Creating an active Interface for MathHub.info

Bachelor Thesis

Johannes-Sebastian See
Supervisor: Michael Kohlhase
Co-supervisor: Tom Wiesing
Friedrich-Alexander University, Erlangen Nürnberg, Germany

July 30, 2019

Abstract

Abstract will be added at the end

Contents

1   Introduction

For many years, the internet has been a crucial part of mathematical education, research and application. With time people created more and more websites with a common goal: spreading mathematical knowledge. Every one of these sites has its own unique approach of imparting this complex subject. For example **Wolfram|Alpha** [Wol] is one of the most popular tools today. It has a large and still growing database of algorithms and methods to compute and represent the result of countless mathematical problems. Furthermore, the **Stack Exchange Network** has two very active Q&A websites where mathematicians assist each other. While **Math.stackexchange** has its focus set on education, the community of **MathOverflow** deals with more advanced questions that come up during research. Even the universally known encyclopedia **Wikipedia** plays an important role, since its often the first place people look up formulas, algorithms etc.. ¹

Wikipedia and Stack Exchange have a rather informal approach in representing knowledge while Wolfram|Alpha lays its focus on formal math. But none of those are really suited for mathematical documents that combine formal and informal knowledge. **MathHub**[Koh] was created to collect documents with varying degrees of formality. To make MathHub a valuable contribution to mathematics it needs an User Interface called MathHub.info. So this thesis describes the individual parts of building an easy and intuitive to use UI that can handle the different levels of formality while being able to adapt to changes in the MathHub library.

It starts with a description of the MathHub system and its previous realization in Section 3. Then Section 4 presents some frameworks that can be used to create a frontend. Following up on this, Section 5.1 and 5.2 take a closer look at React while **??** gives a short introduction to the content of MathHub. In section 6 the inner workings of the frontend are discussed. Afterwards in section 7 and 8 the design of the individual pages are discussed. Section 9 concludes this thesis, while Section 10 takes a look at some plans for the future.

2   Preliminaries - MMT and OMDoc

The abbreviation MMT is short for either <u>M</u>eta-<u>M</u>eta-<u>T</u>heory or <u>M</u>eta-<u>M</u>eta-<u>T</u>ool. Whereby meta-meta-theory represents the theoretical and meta-meta-tool the practical part of MMT. It is a knowledge representation framework that uses a combination of formal and informal languages to create a scalable module system for mathematical theories. That means that in MMT the features of the syntax and the semantics of a language are defined as individual, reusable modules. This way of building individual languages leads

---

¹EDNOTE: more: Zentral Blatt Math; mlfdb; isu-afp.org; Wikidata

to a high degree of abstraction of advanced algorithms.[Rab18]

MMT builds on the OMDoc representation, which is a philosophy about the design of an uniform language for knowledge. More specifically MMT uses an XML format which follows this philosophy.
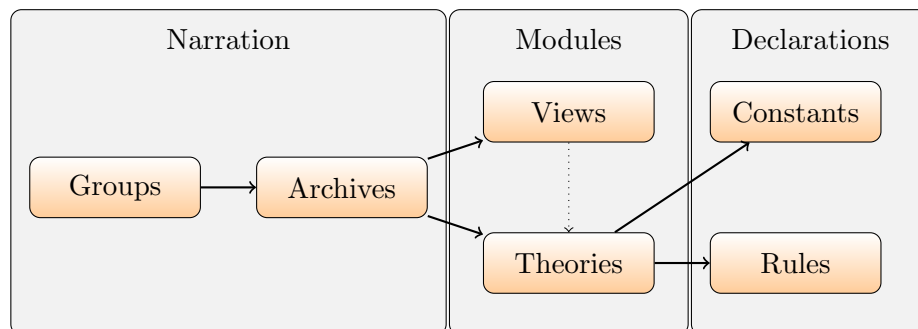

Figure 1: The MMT structure

To create a frontend that displays MMT it is important to get familiar with its structure, that can be seen in figure 1. First up on the highest level are the individual groups. Each group contains several archives. An archive can be described as a collection of documents that typically are equivalent to a software project. So it provides a work flow for a language in MMT. Groups and archives only exist for navigation and narration purposes. The modules that can be found in an archive, make up the actual content. These are either a theory or a view that shows the relation between different theories. A theory is defined by its rules and constants, the so called declarations. [Rab]

3   MathHub

MathHub.info is a portal for the active mathematical documents of MMT. As stated before each one of these documents has a different degree of formal and informal content. The representation of this data, knowledge and structure with varying degrees of formality is called **flexiformal**. But realizing machine support for this approach of representing knowledge comes with a high formalization cost. So it can be quiet challenging to find the right balance between costs and machine support.

