OpenEdu Team C

19 NOVEMBER 2022

Empowering Knowledge

Access to open education resources on any topic anywhere you need

Project Proposal

Prepared by:
Iwona Braun-Nowak
Marcela Rozo
Claudia Porcellato
Carol Hsu
Daniela Pavia Santolamazza
Kevin Zihlmann

Table of Contents

1. Introduction

1.1. Initial Situation

OpenEdu is an open education platform launched in 2020 by wikimedia CH.
Open education platforms allow their users to edit and integrate projects, training tools and news from the world of open education. The goal of the platform is to help educators in the orientation towards finding inspiration for new teaching methodologies or professional updating.
We started a 6-week project on OpenEdu.ch in order to make it more effective and efficient while allowing the user to enjoy using the platform. Our main task was to propose an ontology for storing content metadata, as well as to propose a complete data architecture. The client also wanted us to review and suggest solutions for the uploading procedure.

1.2. Our approach and main goals

Our main challenge in the first phase (discovery) phase was understanding the current OpenEdu structure and its functionalities.

Once the understanding of OpenEdu was clearer and the possible areas of improvements were defined to make the platform more effective and efficient, the approach was focused on the following challenges & end user's needs and problems:

- Usability: It is difficult for the user to find information, resources, or educational material on the website
- User experience: The Interface of the website is overloaded creating cognitive load to the end user
- Value proposition: The aim of the website does not send a clear message

In order to maximize the user experience and provide them with an easy to use platform, the solutions proposed to improve the website are:

- Developed an ontology to provide the database a better structure with the intention of easing the data collection and data processing
- Optimized the Search feature of the website and the related content feature related to the search that has been done
- Maximized the user experience by reducing information on landing page and on a better way to ease the uploading process of new projects

2. Review of initial condition

2.1. Reverse engineering: review of the existing OPENEDU website and functionality

This review is the result of team members using the webpage and back engineering of its usability. Later, some insight came from the interviews with possible users of OPENEDU.

Filters

We find there are some issues within the filters and that these are not so easy to understand. Here some of the issues we considered should be improved:

1. The filters should be implemented in series. If one chooses a given entry on a filter, in the next filter only those entries containing overlapping projects should be available. Ex. If the level beginner is chosen and there are only projects for beginners written in English, then in the language filter only English should appear, and not as for now that all languages can still be chosen.
2. The original language in which a given project exists in the Wikimedia community should agree with the language(s) chosen to show the project.For example the project "READING WIKIPEDIA IN THE CLASSROOM" is shown no matter the language chosen in the filters. However, on the project website, the project information is not given in all the languages. So, the user might have expected to find the material in French, but it is not available. A differentiation must be done specifying in which language the openedu user can find the documentation.
3. The three main filters (training, projects and news) are confusing. The user might get a wrong idea of the capabilities and scope of openedu. We suggest having a cleaner homepage, that is increasing capability and filters as one scroll down the webpage. Maybe something like in this web page https://www.europeana.eu/fr
4. The difference between category and topic is not clear. We suggest keeping only one filter that has the general topics and maybe consider implementing a second filter containing subtopics if when the platform is running the users consider it a useful feature. Further, by improving the search, as will be suggested below, maybe less filters are needed.

Related content

Related content is being assigned manually by the uploading user. If the amount of content gets large this comparison cannot be made by a person anymore. Also, if the uploader is the one suggesting the related content, the results will be subjective to the specific uploader ideology, and the final user will get a biased set of related projects. We suggest using Natural Language Processing (NLP) to automate this task (see chapter NLP)

Search

We did different tests, to analyze how the search is currently implemented. We realized key words are only searched on the title and subtitle. We considered the description of the project should also be considered on the search. Here we also suggest two use NLP to improve the search (see chapter NLP)

User interface

From our experience and the interview with the users, the first arrival to the webpage results in an overload from too much information. The first impression is that it is not clear what one can achieve by searching in Openedu.

2.2. Review OPENEDU database and data structure

The existing structure of OPENEDU is a relational database in postgreSQL. There is a main table summarizing the OPENEDU entries (projects). This table is related by its ID with other tables containing attributes of the openedu material. Further, there is a part of the DB schema dedicated for tracking the users. We identify that these are the standard user tables created with the Django project for deploying the website. We agree on the structure, but we think that some of the attributes should be modified and new attributes should be considered, as the ones we discussed before, and those that are summarized in our proposed ontology. Further, the current schema does not keep track of the users responsible for the OPENEDU entry. We believe this is important information that should be tracked an include in the DB

Further, we consider it a bad practice to store text on the DB including any html syntax. The DB should contain raw data, any editing and formatting should be done while the app y deploy. Ex: the descriptions include characters like <p>, this type of formatting should be done in the backend (ex. Through Django filters)

2.3. Benchmarking

For our analysis we researched existing web pages that offer educational material to serve as orientation for our solution. After some research on other webpages that offer educational material and looking at the existing data structure of openedu. A summary of the webpages considered in this project is given on the table below.

Domain	**Open source platform**
OpenEdu https://openedu.ch/en/	International organization
Oercommons https://www.oercommons.org	International organization
BC Campus https://collection.bccampus.ca	Institution
Europeana https://www.europeana.eu/en	EU organization
MIT OpenCourseWare https://ocw.mit.edu/search/	Institution
Heimshelp https://heimshelp.dese.gov.au/resources/field-of-education-types#Section1	National/Government organization
International Standard Classification of	International organization

Education (ISCED) https://ilostat.ilo.org/resources/concepts-and-definitions/classification-education/	

2.4. User research & User interviews. Creating personas

- The Process

After discussing the business goal and main requirement with the PO, we first reversed the existing website based on four criteria: design, structure, functionality and value to identify initial problems, then we gathered each member's feedback and extract key information to form user interview:

1. **The context**: In what situation educators would visit an educational resource platform

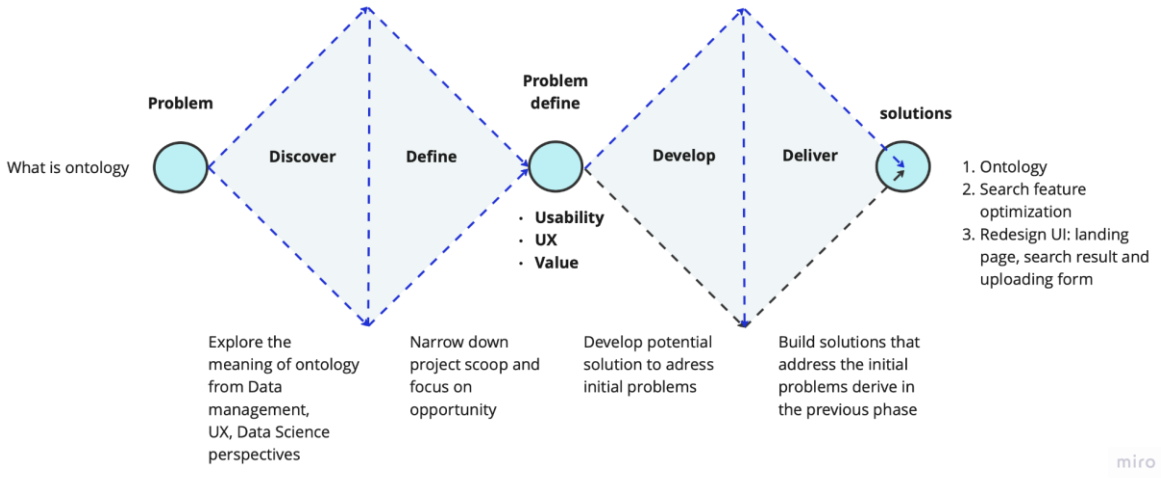
2. **Behavior**: What do educators look for when visiting educational resource platforms

3. **Motivation**: What is the motivation for using an educational resource platform

4. **Expatriation**: What product/ service do you expect to see on an educational resource platform

5. **User Engagement**: What is the motivation for engaging with other users on an educational resource platform

To achieve the business goal, the **Double Diamond Process Model** is applied as this model is not only focused on design elements but taking a holistic view to design a system that can be run technically and visually. The Double Diamond Process Model consists of four phases as follows:

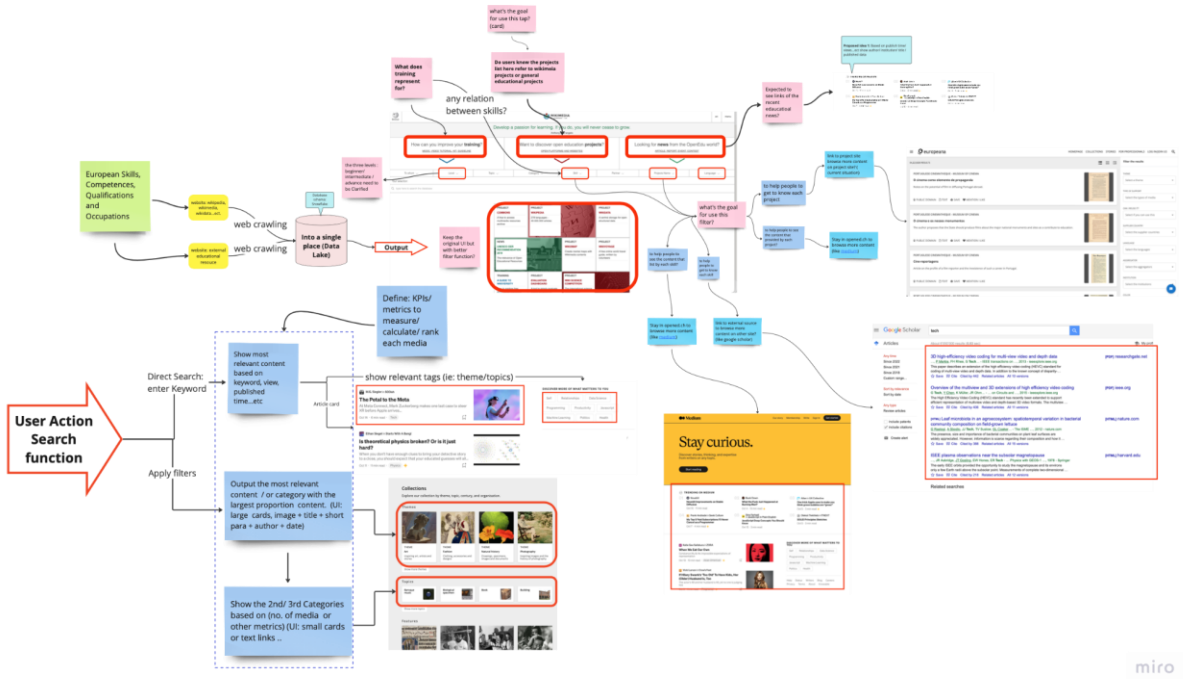


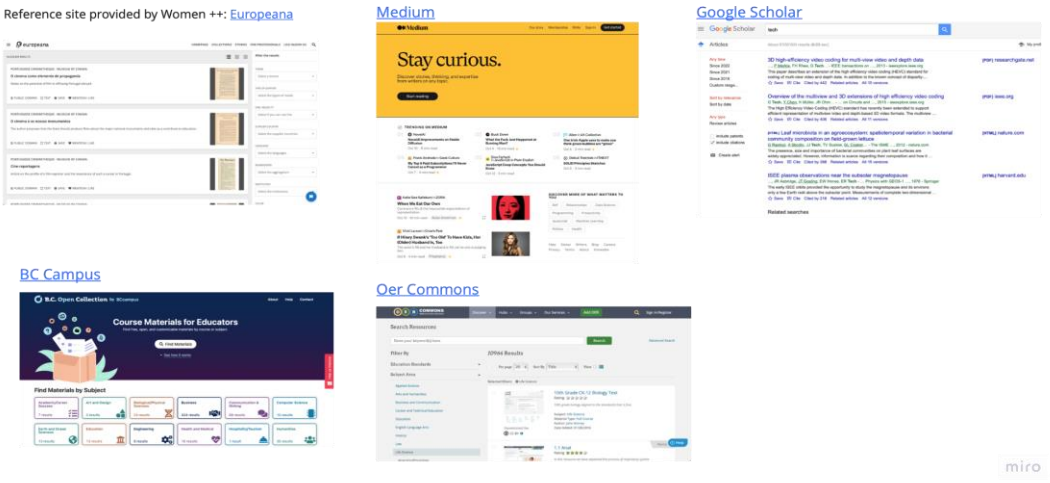
- Discover
    - Understand what is ontology; who are the key stakeholders, and how they will be affected by the ontology.

    - Reverse the existing database structure, and design, function, and service that offer on the current website
- Define:
    - Gathering insights from all team members and defining the domain and purpose of ontology and formulating the structure of the database
    - Identify key problems that need to be addressed now. We decided to focus on optimizing searching, filtering and uploading functions.
- Develop: Once we define our key problems, and common goal, we divided tasks based on member's expertise, which are **Data Engineering**, **Data Science** and **UX|UI**, and each member can focus on developing solutions
- Delivery: Proposed three actionable solutions which are implemented within six weeks
    - Refine ontology, data model and data pipeline
    - Search engine optimization with NLP approach
    - Visual Prototype: redesign landing page, result search page and upload form

Website Assessment

We compared existing Open Education Resource(OER) platforms and media sharing platform : Medium, Google Scholar, MIT, Oercommons, BC campus and Europeana and made benchmarks on information architecture, navigation, search function, filters and search results to get ideas on how to restructure OPENEDU Website.





- Personas

In addition, we also create 2 persona to get better understanding how OPENEDU.CH's user probably look like and be able to come up better solutions to tackle their problems

- User Journey and Interview

We mapped out user journey based on the persona profile and designed eight open questions as follows:

Usage of the educational platform

1. When you go to the educational platform, what was your goal/task that made you visit these sites (the sites you provided)?
2. What functions help you to complete your tasks
3. what struggle you during the process, and what can help you to complete your tasks

Uploading / sharing your work on an educational platform

1. How much time would you like to invest in the uploading process including registering as a user and filling out an information sheet?
2. If you have uploaded files on an educational platform (or any platform), what struggle did you have during the process, and what can help you to complete your tasks?

Collaboration & networking on an educational platform

1. What would motivate you to sign up as a member of an educational platform?
2. What would you expect this platform can offer you?
3. How would you like to interact with other educators?

We conducted user interviews with 5 potential users (one internal and 4 external users).

- 90 % of the Users search bar to start their search on the majority platform
- Online Community, Peer review and students feedback are critical touchpoints
- Having a clear instructions and guidance would help the user to complete the uploading process
- Users wishing to get in touch with other teachers and connect in a simple, easy way

■ Development

We collected ideas from team brainstorming, user interview and and competitive analysis, and we finally selected the most prominent features which are: Search UI, Filters, Uploading Form, and Redesign Landing page that are aligned with solution from our technical team, and be able to visualize in the final prototype

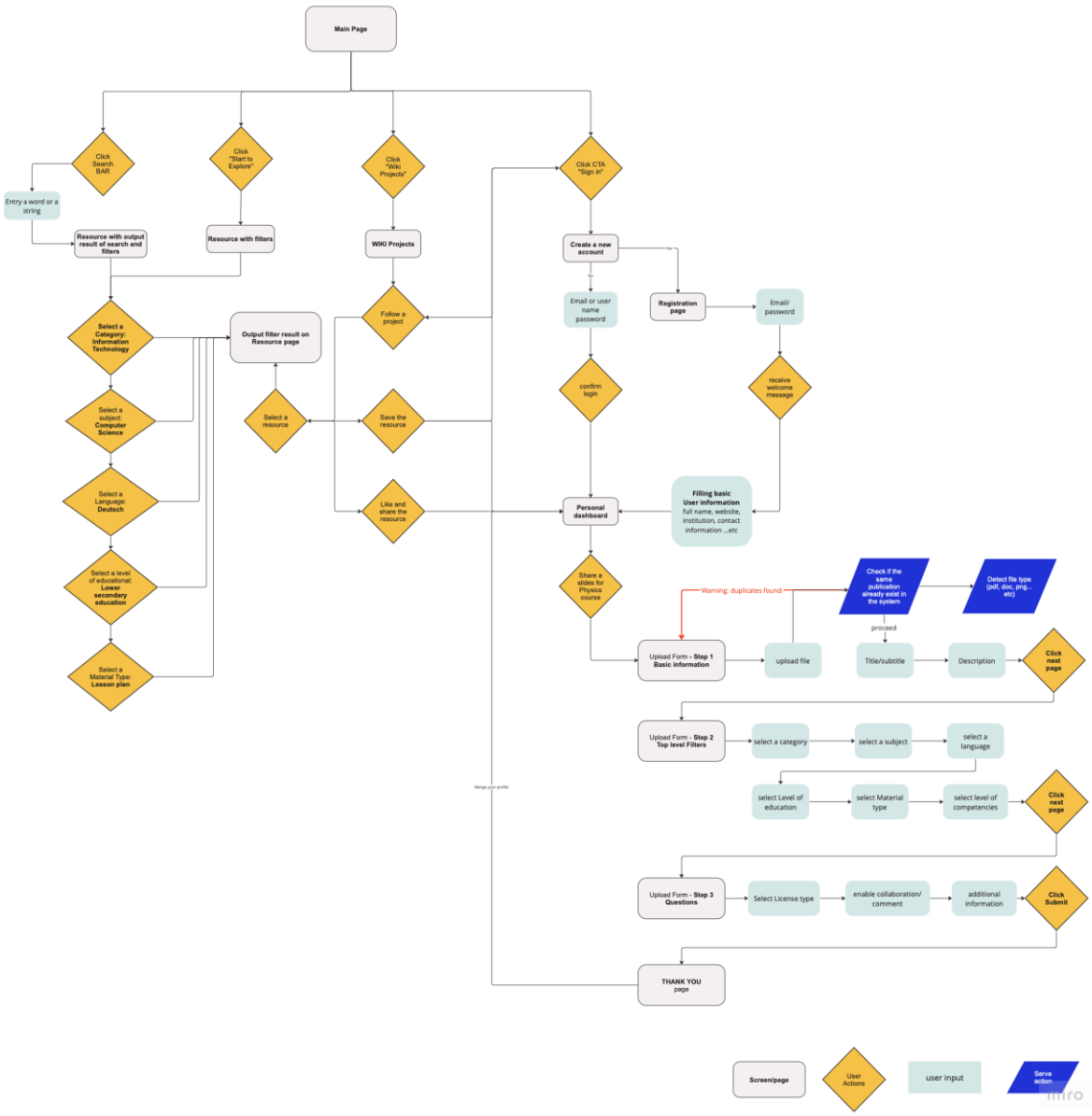
At this stage, we compiled potential solutions and we selected the most reasonable ideas to develop. Here are the decisions:

Features	Actions	Impact	Decision
Search bar	1.Enlarge and ceterned this feature in the landing page	Visible search UI especially on website with heavy information can help user to	Approved

	2.Implement autocomplete function for key word search	find content easily	
Filters	1.Rename & remove unclear or confuse filters 2.Use dropdown function on the filters 3.Use vertical filters section help user to prioritize tasks they wish to do on the site	Eliminate uncertainty and confusion that can deter users motivation to use the website	Approved
Uploading Form	1. Reduce user effort in the input fields 2. Simplify uploading process with an indication and procedure	Reduce the obstacle and potential drop out during the process	Approved
Website Landing page	1.Reduce information overload 2. Create **a clear message**: how people who you are, what do you do and what offers they can expect when visiting the website 3. Highlight **Call-to-Action** to encourage users engagement	Building trust and connection with users	Approved

- User flow

After we defined the key features, I designed the user flow of the search and file uploading process.



3. Results

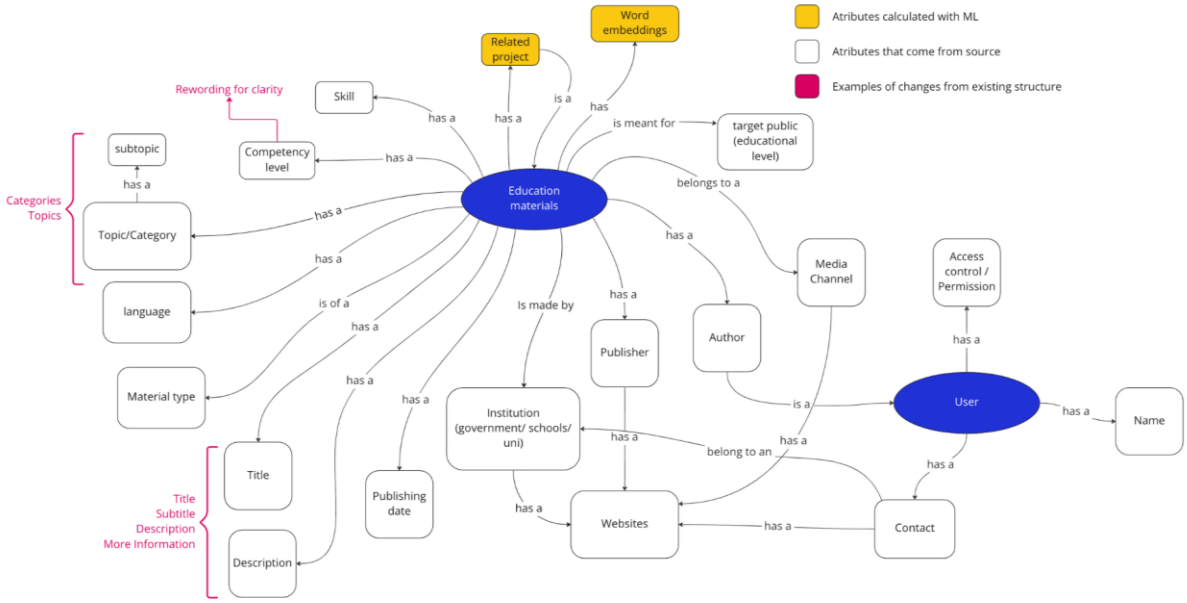
3.1. Ontology and data modeling

We decided to take as a starting point the existing structure. However, we proposed some modifications we considered are required for achieving a better website.

- Ontology

In the following image our proposal for ontology is presented. We arrived at this ontology, by finding the common attributes educational material have. We found confusing the difference between topics and categories. We recommend following a category/subcategory schema. After some research, we came up with the Australian Standard Classification of Education (https://heimshelp.dese.gov.au/resources/field-of-education-types#Section1), which can be considered for categorizing new educational material.

Further, other objects were considered as we started the implementation of some new features for the web page. For example, as is going to be described further in this document, we proposed the search engine to run in a semantic way, for which the word embeddings of the project are required. To save time and not repeatedly calculate the embeddings of each project anytime a new search is carried out, the embeddings must be saved and thus are a characteristic of the educational material. For the case of the related projects, the same applies, which must be recalculated each time a new project is added into the DB or with a predefined periodicity.



■ Data pipeline

In the following figure, we describe the possible data pipeline and the contact of the user with the data. We identify there are 3 types of users, and their connection to the data is explained as follows.

1. Client: these are end users of OPENEDU, which are only allowed to search for new data, and thus, they only have read permission. If wanted they can also have a login, in case reviews or comments are features added to OPENEDU.
2. Uploader: are users interested in posting their projects on the OPENEDU website. This is done through an uploading form. However, their entries should be reviewed by an Admin before accepted and officially saved on the DB.
3. Admin: are administrators of the webpage, incharge of reviewing new entries to OPENEDU, before those are officially published.

We suggest that the review process is done in a semiautomated manner, where some code is written for checking the information uploaded. However, the final check must be done by an administrator of OPENEDU, who will accept or reject the material. Some of the semi automated functions that can be considered are:

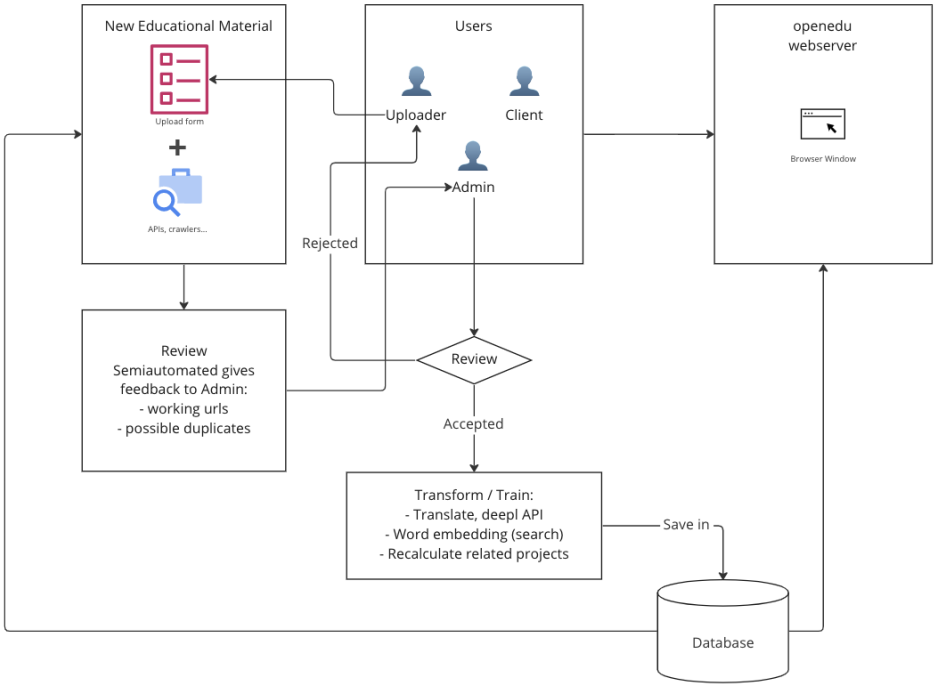
1. Duplicate warnings: Check in the DB for possible duplicates and label as a possible duplicate for the Admin to verify. If it is indeed a duplicate the Admin can suggest

the user to edit the project information in case the information on OPENEDU is outdated.

