



UNIVERSITY OF TWENTE.

**Faculty of Electrical Engineering,
Mathematics & Computer Science**

DMB Interactive Literature Website

Design Report, April 2022

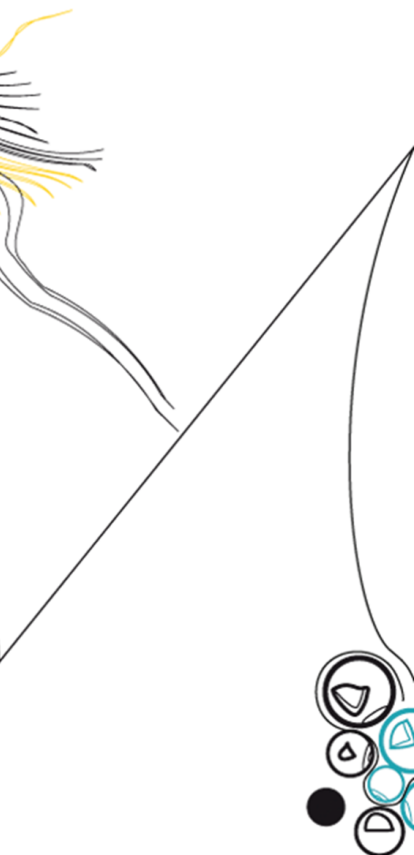
Abdullah Qazi

Ramish Bhutto

Gies Den Broeder

Boris Gerretzen

Frans Schooltink



Supervisor:

Meike Nauta

University of Twente

Faculty of Electrical Engineering,

Mathematics and Computer Science

Zilverling

P.O. Box 217

7500 AE Enschede

The Netherlands

1

Summary

The DMB Interactive Literature Website is based off of the paper, "From Anecdotal Evidence to Quantitative Evaluation Methods: A Systematic Review on Evaluating Explainable AI" which was a collaboration between the Data Management & Biometrics department of the University of Twente and University of Duisberg-Essen. The research team was led by Meike Nauta. Miss Nauta requested a website to be created for her research in order to allow greater accessibility and visualization options as well as enable researchers to contribute to the website by adding new papers. Our development team has tried to materialize her vision as part of the Design Project module of the Technical Computer Science course. The timeline for this project was the 10 weeks allocated for the module.

After deciding on an Agile workflow, the team divided the 10 weeks into 2 week sprints. It was agreed upon with the client, that a meeting would take place every week to provide a forum to share ideas and to answer any questions that might arise from both sides. Every 2 weeks, the latest version of the product would be shown and tested once an appropriate amount of functionality was achieved. In order to commence development, the team conducted interviews with multiple researchers at the Data Management & Biometrics department to define requirements. A low fidelity prototype was created and tested to ensure the client was satisfied with the design choices before implementation began.

Subsequent weeks involved development of the website. An initial tech stack was decided upon, but was changed during the second sprint because the stakeholders preferred the use of GitHub pages over self hosting. The product was then designed with additional GitHub integration in mind. Usability testing was conducted with a group of researchers at the end of the second sprint. From this test useful feedback was gathered and guided implementation in later versions of the product. Throughout the sprints regular communication was had with Meike Nauta over Slack to answer smaller questions that held up production. The final sprint involved a presentation for the researchers, who were given an overview of the design process and a demo of the site.

The final product consists of 4 web pages. A Home page which provides an overview of the purpose of the website and how researchers can contribute to the paper. A Papers page which consists of a table including all the papers that were used in the literature review with various filtering options on top. A Charts page which contains 4 graphs to visualise the data, along with a set of filtering options. Lastly, an "Add Paper" page exists that has a form to input

the details of a new paper. This form generates a JSON output that, once completed, can be copied to add to the db.json file in the user's forked repository on Github. A merge can then be requested with the original repository and if the sufficient number of approvals is met then the paper is added to the official database. It can then be viewed on the website.

Contents

1	Summary	2
2	Introduction	7
3	Project Organization	8
3.1	Division of roles	9
3.2	Planning	10
3.2.1	Sprint 0	10
3.2.2	Sprint 1	11
3.2.3	Sprint 2	11
3.2.4	Sprint 3	11
3.2.5	Sprint 4	11
3.3	Stakeholders	12
3.3.1	User	12
3.3.2	GitHub User	12
3.3.3	GitHub Administrator (Owner)	12
3.4	Requirements	13
3.5	Stakeholder requirements	13
3.5.1	User	13
3.5.2	GitHub user	15
3.5.3	Github Owner	16
3.6	System requirements	16
3.7	Requirements Review	18
3.8	Risk Analysis	19
3.8.1	GitHub Reliance	19
3.8.2	Spam	19
3.8.3	Team Members	19
3.8.4	Addition of False Entries	19
3.8.5	Researchers Leaving	20
4	System Proposal	21
4.1	First Meeting	22
4.2	Researcher Interviews	22
4.3	Lo-Fi Prototype Testing	22
4.3.1	Main Page	23
4.3.2	Review Page	24
4.3.3	Add Paper	25

4.4	Sprint Reports	27
4.4.1	Sprint 0	27
4.4.2	Sprint 1	28
4.4.3	Sprint 2	29
4.4.4	Sprint 3	32
4.4.5	Sprint 4	34
5	Product Description	35
5.1	Use Guide	36
5.1.1	GitHub Pages	36
5.1.2	Renaming the Project	36
5.1.3	Adding papers to the database	37
5.1.4	Adding Additional Tags	40
5.1.5	Adding Additional Datatypes	40
5.1.6	Adding Additional Graphs	41
5.2	Techstack	41
5.2.1	GitHub	42
5.2.2	React and Redux	42
5.2.3	Component libraries	42
5.3	Pages	42
5.3.1	Home Page	42
5.3.2	Papers Page	43
5.3.3	Chart Page	44
5.3.4	Add Paper Page	45
5.4	Known Issues	46
6	Testing	48
6.1	Usability Testing	49
6.1.1	First Usability Test	49
6.1.2	Second Usability Test	49
6.2	Cross Browser Testing	49
6.3	Integration testing	49
6.4	Database Validation and Code Cleanliness Checking	50
7	Future Work	51
7.1	Automatic Tag Parsing	52
7.2	Full Mobile functionality	52
7.3	Additional Visualization Options	52

8	Code Review	53
8.1	Directory tree	54
8.2	File and Folder Descriptions	56
9	Team Contributions and Reflections	58
9.1	Planning Reflection	59
9.2	Team Contributions	60
9.2.1	Frans	60
9.2.2	Gies	60
9.2.3	Abdullah	60
9.2.4	Ramish	61
9.2.5	Boris	61
10	Appendix	62
10.1	Initial Stakeholder Interview	63
10.2	Second Usability Test	66

Introduction

This paper covers the design and implementation of an Interactive Literature Website. It documents the progress, planning and the different iterations this project underwent during the development. It also contains comprehensive documentation on the workings and features of the final product. In January of 2022 a paper on explainable AI was released by a research group from the University of Twente and the University of Duisburg-Essen led by Meike Nauta. This paper provides an in-depth literature review into the topic. With the paper the research team hopes to provide a framework by which explainable AI methods can be evaluated. To better visualize and expand upon the data collected for this paper, Meike Nauta tasked us with creating an Interactive Literature Website. We successfully managed to implement the features requested of us in the 10 week timeframe allocated for this project. Features of the website include multiple graphs, filters for the collected papers and the ability to add papers to the database.

3

Project Organization

This section provides an overview of our project organisation. Sections include planning, task division, stakeholders and requirements, and risk analysis. These sections provided the backbone on which the product was built. During development we did our best to adhere to the goals set during the initial planning stages.

We chose Agile as our main development method for the project because it involves consistent input from our main stakeholders, Meike Nauta and her colleagues. We have divided the time allocated into four 2-week sprints. After every sprint we reflected on the progress made during the sprint, the highlights of these reflections can be found below.

Aside from the sprint retrospectives every two weeks, we had an additional meeting with Meike each week to ask questions and update her on the progress we made during that week. These additional meetings provided us with vital feedback required for the successful development of our product.

3.1 Division of roles

While initial task division split members of our group into a front-end and back-end development team, a simplification of our tech stack led to the reduction of the back-end. The task division that we eventually settled on is as follows:

- Abdullah - UI/Frontend/Report
- Ramish - UI/Frontend
- Gies - UI/Frontend
- Frans - Frontend/Report
- Boris - Frontend/DevOps

3.2 Planning

This section provides an overview of the work flow that the team followed throughout the project. Important deadlines and development stages were set at the start of the project to make sure we remained on track and met the goal of a finished product by the end of the module.

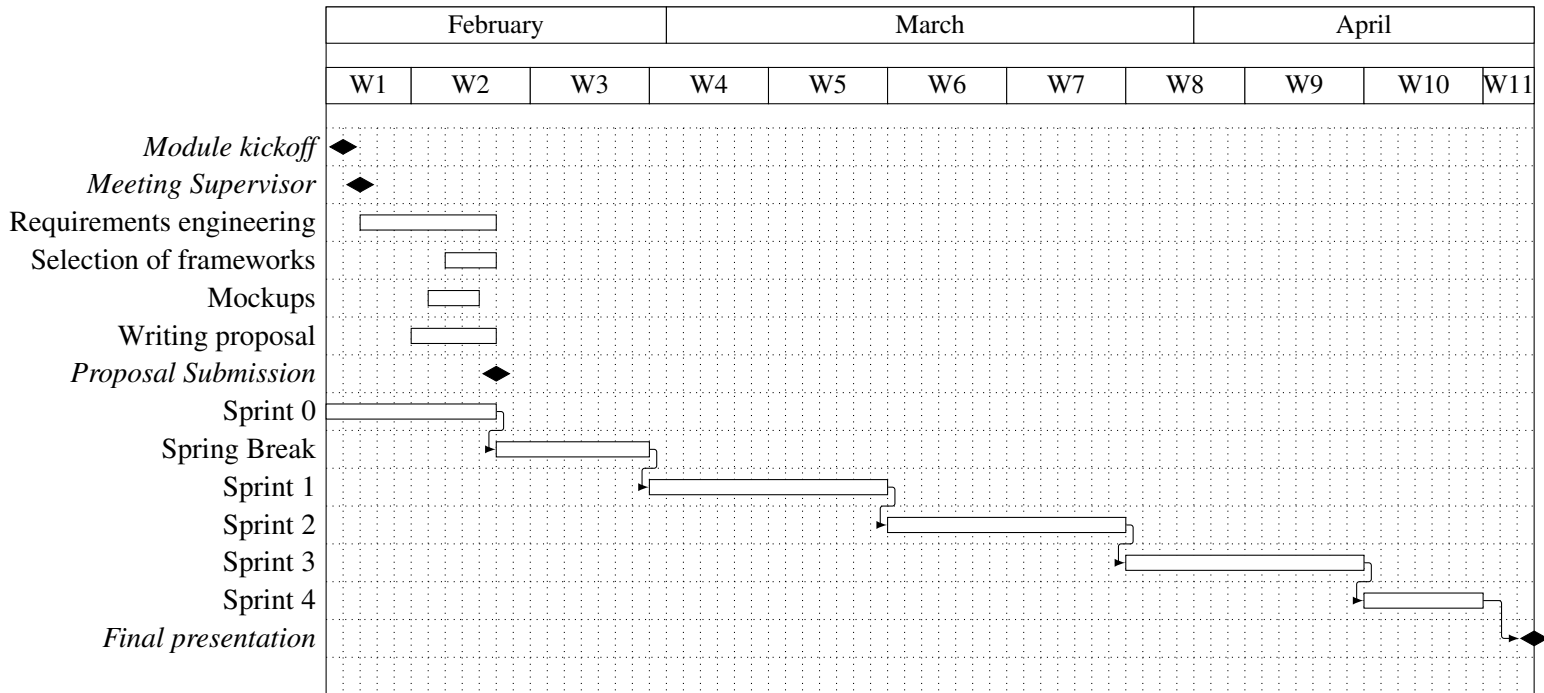


Figure 1: The timeline and set of deadlines for the completion of the design project.

The way we divided the allotted time is as follows:

Week 1-2: requirements, user stories, Lo-Fi prototype

Week 3-8: development, writing report

Week 9-10: Testing and documentation

3.2.1 Sprint 0

In this sprint, the main goals would be to find requirements and finish a lo-fi prototype along with selecting the frameworks to be used in the development stage. This sprint would also consists of meeting the supervisor and stakeholders. An interview with the stakeholders was to be conducted to find and define the requirements. A usability test would be conducted to receive feedback on the lo-fi prototype.

3.2.2 Sprint 1

This sprint would involve the planning of the system architecture. To create a solid framework to build our product on we would need to define clear class- and flow diagrams. In addition, work was to commence on the front end, focusing on functionality instead of visuals. All changes and additions were to be added to the design document.

3.2.3 Sprint 2

Feedback of the previous sprint was to be used to guide further front and back end development. Separate teams would then focus on various aspects of the project. By the end of this sprint we aimed to deliver semi-functional components, to be finalized in sprint 3. Again any changes made during this sprint were to be documented.