3.1   Previous Implementation

Before figuring out how to create a new frontend, it is advantageous to understand the previous realization of MathHub which can be seen in Figure 2. The source documents were stored, edited and converted on the GitLab repository manager. There the documents are organized into their proper

repositories and converted from their source formats into OMDoc/MMT. These documents could be edited by their authors in a working copy of the GIT repository. From there the MMT API could load the documents and send them to the frontend. The frontend itself was implemented with Drupal[KW14], while user interactions were handled by the JavaScript modules in the JOBAD framework. Drupal is an open source content-management framework that also supplies uniform theming and is currently used by millions of different websites. The reason Drupal was used for the User Interface up until April 2018 was that it has an integrated rights-management.
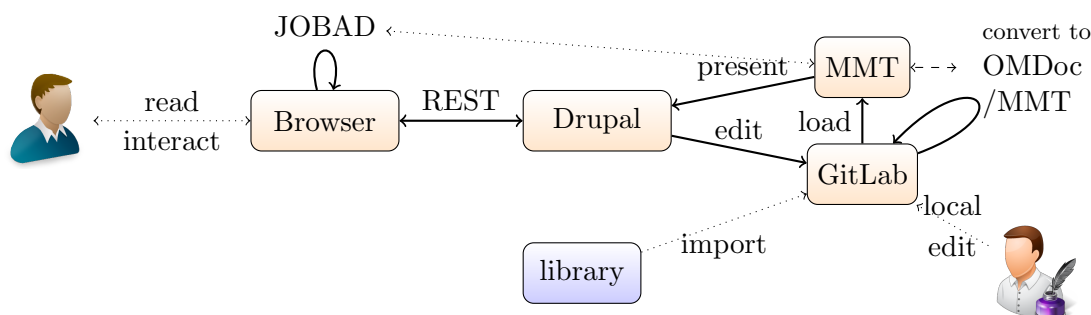

Figure 2: old MathHub architecture

Initially it sounded like a good idea to have rights handled directly by the frontend. But with time it showed that it is easier to use the rights-manager provided by GitLab. So Mathhub.info still had to deal with the huge overhead that is necessary to use Drupal without benefiting from its features. This created a lot of unnecessary and tedious extra work.

3.2   Security Issues

Additionally a critical security flaw in the versions 6 to 8 went public in April 2018. The problem was that Drupal in these versions accepts request parameters without any validation. This means it processes any input from anybody [Tun18]. To exploit this weakness an attacker doesn't even need to log in or have any other privileges on a vulnerable website [McC18]. With this flaw it is possible to inject malicious code and compromise a website in multiple ways. This can be used to access, change and delete private data and create backdoors to make future attacks possible. The Drupal community called this weakness "Drupalgeddon2" while its official name was "CVE-2018-7600". Some code that was injected installed the program XMRig Monero miner, which is a cryptocurrency mining program, as well as deleting other mining programs on the compromised system [Kum18]. The National Institute of Standards (NIST) and Technology gave Drupal a "Highly Critical" Rating because of this vulnerability [NIS]. After this

flaw was discovered a patch was published and a warning to update every website that used a vulnerable version was given.

Since there have been multiple flaws in Drupal before that compromised MathHub.info, the decision to stop using it and rebuild the frontend from ground up with a lighter and easier to use framework to not be affected by future attacks, was made.

4   State of the art - Building an interactive Frontend

Before starting to rebuild MathHub.info a new web framework has to be chosen. Lets take a look at some options.

Polymer [Wika] is an open source JavaScript library developed and maintained by Google. It uses the native technologies of a browser instead of large custom JavaScript libraries. This makes Polymer remarkably fast on Chrome and Safari. It also provides a set of features that makes it easy to create custom elements, that work like standard web components. It is used for several Google services for example Youtube, Google Earth, Google Play Music etc. as well as Netflix, Electronic Arts and many other companies.

Another open source web framework from Google is **Angular**[Wikb]. This TypeScript library has framework architectures that simplifies the development of new web applications. Angular transfers all content of a website at once and leaves the task of building the page to the browser. This significantly speeds up the loading process and search optimization while keeping the communication with the server at a minimum. It also has Angular Material, a collection of UI components that work in browsers, on mobile and desktop.

After using Angular on several Google projects, Evan You decided to create his own JavaScript framework called **Vue.js** [Wikc]. Depending on the project it can be scaled between a framework and a library. Vue.js separates its view layer library from its support libraries for complex applications, to create an easy approach to the framework. It also automatically keeps track of the dependencies between components. So the system knows what exactly has to be rerendered when the page changed[Git].