2. Working urls: if links to existing web pages are given, the functionality of those links should be checked, do the webpages actually exist.
3. Spelling checks: checks of the description submitted can be achieved using grammar APIs.

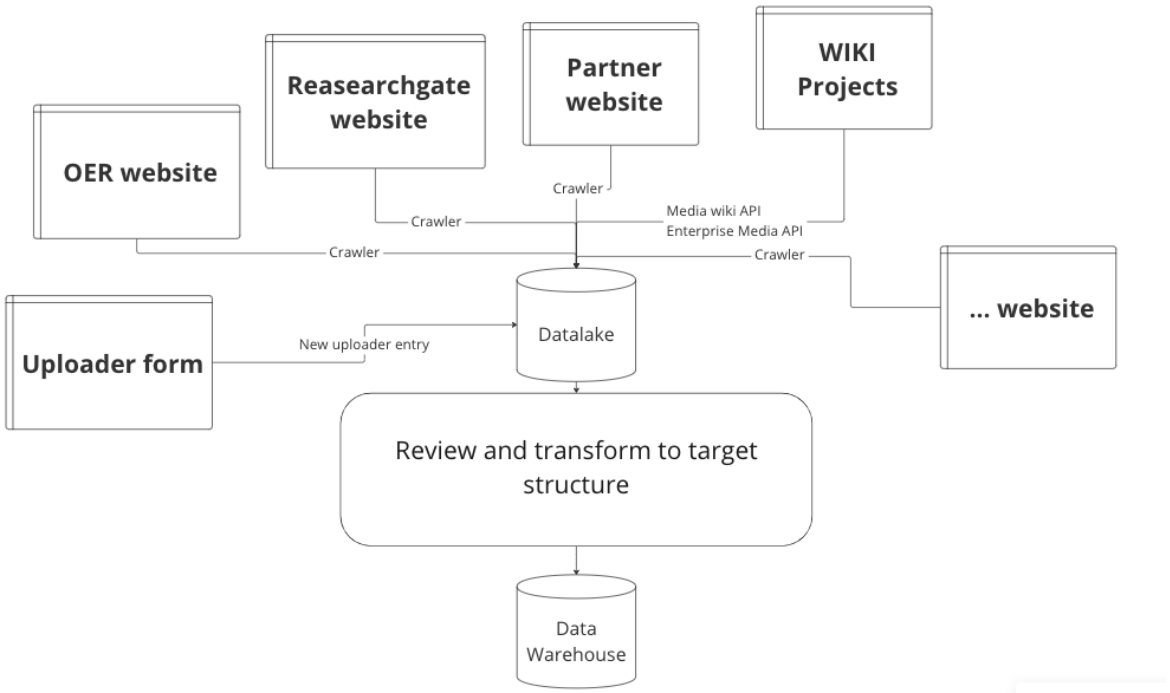
Once the administrator accepts a given material other automated processes need to take place to save the final entry on the DB.

1. The word embeddings of the projects should be calculated and saved on the corresponding table, for facilitating and accelerating the search
2. The related projects assigned to each of the educational material entries should be recalculated, given that the new material can change the similarity space
3. If wanted the content can be translated to other languages, for example using the deepl API.



Database structure

We propose a data schema where a data lake is created, which is the home for any new data crawl from the internet or data uploaded by the users before review. Once reviewed and accepted the data can be saved in the OPENEDU DB and saved in the final database (DB). An overview to the data schema we are proposed is given in the following image:



Further, the structure of the final database where the data display on the openedu webpage is based on the ontology described above and the data pipeline considered. We have the educational material related to different attributes and also users with different permissions that can be related to the educational material in different ways:

- As uploaders/authors: They might or not be the same person.
- As reviewers: this should be someone from the Administrators, the person who accepted the entry
- As a commenter. Any other user that is allowed to comment on an existing educational material

The final database structure is summarized in the schema below. There are some tables not directly related to the upload of a new entry of the educational material. This means, they are independent of the input of the uploader or crawler considered for fetching the data. These tables are created for functionality reasons, although they are related to an existing educational material, they are calculated automatically or will be updated and keep growing as long as OPENEDU exists. These tables are explained in more detail in the following.

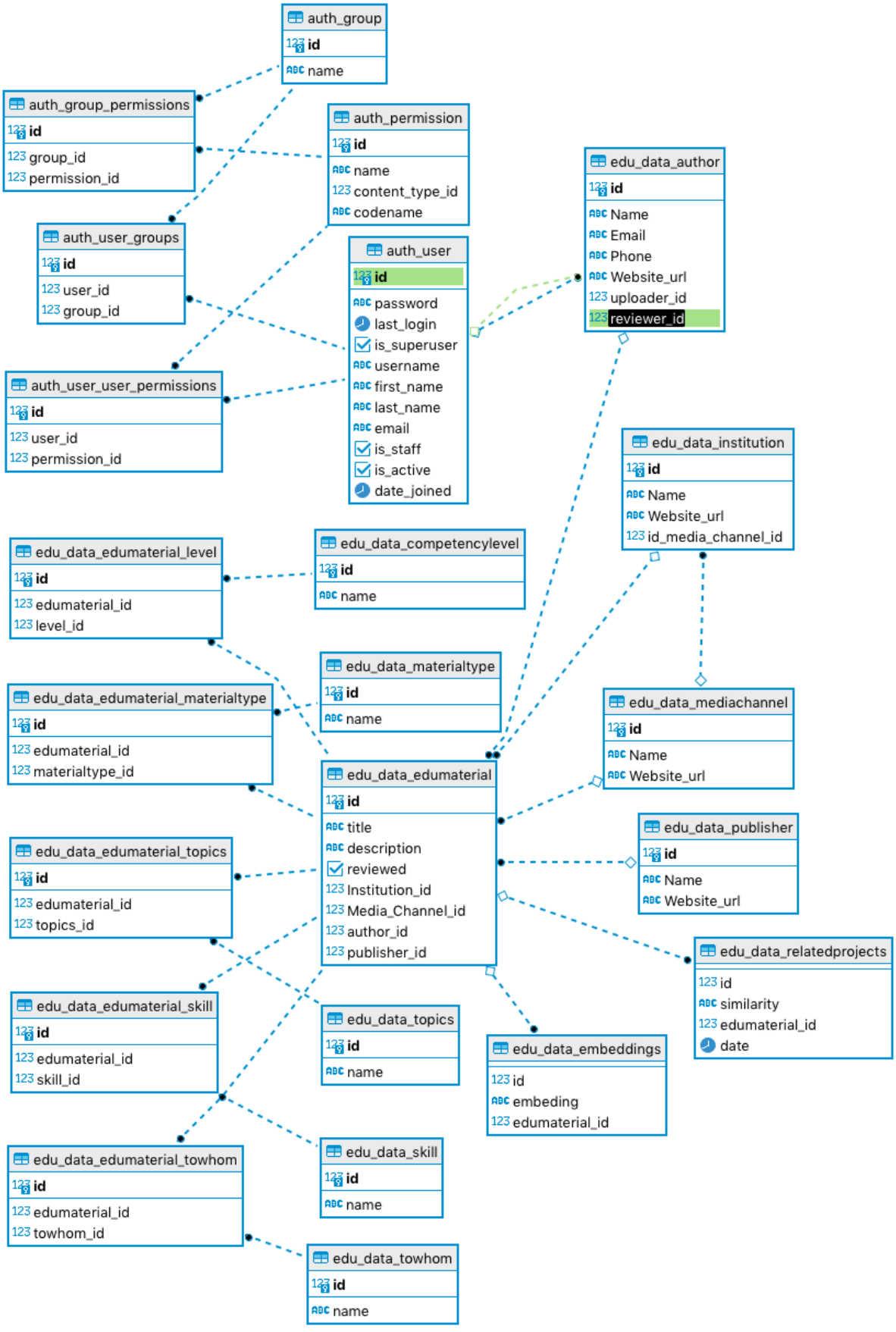
Table embeddings:
This table keeps track of the project embeddings. It is used for doing the semantic search we proposed in the NLP chapter.

Table related projects:
Keeps records of similar projects. As identified using NLP (see Using Data Science to improve website functionality)

This model is created to save the similarities between the projects. A JSONField is chosen given the many to many relationships. A new JSON entry will be generated when new material is added to the DB, or in a given periodicity. Keep in mind that as long as the related projects are not updated, the results won't be considered the new entries. Depending on the required time to calculate the similarity between projects, it can be decided if the update occurs right away or in a given window. If the performance is good and the calculation time is short, the similarity can be calculated after each project upload. Otherwise, this calculation should be done on a daily basis or a weekly basis.

