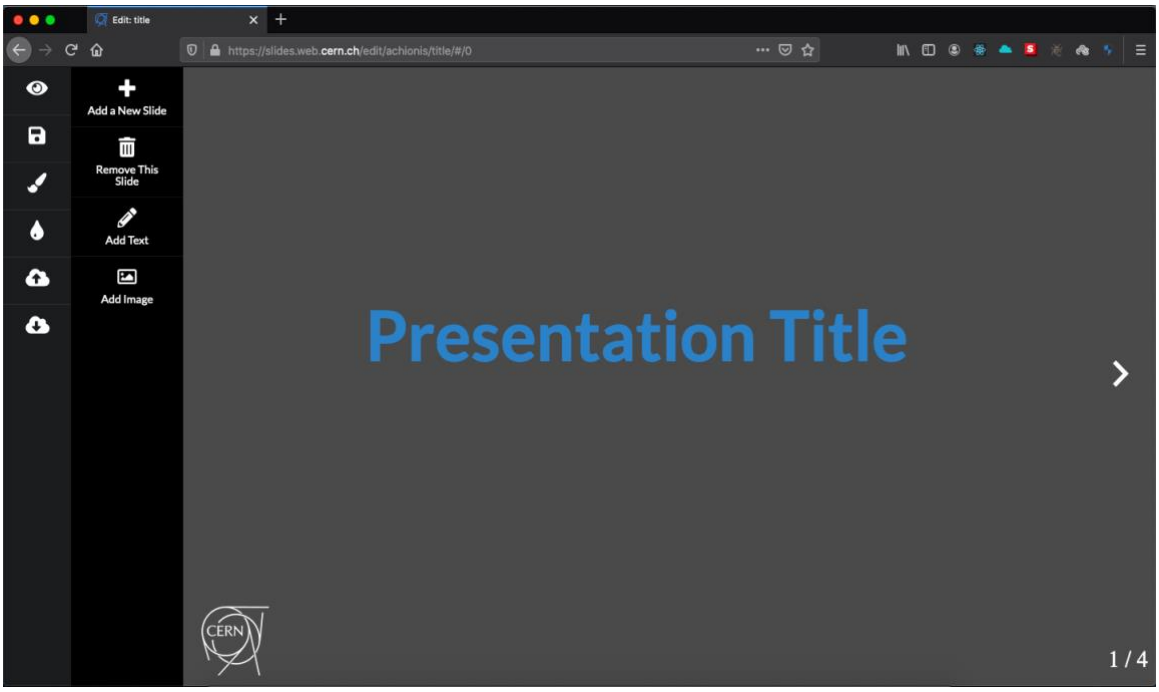

Figure 14: "Changing of the background text color"

4.3 Slideshow

When the users feel that their presentation is ready, they can go in Presentation mode, to view the presentation only, without the sidebars.

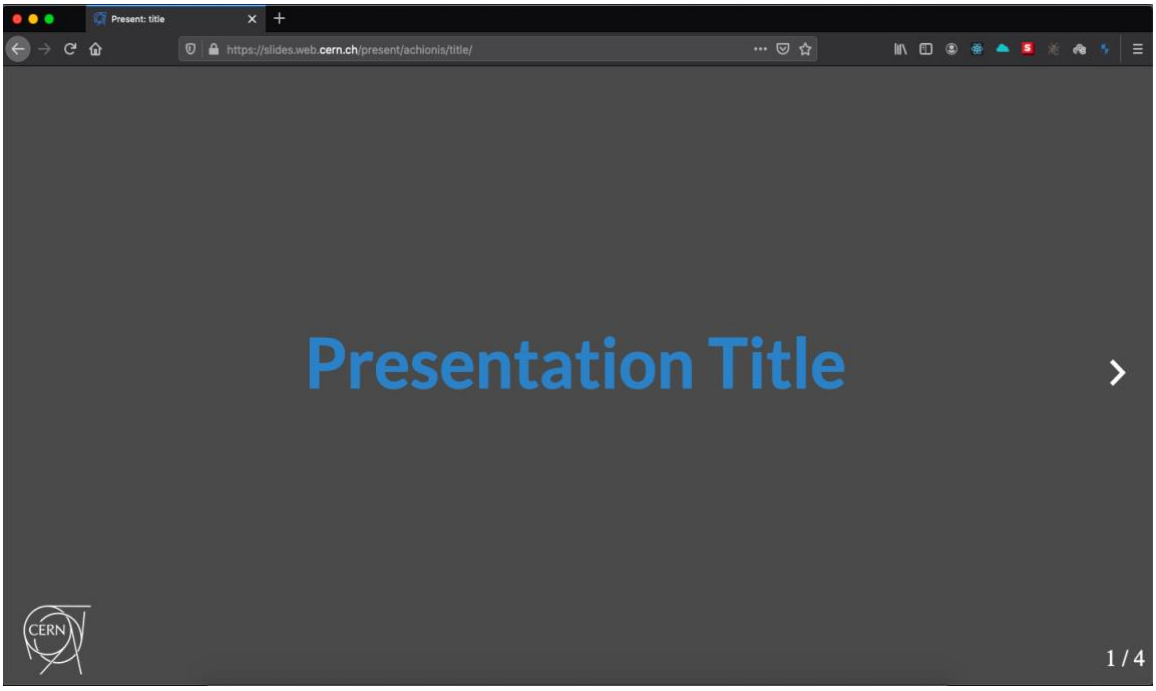

Figure 15: "Slideshow mode of the CERN Slides' App"

4.4 Re-edit presentation

The users may want to make some more changes in their presentation. To return in Edit mode, the users just need to click the back button of the browser and the URL will change again to "https://slides.web.cern.ch/edit/achionis/title/". They can continue editing the presentation until they are happy with the result.

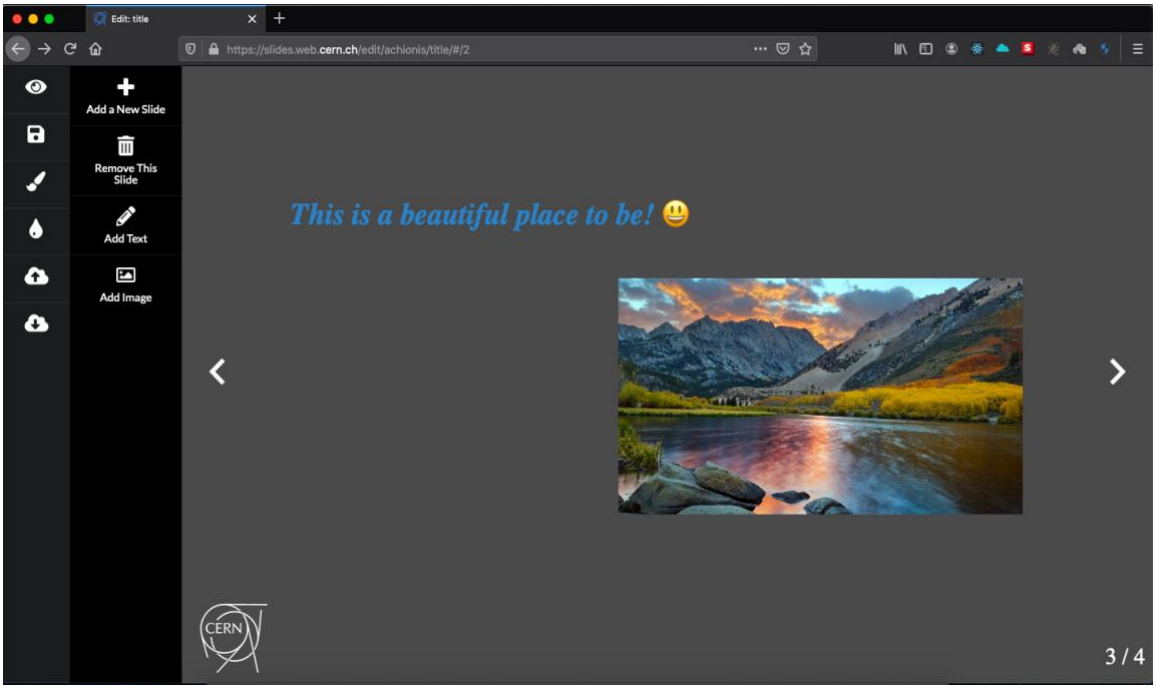

Figure 16: "Slide with text and image"

4.5 Save presentation

It is time for the users to save their presentation, to do so, they need to click on the "Save Presentation" button. Users are prompted to decide on keeping the presentation name or change it. If the users rename the presentation, the presentation name changes also internally in the application and this is reflected in the URL as well, changing the title field.