In the end the decision was made to use **React** developed by Facebook. React is a component based framework that uses a virtual DOM to avoid unnecessary rerendering. It was chosen because the focus on reusable components makes it perfect for a large and still growing system like MathHub. [2] Further details about React and how it is used can be found in section 5.1.   EdN:2

---

[2]EDNOTE: Make reference to the question from the intro

5 React

5.1 The core concept of React

React is an open source JavaScript library owned and maintained by Facebook. It was created to build interactive user interfaces (UI), for example Facebook and Instagram. What makes React unique is its use of a virtual Document Object Model (DOM). The concept of the virtual DOM is that updating a website does not rerender everything. React computes the differences between the last and the next page and only changes the necessary parts. On top of that it also has conditional rendering which means that an item will only be rendered if it is shown. The advantage of the virtual DOM and conditional rendering is that this makes updating a website fast, but it comes with high RAM costs. The actual interface is made up of many different elements and components. A website that uses React can have multiple unique features so it is helpful to build new components to realize these features[Inca]. Further details on how to create components are discussed in Section 5.2.

Since React was designed by Facebook, there was a need for a design pattern that handles a large code base better than MVC (Model View Controller). So they created an updated and improved version of the MVC and called it Flux. The philosophy behind the Flux design is that data flows only in one direction to create unidirectional cycles. [Incb]

React does not have a styling system so its community created the Semantic UI React library to provide an easy way to maintain a consistent theme throughout the frontend. Of course it is still possible to use a different library or create new styling components but MathHub.info mostly uses Semantic UI React.

5.2 Building new components in React

This section provides insight into the work that takes up the majority of the time of creating, updating or amending to a part of the MathHub.info frontend. The actual interface that can be seen in a browser is the combination of many individual components. This way, on an update, the virtual DOM can check each component for differences, so only the affected ones have to be rendered again.

React already has a large external library with a lot of different components, but it is often necessary to make new ones that have a desired functionality. In JavaScript new components can either be implemented as a function or as a class. Their inputs are called props. To ensure the unidirectional design of React and prevent unwanted side effects in other components, props are read-only. In addition to props each component can have an internal state. Whenever a value in this state changes the component is rendered again. Naturally a component can grow big rather quickly.

As it is possible to use components inside other components it is smart to split a big component into multiple smaller ones. This has the advantage that components can be reused in many different locations with various purposes. The actual output often has a tree structure with several levels. In this tree the nodes are the individual components and the leaves are the elements that are directly rendered by the DOM. A good example for a tree with multiple levels is the Glossary Page of MathHub.info (more details about the glossary can be found in section 6.2 and 8.1), which can be seen in Figure 3. In this example the headline can be rendered directly while the Header and the Entries are the child components of the glossary. They both have their own elements and children and so forth resulting in this tree structure.
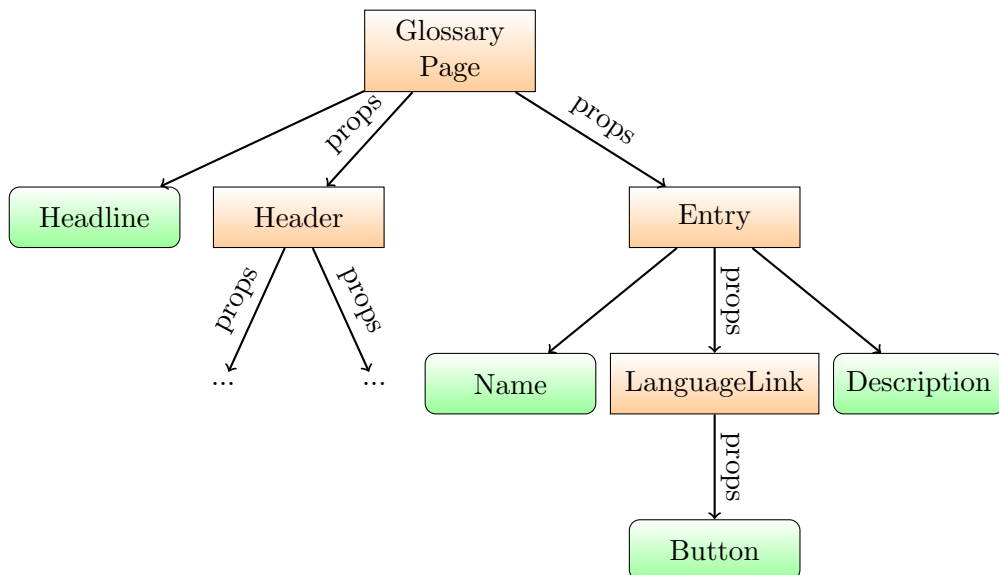

Figure 3: The React component tree of the Glossary Page

The unidirectional design of React has the consequence that an update of a component can only affect its children. But sometimes it is necessary for a component to change something on a higher level in the tree. In this case it is possible to "lift up" the state. This means moving the value that initiates the change to the state of the higher node. This node then has to give it back to its child as a prop.