3.2.4 Sprint 3

Aside from bug fixing, the final product was to be delivered during this sprint. This meant the addition of the last few features and final UI changes. As we would be nearing the end of development, work would also be started on the presentation and poster.

3.2.5 Sprint 4

Any last bugs were to be fixed during this sprint. Finishing the presentation and design report was to be the main focus of this sprint as all deliverables had to be submitted by the end of this sprint.

3.3 Stakeholders

This project has various stakeholders who will be interacting with the system. They will be allocated privileges in accordance to how they will interact with the system. While the project does not have a specific audience it is aimed towards researchers interested in Explainable Artificial Intelligence. Therefore we assume that users have a basic knowledge of the subject.

3.3.1 User

This stakeholder is a person with interest in XAI who visits the React based website. The user is able to make use of the graph visualizations and the table of papers, both of which can be sorted and filtered. Using these tools they are able to gather a deeper insight into the literature review on which the website is based. The site will also provide these users with an ability to create a JSON entry for their own research papers for the database. To update the site's database a user will have to navigate to the site's GitHub at which point he turns into a GitHub user.

3.3.2 GitHub User

To get the full functionality of the the project a user will need to have a GitHub account. The Project's GitHub allows users to submit and review papers and even to fork and create their own version of our website. For the paper submission GitHub users can use the form on the main website to get a JSON entry to update the database with. After a paper has been submitted fellow GitHub users can peer review the new entry by reviewing the merge request.

3.3.3 GitHub Administrator (Owner)

The GitHub administrator overlooks the systems and therefore has full control over the GitHub repository. Administrators can immediately merge papers with the database or revert changes. They are also able to change user permissions.

3.4 Requirements

This section defines the various requirements that have been considered in order to build the system. These requirements have been laid out in the form of user stories for the stakeholders that were defined previously. Along with these user stories, there are system requirements which highlight key features required for the website to be functional.

3.5 Stakeholder requirements

3.5.1 User

Must have

- As a user, I must have the ability to see related papers based on selected papers
The research conducted looks at the patterns and similarities between papers. A graphical visualization displaying these similarities is integral for the website
- As a user, I must have the ability to search papers by title
Being able to search is an important functionality as it makes it easy to find a particular paper
- As a user, I must have the ability to get the link to a paper
The papers are published on specific sites which require a link (Doi) to access them
- As a user, I must have the ability to see a graph of papers over time
New papers get published every year that provide new insight into the topic of Explainable AI. This can be displayed graphically
- As a user, I must have the ability to filter the graphs on type of data
- As a user, I must have the ability to filter the graphs by types of explanation
- As a user, I must have the ability to filter the graphs by type of task
- As a user, I must have the ability to filter the graphs by type of problem
- As a user, I must have the ability to filter the graphs by type of model to be explained
- As a user, I must have the ability to filter the graphs by type of method used to explain
Type of Data, Type of Problem, Type of Model to be explained, Type of Task, Type of Explanation and Method used to explain are all fields used to group the data that was carried out in the research paper. Therefore, one should be able to filter the graphs based on each
- As a user, I must have the ability to filter the graphs by publication year
Publication year provides insight into the research which has been conducted in that time. It should be another field to filter the graphs by
- As a user, I must have the ability to filter the graphs by the venue
The venue is the conference or journal to which the paper was submitted to. It provides another filtering option for the graphs
- As a user, I must have the ability to filter the graphs by author
The author(s) are the personnel which contributed towards the research and writing of the paper. Hence, they are a filtering option for the graphs

- As a user, I must have the ability to reset filters
Filtering options should be ressettable to let the graphs and table to return to the default state. Other filtering can then be explored
- As a user, I must have the ability to search papers by types of data
- As a user, I must have the ability to search papers by types of explanation
- As a user, I must have the ability to search papers by type of task
- As a user, I must have the ability to search papers by type of problem
- As a user, I must have the ability to search papers by type of model to be explained
Type of Data, Type of Problem, Type of Model to be explained, Type of Task, Type of Explanation and Method used to explain are all fields used to group the data that was carried out in the research paper. Therefore, one should be able to filter the list of papers utilizing them to achieve the desired result
- As a user, I must have the ability to search papers by publication year
Publication year provides insight into the research which has been conducted in that time. It should be another field to filter the list of papers by
- As a user, I must have the ability to search papers by the venue
The venue is the conference or journal to which the paper was submitted to. It provides another filtering option for the list of papers
- As a user, I must have the ability to search papers by author
The author(s) are the personnel which contributed towards the research and writing of the paper. Hence, they are a filtering option for the list of papers
- As a user, I must have the ability to combine multiple filtering options
Filters are regularly used together in order to look for a particular result
- As a user, I must have the ability to sort the data based on paper titles
- As a user, I must have the ability to sort the data based on publication year
- As a user, I must have the ability to sort the data based venue
- As a user, I must have the ability to sort the data based author
- As a user, I must have the ability to combine multiple sorting options
Paper title, publication year, venue and author are key identifiers of the research papers. Therefore, they must be able to be sorted in the list of papers
- As a user, I must have the ability to sort newly added papers by their added date
The date can be specified for papers which are newly added into the database. Therefore, they can be filtered according to it

Should have

- As a user, I should have the ability to see the abstract of a paper on the website
The abstract provides an introduction to the paper. This allows a user to get an overview on what the paper is about. Due to the limits of scraping tools available not every paper in the original database has an abstract

- As a user, I should have the ability to see who added the paper (XAI authors, paper author or community)

This can be viewed in the GitHub repository by checking the commit history

- As a user, I should have the ability to see a graph of connected papers

A graph of connected papers shows the common tags which are shared between papers

Could have

- As a user, I could have the ability to see whether the tags of a paper were generated or entered manually

Through natural language processing and machine learning, an algorithm can identify the tags to be generated

3.5.2 GitHub user

Must have

- As a user, I must have the ability to submit my paper to be displayed on the website
Through a form on the site, the user can copy over a JSON entry to the database and make a merge request. If approved, the paper will be displayed on the website

- As a user, I must be able to provide an explanation of why my paper should be submitted

Through the comments section on GitHub, the user can add an explanation as to why the paper is suitable to be part of the research

- As a user, I must be able to provide my papers with the appropriate tags

The form on the website allows the user to select all the appropriate tags and can add comments if a certain tag is not available

Should have

- As a logged in user, I should have the ability to verify the tags of a submitted paper
The user can approve the pull request on GitHub after checking the submitted entry
- As a logged in user, I should be able to provide feedback when verifying submitted papers

The comments section on GitHub allow for a user to provide feedback on merge requests for submitted entries

- As a logged in user, I should be able to distinguish between rejection based on the categorisation and rejection based on the merits of the paper itself, when verifying submitted papers

The review comments on the pull requests will provide insight as to why a paper was approved or rejected

3.5.3 Github Owner

Must have

- As a GitHub owner, I must have the ability to remove a paper
The db.json file can be edited to remove papers
- As a GitHub owner, I must have the ability to edit a paper entry
The db.json file can be edited to alter paper entries
- As a GitHub owner, I must have the ability to approve submitted papers
The GitHub owner can directly approve the pull requests for submitted papers
- As a GitHub owner, I must have the ability to create new columns in the data
The schema.json can be edited to create new columns in the data
- As a GitHub owner, I must have the ability to import new data (rows) into the system
The schema.json can be edited to create new rows in the data
- As a GitHub owner, I must have the ability to override the existing data with new data
The db.json file can be edited to override the existing data
- As a GitHub owner, I must have the ability to export the data from the system as a file
The filtered list can be downloaded from the website as a JSON file
- As a GitHub owner, I must have the ability to disable and enable different features on demand(i.e disable user verification of submitted papers, or paper submissions)
The GitHub owner can change the settings of the repository to enable or disable features
- As a GitHub owner, I must be able to change the amount of positive peer reviews needed for a paper to pass
The GitHub owner can change the number of approvals required to let a pull request take place

Should have

- As a GitHub owner, I should have the ability to block user accounts from the system
The GitHub owner can change permissions for users accessing the repository
- As a GitHub owner, I should have the ability to instantly accept a paper, still to be peer-reviewed, into the database
The GitHub owner can bypass the required approvals and accept a pull request directly

3.6 System requirements

Must have

- The system must allow a user to submit a paper
Having a website allows users to contribute to the extension of the research
- The system must allow the GitHub Owner to remove a paper
In the case an entry is not suitable anymore, GitHub owner should be able to remove it

- The system must allow an admin to edit a paper entry
The admin should be able to make changes to the data set whenever necessary
- The system must allow the GitHub owner to approve submitted papers
The GitHub owner should hold privileges to approve submitted papers
- The system must allow the GitHub owner to add new data
GitHub owner should hold privileges to make changes or add to the data
- The system must allow a user to export data
Exporting data can ensure that a backup of the data exists along with moving data to another program
- The system must allow an admin to enable or disable different features
Admin should hold privileges to change repository features whenever necessary
- The system must have routine backups
Routine backups are necessary to ensure data can be restored from an earlier time in case of an unplanned event such as data loss or corruption
- The system must be easily maintainable
An easily maintainable system will mean that it is reliable and can be easily expanded
- The system must have elaborate documentation
Multiple users are expected to use the system who will contribute to the extension of the research. In addition, it is important in the case of further system developments
- The system must have a dedicated README file
The README file serves as a guide on the usage of the system along with providing the purpose

Should have

- The system should allow users to verify the tags of a submitted paper
Tags of the paper should be accurate and therefore must be checked in order to extend the research
- The system should have the ability to scrape the number of citations of a paper
Citations get updated with time as more papers use it as reference. Timed scraping can allow for these updates

Could have

- The system could be mobile friendly
The system will primarily be used on bigger devices
- The system could have the ability to scrape the abstract of a paper
Accessibility of all abstracts may not be feasible
- The system could have an automatic detection of tags based on the title and abstract of a submitted paper
Natural language processing algorithm can allow for automatic detection but may be developed later on

3.7 Requirements Review

All the requirements were able to be met apart from the ones mentioned below. A description has been provided as to why they were unable to be met.

- The system could be mobile friendly
Due to the limited time frame we focused on generating as much functionality as possible for site. This meant that considering functionality on mobile devices took a back seat. Though the site is still semi functional on mobile devices the layout breaks in a few places
- The system could have an automatic detection of tags based on the title and abstract of a submitted paper
Implementation of this component was not a top priority and as it would have required writing a machine learning algorithm it was eventually deemed out of scope for the project
- As a user, I could have the ability to see whether the tags of a paper were generated or entered manually
Seeing as automatic tag generation was never implemented this requirement became redundant
- The system should have the ability to scrape the number of citations of a paper
An API exists for retrieval of this information, however, the free version only allows for a limited number of look ups per minute. Therefore it is unfeasible to continuously update the database with this information

3.8 Risk Analysis

3.8.1 GitHub Reliance

The project is hosted on GitHub which means that the website will not be functional whenever GitHub faces any major technical issues. As the project is open-source, GitHub is a viable site to be used for hosting purposes especially considering hosting is free as long as the project is open source. On the whole, GitHub is a reliable site and unlikely to face major data breaches and can deal with (distributed) denial of service attacks. As a result, planned maintenance for GitHub will likely be the primary reason the Literature Website is unable to remain functional on the hosting end.

3.8.2 Spam

As the project is open source and hosted on GitHub, anyone on the internet is able to access the website and can get in contact with the people responsible for the research paper at the Data Management and Biometrics department. This could lead to spamming of the people involved in the research. There is no way to prevent the spamming from happening but it is likely to be minimal as researchers involved in the Artificial Intelligence field and people interested in Explainable AI will be the primary users of the website, both of these groups are deemed mature and professional enough to not abuse this feature. Any spam that may be received can be mitigated through utilizing spam filtering tools which will automatically detect and remove spam mail.

3.8.3 Team Members

During the development of the project, it is likely that some team members may become unavailable due to a number of reasons. Reasons can include sickness, extended holidays and other commitments. This can result in small delays in development which risk putting the project behind schedule. As a result, careful planning is key to ensure that the project is completed on time. The iterative approach chosen for this project should mean a regular feedback loop with the client is in place, allowing the team to regularly convey project updates and meet deadlines.

3.8.4 Addition of False Entries

Another possible risk is the one of malicious or wrong entries. With multiple researchers willing to contribute to the research, it is possible that some false entries are likely to be added despite the necessary checks and approvals. In order to mitigate this, the administrators of the system should be thorough in their checks and encourage the people submitting to do the same. In case one does slip through, the repository can be reverted back to a previous one to undo the addition.

3.8.5 Researchers Leaving

The repository will need supervision due to strangers getting involved in the expansion of the research. With administrators having other commitments and motivations to achieve, it is possible that the administrators will be unable to regularly attend to the requests on the GitHub repository. For the long term maintenance of the website, administrator roles should be assigned accordingly so a select few are able to have an outlook on the website. Future administrators should ensure that a healthy level of staff is maintained.

4

System Proposal

This section covers the initial stages of development. It looks at the initial meeting with the product owner where we got an introduction to the research which had been conducted along with a set of resources to consult for the development of the product. This was followed by the interview of other researchers involved in the research and a usability test of the high-fidelity prototype. This initial stage allowed us to fully define the requirements and have a clear idea of what the product should achieve once development began.