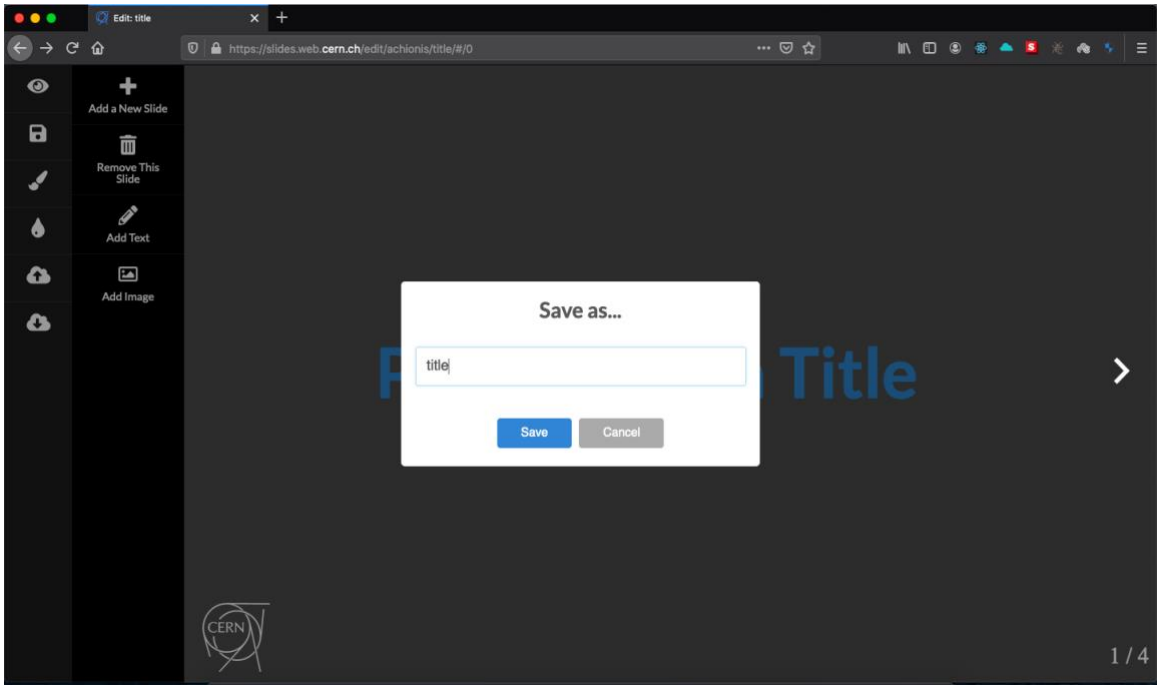

Figure 17: "The current presentation title is shown to the user"

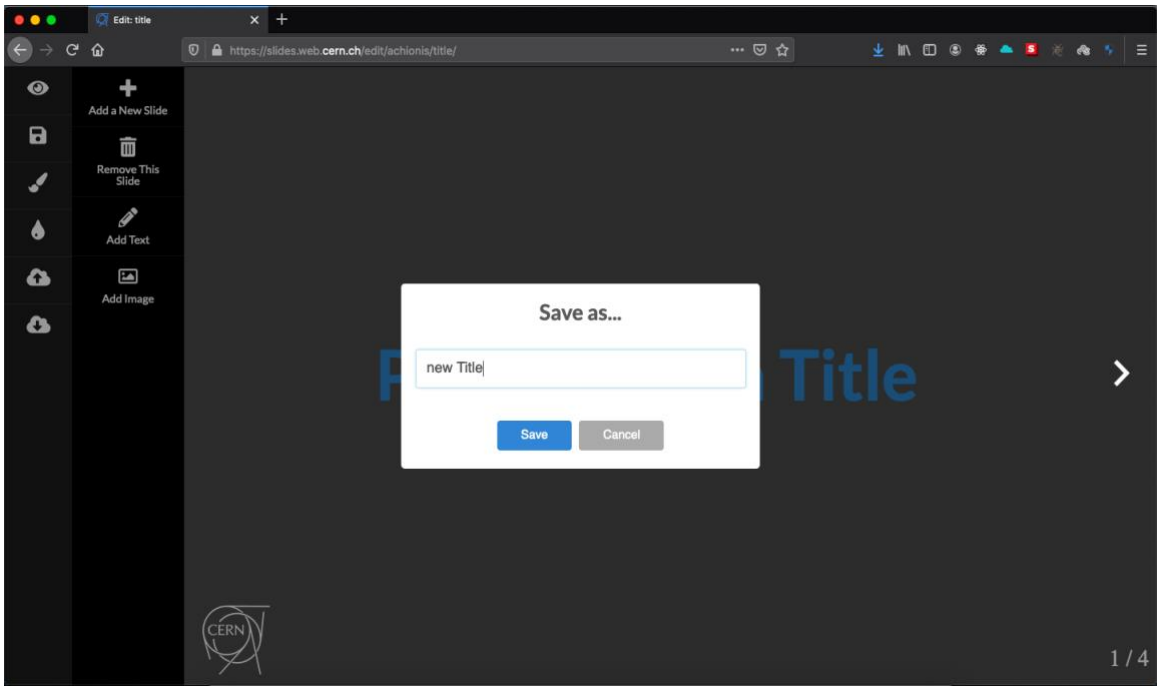

Figure 18: "The user changes the presentation title and clicks 'Save'"

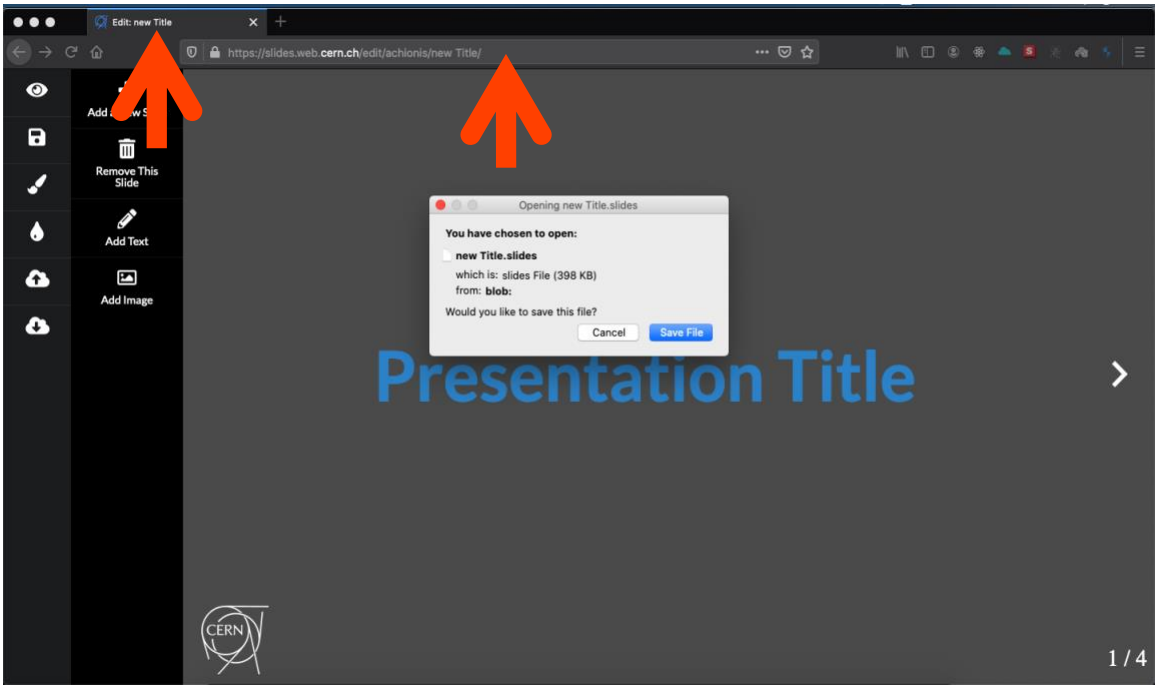

Figure 19: "Notice that the tab title and the URL change, to reflect the new title"