Sometimes an update of a state should affect other components on the same level. As shown in figure 4, this can be done by lifting up the state to a new node that has all the components as children, that are affected by the update. [Incc]
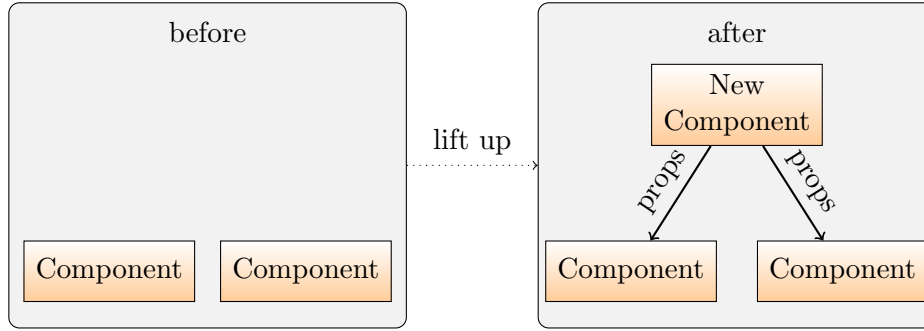
Figure 4: Lifting up the state to a new component

The scalability and the ability to adapt to change of this design makes the React framework perfect for a Math Information System like MathHub.info. Creating new components and lifting up the state makes it easier to to add new functionalities and adjust the representation without creating the need to rebuild entire sections of the frontend.

6   The Architecture of MathHub

6.1   Realization

Now that a new framework has been chosen lets see how this changes Math-Hubs architecture that was introduced in section 3.1. The library is still stored and converted into OMDoc/MMT in GitLab. But now the documents are loaded by a new plugin of an MMT server that provides the user with several semantic services in addition to the actual content. The plugins responses to the requests of the React based frontend follow the constraints set by REST (Representational State Transfer) and use the JavaScript Object Notation (JSON). The use of the Semantic-UI-React TypeScript library provides an universal theme throughout MathHub.info. This results in the new architecture displayed in Figure 5.
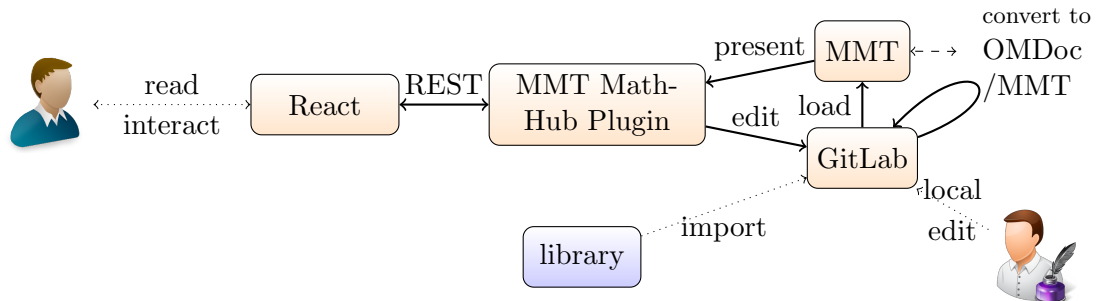