4.1 First Meeting

We had our first meeting with the product owner (and supervisor) Meike Nauta on the 10th of February. The main purpose of this meeting was to introduce the project, familiarize ourselves with the research contents and decide upon a work approach that aligned with our schedules. Multiple resources were provided to us, the first being the research paper, "From Anecdotal Evidence to Quantitative Evaluation Methods: A Systematic Review on Evaluating Explainable AI", for which the website is to be designed. Secondly, we were shown a simple dashboard that was already created in Tableau from which we were able to draw inspiration. Lastly, we were given the dataset on which the website should be build.

4.2 Researcher Interviews

In the second week of sprint 0, an interview was conducted with a few of the researchers involved in the research paper. This was done in order to gain a clearer insight into the various ideas the researchers had in terms of tools, visualization options and other relevant features for the website. In addition, the presence of multiple researchers meant that the ideas came from more than one person and therefore less bias would arise in terms of requirements. The interview was organized in a discussion format where a question would be posed and each of the researchers was allowed to express their opinion. The questions and answers can be found in the appendix. Meike Nauta, Jan Trienes and Elisa Nguyen took part in this interview.

4.3 Lo-Fi Prototype Testing

At the end of the initial two weeks (Sprint 0), we had finished a Lo-Fi prototype of the website. The main focus of the Lo-Fi was to create a design which satisfied the requirements that were identified during the initial interview and the second interview with the researchers. Along with testing the prior identified requirements, it allowed for a first usability test to be conducted with the client. This test was designed with the purpose to gain feedback on the prototype design. This feedback would later be used to guide the design of the our website.

4.3.1 Main Page

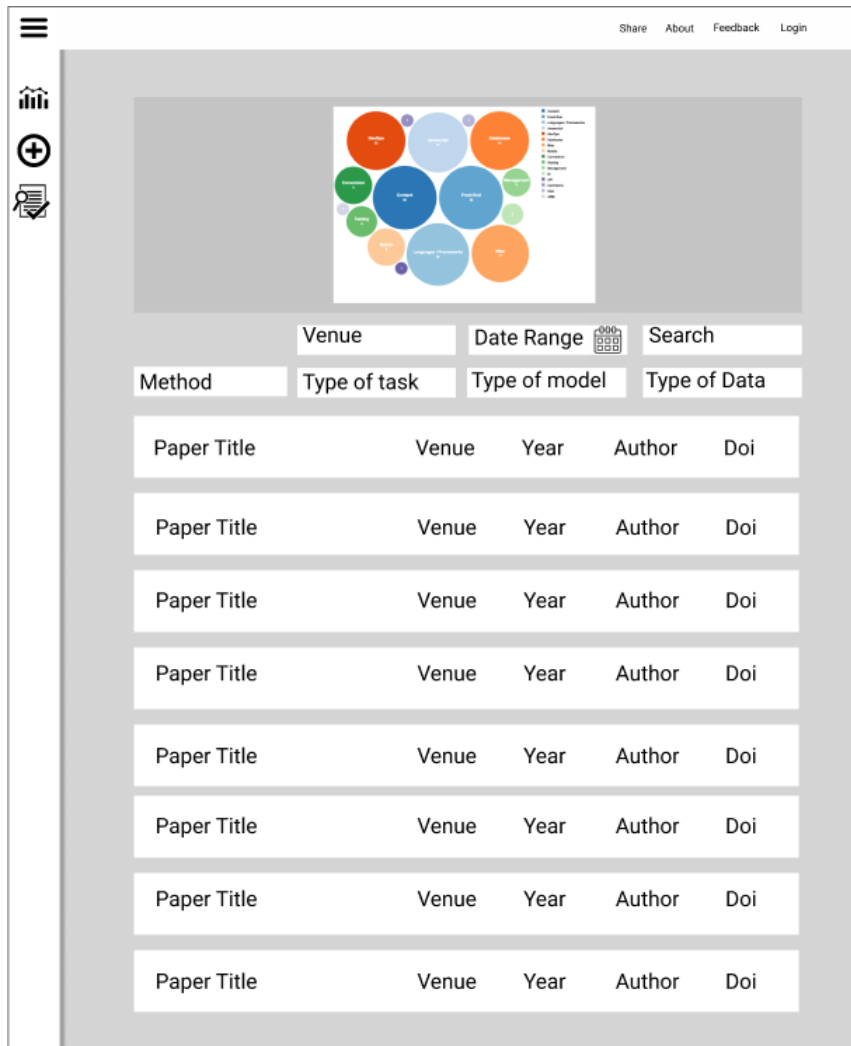


Figure 2: Main Page

The main page, or landing page, as seen in Figure 2 was chosen to include a table which has columns similar to the Tableau page that was shown to us at the start of the project. The top of the page is the designated place for the various graphical options. The graphical options were not a focus for the low-fidelity prototype so they are represented by a placeholder.

It was understood that the filtering options would update the graph and tables accordingly. Once expanded as seen in Figure 3, the abstract for the paper can be viewed. The paper can be removed using the red "x" icon on the bottom right. The feedback was that only one author should be in the author column and the Doi should be a link rather than a column. The abstract was seen as useful along with clicking again to close the expanded paper. The

removal button was not that intuitive and the user felt it was better to turn the removal into another tab as it is unlikely to occur often.

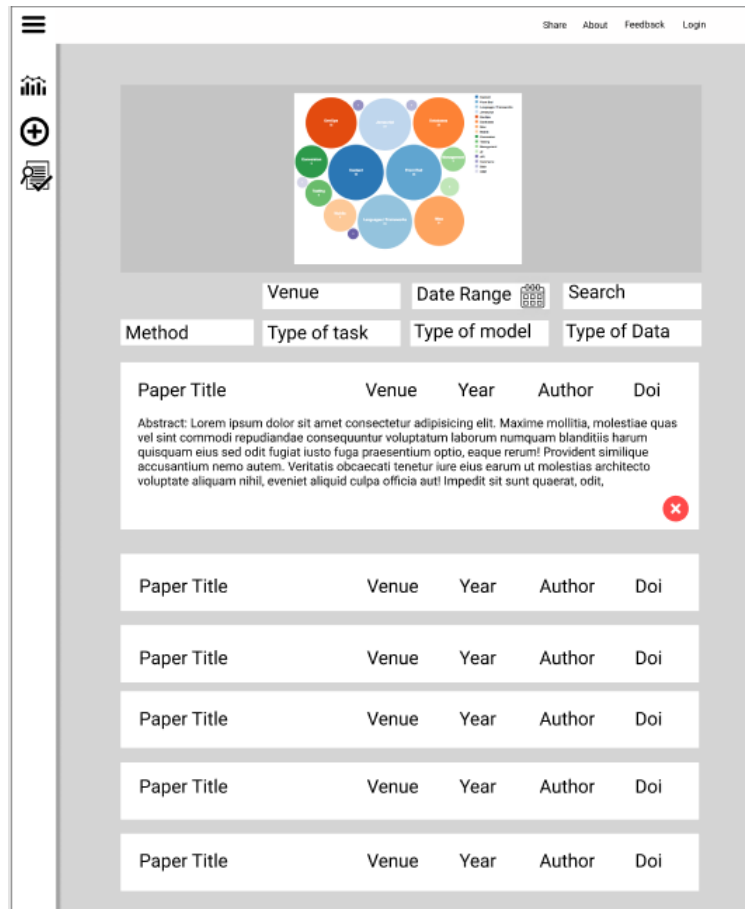


Figure 3: Main Page Expanded

4.3.2 Review Page

The Review Page as seen in Figure 4 is similar to the Main Page with the table design. This is where all the submitted papers are going to be displayed so that users can review them. Depending on the outcomes of the review process, the paper will be accepted or not. The 'thumbs up' icon as seen on Figure 5 was designed to submit a review.

The 'thumbs up' icon was deemed confusing and should be replaced by a simple 'yes' or 'no'. A system was proposed for the reviewing process where if there was a difference of 3 acceptances at any point for a paper, then it should be fully accepted. The same process was to apply for rejections. It was also proposed that the administrator should be able to directly accept or reject papers along with being able to change the number of required reviews and the difference to qualify a paper for acceptance or rejection. In addition, a motivation section should also exist for reviewers where they can explain their decision to accept or reject a submitted paper.

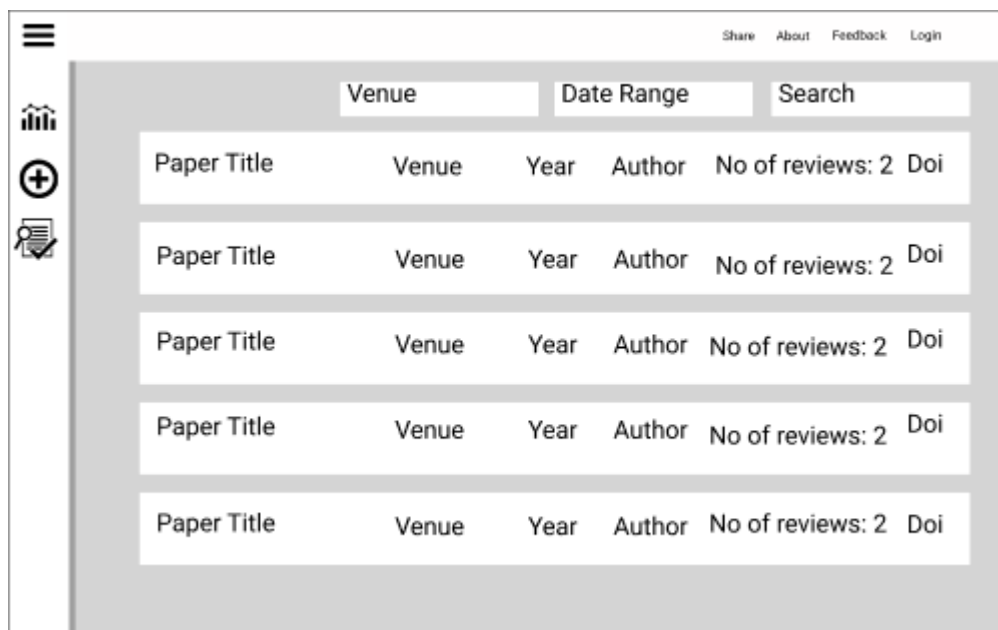


Figure 4: Review Page

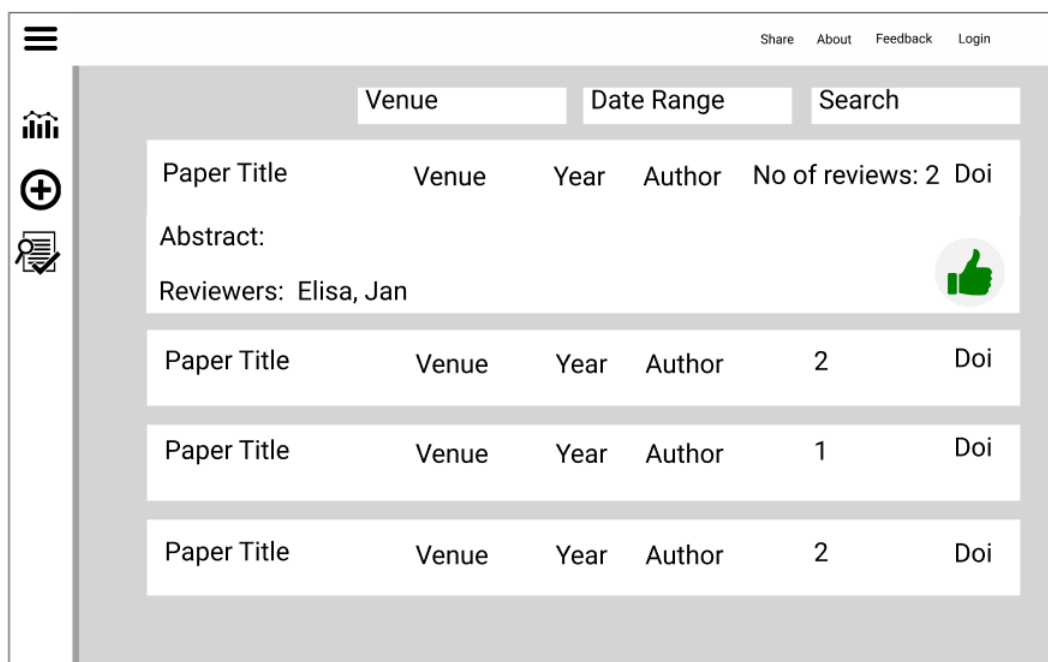
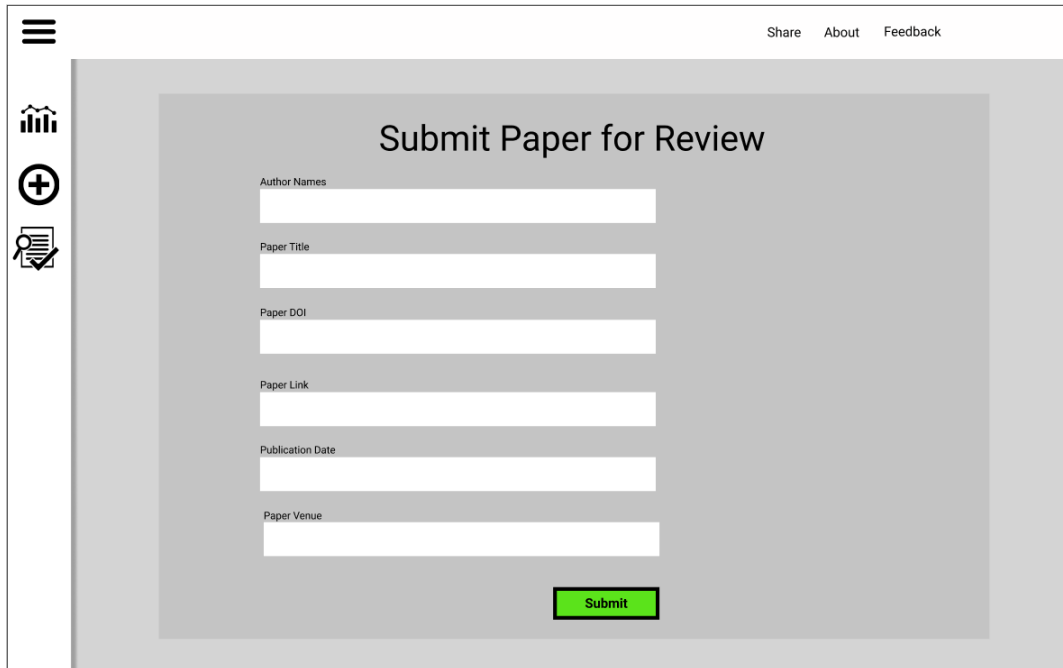