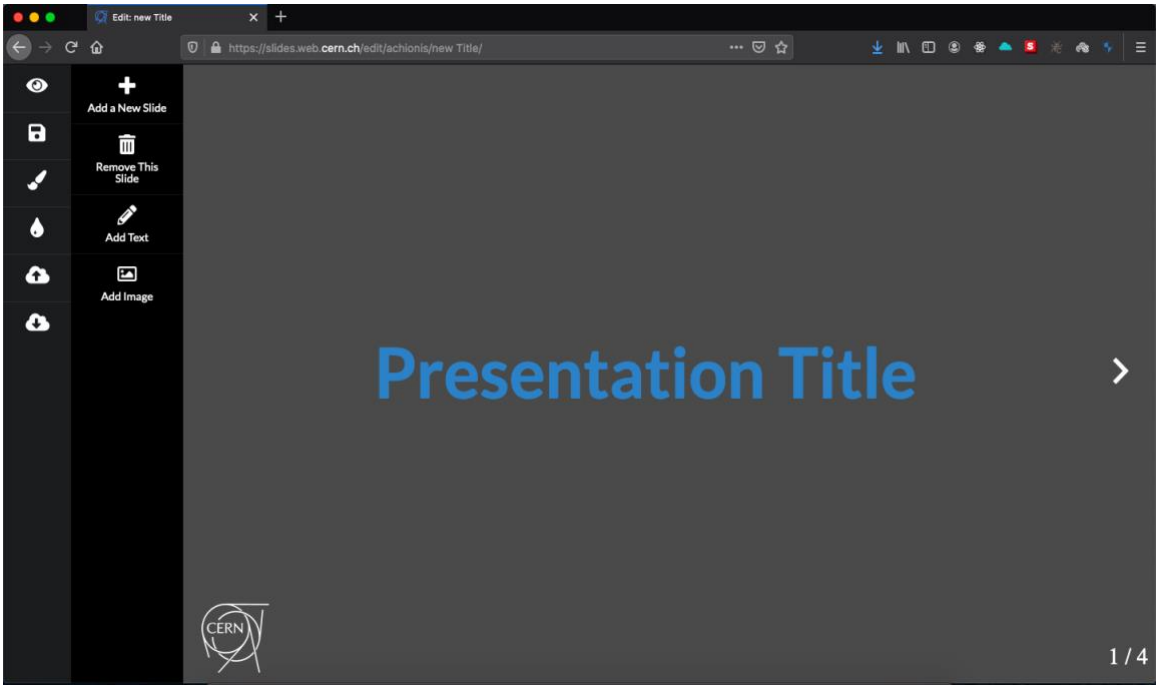

Figure 20: "User clicks on 'Save File' and the presentation gets downloaded"

4.6 Upload presentation

If the users want to continue editing an existing presentation, they need to upload it from the application's homepage:

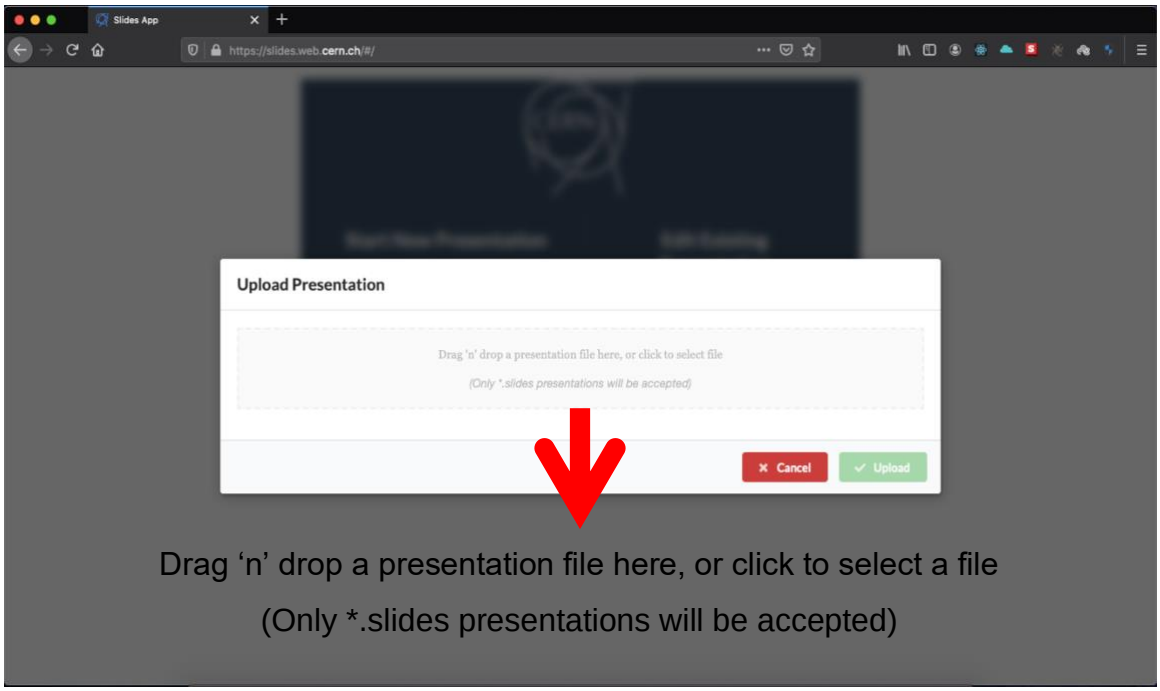

Figure 21: "User clicks on upload existing presentation button"

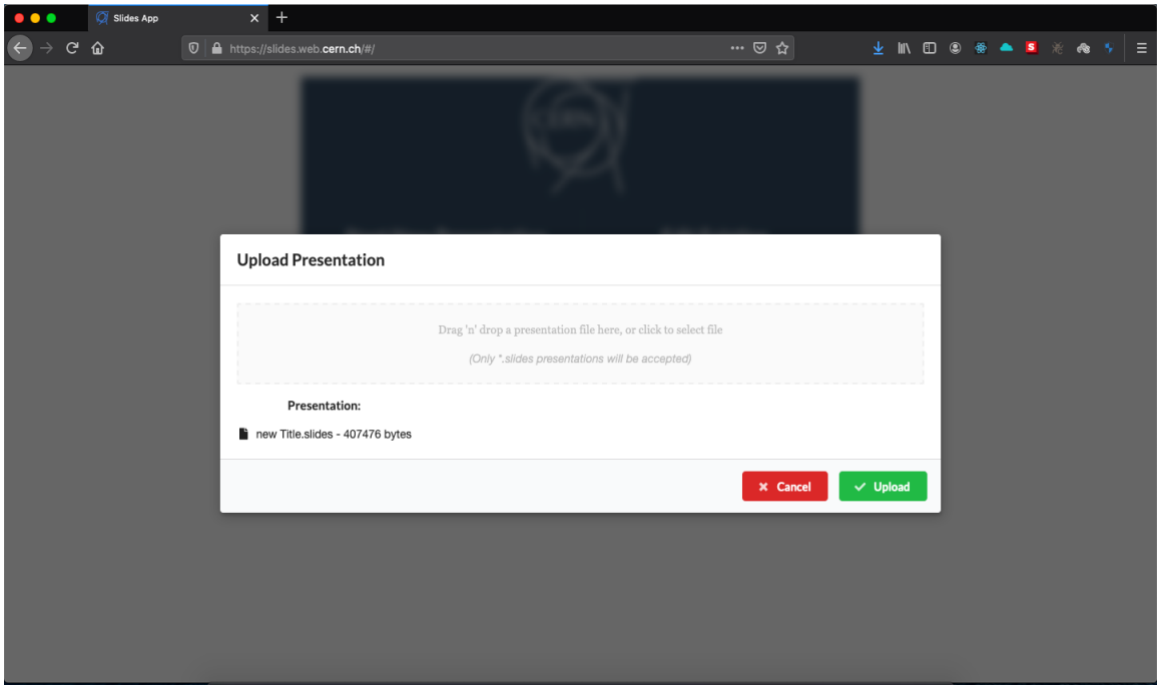

Figure 22: "The 'new Title.slides' file is selected to be uploaded"