Table comments (optional):

If users are allowed to interact and leave comments on the educational material. This should be included in the DB.


auth_group
id
name
auth_group_permissions
id
group_id
permission_id
auth_permission
id
name
content_type_id
codename
edu_data_author
id
Name
Email
Phone
Website_url
uploader_id
reviewer_id
auth_user_groups
id
user_id
group_id
auth_user
id
password
last_login
is_superuser
username
first_name
last_name
email
is_staff
is_active
date_joined
auth_user_user_permissions
id
user_id
permission_id
edu_data_institution
id
Name
Website_url
id_media_channel_id
edu_data_edumaterial_level
id
edumaterial_id
level_id
edu_data_competencylevel
id
name
edu_data_materialtype
id
name
edu_data_edumaterial_materialtype
id
edumaterial_id
materialtype_id
edu_data_mediachannel
id
Name
Website_url
edu_data_edumaterial
id
title
description
reviewed
Institution_id
Media_Channel_id
author_id
publisher_id
edu_data_publisher
id
Name
Website_url
edu_data_edumaterial_topics
id
edumaterial_id
topics_id
edu_data_relatedprojects
id
similarity
edumaterial_id
date
edu_data_edumaterial_skill
id
edumaterial_id
skill_id
edu_data_topics
id
name
edu_data_embeddings
id
embeding
edumaterial_id
edu_data_edumaterial_towhom
id
edumaterial_id
towhom_id
edu_data_skill
id
name
edu_data_towhom
id
name

3.2. Django and Dummy UI

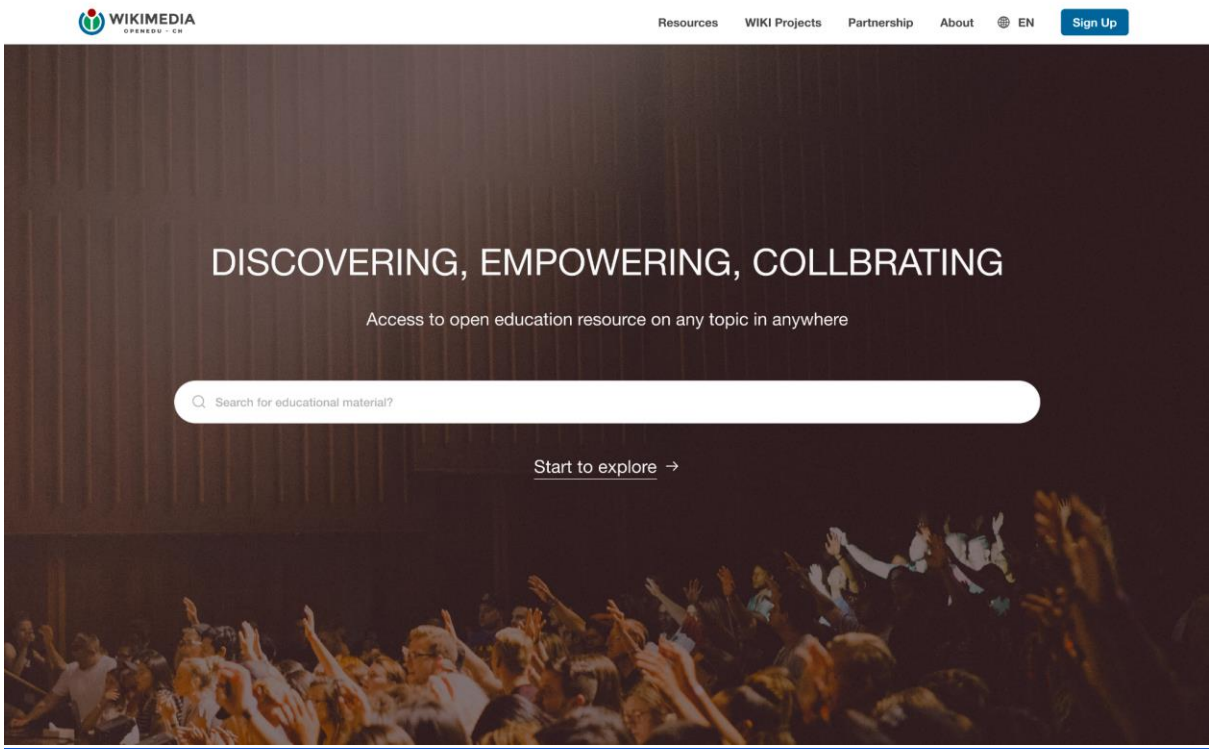
For testing the database, the search, the filters and the related projects we programmed a dummy UI to be able to test our results. We used django to create the database structure and for connecting the backend and front end. However, this Dummy UI was just programmed for testing the functionality of our developments, but does not represent the actual visual design we are proposing. Our recommended design is given separately.

The code for running the Dummy UI can be found in the repository of this project. in the django_openedu folder. Information on how the Django project was created is given in a separate document.

3.3. UX | UI: Visual Prototype

Many users expect what they believe will interact with the product and service and therefore, making the website similar to their competitor can help users to easily navigate on the site without hesitating. In the design phase, we implemented the Mental model and design principle to reduce the cognitive loading on the web page and create pleasant experiences for the user, as well as help them to easily navigate on the website.

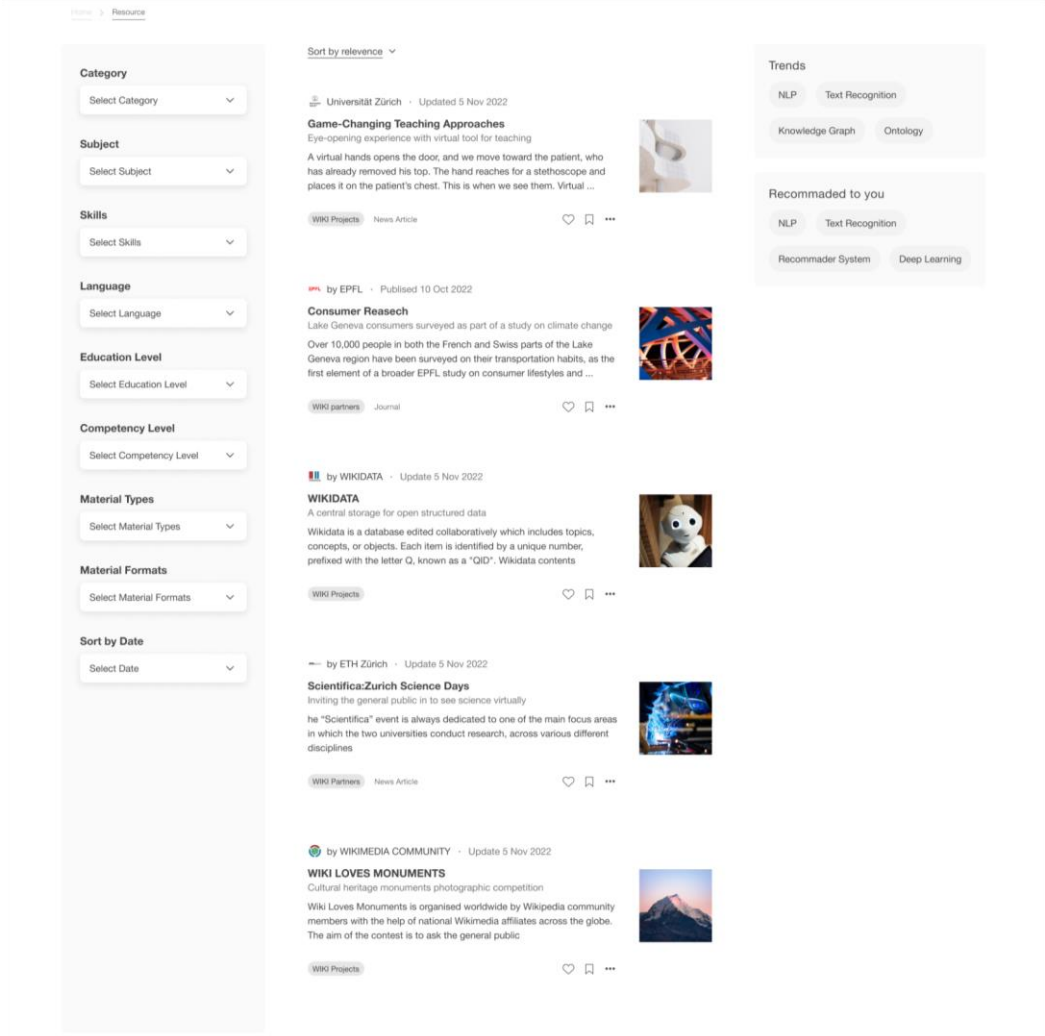
- Redesigning landing page



  - Core message: a short and power slogan tell users Who we are, what we do and what service and products offers on this site
  - Enlarge Search UI: make the search bar visible
  - CTA button: invite people to sign up and explore the site