Figure 5: Review Page Expanded

4.3.3 Add Paper

The Add Paper page as seen in Figure 9 serves as a form in order to submit entries into the data set. Once approved, the entries will be visible on the website.

Overall the page was intuitive. Feedback came in the form of extra fields to allow the user to input more necessary data. This included a field for the link of the paper and a field for motivating the choice of adding a certain paper. A field for choosing a specific set of predefined labels was also wanted along with an option for an 'other' label in case the paper had labels which were not already in the database.



The screenshot shows a web interface for submitting a paper for review. On the left is a vertical sidebar with four icons: a hamburger menu, a bar chart, a plus sign in a circle, and a document with a checkmark. The top right of the page has links for 'Share', 'About', and 'Feedback'. The main content area is titled 'Submit Paper for Review' and contains a form with the following fields: 'Author Names', 'Paper Title', 'Paper DOI', 'Paper Link', 'Publication Date', and 'Paper Venue'. Each field is represented by a white input box on a light gray background. A green 'Submit' button is located at the bottom right of the form area.

Figure 6: Add Paper Page

4.4 Sprint Reports

This section provides a summarisation of the progress made in each sprint. These have been organized in separate sprint reports which detail the developments made in each two week period from the start of the module to the end.

4.4.1 Sprint 0

Sprint 0 marked the start of the project. During this sprint we undertook the first steps of the design process. Multiple interviews were conducted with the stakeholders to discover their initial wishes for this project. Using these interviews a list of requirements could be created. These requirements steered the selection of our tech stack and the design of the Lo-Fi prototype.

Milestones during this sprint

- Created use case requirement list according to MoSCoW method
- Decided on tech stack (see section 5.2)
- Made a Lo-Fi prototype using Figma
- Did usability testing with stakeholder using the Lo-Fi prototype
- Creation of 2 flow diagrams for paper submission system

Lo-Fi prototype

A low fidelity prototype was designed using the prototyping tool Figma. Inspiration for the layout was taken from the tableau site provided to us by the client along with browsing for standard dashboards designs. Focus was placed on meeting the requirements and therefore the design was divided into several pages with each focusing on a key feature the client wanted to have. Graphical visualization was not considered for the prototype as it was a feature that would be easier to focus on during development. A usability test was then conducted with the client where new requirements were identified along with receiving feedback for the design which would be refined during the development stage.

Usability test

To refine the requirements and the concept we had for the site, a usability test was conducted. For further details on the results of this testing please see paragraph 6.1.1.

Initial tech stack

In addition to the list of requirements and the Lo-Fi prototype we also decided on our initial tech stack. While the tech stack got changed in sprint 1 some of the elements have remained the same.

Component Library: Ant Design
Front end: React + Typescript
State management: Redux Toolkit

Docker for containerization
Hosting: UT
Database: MongoDB
Repository: Github
Backend: Flask

4.4.2 Sprint 1

The first sprint meant the start of actual development. Due to the chosen tech stack, the first week was focused on setting up the repository and ensuring that all required modules were installed in order to commence on implementation. However, at the end of the first week a progress meeting was held with the client and her colleague Jan Trienes in which the decision was made to switch to GitHub Pages for deployment. This meant changing the tech stack and rethinking part of the implementation methods. This resulted in a greatly simplified tech stack, which remained unchanged for the rest of the project.

Milestones during this sprint

- Basic site layout
- Flow charts for reviewing and adding papers
- Implemented Table with research papers
- Implemented connected papers graph
- Sorting options for table
- Set up Github and React
- Reformatted Database from CSV to JSON

Site Layout

During the first sprint we had designed a main page. To implement this page, the Ant Design Library was used. The Ant Design table components were used to create a filterable list of all the papers stored in the database. The connected papers graph was created using the Connected Papers Components from Graphin, an Ant Design sub Library. Work was also started on creating a dark mode for the site, this has sadly remained unfinished due to technical difficulties.

Tech Stack

As mentioned previously the decision was made to switch tech stacks midway through the sprint. Both the original tech stack as well as the new shortened variant were set up by Boris. During this sprint he also wrote the GitHub scripts for Database Validation, Code Cleanliness check and Automatic Deployment.

Product Update

By the end of the sprint, basic functionality was implemented for the website. As can be seen

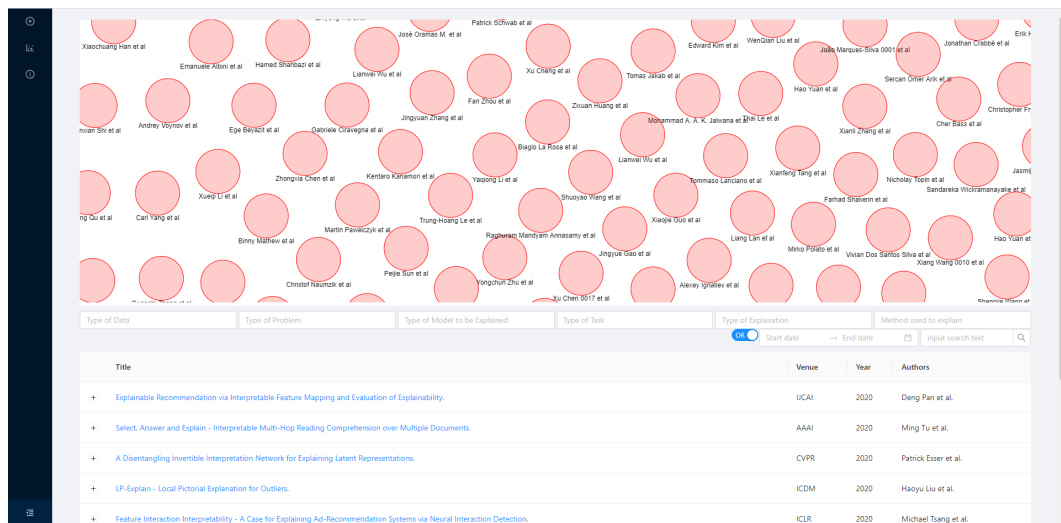


Figure 7: Main Page end of Sprint 1

on Figure 7, the website comprised of a single page which contained a chart at the top and a table below it. The design was inspired from the low fidelity prototypes which was created in the sprint prior. The filtering options included being able to select specific tags, a date range picker along with a search bar. A switch button also allowed a user to select "AND" where papers that include all the specific tags would be displayed along with "OR" where all papers containing either of the selected filters are displayed.

The graph on top of the page was in early development. The nodes refer to a different paper that exists in the data set. The connected graph was linked to the filters and therefore, nodes (papers) would be displayed depending on the filters chosen.

4.4.3 Sprint 2

Milestones during this sprint

- Implemented a Circle Packing Chart
- Implemented a Line Chart
- Implemented a Race Chart
- Separation of the Charts and the Papers into separate pages
- Added Add Paper page
- Added a form to the Add Paper page which outputs a paper in the correct JSON format
- Added an About Page
- Many smaller quality of life changes
- Conducted second Usability Test

Product Update

By the end of the second sprint, we were able to have reached a minimum viable product. Most of the important features were implemented and it was in a suitable state to conduct a usability test with multiple members of the Data Management and Biometrics department. The results of the usability testing can be found in the appendix.

The charts and the table were separated into 2 pages on request of the client, Papers as seen in Figure 8 and Charts in Figure 10. 4 different charts were added which included the connected graph, line chart, race chart and a "tableau graph" that was based off one of the graphs that was part of the tableau dashboard for the paper. Just like the Papers page, the charts are responsive to the filter selection which are identical in design and functionality.

In addition, an Add Paper page was implemented as seen on Figure 9. It consisted of a form that included all the fields that are a part of the db.json file. Filling out the entries outputs everything into JSON text on the right. The JSON formatted text is meant to be copied and then pasted into the db.json file in order to add a new entry. Lastly, an About page was added that included information about the purpose of the website, the abstract, email pop up and the bibtex citation.

A second usability test was also conducted during this sprint. This test allowed us to get valuable feedback on our product thusfar and guided future development to a great degree. Please refer to the testing paragraph and the appendix for further information.

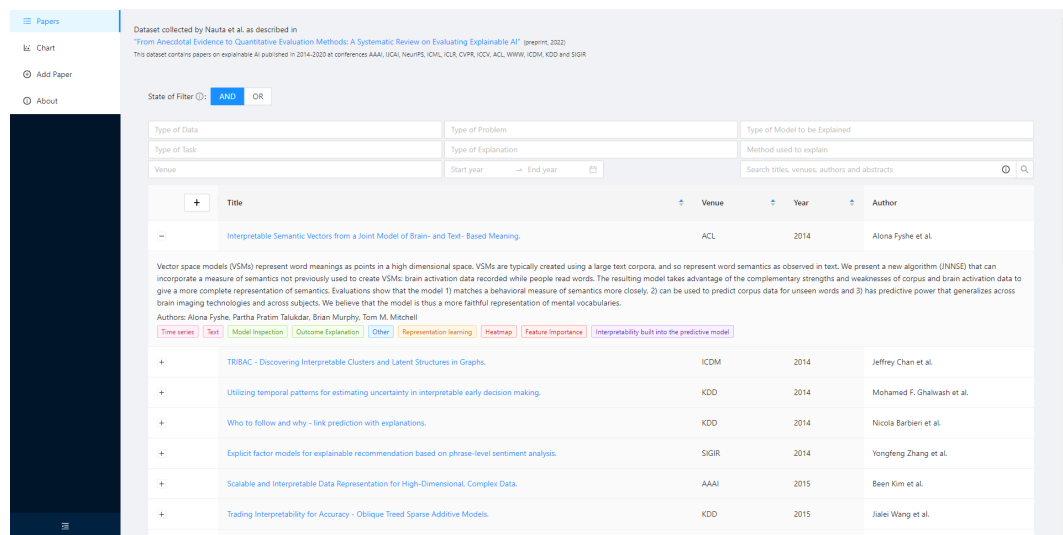


Figure 8: Papers Page end of Sprint 2

Papers
Chart
Add Paper
About

Dataset collected by Nauta et al. as described in
["From Anecdotal Evidence to Quantitative Evaluation Methods: A Systematic Review on Evaluating Explainable AI"](#) (preprint, 2022).
This dataset contains papers on explainable AI published in 2014-2022 at conferences AAAI, UCAI, NeurIPS, ICML, ICLR, CVPR, ICDCV, ACL, WWW, ICDM, KDD and SIGIR.

Paper Submission

The original paper already includes classification of over 300 research papers. Of course this list is not exhaustive and therefore we would like to ask the wider research community to aid in extending the original research.

You can contribute to this project by going to the project's GitHub (<https://github.com/BorisGerstzen/DMLiteratureWebsite/blob/master/src/db/db.json>). This page shows the current contents of the database. To add a new entry simply fork the project and append the db.json file using the following template.