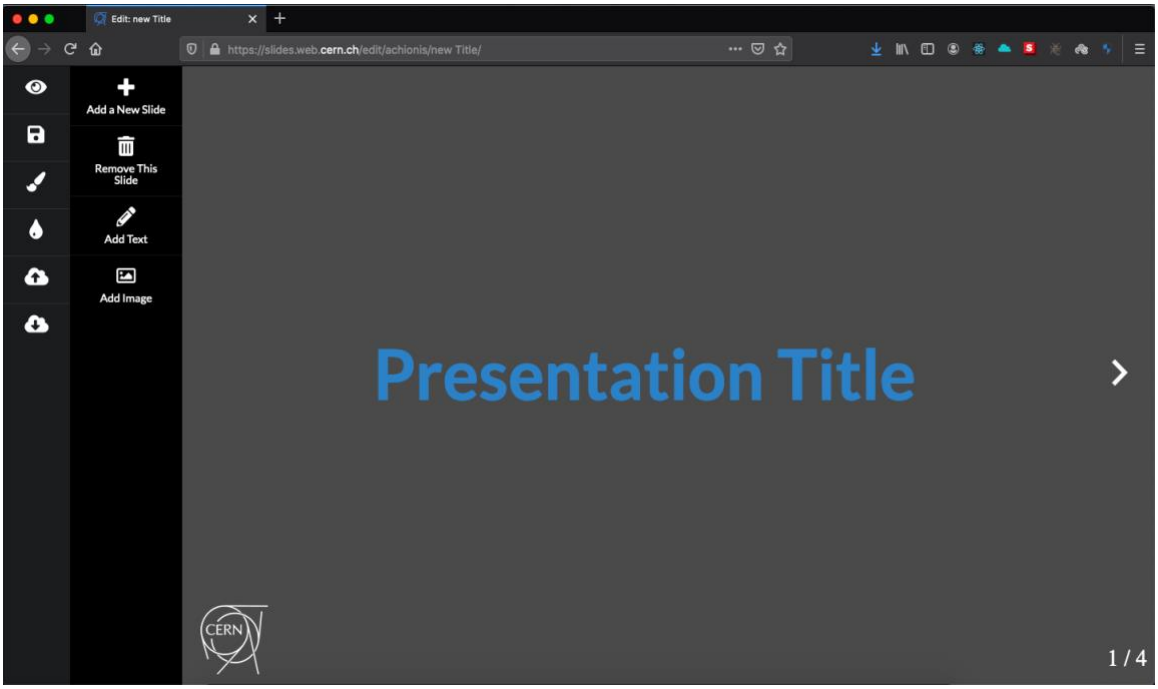

Figure 23: "The presentation is loaded and the CERN Slides' App is in edit mode"

5. TEXT EDITOR

One of the most important attributes of a presentation software is undoubtedly its text editor. The text editor is so important, because it enables the users to give a personal note to their presentation, by selecting the appropriate text style. Like the font-family, font-size, text color, text alignment, typography and more.

5.1 Investigation

First step was to try to find which the options were, when considering a text editor. Then, spectacle-editor came into perspective. This project is a joint effort between the companies Formidable and Plotly. It actually is an Electron based app, for creating presentations, using Spectacle components. The project is not maintained anymore, and it was difficult to get it running, but the ideas they were using in their codebase seemed interesting. So even though the project could not be used as is, the code could be inspiring. The integration of some spectacle-editor code started, it was tedious and after a lot of effort, the idea was abandoned, because of the implementation complexity.

Electron is an open source software framework developed and maintained by GitHub. It allows for the development of desktop GUI applications using web technologies, it combines the Chromium rendering engine and the Node.js runtime.

Other text editors that were tried and rejected include: CKEditor 4 & 5 [10], Froala [11], TinyMCE [12], Quill [13], Summernote [14] and more. All of them were tried, but none was satisfying all of the requirements: a text-editor that is free and open source, feels modern, is customizable, works well with ReactJS and can easily save the text content in the Redux store.

CKEditor was the editor that was tested more than others, even though its customizability was limited, the rest of the points were met. Froala editor is branded and not customizable, imposing a "Powered by Froala" label at the bottom of the editor. The TinyMCE and Quill editors had issues integrating with Redux. The summernote editor was not easily customizable and a lot of basic text formatting functionalities were not working.

The text editor was definitely the most challenging choice to make for the CERN Slides' app.

5.2 Present

After almost 3 months of evaluating and rejecting text editors, a solution finally came into light. React-draft-wysiwyg [15], this is an open source project, which offers ReactJS integration for the draft-js [16] editor. This editor is created by Facebook and still remains open source. The React-draft-wysiwyg variation of this project is a more customized version of the original project and is the perfect fit for the CERN Slides' app. It is free, customizable, ReactJS compatible and works well with Redux as well. So, after some adaptation actions in order to match well with the project's needs, this solution was adopted.

6. IMAGES

The second most important component in presentations is definitely the images. Users should have a way to upload, move and resize images easily in their presentations. Also, it is important to decide where the images will be stored, in what format and how they will be rendered and presented to the user.

In web applications two are the main ways to handle images. One is to encode them as base64 strings and embed them directly in the HTML, CSS or JavaScript. The second way is to store them as binary files and render the images by referencing the path where the image is located. Both ways are good but in different applications. In most cases, using the original image files is more efficient.

The use of the base64 string encoding, makes the files ~33% larger than their original binary representations. This means more data down the wire, which can be exceptionally painful on mobile networks. The base64 encoding uses the data URL scheme to function, which is not supported by the Microsoft browser Internet Explorer versions 6 & 7. Additionally, base64 encoded data may possibly take longer to process than binary data. For these reasons base64 string encoding was not adopted.

Using images as binary files is a better approach for the CERN Slides' app. All images are transferred from the client to the server using POST requests and then the server saves them in a specified folder. Specifically, in the server there is a dedicated Ceph [17] storage volume, mounted on the "/mydata/presentations/" path.

Ceph is an open-source software storage platform that implements object storage on a single distributed computer cluster and provides 3 in 1 interface for: object-, block- and file-level storage. Ceph aims primarily for completely distributed operation without a single point of failure, scalable to the exabyte level, and freely available.

When a user uploads an image to the CERN Slides' App, the server saves it in the Ceph volume storage, in the path:

"/mydata/presentations/<username>/<title>/assets/<image_name>".

The image name that is used when saving an image in the server, is not simply the name of the image. Instead, an md5 hashing is performed in the contents of the image and this hash is used along with the image name to construct the final name of the picture. For example, having an image named "aristophanes.png", can result in being named: "95c085fd173a31f9375d878fc4c9bbf1_aristophanes.png". The first part is the file's md5 hash value. The purpose behind this is to make each image name unique and avoid overwriting of images because of the same name.

All the valid image file formats are supported, like ".jpg", ".jpeg", ".png" and also all animated GIFs as well (".gif").

After the successful saving of the image, it has to become available to the client side. To do that, the server serves statically the mounted Ceph storage volume ("/mydata/presentations/").

Then the client side is responsible to keep information about:

- The server's URL.
- The username and title of the current presentation.
- The image name.

With all this information, the appropriate URL is generated like that:

const image = "<server_path>/static/<username>/<title>/assets/<image_name>". This URL is used when the client side presents the image to the user. For example:

.

Finally, using the images as files, enables the browser to download each image after the first request and then save it in cache. This saves bandwidth because less requests need to take place.

7. RESIZE AND MOVE

Giving the user the functionalities of adding text and uploading images in their presentation is basic and crucial. The next step is to enable the users to resize and move both the text and the images in their presentations. The CERN Slides' App is handling both image and text as a generic element type and on this type, the resizing and moving functionalities are applied.

7.1 Investigation

In the beginning, existing work was explored, in order to get some examples and help from existing projects and libraries. The two main libraries that were tried out for the resize and move functionalities were, the react-rnd [18] and react-grid-layout [19], but both were rejected as there were troubles syncing with the Redux store and the position of the elements did not persist when a new slide was loaded.

7.2 Implementation

After some research, reactablejs [20] came into light, a react high-order component for interactjs [21]. With this library, offering a lot of flexibility and customization options, as much as good integration with the Redux store, the users of the CERN Slides' App could finally resize and move the elements (images and text boxes) inside a slide and also keep the new coordinates well updated in the Redux store.

7.2.1 Calculation of coordinates

To calculate the new coordinates (x, y) of an element, after a move event, a mathematical equation had to be used.

- const x = e.client.x - e.clientX0 + coordinate.x;
- const y = e.client.y - e.clientY0 + coordinate.y;

Explanation:

1. const x and const y, are the variables to be stored.

2. e.client.x and e.client.y are the final coordinates of the item as recorded by the library.
3. e.clientX0 and e.clientY0 are the coordinates from where the item started its dragging.
4. coordinate.x and coordinate.y are the most recent values of x,y as stored and used in the element's component state.

8. THE CODE

When starting out to design the application, the idea was to follow the principles, look and feel of Slides (https://slides.com/) as discussed above. Their UI includes two sidebars on the left and the presentation deck of slides that takes up the rest of the space.

The first sidebar on the left is for "presentation-wide" functions (Export, Import, Download, Preview, etc.) and the second one for "slide-wide" functions, adding different types of elements in each slide (Text, Image, Shape, etc.).