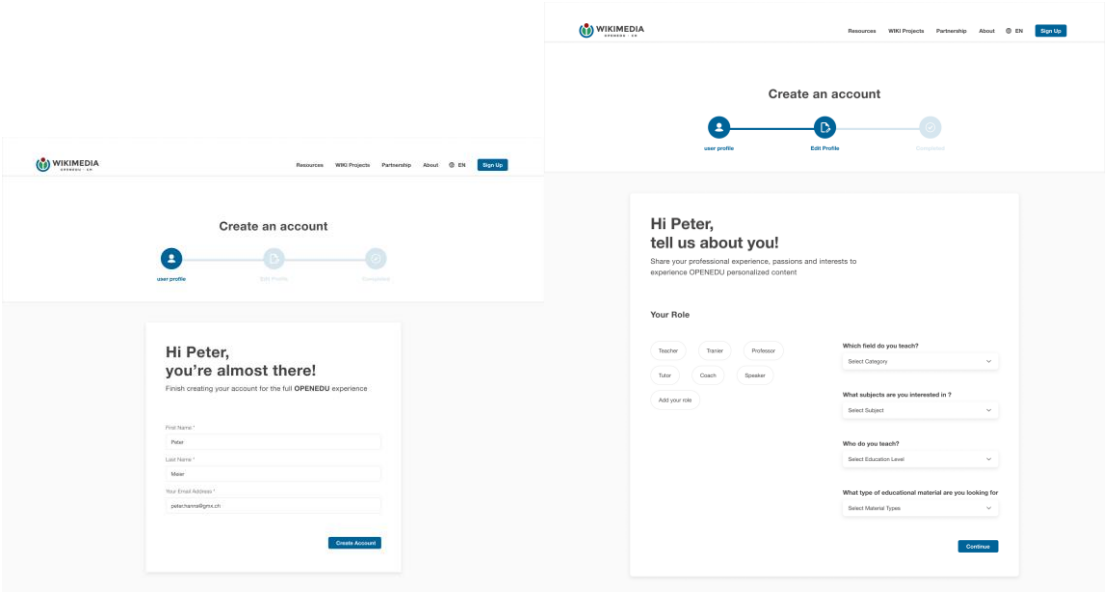
- Clear navigation: make the navigation visible to navigate users to visit different pages

- Resource page: Use Dropdown animation in the filter feature



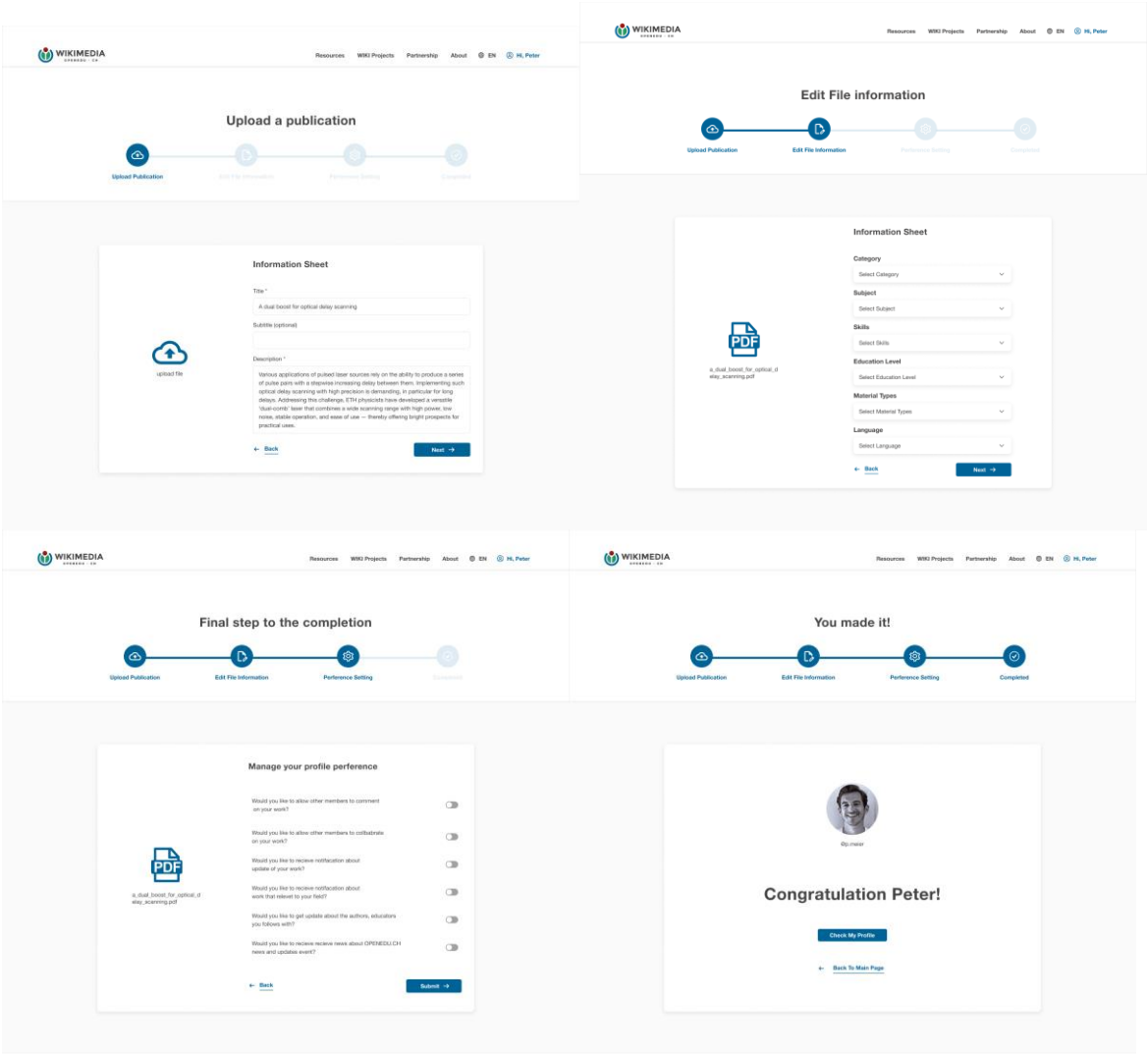
- Sign up pages

In the sign up page, we want to collect more user basic information, not just their name and email. However, to make the user feel comfortable and share their information with us, we implemented some tags and a drop down menu to help users to fill out their information in a simple way.



- Uploading Pages

We make the uploading form simpler and divide the process into four steps, to make the user informed and know which section they are and be able to estimate how much time they need to complete this task.



- User Upload

We suggest having on the one hand a user registration form, where basic information about the user should be saved on the DB. This will reduce the amount of information users need to give each time they are uploading a new educational material.
Further, the upload formula should suggest possible categories and entries as the user is making new entries.

3.4. Natural language processing (NLP)

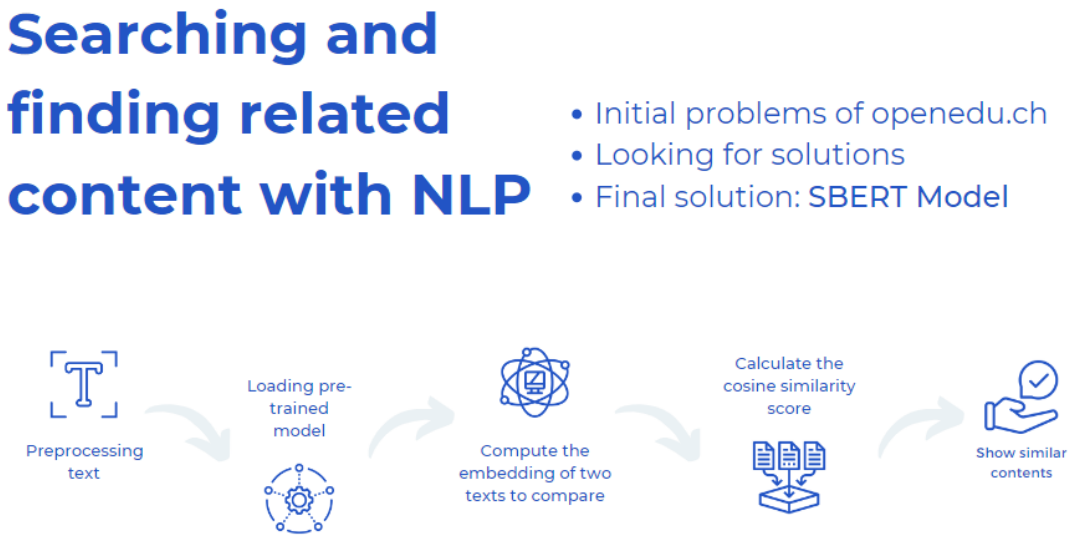
We asked ourselves in the beginning of the project how we can use NLP techniques to improve the website. We found two features that we really wanted to improve:

- suggesting related content automatically
- semantic search tool

- Finding related content by semantic textual similarity (STS) analysis

The first feature is the related content. The website shows related content to the user. We liked this feature but on the OPENEDUwebsite the related projects are manually assigned. As the content of the website will grow, this will not be feasible anymore. So we automated this process by semantically comparing the descriptions of the projects.