Title:
Doi:
Year of Publication:
Authors:
Venues:
Type of Data:
Type of Problem:
Type of Model to be Explained:
Type of Task:
Type of Explanation:
Method used to explain:
Abstract:

Your JSON:

```
{
  "Title": "",
  "url": "",
  "Year": "2020",
  "Venues": [
    "isOld": true,
    "value": ""
  ],
  "Authors": [],
  "Type of Data": [],
  "Type of Problem": [],
  "Type of Model to be Explained": [],
  "Type of Task": [],
  "Type of Explanation": [],
  "Method used to explain": [],
  "Abstract": ""
}
```

Figure 9: Add Paper Page end of Sprint 2

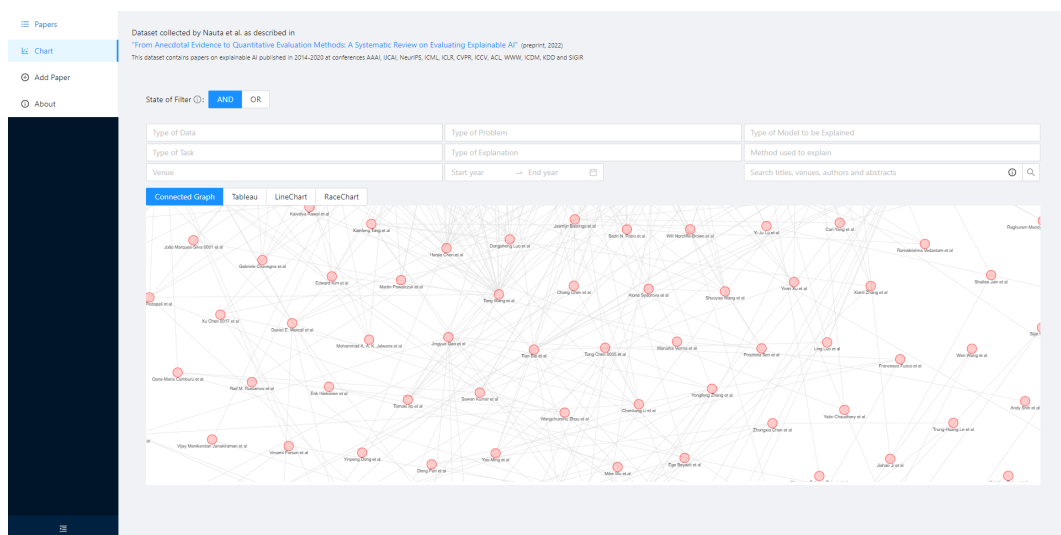


Figure 10: Charts Page end of Sprint 2

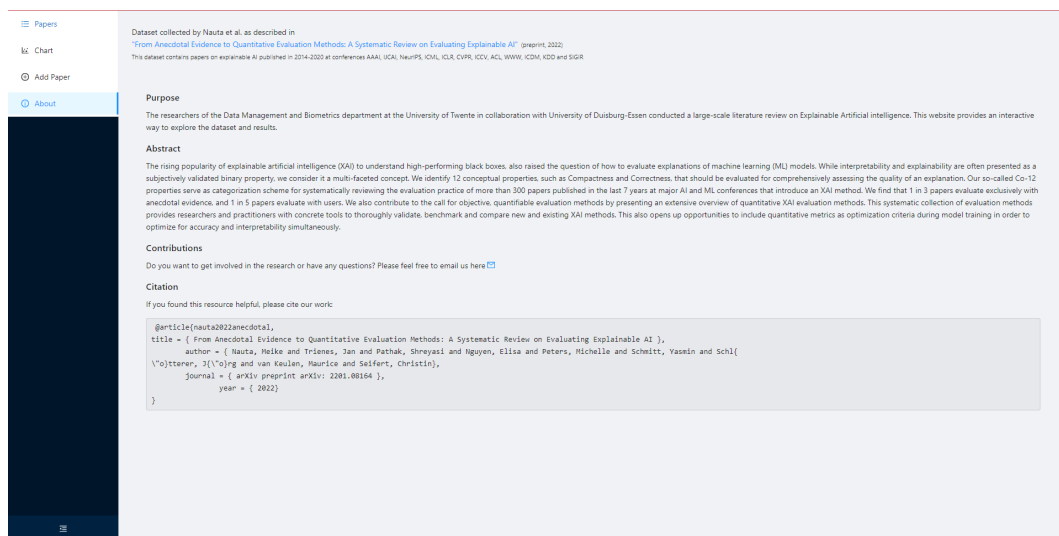


Figure 11: About Page end of Sprint 2

4.4.4 Sprint 3

Milestones during this sprint

- Combined "Papers" and "About" pages into a single page
- Added download button for the data set that functions with filters
- Added commentary field to database
- Proper labelling for all relevant sections
- "Remove all filters" button added in overview
- Tags to distinguish original data set and newly added papers
- Various bug fixes

Product update

The major overhaul in this sprint was the combination of the Papers and About page into a single page under the name of "Papers". This was done in order to create a default landing page which provided users with the necessary information rather than switching to another page to discover the purpose.

Furthermore, the logos for the University of Twente and University Medicine Essen were included for each page. The Papers page had a few additions such as the download and reset filters buttons. An additional data type was added that allowed a user to distinguish from the original data set and newly added papers. This was tested by adding a new paper following the process specified in the README. The "tableau" graph was replaced with a circle packing chart after receiving feedback from the usability testing. Along with that, all graphs were given a description to offer more clarity on what type of visualization was being shown.

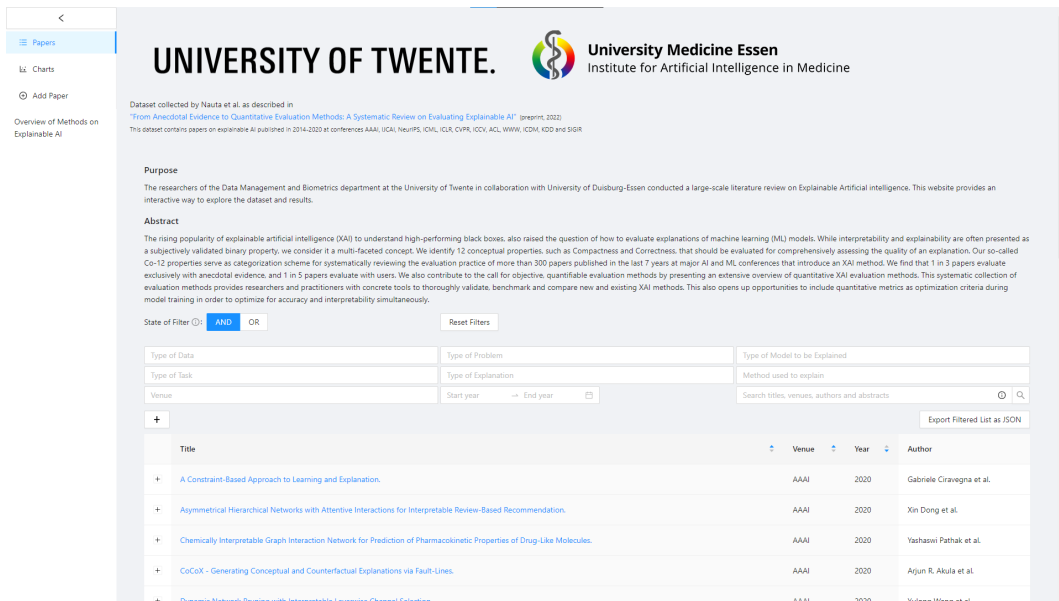


Figure 12: Papers Page end of Sprint 3

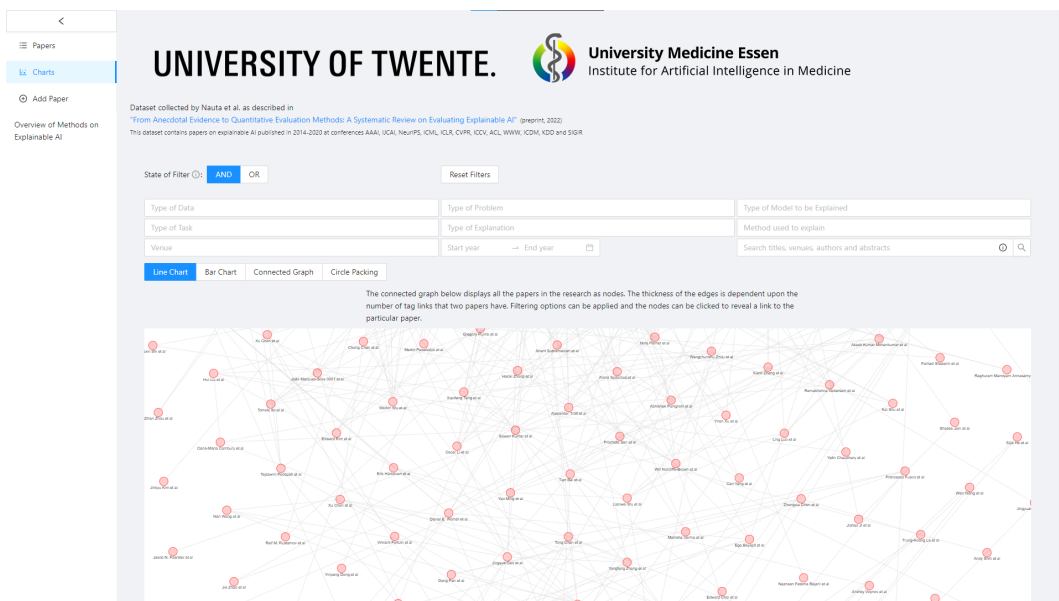


Figure 13: Charts Page end of Sprint 3

Product Description

In this section, an in depth description is provided for the finished product. A user guide exists which outlines the way the product is meant to be used along with instructions to add to the existing data types and graphs. In addition, the utilized software tools are also provided.

5.1 Use Guide

5.1.1 GitHub Pages

For the deployment of our Project we rely on GitHub Pages. GitHub Pages is a free service offered by GitHub which allows users to deploy their GitHub repository without requiring additional software. To setup up the GitHub pages the following steps have to be undertaken:

- First clone the GitHub repository to your own GitHub account. This can be done from <https://github.com/BorisGerretzen/DMBLiteratureWebsite>.
- Go to `package.json` in the main folder of the repository. and change the 4th line from `"https://borisgerretzen.github.io/DMBLiteratureWebsite/"` to `"https://youraccountnameinlowercase.github.io/DMBLiteratureWebsite/"`.
- If you wish to populate the database with the papers from the original XAI paper, make sure you have Python installed and navigate to `src/db/`. Then run the file `create_json.py` with python by executing `'python create_json.py'`. If this does not work for you, try `'py create_json.py'` or `'python3 create_json.py'`.
- Go to the GitHub *settings* → *Pages*. Then set the source to `gh-pages`.
- (For systems older than Windows 10) Install a program called Putty. Putty can be found at <https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>.
- generate a SSH keypair using either the `"ssh-keygen"` command in the command line on Windows 10 or newer or Putty on older versions.
- Go back to the GitHub *settings* → *deploy keys*. Add a new deploy key with the name `gh-pages` and the public SSH key in the field, be sure to click "allow write access".
- Now go to *secrets* → *actions* and click add new repository secret. The name should be `"ACTIONS_DEPLOY_KEY"` and the value should be the private SSH key you generated.
- You are now set to deploy the Interactive Literature Website. Go to the actions menu on the main GitHub page and go to Deploy. Click workflow, this will deploy your site, once this is done you can visit the site by going to `"https://youraccountnameinlowercase.github.io/DMBLiteratureWebsite/#/"`

5.1.2 Renaming the Project

The project has largely been build to accommodate easy name changing. Once the project has been forked, the name of the repository can be changed. After this, the only line that has to be changed to maintain functionality is in `package.json`.

`"https://borisgerretzen.github.io/DMBLiteratureWebsite/"` has to be changed to `"https://borisgerretzen.github.io/NewName/"`.

5.1.3 Adding papers to the database

To add a paper to the database please follow the following steps:

- Follow the link in the about of the GitHub to the main site.
- Navigate to the add paper page using the sidebar.
- Fill in all the required fields with the information from the paper you wish to add.
- Copy the generated JSON on the right.
- Go back to the project GitHub.
- Fork the project GitHub.
- In your fork navigate to `src/db/db.json` and click on the edit button on the right.
- Scroll to the bottom of the document.
- After the final entry's `}'` (and before the final `']'`) append the JSON code copied from the literature website.
- Commit the changes, please do not change the title of the commit from "update db.json", in the commit message please include the title, the link to the paper, and the authors of the paper you added. *The name and contents of the commit are purely a naming convention, the site and the pull request will still function if different information is given.*
- Create a new pull request for the main GitHub.

Once your pull request has been created you will have to wait for people to review your update. Once three people accept your change it will get added to the official database.