8.1 Code structure

The CERN Slides' App frontend is a ReactJS application. In the edit mode of the application, the user sees two sidebars with buttons and the Spectacle <Deck /> component including an array of <Slide /> Spectacle components. Each one of the <Slide /> Spectacle components contains an array of elements, text or image elements.

The leftmost sidebar has buttons that, once clicked, they set variables in the Redux store to "true". Then, the responsible components, that are already loaded, are waiting to get triggered by their respective Redux store variables, catch the change in the Redux store, execute the required actions and set the Redux store variables back to "false". In this way everything stays synchronized, clean and the ReactJS standards are followed. All communications between components should happen through the Redux store and not by passing variables around, from function to function. An example is the "Save Presentation" button. Once clicked, an action from the Redux store is triggered, in order to set the "saveRequest" variable to "true". The SavePresentation component, which is loaded from the beginning, catches the change in this variable's value and calls the "Save()" function in order to execute all the requests and create the ".slides" presentation file.

The second sidebar uses exactly the same logic.

8.1.1 Redux

Redux is an open-source JavaScript library for managing application state. Redux is widely used in ReactJS applications and has become a standard way of establishing communication between ReactJS components and sharing code variables.

In the CERN Slides' App, Redux is used and it contains 4 fields:

- keycloak
- presentation
- deck
- router

The presentation and deck are the fields that are customly created and handled by the CERN Slides' App. There lives all the vital information of each presentation.

8.1.1.1 Keycloak

Keycloak holds all the authentication information and is added to the Redux store by the keycloak wrapper. The most important information being held here is the username, the user authentication state (authenticated or not), the Bearer authentication token [22] and the refresh authentication token.

8.1.1.2 Presentation

Presentation holds general information about the current presentation. Specifically, it is described using the following attributes:

- isPhoenixMode: is a boolean value, used to determine if the application is in Phoenix mode.
- presentationMode: is a boolean value, used to determine if the application is in Presentation mode.
- exportMode: is a boolean value, used to determine if the application is preparing the PDF file of the presentation.
- theme.
- title.

- backgroundColor.
- isReady: is a boolean value, used to control if the splash screen is shown or not.
- assetsPath: is a string value, where the backend URL is stored, changes between two values, depending on the environment, development or production, for the latter the value is "https://slides-backend.web.cern.ch".
- imgUploadRequest: is a boolean value, which becomes "true" when users want to upload an image to their presentation.
- saveRequest: is a boolean value, which becomes "true" when users want to save their presentation in a ".slides" presentation file.
- loadRequest: is a boolean value, which becomes "true" when users want to upload a ".slides" presentation file.
- styleRequest: is a boolean value, which becomes "true" when users want to change the background color of their presentation
- themeRequest: is a boolean value, which becomes "true" when users want to change the presentation's theme.

8.1.1.3 Deck

The deck includes all the information about each slide, what are the contents of each slide and which one is the current slide. The deck field contains:

- currentSlide.
- slides, an array of slides. Each slide contains an ID and an itemsArray. The itemsArray is an array of items which can be either text or image.
  - Text items are objects described by these attributes:
    - ID.
    - Position (x, y) in pixels.
    - Size (width, height) in pixels.
    - Focused: boolean value to determine if this item is clicked and focused.
    - type: this differentiates if the element is a text or image, for all text boxes it has the value: "TEXT".

        - Data: this is "text-specific" attribute that holds the text that is typed in the text box, in an HTML format, example: " Hello world ".
        - Edit: boolean value that controls if the text editor is visible or not.
    - Image items are objects described by these attributes:
        - ID.
        - Position (x, y) in pixels.
        - Size (width, height) in pixels.
        - Focused: boolean value to determine if this item is clicked and focused.
        - type: for all images it has the value: "IMAGE".
        - Src: this is the name of the image stored in the server, example: "95c085fd173a31f9375d878fc4c9bbf1_aristophanes.png".

8.1.1.4 Router

The router field is created by the react-router and holds routing information.

8.2 Libraries & dependencies

Many open source libraries were used in order for the application to have all these functionalities. The yarn package manager was used in order to install and manage all the dependencies. In order to better set up Slides' package architecture, the yarn workspaces were adopted. In this architecture, the "packages/" folder on the root of the project's directory holds the different parts of the application, as standalone packages of code. Using yarn workspaces makes it easy to install new dependencies and keep everything up-to-date and optimized.

8.2.1 Frontend

The frontend part is built on top of the Create React App environment, which provides the most essential set of packages to start a ReactJS application from scratch. A list of the most important complimentary packages used is:

- Spectacle: the presentation library [23].

- keycloak-js-react: the authentication adapter [24].
- draft-js: the text editor [16].
- axios: the facilitator of HTTP communication between backend and frontend [25].
- reactablejs [20] & interactjs [21]: the resizing and moving items library.
- semantic-ui-react: the UI design library [26].
- sweetalert2: the alerting system library [27].
- react-color: the color palette library [28].

8.2.2 Backend

The backend part is smaller, it is a NodeJS server. A list of the most important packages is:

- express: the web applications and APIs maker [29].
- axios [25] & cors [30]: the facilitator of HTTP communication between backend and frontend.
- fs-extra [31]: the file system library that executes operations like create, delete, move files and folders.
- keycloak-connect [32]: the backend authentication adapter.
- zip-folder [33]: the library used for zipping folders.
- extract-zip [34]: the library used to extract zipped folders.

8.3 Code Documentation

The best way to write code documentation for JavaScript projects appears to be the following:

- using JSDoc [35], a standardized way of writing comments in the code, in order to describe functions, classes, methods and variables. Example:

```
/**
 * Represents a book.
 * @constructor
 * @param {string} title - The title of the book.
 * @param {string} author - The author of the book.
 */
function Book(title, author) {
}
```

Figure 24: "JSDoc code comments example"

- then, Documentation.js [36] has to be used, in order to generate documentation based on all the comments written in the project. This tool takes the JSDoc code comments and transforms them into readable HTML or Markdown documentation.
- This Markdown documentation can be served by Mkdocs [37] (more on this on the Documentation chapter).
- Also, a tool called Flow [38] can be used, to make writing documentation comments easier. Using Flow, one can avoid writing the types of every function, because it extracts this information automatically. There is a ReactJS integration for Flow, which means that CERN Slides' App ReactJS frontend can use it in order to generate all the JSDoc comments, saving a lot of time from manual writing of comments.

9. USER AUTHENTICATION

User Authentication in a web application is a milestone that has to be implemented with great attention to detail, because it is security related and helps prevent unauthorized users from accessing server's resources. Authentication should never be implemented by the application itself, but instead all enterprises are supposed to establish a central authentication system that all of the applications will be using. At CERN, the authentication service is built on top of an open source software, Keycloak, which provides all of the basic and advanced features an authentication system needs. Both the SAML and the OIDC authentication protocols are supported. The CERN Slides' App is using the OIDC protocol. The new standard, and recommended by the authentication team, way of communicating with the CERN authentication.

9.1 Keycloak

Keycloak is an open source software product to allow single sign-on with Identity Management and Access Management aimed at modern applications and services. CERN is using this product, for managing all its authentication needs, which include:

- User registration.
- Social login.
- Single Sign-On/Sign-Off across all applications belonging to the same Realm.
- 2-factor authentication.
- LDAP integration.
- Kerberos broker.

9.2 OIDC

OpenID Connect (OIDC) is an identity layer on top of the authorization framework OAuth2.0. It standardized how authentication assertions are constructed and sent from the identity provider (CERN SSO) to the relying party (web application).

Applications register with the Identity Provider to establish trust. Information about the authenticated user is typically contained in JSON Web Tokens [39].

9.3 Client-side authentication

To add the authentication system to the CERN Slides' App, the application had to:

1. Be configured to require authentication with OIDC.
2. Be added to the Authorization Service.
3. Get its permissions scheme defined.
4. Be registered to use CERN SSO.

After all of this configuration was done, a React/Redux wrapper for the keycloak-js package had to be used, in order for the application to require authentication before it is accessed.

9.4 Server-side authentication

After the client side of the application is protected, it is time to secure the server as well. The REST server of the CERN Slides' App is fulfilling the requests of the client side, like 'Save Presentation', 'Load Presentation', 'Upload an Image', 'Delete an Image'. These are all sensitive operations and should all be executed only from the person who is editing the presentation and has all the permissions to do so. In order to ensure that no malicious actors can send these types of requests to the server, a keycloak adapter had to be used. Keycloak offers an adapter especially for NodeJS and this was used for the CERN Slides' App server protection.