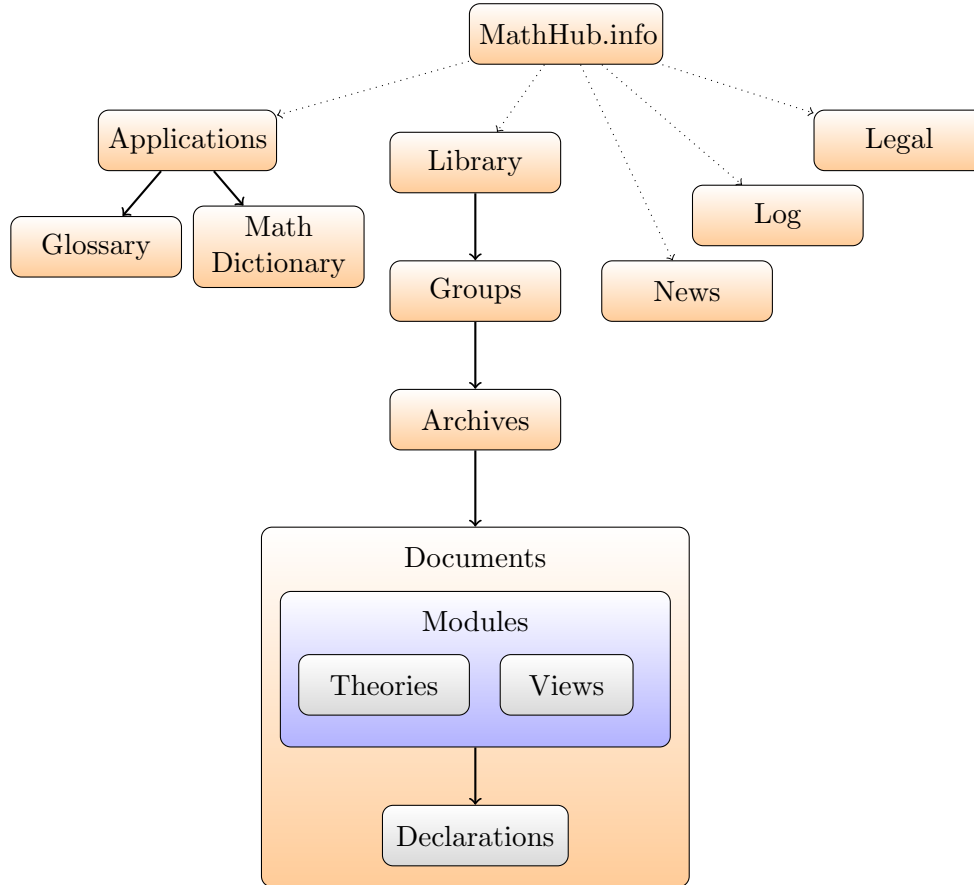Figure 5: MathHub architecture

6.2 MathHub.info Routes

Figure 6: MathHub routes

To create a logical navigation system MathHub.info uses routes to divide its content into numerous pages, that can be accessed from the **Homepage**. The structure of these routes can be seen in figure 6.

First of is the main content of MathHub, the **MMT library**. It starts with the different **groups**. The next page displays the **archives** that belong to a specific group. Groups and archives exist only for narration and navigation purposes. So they don't have any actual content only meta information like names and descriptions. The actual content can be found in **documents**. These document consist of multiple modules (i.e. theories and views) as well as opaque elements that give the user additional information about the theories. So the archive-page has a list of all the documents in an archive. On the document-page the modules are displayed with all their declarations.

With the huge mathematical library of theories there are a lot of technical

terms. The **glossary** is a collection of these expressions. There wouldn't be much value in to just having a collection without any additional features. So the glossary also provides a definition for each term. Over the time many different authors have contributed to the theories, so it can happen that there are different terms that share a meaning. These synonyms can also be found in the glossary. Since many theories exist in multiple languages it makes sense to have a glossary available for every used language. Currently the biggest collection of terms are in the English glossary, followed by German and French. Smaller collections for Turkish and Romanian are also available as well as simplified and traditional Chinese.[Gin+]

Most of the times a user does not want to browse through the gigantic glossary to just find a single term. This is the reason why the **Math Dictionary** is a useful extension of the glossary. The main purpose of the Math Dictionary is to translate a term into another language and look up a definition of a specific expression.

There is also a page with all the latest **news** regarding MathHub, as well as a couple of static pages like **licenses** and the **privacy policy**. At last there is a **Log** with the most recent messages from the backend.

6.3   Communication with the Backend

As previously mentioned the frontend does not have any actual content. It gets the data from the MMT backend. The frontend has several clients that each communicate with a part of the server when their specific functionality is needed. The different clients are:

- a Library Client for everything related to the content of the library

- a Glossary Client that gets all the terms in a specific language

- a Translation Client that is used by the Math Dictionary to search for a translation of a term

- a News Client

- a Log Client

The answers that are received from the backend use the JSON format. The data objects in JSON consist of attribute-value pairs and arrays. The frontend then uses these objects as props for React components.

6.4   Layout

Every page consists of three parts: A header, a footer and the actual content in between. This is illustrated with the Homepage as an example in figure 7.
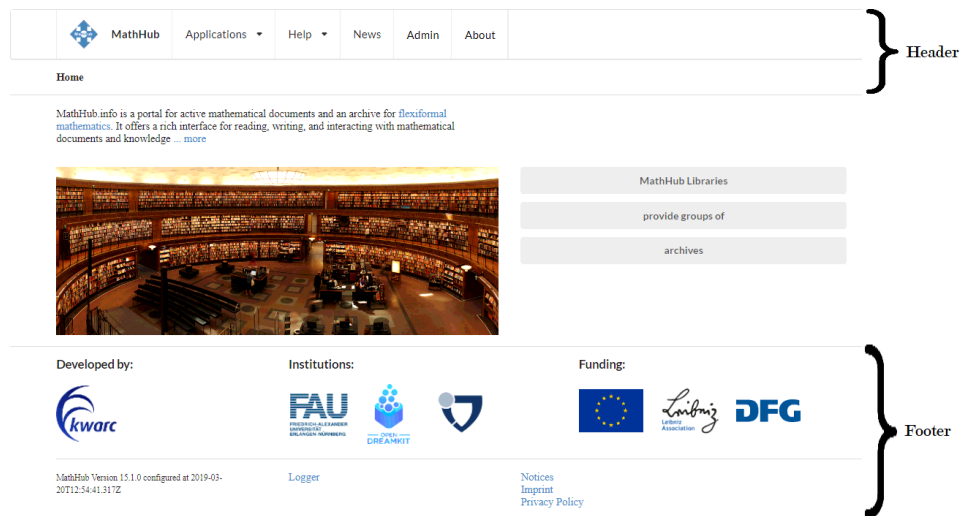
Figure 7: The Homepage

The header is a menu with the routes to Home, the news, the glossary and the Math Dictionary as well some external links. Under the menu there are breadcrumbs for an easier navigation. In the footer the logos of the institutions that are involved in MathHub can be found. There also are the routes to the Log, the licenses, the imprint and the privacy policy. The body itself depends on the page the user is currently on.