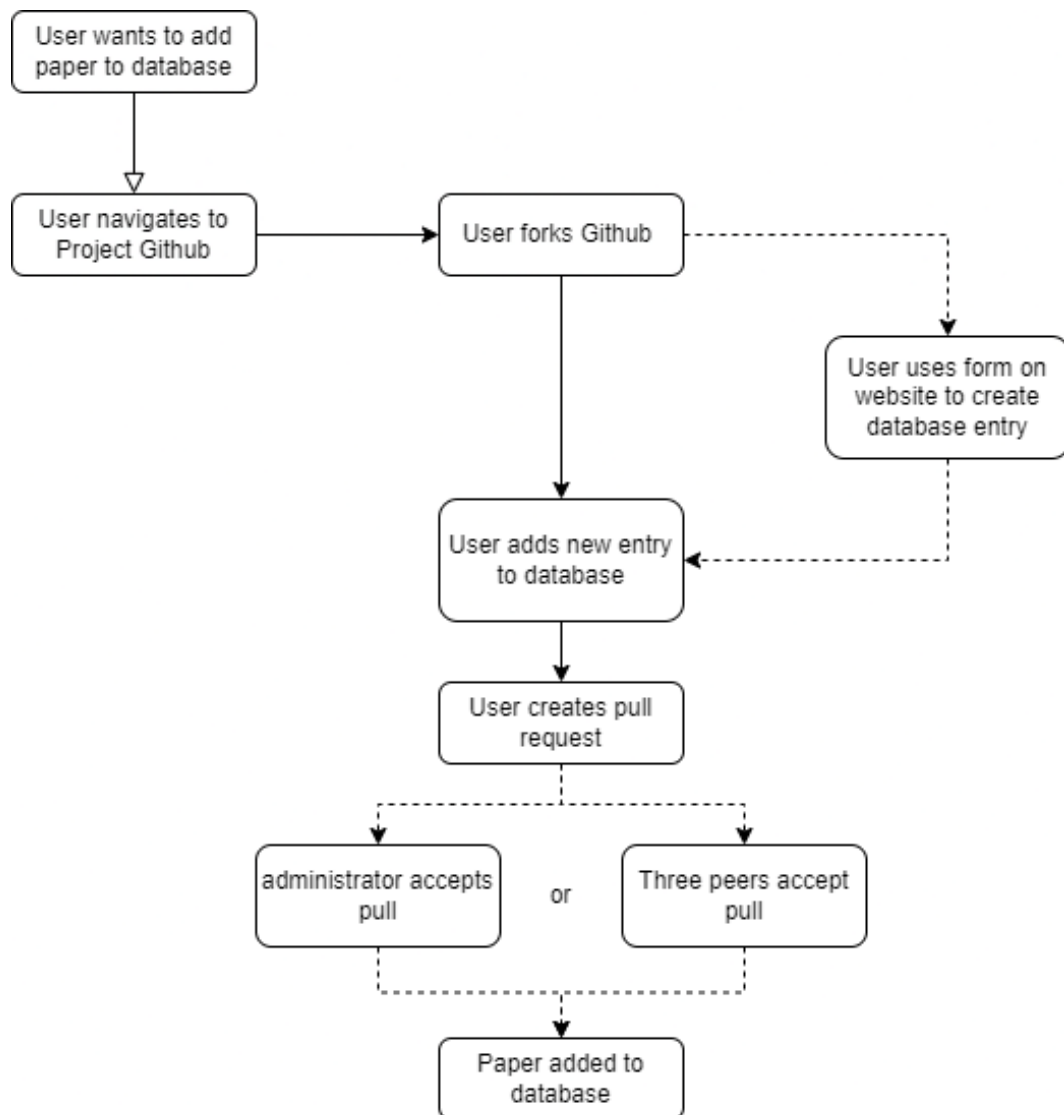


Figure 15: Diagram of Paper Submission

Reviewing Paper Submissions

To enable the requirement that changes made to the database need to be reviewed (and thus paper reviewing) please do the following:

- Go to your project's GitHub.
- Go to *settings* → *branches*.
- Add a new branch rule for the master, be sure to name this branch rule "master".
- Click require a pull request before merging, click require approvals and select how many reviews will be required.

- Click require conversation resolution before merging so a reviewer can block a submission if he deems something is incorrect.
- Click require branch to be up to date before merging.

After following the steps above all new pull requests will have to be reviewed before they are accepted into the database. To review a pull request users should do the following:

- Go to the pull request section of the GitHub.
- Click a pull request that still requires reviews with "update db.json" as the title.
- Go to the Files Changed section of the pull request.
- Check if the entry into the database is correctly formatted and has the correct tags.
- Click the review changes button, if everything is correct click approve, otherwise reject and provide an explanation why the proposed changes were rejected.

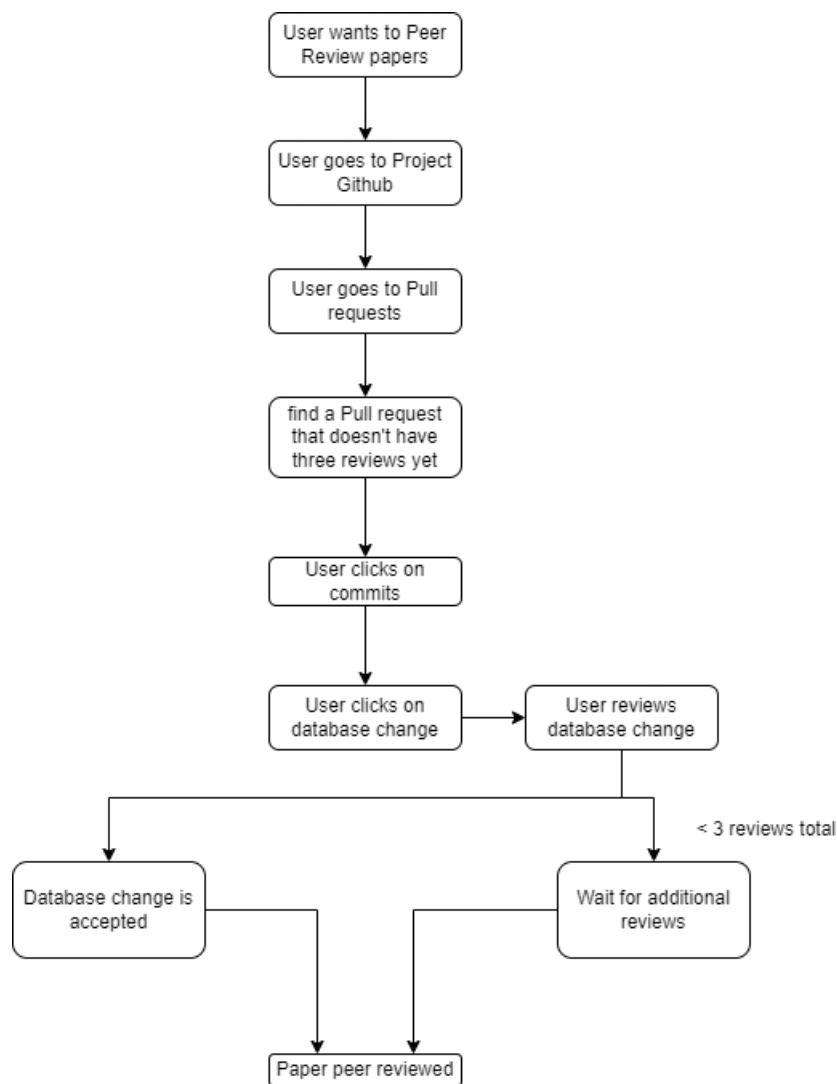


Figure 16: Diagram of Peer Review

5.1.4 Adding Additional Tags

Adding a new tag to an already existing filtering aspect is very easy. Go to `src/db/schema.json` and add the tag to the appropriate enumeration. These enumerations are used to create the relevant types in the react code and the website will then automatically be able to use it. Note that papers with the new tag will only be accepted by the database validation after the `schema.json` is changed.

5.1.5 Adding Additional Datatypes

Adding additional datatypes like "type of problem"/"type of explanation" is possible for our project. This does require a lot of files to be changed. To add a new datatype please change the following files.

- Go to `src/db/schema.json` and add a new entry, look at the other types for an example, if the new datatype is required add it to the required list at the bottom.
- Go to `src/types/paper` and add a new type object, it's easiest to copy over one of the other entries and change them appropriately. Also add a new entry to the `Paper` interface and specify the correct type. If the new datatype is not required add a question mark to the name, this tells javascript that key can be undefined in any `Paper` object.
- Go to `src/utils/paper` and add new entries to the `enumKeyMap` and `typeArray` objects. If the datatype is required, also add it to the `isPaper` function. Add a new case to the `getColor` function at the bottom, for available colors please refer to: <https://ant.design/docs/spec/colors>.
- Go to `src/redux/slices/filters.ts` and add a new reducer according to the examples already present. Also add the new datatype to the `Filters` type and the initial state of the `Filters`.
- Go to `src/components/filters.tsx` and add a new handler and select element according to examples already present, the handler takes an argument with the previously defined redux state.
- Go to `src/redux/slices/form.ts` similarly to `filters.ts` add a new reducer and change the default form state.
- Go to `src/pages/addpaper.tsx` and add a new handler which again takes the previously defined redux state. Also, in the "addpaper" component add a new item field to the form according to examples already present. If the datatype is not optional, it also needs to be included in the boolean that determines whether the json is shown.

If done correctly above steps will result in a new datatype being added to the website. It should be noted that if the new datatype is required all the older entries in the database will have to be updated before the website functions again.

5.1.6 Adding Additional Graphs

Our project offers the ability to add additional graph. Although instructions on how to add them are included it should be noted that each graph has its own requirements for implementation. Therefore we can only provide a general overview of how graphs should be added but cannot specify the fine tuning of the graphs itself. To add a graph to the website follow the steps below.

- Create a new TypeScript React (.tsx) file in the `src/components/charts` directory and give it the name of the graph.
- Import the graph from the component library of your choice. Graphin and Nivo were used for the other graphs but other component libraries can be used if desired.
- Copy over the following imports:

```
import { useFilteredPapers } from "../../hooks"
import { Paper } from "../../types"
import { NoDataChartText } from "../index"
```
- Parse the papers from `useFilteredPapers` according to the input requirements of the graph, for examples on how this can be done please review the documentation of the other graphs included in the project.
- Go to `src/pages/Charts.tsx`
- Add your graph to the imports at the top.
- In the definition of `graphMap` add a new entry.
- `withSelect` determines if a graph should have the data type selector at the top right, if so then type should be one of the "props" (arguments) of the new chart components. The element determines what will be rendered as a chart.

After following the steps above you should now have a new working graph on the website. This graph may still require fine tuning of the settings for it to work correctly.

5.2 Techstack

The software tools that we utilized in order to develop the Interactive Literature Website are:

Component Libraries: Ant Design + Nivo

Front end: React + Typescript

State management: Redux Toolkit

Hosting: GitHub pages

Repository: Github

It should be noted that this project is open source on GitHub to meet the demands for creating a collaborative community.

5.2.1 GitHub

For our project we heavily rely on GitHub. As discussed, we use GitHub pages to host the website. Because we host our website on GitHub and because our stakeholders requested it of us we also host the database on GitHub as well. Users that want to add papers to the database can do so by editing the database file in the repository. We also set up GitHub workflows to automatically check and test the correctness of the database and code, while another workflow actually redeploys the website after changes have been made to the repository.

5.2.2 React and Redux

For the project we used React as a base for the front-end. For the routing of the pages React-Router was used, as this is a popular and proven React library for routing. Redux is a supporting library used for global state management. There is global state management available for React in the Context API, however, Redux provides a much cleaner solution at little to no performance cost. Redux caches the data so it can be used between different pages and so it only has to be initialized once. Normally Redux requires a lot of boilerplate code to be used properly, though Redux Toolkit solves this issue. In addition Redux Toolkit allows for making API calls and hooking those into the state management seamlessly, if it is ever decided to start using APIs for any part of the site.

5.2.3 Component libraries

For our project we are using two different component libraries Ant Design, and its sub-library Graphin, and Nivo. Graphin was used to create the Connected Papers Graph on the Chart Page. Nivo was used to create the other graphs. The rest of the website was constructed using Ant Design, by using a single library to construct our website we could ensure a sleek looking and uniform design.

5.3 Pages

5.3.1 Home Page

The home page is the standard landing page of the website. It provides information on the original research paper released, as well as the website itself. Short introductory explanations are given for how the website functions and how it should be used. At the bottom of the page the contributors and an easy way to cite the original paper can be found.