10. STORAGE

The CERN Slides' App uses a dedicated Openshift project for its backend server. This backend server is responsible for all the storage related operations. These include:

- Upload an existing presentation.
- Deleting an existing presentation.
- Upload an image to be used in a presentation.
- Delete an image from a presentation.
- Check if a title already exists for this user.
- Rename a presentation.
- Save a presentation.

All these actions can happen only after a successful chain of events has taken place. For every operation, the user clicks a button in the UI (client-side), the application is gathering the required information. Depending on the action, this can be the username, a ".slides" presentation file, an old and a new title and always the authentication Bearer token. Right away, a REST request is formed (POST or DELETE) and sent to the server. There the server is accessing a Ceph mounted volume storage in the path, "/mydata/presentations" and executes the required operations, sending a response to the client side with the result.

10.1 Save presentation

To save a presentation, when the users click the "Save" button, the application will ask them if they want to change the presentation title, then a request will be sent to the server to check if there are not any other presentations from this user, with the same title. If everything is good, then the current presentation title changes internally in the application state, the redux state is stringified, and only the presentation and deck fields are kept (the keycloak and the router fields are discarded as they should not be stored in the ".slides" file). This stringified Redux state, that also contains the new presentation title (if updated by the user), along

with the username and the Bearer authentication token are sent as a POST request to the server.

When the request reaches the server, the stringified state is extracted along with the username.

1. The state gets written as a string in a new file with name: "/mydata/presentations/<username>/<title>/tmp/presentation.JSON".
2. Then the assets folder is copied from: "/mydata/presentations/<username>/<title>/assets" to "/mydata/presentations/<username>/<title>/tmp/assets".
3. The folder "/mydata/presentations/<username>/<title>/tmp" gets zipped and renamed to "/mydata/presentations/<username>/<title>/<title>.slides".

This is how the ".slides" file is created in the server side. Once it is ready, it is transferred to the client side as a response to the previous POST request, in a buffer format using the function (fs.readFileSync()). Once the response is sent, the ".slides" presentation file is deleted from the server.

Back to the client-side, if the request is successful, the file is converted to a blob, using the constructor (new Blob([<fileAsBuffer]>)) and is becoming available for download from the browser, showing a success alert to the user.

10.2 Upload presentation

After the users have downloaded their ".slides" presentation, they may need to re-upload it to the application at a later time, in order to continue editing the application or present it to an audience. There are two buttons that enable the user to upload an existing presentation, one is on the landing page and the other one in the editing mode of the presentation.

Uploading a ".slides" presentation file consists of the following steps:

1. The user selects the file to be loaded.
2. The client-side sends a POST request to the server, containing the username, the ".slides" file and the authentication Bearer token.
3. The server-side, saves the received ".slides" file as "/mydata/presentations/tmp-folder/<file.md5>_<file.name>". The file.md5, is

the md5 hash of the contents of the ".slides" file. The md5 hash of the file is used to avoid creating files with the same names, for presentations from different users that have the same presentation title.

4. The file is extracted in the folder: "/mydata/presentations/extract-folder_id_<file.md5>".
5. The presentation.JSON file is located in this folder, the server finds it and parses the JSON data into a JavaScript object.
6. The title is extracted from the object and the proper path is constructed in order for the presentation folder to be created. This path is: "/mydata/presentations/<username>/<title>"
7. The assets subfolder is copied over, from the extracted folder to the presentation folder.
8. The loaded Redux state object, that was read from the presentation.JSON file, is sent as a response to the POST request and reaches the client-side.
9. The unwanted files and folders are deleted from the server, like the: "/mydata/presentations/tmp-folder/<file.md5>_<file.name>" and the "/mydata/presentations/extract-folder_id_<file.md5>".
10. The client-side has to update its Redux state. For this reason, the deck and the presentation fields of the Redux state are updated with the new information from the server's response. This information is about the names of the images in each slide, the text that the slides contain, the background color, the presentation title and much more.
11. Lastly, the URL is also updated to match the new presentation title.

11. SLIDESHOW

The CERN Slides' App has two modes, the edit mode and the presentation mode or Slideshow. To go in the presentation mode, one needs to click the "eye" button, this changes the URL from: "https://slides.web.cern.ch/edit/<username>/<title>/" to "https://slides.web.cern.ch/present/<username>/<title>/".

Users are not allowed to change the URL manually to trigger this change of modes for two reasons. Firstly, because the goal of this application is to make everything as easy as possible for the users and clicking the "eye" button is the easiest way to enable presentation mode, so users should stick only to this way. Secondly, all the information about the presentation lives on the Redux store of the application, reloading the window or changing the URL and loading the "/present" route will erase all the information for the current presentation.

When the user is in presentation mode, the two sidebars disappear and only the Spectacle <Deck /> component which includes all the presentation slides is shown. Also, all the items stop being clickable, deletable or interactable in any way. The text editor is not showing up when the user double clicks the text area, and the images are not resizable and movable anymore. The user is now essentially seeing his spectacle presentation only.

12. EXPORT

One of the most wanted features from the user community, has been the export functionality. Creating, editing and presenting a ".slides" presentation with the CERN Slides' App, is not enough. People need a presentation, in a form that they can use offline and they are sure it is always going to stay like it is, without any changes. The first file formats that we thought of were the HTML and the PDF file.

When the Spectacle presentation library was selected, the export functionality was in the project's requirements list already. So, while reading Spectacle's documentation, figuring out that they had a way to export their presentations in PDF files, was a great relief.

From the beginning of the CERN Slides' App, until now, the Spectacle library, made a major version upgrade, which means that a big part of their code was rewritten. The new version is not backwards compatible. That means upgrading to the new version will not work out-of-the-box, some adjustments have to be made, after the new version is adopted. The export-to-PDF functionality is quite different, between the version of Spectacle currently being used in the CERN Slides' App, and the latest available.

12.1 Export in Spectacle

To export a spectacle presentation into a PDF file, according to ("https://github.com/FormidableLabs/spectacle/tree/v5.7.2#pdf-export"), one must:

1. Add "?export" after the "/" on the URL of the page the application is running on, e.g.: "http://localhost:3000/#/?export".
2. Bring up the print dialog (ctrl or cmd + P)
3. Change destination to "Save as PDF", as shown below:

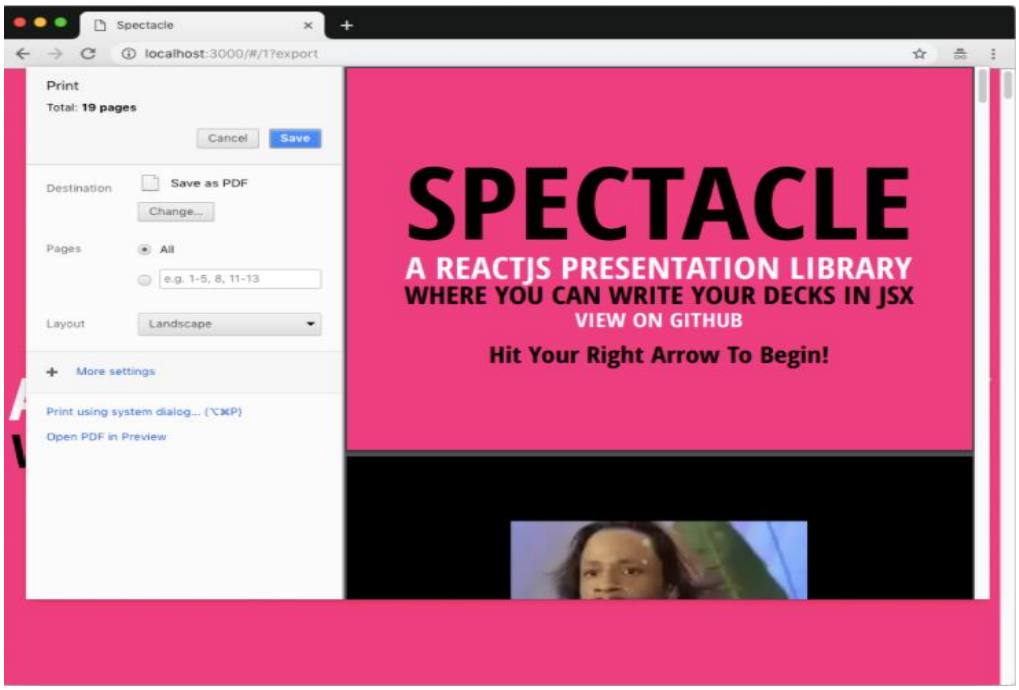

Figure 25: "Export a presentation to PDF using Spectacle"

This is now changed and with the latest version of spectacle instead of the "?export" URL parameter, one should add "?exportMode=true" and then continue with the same steps as above.