Method



1. *Pre-processing:* Before we can give the texts to the NLP model we need to pre-process the text.
   - concatenating title, subtitle, description and more information (without html-tags).
   - removing any digits, special characters and symbols,
   - lowercasing all characters
   - removing all the stop words (e.g. “a”, “the”, “of”)

2. *Loading the pretrained SBERT model*: We use the SentenceTransformer model ‘stsb-mpnet-base-v2’ and the bi-encoder (see https://www.sbert.net/ for more information).

3. *Calculation of the embeddings:* Next the SBERT-model and the pre-processed text are used to calculate the embedding vectors of all the projects. Now we have for each project an array consisting of 768 numbers that hold the meaning of the text.

   Note: We also save the embeddings in the database so we can use them without having to recalculate them for the semantic search tool

4. *Calculation of the similarity scores:* We use the cosine similarity function to calculate the similarity score of each project-project combination. The score is a value

between 0 and 1 indicating how similar the two project descriptions were. The result of this step is a similarity matrix of size n times n. n is the number of projects in the database

5. *Finding the three highest scores:* The last step is to go through every project and find the three highest scores. This gives us the related content that is shown on the website.

This calculation of the related content (step 1 to 5) is done if:

- the project description in the database is modified or
- a new project is added

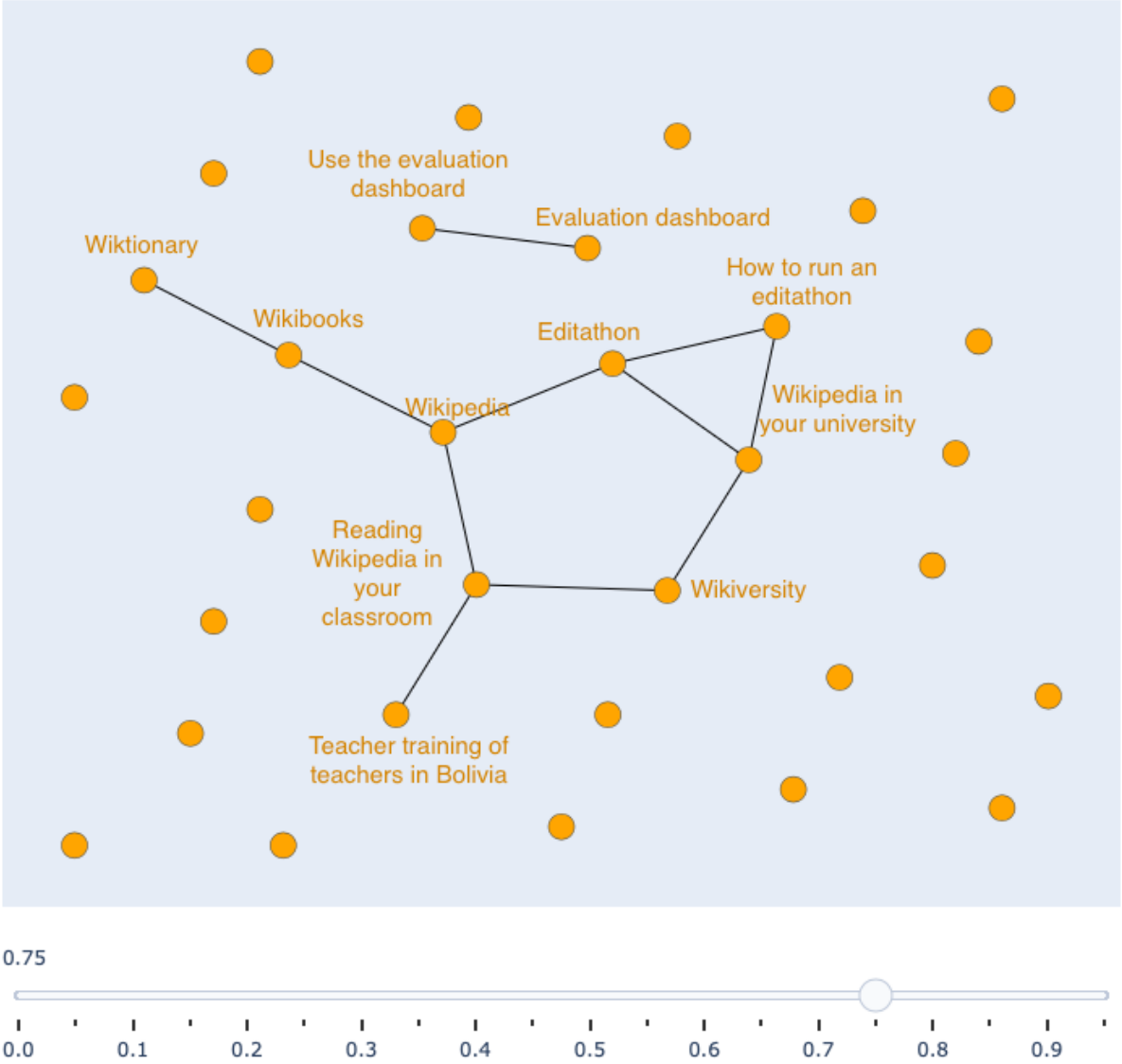
Tests and results

The calculation of related content using the SBERT model works very well. We initially tried three different methods to calculate the similarity scores. These were the:

- the TF-IDF model described in the NLP for openedu workshop
- SBERT model using the bi-encoder
- SBERT model using the coss-encoder

The SBERT model using the bi-encoder gave the best results in a reasonable computation time. The cross-encoder took too long and the TF-IDF model was not able to fully catch the semantic meaning of the texts.

The similarity matrix can be plotted graphically by combining related content with a line. In the graph all the projects with a relative similarity score higher than 0.75 are connected with a line.



■ Implementing a semantic search tool

We have implemented a function using an NLP method, that is SBERT semantic model, to compare the semantic similarity between the search key and the projects, which are already stored in the DB.

The BERT Model uses a neural network to convert texts to embeddings. An “embedding” vector is a numeric representation of our natural language texts so that our computers can understand the context and meaning of our text. We construct these embeddings so that semantically similar texts cluster nearer to each other while dissimilar texts are further apart. After loading a pre-trained model (pre-trained with this approach), we can find the similarity between two texts by mathematically computing the cosine similarity between their vectors.

On the basis of this similarity score, we can easily get all the projects, which are related/similar to our search.

Finally this function has been integrated in the dummy UI.

Advantages of BERT Model:

- it allows you to work with an already pre-trained model on a large amount of data, which was not possible for us.
- it allows to recognize the similarity between texts in a semantic way, transforming the text into embeddings

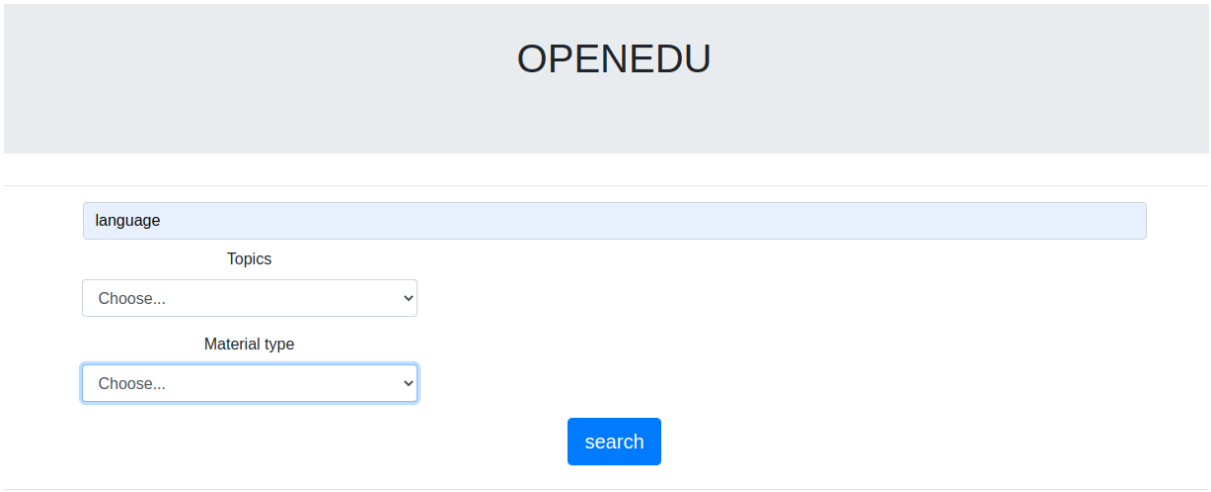
Tests and Results

To set the correct threshold we performed tests with 18 different search-keys. 9 search-keys contain words that occur in the database description: *science competition, language, encyclopedia, database, travel, school, medicine, university, research.* The other 9 search-keys do not occur in the description and should not yield any results. These 9 words are: *jungle, kitchen, rain, elephant, grass, cars, sun, children, chair.*