7   MathHub Library Components

In this section the implementation of the different components of the Math-Hub library is discussed. The first page of the library is a list of all the groups of MathHub. Every group is rendered in its own React component that has the name of the group, a short teaser and links to the corresponding group-page.
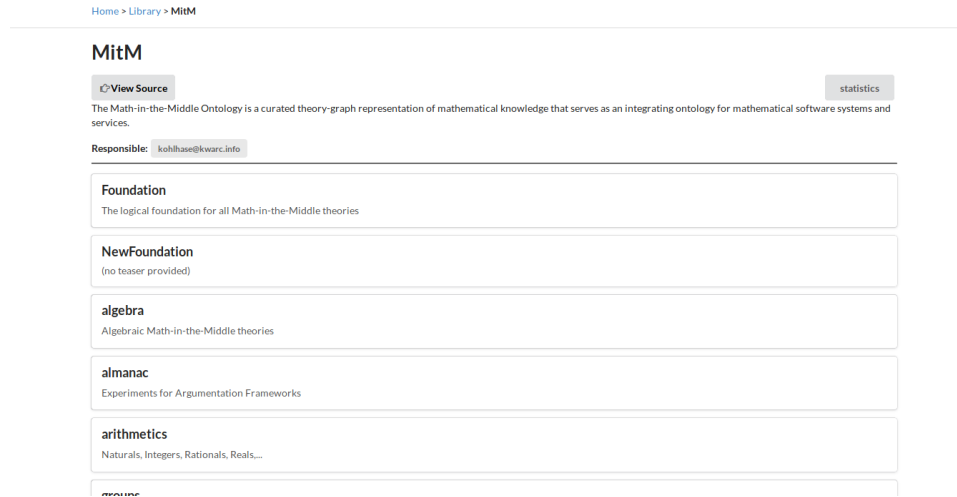
7.1 Groups

Figure 8: a group in the frontend

As an example figure 8 shows a screenshot of the MitM group as it is currently depicted on MathHub.info. Above the numerous archives the group page starts with a header that has the same structure for every group. It begins with a button that links to its source files on GitLab. After that follows a description that gives an overview of the groups content. The header ends with a list of e-mail addresses of the people that maintain the group.

Beneath that there is a list of all the corresponding archives. Every archive is its own React component that consists of a name and a short teaser that summarizes its content for the user. By clicking on an entry the user is taken to the corresponding archive-page.
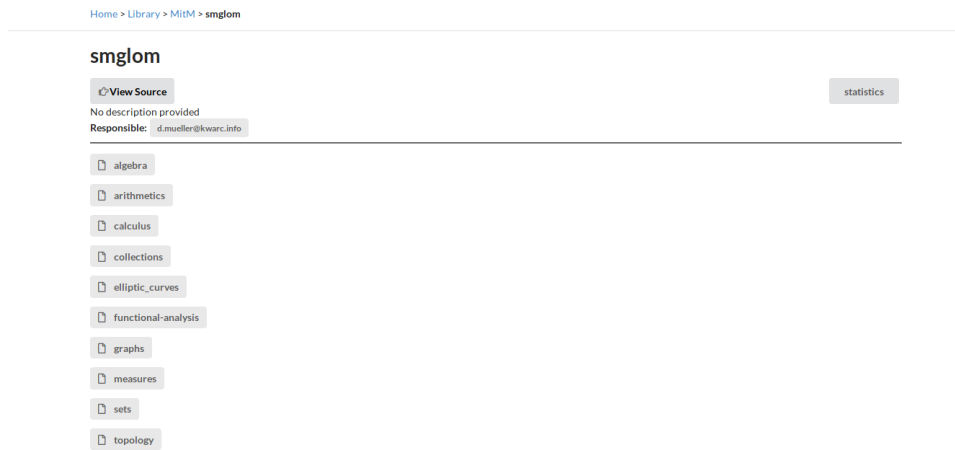
7.2 Archives

Figure 9: an archive in the frontend

Figure 9 shows that the structure of the archive-page is very similar to the group-page. It also starts with a header that consists of a button that links to the source files, a description of the archive and the e-mail addresses of the responsible people.

After the header follows a list of the documents in that archive. There is the main difference between the group-page and the archive page. The document entries do not have a teaser text so they just link to the document-page.