12.2 Export in the CERN Slides' App

The ideal for the CERN Slides' App, would be to create a route, which will only renders the Spectacle <Deck />, exactly like in Slideshow mode, and add the export URL parameter to the route as well, and automatically, do the procedure of opening the print dialog, pass the proper print parameters in order for the PDF to be exported nicely, and download the PDF file of the presentation in the local disk of the user, without any interaction by them. This feature is currently under active development.

13. ENVIRONMENT

For the CERN Slides' App to be available to the audience, it should go to production. This means that the frontend ReactJS application and the NodeJS application should be running in 2 docker [40] containers on a cluster management system.

13.1 Openshift

At CERN the Openshift Platform as a Service (PaaS) product is being used to orchestrate and manage docker containers using Kubernetes [41]. CERN has its own self-hosted/self-managed Openshift instance to host and serve CERN applications as docker containers. One can synchronize its application releases to send a new version of the application in the form of a docker container to Openshift which will then serve the new version to the users that navigate to the application URL. In Openshift one can mount storage of different types, store application secrets and environment variables safely and manage application's URLs, aliases and more.

13.2 GitHub

GitHub is a company that provides hosting for software development version control using Git. In the CERN Slides' App, it is used alongside GitLab, keeping a mirror of the git project. This is useful, because the code in GitHub is open to the public, making the CERN Slides' App an open source project. The code can be found here: https://github.com/CERN/slides.

13.3 GitLab

GitLab is a web-based DevOps lifecycle tool that provides a Git-repository manager providing wiki, issue-tracking and continuous integration/continuous deployment pipeline features, using an open-source license, developed by GitLab Inc. CERN has its own self-hosted/self-managed GitLab instance running on CERN servers

and is only accessible to CERN users. This is the platform most of the projects are using because of its custom features tailored for CERN applications, like the ci/cd pipelines and the good integration with the Openshift platform. In the CERN Slides' App git repository, there is a master branch and a dev branch, where the new features are pushed tested and only when everything is working as expected, the code is transferred to master. Once the code is at the master branch, the production version of the application gets updated.

13.4 Deployment Process

A gitlab-ci.yml file is written, in order to construct and fire the pipeline. It consists of two stages, one is the docker_build and the next is the deploy. In the docker_build, the instructions of the frontend.Dockerfile and backend.Dockerfile are used, in order to create two environments (containers) and start the two applications, the frontend and the backend accordingly. When they are built successfully, the script takes them and sends them as a new version of the application in both of the Openshift projects (frontend, backend). There, the two containers expose the applications running in the two ports, and are made publicly accessible from the URLs:

- https://slides.web.cern.ch
- https://slides-backend.web.cern.ch

For them to communicate, the frontend has hardcoded the backend URL in its code and is using this URL in order to send API requests.

14. DOCUMENTATION

At CERN, the norm is that all of the applications should have two websites for user facing documentation with names:

- <application_name>.docs.cern.ch
- <application_name>-admin.docs.cern.ch

The first one is for the user guide of the application and the second one is documentation targeted to the developers and maintainers of the project. In the CERN Slides' App, the two documentation websites are:

- https://slides.docs.cern.ch
- https://slides-admin.docs.cern.ch

14.1 MkDocs

The documentation websites are built with MkDocs, a static site generator designed for building documentation sites. Documentation source files are written in Markdown and configured with a single YAML configuration file. Openshift has an integration for MkDocs sites, so it is very simple to host the project's documentation following a simple guide. Markdown syntax is straightforward, and the result of the documentation is easy to read and to navigate among the different chapters.

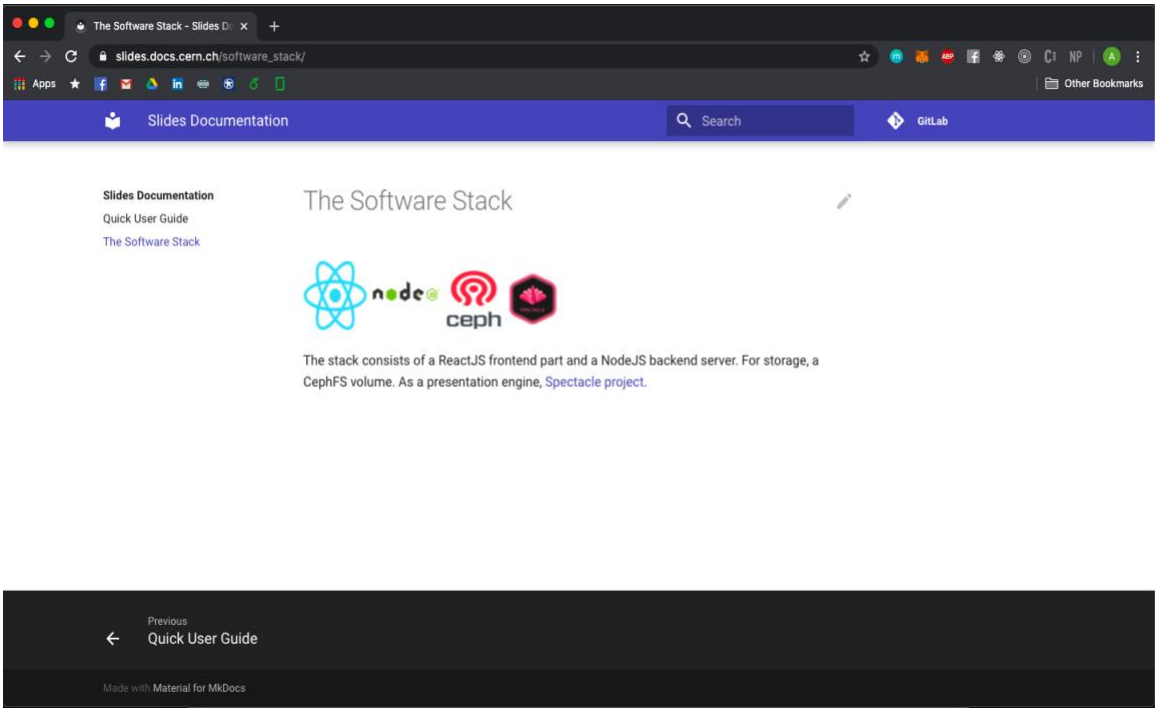

Figure 26: "Documentation website screen"

14.2 E-groups

At CERN, e-groups is the term for mailing lists. The CERN Slides' App has its own e-groups:

- "slides-support", which intends to represent the group of developers, maintainers and project owners as an entity. People can send feedback and questions to the "slides-support@cern.ch" and are redirected to the respective emails of the ones responsible for the project.
- "slides-admin", is the e-group which contains the project owners, who have all of the admin privileges to execute actions for the CERN Slides' App. This e-group is the one that has privileges to change the authentication configuration from the application-portal. Changes there reflect to which logins are valid to access the https://slides.web.cern.ch application.

15. FUTURE WORK

The CERN Slides' App is progressing, and more work is needed in order to become an appealing product that users will prefer to use over other alternatives such as CodiMD slides, Microsoft PowerPoint and LibreOffice Impress.

15.1 Open source

The CERN Slides' App is built exclusively using open source libraries and the purpose is to make it an open source application that other institutes, organizations or individuals will want to contribute and improve. This mindset of community and openness is the one that CERN promotes and supports. Opening the application to the world and seeing others collaborating and adopting it, would be the ideal future of the CERN Slides' App.

15.2 Additional features

A list of features for future development include:

- Improvements and new CERN themes.
- Add Image as a slide background.
- Master slide and default slides (text-only, text-image, image-only).
- Insertion of tables and shapes.
- Embed video in a slide and play it without exiting the presentation mode.
- Import images from users' CERNBox.
- Ability to include presentation notes for the presenter.
- Header and footer in a slide.
- Code and mathematical expressions inclusion in a slide.

15.3 Integrations

Two are the main integrations that are planned to happen in the near future for the CERN Slides' App.

15.3.1 Phoenix integration

Completing this, means that the users will be able to create presentations with the CERN Slides' App directly from their cloud storage space. The ".slides" files will be stored and backed up automatically online, giving the users ease of mind and quick access to their presentations from different devices.

15.3.2 Indico integration

Indico [42] is used excessively throughout CERN for event planning and material sharing. After the integration, users will be able to upload ".slides" presentations as materials in Indico events, and the participants could just click the file and be redirected to the CERN Slides' App in presentation mode. This saves users a lot of trouble and time from downloading the ".slides" file from the Indico event, uploading it on the CERN Slides' App and triggering the presentation mode manually.