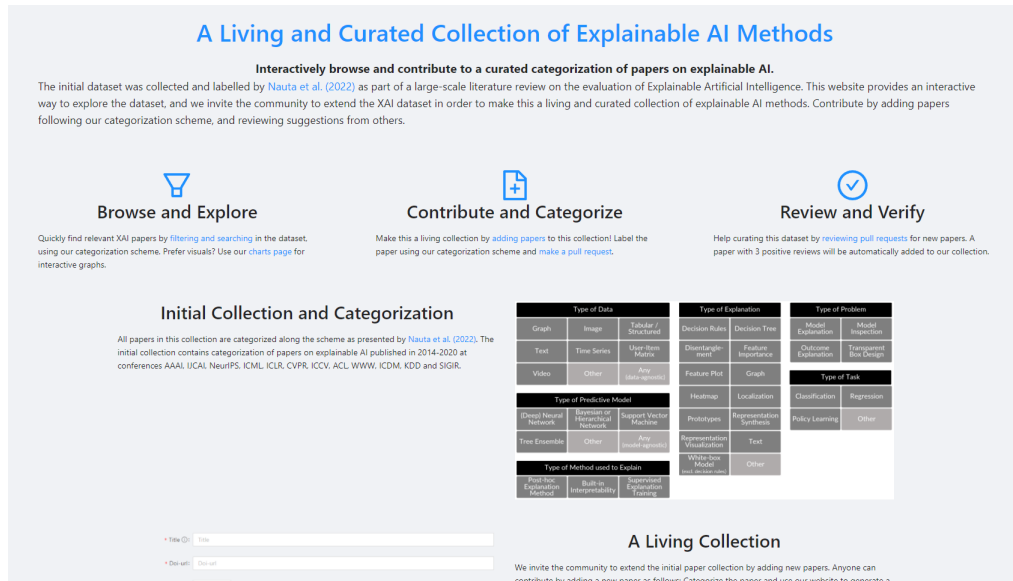


Figure 17: Final Version Home Page

5.3.2 Papers Page

The papers page can be seen as the main component of our site. It features a list of all the papers included in the current version of the database as well as options to filter them. From the top to bottom the features include:

- An And/Or toggle for the filters, as the tooltip explains it is used to switch between selecting all papers with the selected tags or just one of the selected tags.
- A reset filter button to reset all currently selected filters.
- Filters for all the categories included in the original study.
- A year selector to filter the papers on their year of publishing.
- A search field which can search in all text or in titles, venues, authors or abstracts specifically.
- An expand all button to expand/collapse all papers on the current page.
- A button to export the currently filtered list as a JSON, this can then be used in your own research.
- The ability to sort papers based on title, venue, and year.
- Each paper unexpanded shows its title, venue, year, and author, as well as if it was in the original dataset or not.
- Each paper can be expanded to show the abstract (if available), the authors and the paper's tags. If a paper was submitted at a later date it will also show when it was submitted to the database.

Note that filters selected here carry over to the Chart page.

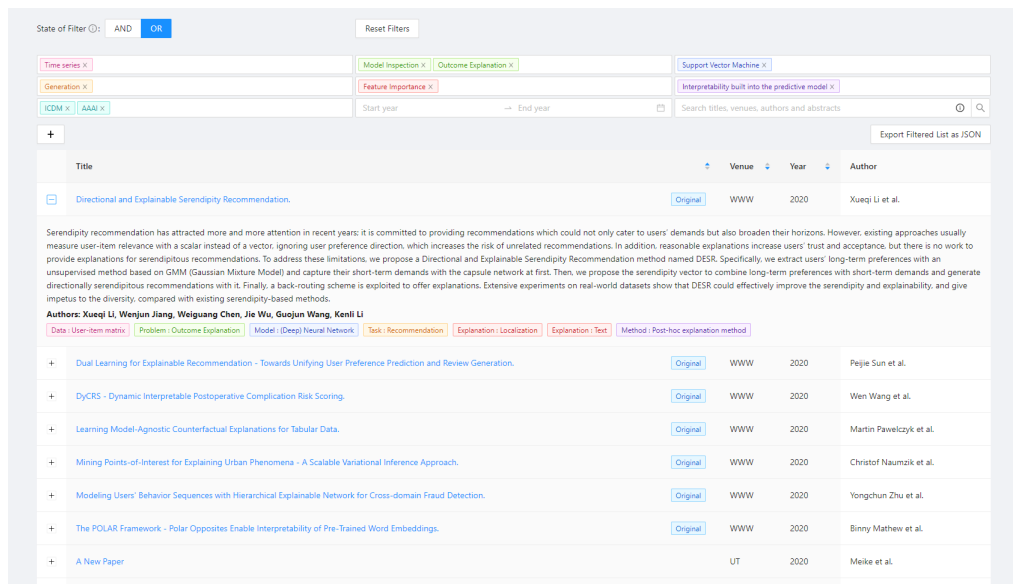


Figure 18: Final Version Papers Page

5.3.3 Chart Page

The charts page offers visualisation for filters you have selected. The final version of this page offers four different charts:

- The line chart shows the increase in tags over time, different categories can be selected in the top right.
- The Bar chart shows the amount of tags of a given type per category, as with the line chart different categories can be selected in the top right.
- The Connected Papers chart shows how the papers from your current selection are connected, the edges between papers thicken depending on the amount of shared tags.
- The Circle Packing chart shows the distribution of tags and papers belonging to each category, the selected category can be selected on the right.

Note that filters from this page carry over to the Papers page.

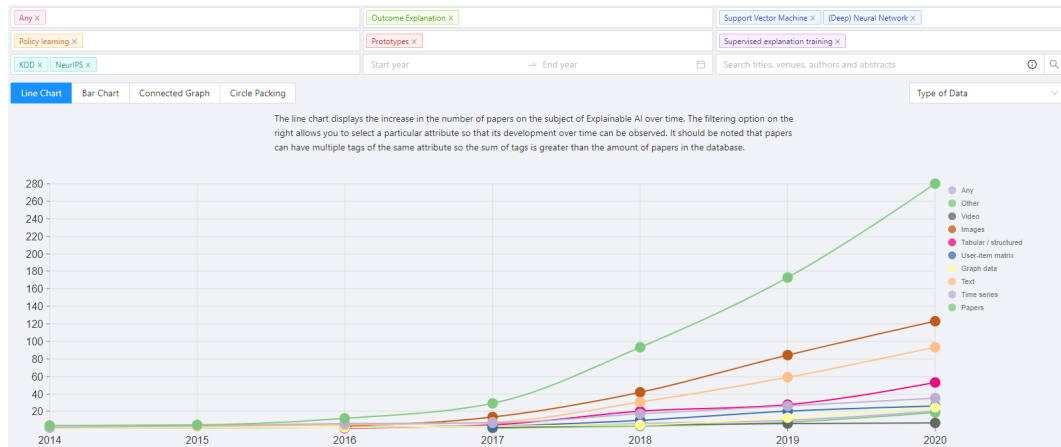


Figure 19: Final Version Charts Page

5.3.4 Add Paper Page

The Add Paper page offers users the ability to add their own papers to the database. By filling in the form a JSON template is generated that can then be entered into the GitHub Database JSON file. Each of the fields that needs to be filled out has a explanation for what should be entered and what the tags mean.

* Title ⓘ:

* Doi-url:

* Year of Publication:

* Authors ⓘ:

* Venue:

* Type of Data ⓘ:

* Type of Problem:

* Type of Model to be Explained ⓘ:

* Type of Task ⓘ:

* Method used to explain:

* Type of Explanation:

Figure 20: Final Version Add Paper Page

5.4 Known Issues

Though we would have liked to deliver a flawless product that meets all our stakeholder's requirements there are some features that still need improvement or expansion.

- Not all papers have an abstract due to not all DOI links being present in the original database.
- Abstracts are not automatically filled out if a user enters a paper into the database without abstract. Though abstracts can be retrieved using some APIs, not all abstracts are included here. In addition, this API has a limit to the amount of calls that are

allowed to be made. Though this limit might be circumvented, we were not able to find a solution during the timeframe.

- Though it is relatively easy to add extra graphs and tags to the website, adding additional filtering options is quite difficult. This is due to a large number of files that have to be changed in order to accommodate this new filtering option.

6

Testing

Below you will find an overview of the testing done during the development of this project. During the project we did two usability tests to guide development. We implemented multiple checks for code cleanliness and database validation. At the end of development we also tested cross browser functionality and set up integration testing.

6.1 Usability Testing

Usability testing involves testing user interaction with the system. Test participants were asked to complete a set of tasks which pertain to various functionalities of the product. We recorded their reactions and checked if the system was intuitive to regular users. Using the results of these tests we were able to guide the design to be the most user friendly possible.

6.1.1 First Usability Test

At the end of sprint 0 we conducted our first usability test using our Lo-Fi prototype. This was a very successful test and we were able to glean many interesting insights that were used to guide the design of our actual site. Many of the features that we drew up for the prototype saw their return and expansion in the final design of the website. For this test we also designed a page for reviewing papers, this was before we decided to switch to GitHub for deployment. After the switch this page was dropped and saw no return in the final design. For this first Usability test we interviewed our supervisor, Meike Nauta.

6.1.2 Second Usability Test

At the end of sprint 2 we conducted a second usability test. For this second test we used the site hosted on GitHub pages. For the exact questions that we asked our participants and the responses that we got please see the appendix chapter 10. For this test we had four different participants which really helped us get a complete overview of all changes and additions that still needed to be made before we finalized our product. The four participants for this test were: Meike Nauta, Jan Trienes, Shreyasi Pathak and Michelle Peters, all of which worked on the original research paper.

6.2 Cross Browser Testing

The expected functionality of the website was tested across different browsers. The browser on which we tested our website were Google Chrome, Microsoft Edge, Brave, Vivaldi and Fire Fox. All the features functioned fully on all of these sites and therefore our Cross Browser Test was deemed a success.

6.3 Integration testing

To ensure that different parts of our website work together in the way we expect them to, we will be making use of integration testing. These tests will be run every time a pull request is created to make sure no unintended behaviour ends up in the codebase.

6.4 Database Validation and Code Cleanliness Checking

Two scripts were programmed to test the validity of the database and the cleanliness of the code. Both of these scripts can be found in `.github/workflow` sub-folder of the project. The code cleanliness is checked using ESLint and if the code is deemed unclean the build of the website will fail. The script for the code cleanliness is named `linter.yml`. Database Validation is done by the `main.yml` script and again fails the build if the database does not adhere to the schema specified in `src/db/schema.json`. In this section, we look at the various testing which was conducted during the development of the project. Most of the tests are incorporated with GitHub and therefore are checked once a pull request is made.

7

Future Work

This section looks at any future developments that the product can incorporate. Due to the time constraints, not all features could be implemented. Developers can utilize this section to aid in any updates they want to make to the website.

7.1 Automatic Tag Parsing

The client requested a feature to allow automatic parsing of the tags for a new paper entry. The finished product currently allows users to manually select the tags when adding papers. Therefore, there is a lot of emphasis on user input which can result in the submission of wrong entries. Through the combination of natural language processing and machine learning, the human user input element can be removed. Future iterations of the website can incorporate those elements and means a person can add a new paper which will be scanned by an algorithm in order to determine all the relevant tags and automatically fill out all the field entries. With more researchers likely to get involved to expand the research, automation in adding a new paper would prove to be useful.

7.2 Full Mobile functionality

The website can also be extended to account for mobile devices. A could have requirement existed but due to time constraints this feature was not able to be implemented. In addition, we imagined users will primarily be using the website on personal computers rather than mobile devices. The process of adding a new paper is much more convenient if done on a PC rather than a phone considering the switch to the GitHub repository. If more people end up using the website, then there could be greater demand for complete mobile functionality. Therefore, it can be added later on.

7.3 Additional Visualization Options

The current iteration of the product includes 4 graphical options. Depending on the feedback from users, more graphical options can be added. Additionally, different graphical libraries can be explored which can potentially improve upon the functionality and design of the current graphs.

8

Code Review

This section provides an overview of the file structure. A directory tree has been drawn up along with descriptions of the most important files and folders in the project.

8.1 Directory tree

```
github/workflows
├── deploy.yml
├── linter.yml
├── main.yml
└── test.yml
public
├── explanation_types_examples
│   ├── decision_rules_16.png
│   ├── decision_tree_19.png
│   ├── disentanglement_5.png
│   ├── feature_importance_22.png
│   ├── feature_plot_9.png
│   ├── graph_27.png
│   ├── heatmap_25.png
│   ├── Localization_22.png
│   ├── prototypes_14.png
│   ├── representation_synthesis_33.png
│   ├── representation_visualization_29.png
│   ├── text_21.png
│   └── white_box_model_28.png
├── add_paper_form_half.png
├── add_paper_form.png
├── IKIM_Logo.png
├── index.html
├── manifest.json
├── protocol_categories.png
├── pull_request_unreviewed.png
├── UT_Logo_Horizontal_Black.png
├── UT_Logo_Horizontal_White.png
├── UT_Logo_Vertical_White.svg
└── UT_Logo_Vertical_Black.svg
src
├── _test_
│   ├── Filters.test.tsx
│   └── Papers.test.tsx
├── components
│   ├── charts
│   │   ├── BarChart.tsx
│   │   ├── CirclePackingChart.tsx
│   │   ├── ConnectedChart.tsx
│   │   ├── GrowthLineChart.tsx
│   │   ├── index.ts
│   │   └── NoDataText.tsx
│   ├── Filters.tsx
│   ├── Select.tsx
│   └── TagList.tsx
```

```
├── index.ts
├── db
│   ├── .gitignore
│   ├── create_json.py
│   ├── db.csv
│   ├── db.json
│   ├── results.json
│   └── schema.json
├── hooks
│   ├── expand-all-table.ts
│   ├── filtered-papers.ts
│   ├── index.ts
│   └── redux.ts
├── pages
│   ├── AppPaper.tsx
│   ├── Charts.tsx
│   ├── Home.tsx
│   ├── index.ts
│   └── Papers.tsx
├── redux
│   ├── slices
│   │   ├── filters.ts
│   │   ├── form.ts
│   │   ├── index.ts
│   │   └── papers.ts
│   ├── index.ts
│   └── store.ts
├── types
│   ├── index.ts
│   ├── navigation.ts
│   ├── paper.ts
│   └── select.ts
├── utils
│   ├── index.ts
│   ├── navigation.ts
│   ├── paper.ts
│   ├── select.ts
│   └── utils.ts
├── index.tsx
├── setupTests.ts
├── .eslintignore
├── .eslintrc.json
├── .gitignore
├── README.md
├── package-lock.json
├── package.json
└── tsconfig.json
```

8.2 File and Folder Descriptions

`.github/workflows` : It contains all the github actions files that are used for the different checks that happen for every pull request as well as the automatic deployment of the site.