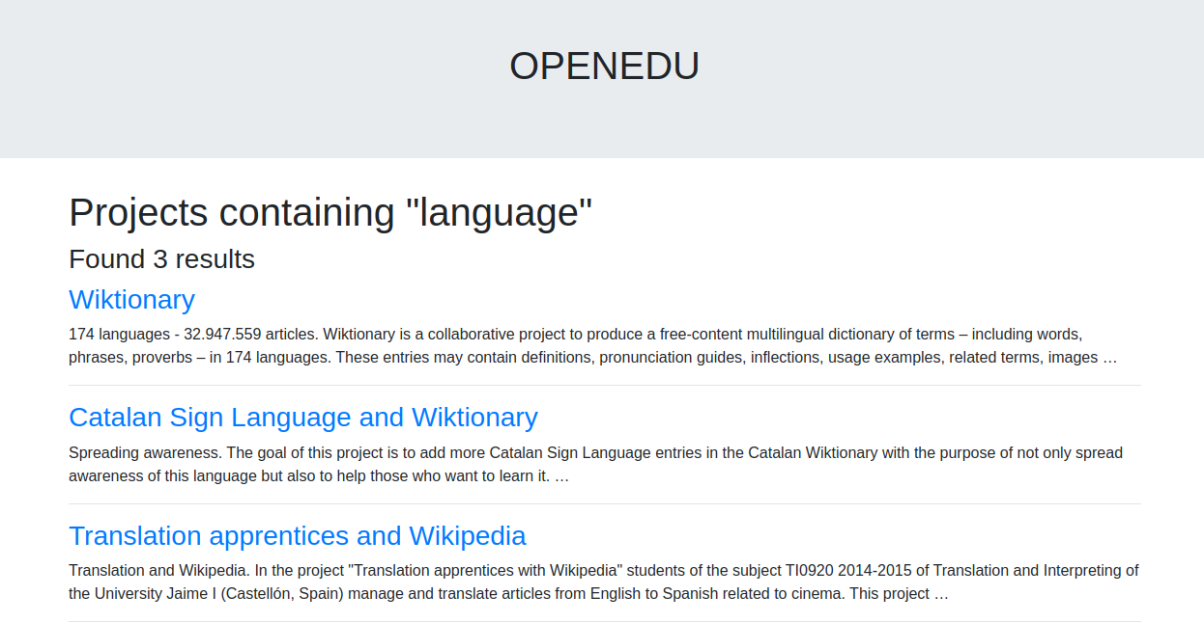
We found that when setting the threshold to 0.28 we get the best results. On one hand, the search-keys that don't occur in the database do not yield any result. And on the other hand, the search-keys that do occur in the database do yield almost all the relevant results. The full test can be found on github in the folder NLP.

One example of the semantic search function integrated in the dummy UI will be displayed below.

The search key is "language":



The results of the search with our semantic search function are the following:



The three projects are clearly related to the search key. Interesting is the fact that, even if the third article does not contain the search key 'language', semantically the tool was able to recognize its similarity.

4. Conclusion and next steps

4.1. General achievements

We came up with an ontology, which was refined all the way through the project as we saw it was needed to. The ontology was tested while creating a Dummy UI in Django with the objective of evaluating the functionality of various features we suggested for improving OPENEDU.

In this project we had the opportunity to improve the search engine of OPENEDU by means of a pretrained model (SBERT), which successfully allowed us to search for keywords and retrieve projects associated in a semantic way to this word. Using this pretrained model, has the advantage that the dataset used is larger than the one we had. Additionally, we proposed a better solution for defining what is to be shown in the OpenEdu website as similar projects, it is defined through a semantic similarity analysis of the project's description. This prevents similar projects from being selected according to the uploader's subjectivity, which would be a biased result.

Further, we developed a concept to improve the user journey through the website. Our UX|UI is based on clear instructions and guidance to help the user to complete their tasks. It includes the possibility for users to get in touch if wanted and create a network with other OPENEDU users.

4.2. Outlook and further challenges

As further implementations for the OPENEDU project we recommend the following features:

- Increase the current educational data retrieved in the OPENEDU website. Data collection by connecting to API and implementing web crawling.
- Deepl API for translating existing content
- Improving the uploading procedure by:
    - Creating a function that suggests categories, skills, competency levels, etc. By means of a similarity approach, maybe the same is used for finding the related projects proposed in this document, some of the uploader fields can be suggested, taking those from similar projects. Here the uploader should also review and add further categories.
    - Allowing automated abstract generation from uploading documents. If a video of a lecture is included an API can be used for looking at the subtitles (ex. youtubeAPI). These subtitles can be saved as a txt document, which can be the entry to a function generating the abstracts. However, summarization algorithms tend to be imprecise, thus, if implemented it should be added as a suggestion. The uploader should verify the recommended abstract and edit as considered.
- Implementation of a help for the reviewer where for example the following functions are considered:
    - Duplicate warnings: Check in the DB for possible duplicates and label as a possible duplicate for the Admin to verify. If it is indeed a duplicate the Admin can suggest the user to edit the project information in case the information on OPENEDU is outdated.
    - Working urls: if links to existing web pages are given, the functionality of those links should be checked, do the webpages actually exist.
    - Spelling checks: checks of the description submitted can be achieved using grammar APIs.
- Give the users the possibility to suggest a merge if they found out that there are duplicated projects. This should, however, be reviewed by an Admin. To ensure that real duplicates are deleted
- Taking into account a growing trend and need for collaboration, exchanging experiences and knowledge, participating actively in some forums we recommend connecting users to each other through OPENEDU which will make it possible for peer 2 peer learning or networking.

Further, we want to mention some challenges we faced and could be faced while the implementation of our proposal

■ Challenges concerning the ontology

The biggest challenge concerning the ontology and the data structure was on understanding the scope of the project and defining what our goals and objectives were. We decided to take as a starting point the existing data structure and migrate the existing data with our suggested modifications into a new DB, with our proposed structure. However, given that

we recommended new attributes, the relation between the different educational material entries and the had new attributes.

We recommend reviewing the ontology and the DB as more educational material is added to OPENEDU. For example, we assumed that english is the only language considered for OPENEDU. Thus, if new languages are implemented, the DB must be updated to include them. Further, if a users network is allowed as a feature to OPENEDU, the DB must contain a table to keep track of the user comments and be related to the users table by their id.

- Challenges concerning the UX|UI

The purpose for this project is to provide tangible and actionable solutions for the client, although the redesign website was not required by the people, we believe having a website prototype that can visualize our solution in a feasible way is more convincing when pitching our idea, There were some flaws in the process and improvement that need to be addressed accordingly based on the data collection in the future development.

- Challenges concerning the semantic search

We implemented the two NLP features limiting ourselves to English. Since Swiss users will most likely search in German, French or Italian, the implementation has to be expanded to these languages. In addition, the website might have to automatically detect the language of the search key. We are aware of search cases where the algorithm is not listing the project even though the project description contains the search-key. For example the projects “Wiktionary” and “Wikibooks” contain the word languages but are not getting listed by the algorithm when searching with the search-key “language”. This is because the similarity score of the comparisons are lower than the threshold. A possible solution can be to combine the semantic search with a traditional search query in the database. Also a search query should take no longer than 1 second. Right now a search query takes approx. 8 seconds because the code is running locally and a connection has to be established to the database on the azure server. We believe once the python script of search and the database are both located on the server the execution time will be less than a second since most embeddings are pre computed.

4.3. Our learning journey

Our individual & collective learning experience

Participating in the project was very meaningful for us on many different levels, among which are:

- We received many valuable lessons on **project management** - in terms of the process, tools, dynamics of the project, collaboration on the project, roadmap, stakeholders etc
- We learnt and practiced working in an **agile** way, using **scrum**, we did all scrum ceremonies, we practiced backlog management and learnt useful tools
- We had a chance to experience a **real remote team work**, not only in a **cross-functional**, but also **international** team, including different time zones
- We received a great lesson on UI/UX including creating personas, customer journey, user flow, user interviews etc
- We also extended our technical knowledge on databases, data structure, NLP, Azure, Python, django etc.
- We had a chance to learn and effectively use different tools: GitHub, Slack, Miro, Google Workspace etc.
- We gained a lifelong lesson that **open communication and building trust** among the team members is crucial in a successful product delivery
- We are finishing this project convinced that whatever project we do in the future - we should always **keep our end-user's needs and point of view in mind** and there will never be enough questions we can ask our (potential) users.