16. OPERATIONAL REQUIREMENTS

To summarize, the CERN Slides' App has reached a state that the future looks optimistic, with a working proof of concept and users that have created presentations with it. To process to operation several questions remain open:

1. Under which hierarchical authority at CERN the application will be supported and maintained?
2. Which method should be used to attract users?
3. How many additional features can we afford to accept from the feedback received?
4. What if the open source community requires technical exchanges for the product evolution and we have no resources for these creative activities?
5. What if dependencies, prerequisite packages and plugins move on and things break in the code?
6. What if security vulnerabilities arise and nobody discovers them on time?
7. What if new browser versions create broken views?

17. CONCLUSIONS FROM THIS EXPERIENCE

17.1 Conclusions for the application

Creating a software application from scratch can be tricky. There are a lot of parameters that the developers should take into consideration, different teams that have to cooperate and challenges that need teamwork to be solved. Asking for feedback from the users of the application is crucial to pave the way of the application development. Also, be adaptable to new user requirements and software limitations is something to always keep in mind. As it always happens with web applications, the user views received sometimes conflicting feedback due to personal taste and habits, so the optimal solution was not always obvious. Another challenge that was faced, was the pessimism of the environment. In the beginning, some colleagues thought the task is gigantic, undoable and unachievable considering the time and resources available.

17.1.1 Time consuming

The most challenging parts of the application development were the Text Editor, the resizing and moving of the text boxes and images, and the image storage in the production server. These milestones were necessary but difficult to be implemented. The CERN Slides' App, as a full stack web application required work in many areas, from frontend ReactJS code, to CSS, to API design, to proper authentication and securing of the application, to image transfer and storage in the server, to deployment and docker container knowledge. It touched many areas of expertise, which had to be learned. The best coding practices had to be followed in order to assure the longevity of the project.

17.1.2 Good result

The most important thing is that the initial idea/inspiration of the application design turned out to be possible. Rendering a Spectacle <Deck /> component alongside two sidebars that control what is shown in the <Deck /> actually worked. The use

of Spectacle library is the best solution and the approach that was taken in the beginning was the best possible. The result is a working prototype of a web application that one can use to create presentation slides, present them using a web browser and export them as PDF files for offline presentations. The initial goal has been achieved and the future seems bright, a lot more improvements and features are yet to come.

17.2 Personal conclusions

This experience has been very powerful to me. I feel that I grew as a person and as a professional. This experience required a lot of technical knowledge that had to be acquired along the way, it was a learning journey and my supervisor as well as my work colleagues were there with me, to support my effort.

From the first meeting at CERN, I felt that my opinion matters and even though I had an intern position, the team was asking to consult my point of view. That was very motivating and kept me focused on doing a good job.

During my contract with CERN, I had to manage my time effectively, to learn how to be productive working both in a team and alone, to find creative solutions to difficult problems, to find the right people to ask for help and to not give up when things get tough.

After this experience, I feel more comfortable about my technical and personal skills. I feel so lucky to have worked at CERN, in this specific project.

ABBREVIATIONS – ACRONYMS

ΕΚΠΑ	Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών
NKUA	National and Kapodistrian University of Athens
CERN	European Organization for Nuclear Research
IT	Information Technology
PDF	Portable Document Format
UI	User Interface
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
LHC	Large Hadron Collider
PB	Petabytes
API	Application Programming Interface
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
OIDC	OpenID Connect
JSON	JavaScript Object Notation
MD5	Message Digest algorithm
WYSIWYG	What You See Is What You Get
URL	Uniform Resource Locator
GIF	Graphics interchange Format
App	Application
SAML	Security Assertion Markup Language
LDAP	Lightweight Directory Access Protocol
YAML	Yet Another Markup Language

REFERENCES

[1] "ReactJS," [Online]. Available: https://reactjs.org/. [Accessed 29 06 2020].

[2] "NodeJS," [Online]. Available: https://nodejs.org/en/. [Accessed 29 06 2020].

[3] "CodiMD," [Online]. Available: https://demo.codimd.org/. [Accessed 29 06 2020].

[4] "Phoenix," [Online]. Available: https://github.com/owncloud/phoenix. [Accessed 29 06 2020].

[5] "Owncloud," [Online]. Available: https://github.com/owncloud/core. [Accessed 29 06 2020].

[6] "WOPI Server," [Online]. Available: https://github.com/cs3org/wopiserver. [Accessed 29 06 2020].

[7] "CS3-REVA," [Online]. Available: https://github.com/cs3org/reva. [Accessed 29 06 2020].

[8] "CS3 APIS," [Online]. Available: https://github.com/cs3org/cs3apis. [Accessed 29 06 2020].

[9] "EOS," [Online]. Available: https://github.com/cern-eos/eos. [Accessed 29 06 2020].

[10] "CKEditor," [Online]. Available: https://ckeditor.com/. [Accessed 29 06 2020].

[11] "Froala," [Online]. Available: https://froala.com/wysiwyg-editor/. [Accessed 29 06 2020].

[12] "TinyMCE," [Online]. Available: https://www.tiny.cloud/. [Accessed 29 06 2020].

[13] "Quill," [Online]. Available: https://quilljs.com/. [Accessed 29 06 2020].

[14] "Summernote," [Online]. Available: https://summernote.org/. [Accessed 29 06 2020].

[15] "React-draft-wysiwyg," [Online]. Available: https://github.com/jpuri/react-draft-wysiwyg. [Accessed 29 06 2020].

[16] "draft-js," [Online]. Available: https://draftjs.org/. [Accessed 29 06 2020].

[17] "Ceph," [Online]. Available: Available: https://ceph.io/. [Accessed 29 06 2020].

[18] "react-rnd," [Online]. Available: https://github.com/bokuweb/react-rnd. [Accessed 29 06 2020].

[19] "react-grid-layout," [Online]. Available: https://github.com/STRML/react-grid-layout. [Accessed 29 06 2020].

[20] "reactablejs," [Online]. Available: https://github.com/beizhedenglong/reactablejs. [Accessed 29 06 2020].

[21] "interactjs," [Online]. Available: https://interactjs.io/docs/#. [Accessed 29 06 2020].

[22] "Bearer token," [Online]. Available: https://swagger.io/docs/specification/authentication/bearer-authentication/. [Accessed 29 06 2020].

[23] "Spectacle," [Online]. Available: https://github.com/FormidableLabs/spectacle. [Accessed 29 06 2020].

[24] "keycloak-js-react," [Online]. Available: https://www.keycloak.org/docs/latest/securing_apps/#_javascript_adapter. [Accessed 29 06 2020].

[25] "axios," [Online]. Available: https://github.com/axios/axios. [Accessed 29 06 2020].

[26] "semantic-ui-react," [Online]. Available: https://react.semantic-ui.com/. [Accessed 29 06 2020].

[27] "sweetalert2," [Online]. Available: https://sweetalert2.github.io/. [Accessed 29 06 2020].

[28] "react-color," [Online]. Available: https://casesandberg.github.io/react-color/. [Accessed 29 06 2020].

[29] "express," [Online]. Available: https://github.com/expressjs/express. [Accessed 29 06 2020].

[30] "cors," [Online]. Available: https://github.com/expressjs/cors. [Accessed 29 06 2020].

[31] "fs-extra," [Online]. Available: https://www.npmjs.com/package/fs-extra. [Accessed 29 06 2020].

[32] "keycloak-connect," [Online]. Available: https://www.npmjs.com/package/keycloak-connect. [Accessed 29 06 2020].

[33] "zip-folder," [Online]. Available: https://www.npmjs.com/package/zip-folder. [Accessed 29 06 2020].

[34] "extract-zip," [Online]. Available: https://www.npmjs.com/package/extract-zip. [Accessed 29 06 2020].

[35] "JSDoc," [Online]. Available: https://jsdoc.app/. [Accessed 29 06 2020].

[36] "Documentation.js," [Online]. Available: https://documentation.js.org/. [Accessed 29 06 2020].

[37] "MkDocs," [Online]. Available: https://www.mkdocs.org/. [Accessed 15 07 2020].

[38] "Flow," [Online]. Available: https://flow.org/. [Accessed 29 06 2020].

[39] "JSON Web Tokens," [Online]. Available: https://jwt.io/. [Accessed 29 06 2020].

[40] "Docker," [Online]. Available: https://www.docker.com/. [Accessed 15 07 2020].

[41] "Kubernetes," [Online]. Available: https://kubernetes.io/. [Accessed 15 07 2020].

[42] "Indico," [Online]. Available: https://getindico.io/. [Accessed 29 06 2020].