`public` folder : It contains all images used in the website as well as the root `index.html` where the website is injected into by react. The `<head>` tag of the `index.html` can be changed as desired and works like a normal `index.html` file. The `<body>` tag should remain the same as all components and other website parts are handled through react, not html.

`src` folder: It contains all source code for the website.

File `src/index.tsx`: It is the entry point of the React code and has the components that are needed for every part of the site to function, including the Redux store and the HashRouter from React-Router. In addition it defines the layout of the sidebar as this is shared between all pages.

`test` folder: It contains the different test files that are run during the test part of the github actions check.

`db` folder: It contains the original csv version of the database and the python script that turns it into the desired json file. In addition the `db.json` is also in here, along with the `schema.json` which is used to determine the correctness of the `db.json` for a pull request.

`redux` folder: It contains the files for the redux store and the different "slices" of redux state. These redux state slices are cached and are mostly used to keep information in memory between different pages (i.e the state of the filters are in a redux slice and so are not lost when navigating between the Papers and Charts pages). The `papers` file in the `slices` folder reads the `db.json` and imports all papers into the redux store using the correct typing.

`pages` folder: It contains the different page components such as Papers and Charts.

`components` folder: It contains the different smaller components that are reused throughout the site. Examples include the different charts in the `charts` subfolder and the `Select.tsx`, which specifies the desired behaviour of the Ant Design Select component to make it easier to reuse.

`hooks` folder: It contains some custom React Hooks. Redux requires some custom hooks so typescript can be utilized. Also the filtering of the papers happens inside a hook, so all components that require the filtered papers list can easily share this functionality.

`utils` folder: It contains some utility functions for different parts of the website. Especially the `navigation` file is important here, as all the github base url is defined here, as well as some other useful functions.

`types` folder: It contains some of the different custom types. Especially the `paper` file is important here, as the `paper` file reads the `schema.json` and creates objects that define the types for the different tags. Ideally these actually would be types, instead of objects, however due to typescript limitations we can not create types at runtime. If the types would instead be hardcoded (instead of read from `schema.json`), then any change to the tags would require changes at 2 different places.

Team Contributions and Reflections

In this section, we reflect on the planning that was done at the beginning of the module. In addition, a detailed description of the contributions of each team member is also highlighted.

9.1 Planning Reflection

Adopting the SCRUM methodology proved to be successful for the completion of this project. Dividing our work in two week sprints meant there was sufficient time for development of new features for which we could receive suitable feedback for. By choosing an iterative approach, the design a lot from sprint to sprint but it ultimately led to a final product which the stakeholders were satisfied with.

Sprint 0 was designated for the collection of requirements and initial design. Interviews were conducted with the stakeholders to understand what types of functionality they were looking for so that we could make a decision regarding our choice of software tools. A low fidelity prototype was also created and tested with the client. The goals of the sprint were reached as a solid understanding was developed on what the product should look like and the types of features it should contain.

Sprint 1 focused the attention towards the development of the product. The first week was spent setting up the project. However, after having a meeting with the stakeholders, the tech stack had to be modified as the client expressed preference towards utilizing Github pages. This meant the work done towards setting up was redundant but as no actual development had started, the tech stack change did not result in any problems. The goals of the sprint were met in the following week as basic design and functionality was implemented.

Sprint 2 involved us just continuing the development that was started in the previous sprint. More pages were added with most of the requested functionality. A thorough usability testing was also conducted at the end of this sprint. We were on track and had been able to gain feedback from multiple users which would aid us in implementing the changes over the last few weeks.

Sprint 3 focused on addressing the feedback from the usability testing that had taken place along with the addition of a separate landing page. Testing was also of high priority. Along with that, the integration with Github was also tested in order to ensure adding new papers was a seamless process. The changes and testing were all achieved on time and meant that we could focus our attention towards the final presentation to the stakeholders along with designing the poster.

The last sprint meant the conclusion of design changes and displaying the work done throughout the course of the module. A fully functional product was shown and the members of the DMB group were able to interact with it. Minor changes were requested after the presentation but as everything else was on track, they were implemented with ease.

In conclusion, the initial planning was able to be followed with ease. The team was able to work effectively and all deadlines were met as a result. Combined with the iterative approach, changes were always able to be made on request with the client. Weekly meetings meant the client could regularly test any updates that were implemented whilst communication through Slack allowed for swift exchange of ideas and feedback. All this lead to a final product that the group is proud of.

9.2 Team Contributions

9.2.1 Frans

Frans was involved in writing up the design and reflection reports with Abdullah. He also shared presentation duties during the peer review sessions whilst also taking part in the usability testing. On the final presentation as he provided an overview of the design process. On the development end, Frans worked out the process of adding a new paper to the repository along with properly citing relevant papers for informative purposes on the "Add Paper" page.

9.2.2 Gies

Due to his previous experience with development using React, Gies was heavily involved in the front end development. He was responsible for the layout and functionality of the filters, the papers table and the functionality of the add paper form. In addition, he also integrated the data to the charts and created the landing page towards the end. Further contributions include setting up the Redux which is pivotal for all the state management that occurs on the site. Furthermore, he was involved in various bug fixes and assisting in any typescript issues that came up. For the report, he wrote up the Code Review section which outlines the purpose of the important files and folders. Lastly, he ensured that all the code was properly documented for future maintenance.

9.2.3 Abdullah

Abdullah created the low fidelity prototype in the first few weeks and conducted the initial interview for the collection of requirements. On the front end, he was responsible for table and form design along with the creation of the About page which was converted to the Home page at the end. He was actively involved in the usability testing along with sharing presentation responsibility during peer review sessions. For the final presentation, he provided a demo of the website along with the process of adding a new entry. He had a major contribution towards the design report whilst also writing up the reflection report with Frans. Lastly, he aided in the poster design.

9.2.4 Ramish

Ramish was responsible for the development of the charts. He discovered the various libraries which gave access to the different visualization options. He proceeded to take part in implementing and integrating the four charts with the data-set. In addition, he also aided in the design for the low fidelity prototype and with the layout and design of the website.

9.2.5 Boris

Boris was primarily responsible for the workings on the database end. He setup the Github repository along with ensuring the functionality with GitHub actions. Along with that, the repository also included various tests such as code cleanliness that checks proper code formatting and database validation which checks the formatting of the db.json file. In addition, he also conducted integration testing on various front end components as well as writing the algorithm for the connected graph.

10

Appendix

This section included the appendices of the project. It comprises of the initial interview and the results of the usability testing.

10.1 Initial Stakeholder Interview

1. When selecting tags for a new paper being added to the system should tags be selected from a list or can a user input their own tags?

Elisa: Type

Meike: There will need to be a check on new papers getting added. Concerns about time consumption. Current scope: Only works with current available tags. Good idea for future implementation. Maybe train a model on title and abstract for type of explanation. Maybe make a distinction between manual checks and automatic checks. Type of data and type of explanation devised by machine learning. Use something existing (GitHub accounts) instead of making our own account system

Jan: Too little time to manually check. So let submitters submit their own tags. Maybe allow peer reviews on paper tags. Everyone should be able to contribute

2. Is it interesting to filter papers based on the author and/or venue?

Elisa: Agree

Meike: Very important to filter on venues, this can determine the importance of papers. Ability to search for authors would be very useful as well

Jan:

3. What are the intentions for the submitted papers, (fully available text, abstract)?

Elisa: Maybe check if all venues are in scopus

Meike: in data only allow link. So we would have to scrape the abstract. Might be hard to do. Might be very useful when training a model on abstract. Making the scraping function very useful. Find the amount of uses for papers

Jan: DOI is a minimum requirement. Lots of different venues. Maybe find a database for abstracts

4. Can a paper have multiple tags?

Elisa:

Meike: Papers can have multiple tags

Jan:

5. In the case of connected papers, how should the distance/size of the papers be determined?

Elisa: If users want to create heatmap

Meike: If papers have similar tags distance zero, find out ourselves. No currently set papers. Show first x results of other websites

Jan:

6. Do users need to create an account to submit a paper or do they simply supply their contact information?

Elisa:

Meike: Users and admins have accounts. But don't let people make accounts. Use something that is already available. Maybe make a user interface on github pages. Also done to disallow spam

Jan:

7. Would the admin like to be able to extract the data from the website, like export it to a csv or other file?

Elisa:

Meike: Yes export and import. Both complete replacement as well as addition/subtraction. Add checkmark for full replacement

Jan: Maybe work using a github csv file. Json might be a better data structure

8. Would the admin like to be able to add and/or remove tags?

Elisa:

Meike: Yes, and add ability to columns

Jan:

9. Are there any requirements that you feel are "nice to have" but not integral for the successful implementation of the product?

Elisa:

Meike: Add and disable functionality. So modular design?

Jan: Once the project is over, maintainability is very important so good documentation. Good documentation of other packages and software to make fixing things easier. Ability to change the current dataset easily

10. Is mobile friendliness an important feature?

Elisa:

Meike: nice to have, not needed. Notification is easier on the desktop. Look at if easy

to implement

Jan:

11. What visualizations would you like to have in addition to the connected papers one and growth of tags over time? (Bar/Line/Other Graphs)

Elisa: For anyone looking for papers, ability to generate graphs based on their own criteria

Meike: Graph over time of all papers, filter on tags. Deeper look after iteration

Jan: predesigned options better than complete flexibility in graphs

12. Should the product automatically make a backup of the database?

Elisa:

Meike: Yes, backup

Jan:

13. If yes, do you want to control the time interval?

Elisa:

Meike: whenever new paper is added, make a backup. Keep backups for a week or a month then dump. Github as well. Maybe limited functionality

Jan: Maybe look into GitHub for data

14. Should a researcher/user be able to request removal of their previously submitted paper?

Elisa: Don't know about legal authority about removal of data. Maybe edit history

Meike: Only allow authors to add papers or allow all users. Having the option to delete data is very important. Should not allow full paper to be database due to legal issues. Maybe allow people to indicate the author, XAI authors or others. Only the admin can actually delete papers

Jan: Create a separate tab for peer review. So papers can be added to the database

10.2 Second Usability Test

1. Search for the papers with the author Zhang

Michelle Peters: Found them easily

Jan Trienes: Found them easily

Shreyasi Pathak: Found them easily

2. Search in the abstract, the keyword “New” and tell us the name of the first entry→
Explaining Black Box Predictions And Unveiling Data Artifacts through Influence Functions

Michelle Peters: Used the info icon to get a description of the search functionality. Was able to find the paper afterwards

Jan Trienes: Utilized the info icon to get more information and then found the paper

Shreyasi Pathak: Utilized the info icon to get more information and then found the paper

3. Find the papers with either data type: images OR type of problem: Model Explanation

Michelle Peters: Found them without issue

Jan Trienes: Found them without issue

Shreyasi Pathak: Found them without issue

4. Find the bibtex citation

Michelle Peters: Found it by checking the page titles and then navigating to the bottom of About

Jan Trienes: Found it by checking the page titles and then navigating to the bottom of About

Shreyasi Pathak: Found it by checking the page titles and then navigating to the bottom of About

5. Change the connected papers graph to show only the ones with Type of Task: Regression OR Type of Explanation: Feature plot

Michelle Peters: Managed with ease

Jan Trienes: Managed with ease

Shreyasi Pathak: Managed with ease

6. Change the connected papers graph to show only the ones with Type of Explanation: Text AND which were published in the AAAI venue

Michelle Peters: Managed with ease

Jan Trienes: Managed with ease

Shreyasi Pathak: Managed with ease

7. See if they can find out how a connected chart works, as in how they are connected.

Michelle Peters: Thought of the tags but was not sure.

Jan Trienes: Was confused as to how the edges worked Saw a certain paper as a central node and thought it was citations

Shreyasi Pathak: Working in similar fields but thought maybe also citations. Then said common values in the field

8. Switch to the Line Chart and tell us how many papers had Type of explanation: Text in 2016 → None

Michelle Peters: Managed to find it

Jan Trienes: Managed to find it

Shreyasi Pathak: Managed to find it

9. A new paper has been added with an any tag (Venue), look for it and tell us what the title of the paper is and who wrote it

Michelle Peters: Paper was not appearing but after refreshing and retrying, was able to find it

Jan Trienes: Paper was not appearing but after refreshing and retrying, was able to find it

Shreyasi Pathak: Paper was not appearing but after refreshing and retrying, was able to find it

10. Add a new paper and see what the formatting of the output is

Michelle Peters: Found it easy to add the entries. Suggested specifying fields for the other data sets. May make it less clean but allow more options for the users

Jan Trienes: Suggested adding a general comment field to specify further additions along with having a collapsible prior to entry fields to provide more information if necessary

Shreyasi Pathak: Thought it was easy to use