7.3 Documents

Figure 10: a document in the frontend

Depending on the document, the its page can be a combination of modules and opaque elements as well as links to further documents. Figure 10 displays a document in the OMDoc format. There is also a button that links to a documents source file on GitLab[3] if it is in the OMDoc format. EdN:3
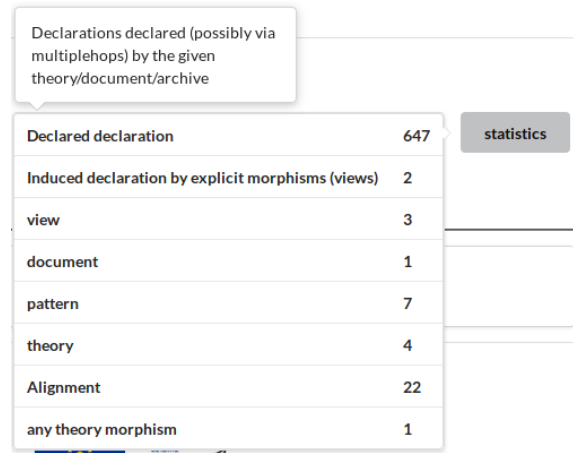
The modules are expandable React components that consist of a name, a type, either theory or view and a button to show further details. Clicking on the button shows all the declarations and nested theories inside of that module. These declarations (structure elements, constants, rule constants etc.) and theories can be further extended to show their own content.

In between the modules there can be opaque elements. These are just text that help the user to understand the content of the document. So opaque elements make up the informal part of theories in MMT.

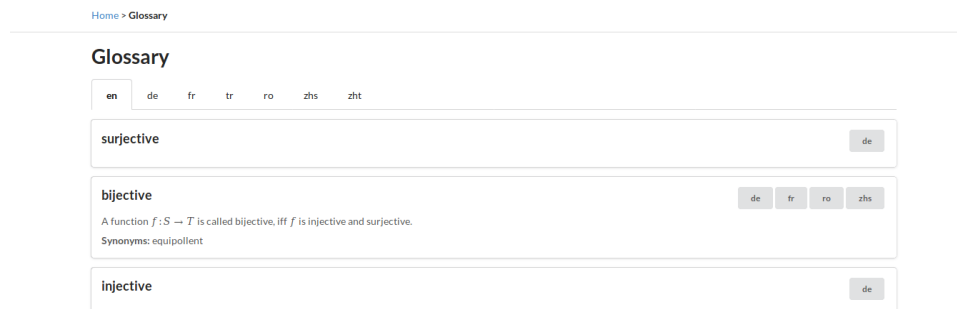
[3]EdNote: + jupyter

7.4 Statistics

Figure 11: statistics in the frontend

On every group-, archive- and document-page there also is a statistics button that shows some available statistics about that particular group, archive or document. When the cursor hovers over a keyword of a statistic that keyword is explained in a pop-up as can be seen in figure 11.

8 The Applications of MathHub

8.1 Glossary

Figure 12: the glossary in the frontend

As shown in 12, the glossary-page has a tab for every available glossary. At any given time only the terms are rendered that have an entry in the currently selected language. Thus there are two possible ways to change languages. First by changing the language tab of the glossary or second

by clicking on a language-button inside an entry. If such a button with a different language is available, this means this particular entry also exists in that language.

To create a better overview the definition of a term is not immediately shown. By clicking on an entry its definition becomes visible.
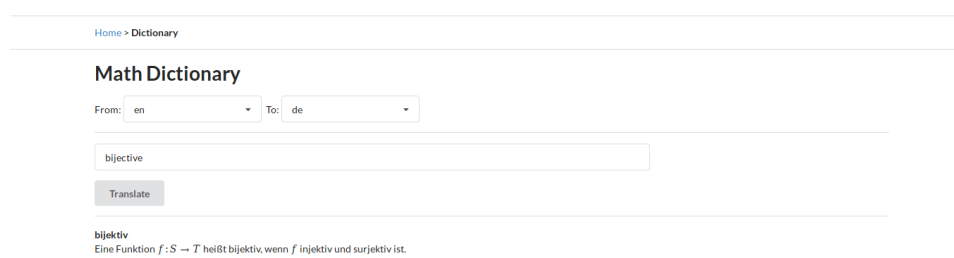
8.2 Math Dictionary

Figure 13: the Math Dictionary in the frontend

Figure 13 displays the Math Dictionary of MathHub.info. To translate a term with the help of the Math Dictionary the user has to select the language from a dropdown menu in which the term currently is and also the language to which it should be translated into Pressing the "translate" - button sends a translation request to the server. Until the servers responds, the message "translating" is shown and the button is disabled to prevent sending to many translation requests at once. If a translation exists then the translated term, its definition and potential synonyms are shown and the button is enabled again. By selecting the same language for "from" and "to" the Math Dictionary can also be used to get the definition for an expression without searching the glossary.

9 Conclusion

⁴

10 Future Work

Obviously the work on a project like MathHub.info is never completely finished. The biggest expansions that will be added to MathHub.info in the near future are TGView and MathWebSearch.

---

[4]EdNote: re-iterate everything

10.1 TGView

TGView is a graph viewer with the task to visualize the relations between multiple theories to give a better overview over a system like MathHub. The distinctive feature of TGView is that its backend sends all necessary data at once so every interaction can be handled by the frontend. That means that its graphical interface is build on the client side to avoid sending a server request every time the user interacts with the interface, eg move or hide nodes. Otherwise it would be required to refresh the page on every single change to the graph[RKM].

Even though the old MathHub frontend did have static graphs to show the relations between theories, it never integrated TGView.

10.2 MathWebSearch

Since MMT is a combination of formal and informal languages there is a need for a special search engine that can deal with this library. The search engine that will be used within MathHub.info is called MathWebSearch[HK]. It is able to search documents that have formulae of a certain shape. For example searching the formula "?x+1" would return documents with formulae like "1+1", "x+1" etc. with that shape. For Mathhub.info a combination of MathWebSearch and normal text search is needed. This combination is called Themasearch and the goal is to make it possible to use it via the User Interface.

10.3 Subset Frontend

Currently MathHub.info is a place to showcase every MMT group. But some of these groups could benefit from specialized additional features or a slightly different structure. So it makes sense to create subset frontends based on MathHub.info. This way the subset frontends can be designed to perfectly fit their group without taking other groups into account. In addition to new features they also should have a new Homepage to give the user a more detailed introduction than the short teaser that is shown on MathHub.info.

To summarize, the subset frontends will have a similar makeup as MathHub.info but will be adapted to better represent their content.

10.4 Issue report: MathHub.info and content

As of right now there is no direct way for a user to report an issue, a bug or request a new feature. When this function is integrated into MathHub.info it would be appropriate to distinguish between two different kind of reports.

First, the user should be transferred to GitHub to open a new issue, if they have a problem with the frontend.

Second, if the user finds an issue within the content that is displayed on MathHub.info, they should be redirected to the corresponding GitLab page.

10.5 Jupyter Integration

References

[Gin+]   Deyan Ginev et al. *The SMGloM Project and System. Towards a Terminology and Ontology for Mathematics*. URL: http://kwarc.info/kohlhase/papers/icms16-smglom.pdf.

[Git]   Github. URL: https://github.com/vuejs/vue.

[HK]   Radu Hambasan and Michael Kohlhase. *Faceted Search for Mathematics*. URL: http://ceur-ws.org/Vol-1458/D05_CRC1_Hambasan.pdf.

[Inca]   Facebook Inc. URL: https://reactjs.org/.

[Incb]   Facebook Inc. URL: https://github.com/facebook/flux.

[Incc]   Facebook Inc. URL: https://reactjs.org/docs/getting-started.html.

[Koh]   Michael Kohlhase. *MathHub Documentation*. URL: https://github.com/MathHubInfo/Documentation/wiki.

[Kum18]   Mohit Kumar. "Hackers Exploiting Drupal Vulnerability to Inject Cryptocurrency Miners". In: *https://thehackernews.com/* (2018). URL: https://thehackernews.com/2018/04/drupal-cryptocurrency-hacking.html.

[KW14]   Michael Kohlhase and Tom Wiesing. *In-place computation in active documents*. Jacobs University Bremen, 2014.

[McC18]   Kieren McCarthy. "Running Drupal? You need to patch, patch, patch right now!" In: *https://www.theregister.co.uk/* (2018). URL: https://www.theregister.co.uk/2018/03/28/drupal_urgent_security_software_patch/.

[NIS]   NIST. URL: https://nvd.nist.gov/view/vuln/search-results?cpe=cpe%3A%2Fa%3Adrupal%3Adrupal%3A7.38&page_num=0&cid=3.

[Rab]   Florian Rabe. *The MMT Language and System*. URL: https://uniformal.github.io/doc/.

[Rab18]   Florian Rabe. *MMT: A Foundation-Independent Logical Framework*. Online Documentation. 2018. URL: https://kwarc.info/people/frabe/Research/rabe_mmtsys_18.pdf.

[RKM]   Marcel Rupprecht, Michael Kohlhase, and Dennis Müller. *A Flexible, Interactive Theory-Graph Viewer*. URL: http://kwarc.info/kohlhase/papers/mathui17-tgview.pdf.

[Tun18]   Liam Tung. "Update Drupal ASAP: Over a million sites can be easily hacked by any visitor". In: *www.zdnet.com* (2018). URL: `https://www.zdnet.com/article/update-drupal-asap-over-a-million-sites-can-be-easily-hacked-by-any-visitor/`.

[Wika]   Wikipedia. URL: `https://en.wikipedia.org/wiki/Polymer_(library)`.

[Wikb]   Wikipedia. URL: `https://en.wikipedia.org/wiki/Angular_(web_framework)`.

[Wikc]   Wikipedia. URL: `https://en.wikipedia.org/wiki/Vue.js`.

[Wol]   WolframAlpha. *About Wolfram—Alpha Making the world's knowledge computable.* URL: `https://www.wolframalpha.com/about/`.