Ilumni: Study Assistant

Fedor Moiseev Artem Lukoianov Davit Martirosyan

Initial idea

Choosing the right project to work on was really tough. After careful consideration of possible data visualization projects that we could do based on standard datasets, we decided to take a different route by collecting our own data and building a website that would help **EPFL master students** visualize all information about their studies in one place and also plan their future studies.

The motivation for this project came from our own experience of finding it difficult to quickly see the useful information within our grade transcripts:

1. There is no way to compute the GPA other than manually, and each student does that when updating their CV.
2. Grade distribution at EPFL is very different compared to other universities and thus it is much better to put percentiles in CV (top 5% of students sounds much better than 5.5/6 GPA). This information is not available at all.
3. It is not easy to track all the requirements of your program.
4. Because there isn't any readily available course statistics over the years, it is very hard to get a sense of how easy/difficult the course is and where a student ranks based on their grade, etc.

Inspired by our project idea, we framed the logics (e.g. the flow of events) of our website and created our first sketches of visualizations which are presented in Figure 1 shown below. . Entering our website a student is prompted to upload their grade transcript available on IS-Academia to our website which we then parse to get the relevant data. Afterwards, the student is navigated to the main page of our website (see the left sketch of Figure 1), which is basically a flower-like plot, where petals represent the classes completed by the students and the circle at the core of the plot contains general information such as the cumulative GPA, the

number of completed credits, etc. The length of each petal is proportional to the grade obtained by the student and the width of each petal is proportional to the number of credits of the given course. Note that the total number of credits required to complete the program forms a full but bumpy circle unless the student has got 6 on all his/her courses. Besides the flower-like plot, we also thought to have a part on the main page where students could see how many of their program requirements (e.g. semester project, internship, etc.) have been fulfilled.

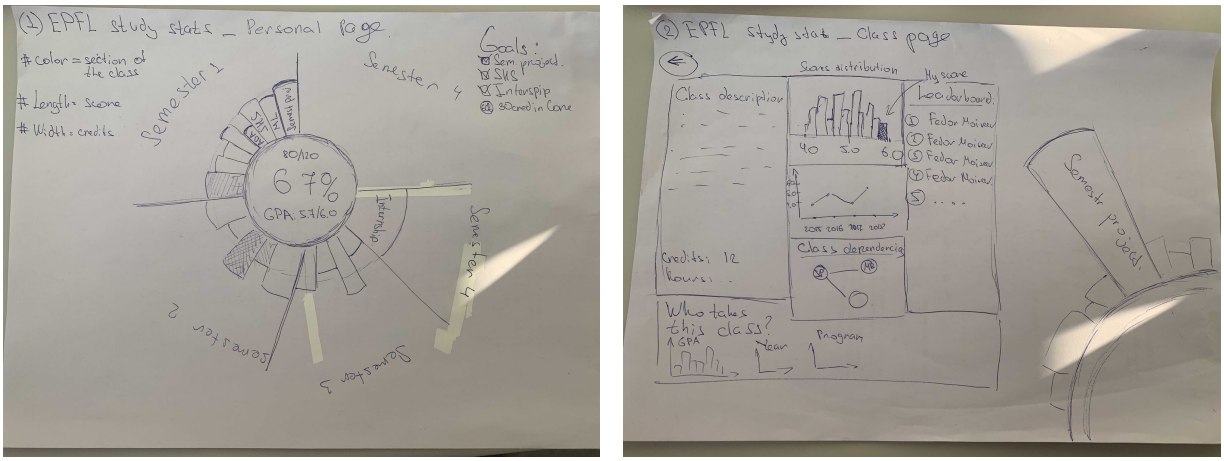


Figure 1: First sketches of visualizations. Classical pen and paper are the most convenient tools for us when it is needed to be creative =)

According to our initial sketch, after clicking on a petal the student is navigated to another page (called “Class page”, see the right sketch) where they are presented with a number of details about the corresponding course such as its description, the mean average cumulative GPA of students taking that course over the years, a graph showing which courses are prerequisites for that course and which courses can be taken afterwards, and much more. While some of the visualizations in the “Class page” that we planned to make could be done based on the information publicly available on EPFL’s website, the ones related to showing grade distributions required the info of EPFL students’ transcripts. To overcome this problem, we initially thought to ask a number of EPFL students to share their transcripts to gain the necessary statistics, however due to the privacy issues involved our mentor TA suggested generating “placeholder” data and using that instead.

Another problem that we faced was connected with the graph plot showing course dependencies. We wanted to scrape the necessary information from EPFL’s website, however,

the problem was that for some courses the prerequisite courses were available, while for some others they were absent or written in a way almost impossible to parse any useful information from. Thus, we made the decision to abandon this idea.

Initial idea refinement

After discovering the problems described above, we decided to think on new visualizations and features to add to our website. We came up with the following ideas:

Skills plot: Almost every masters course at EPFL belongs to several programs at once, hence completing a masters course contributes to one's skills in several directions (e.g. Machine learning contributes to the skills necessary in Data Science, Computer Science, Cybersecurity etc). Having this idea, we decided to visualize the skills of each student based on their completed courses in the form of a spider chart. Since depending on the master program, a course can be core or optional, we decided to add 2 plots on the same graph: one based on all courses (coloured in green) and one based only on the completed core courses (coloured in red). A preliminary version of the plot we had is shown in Figure 2.

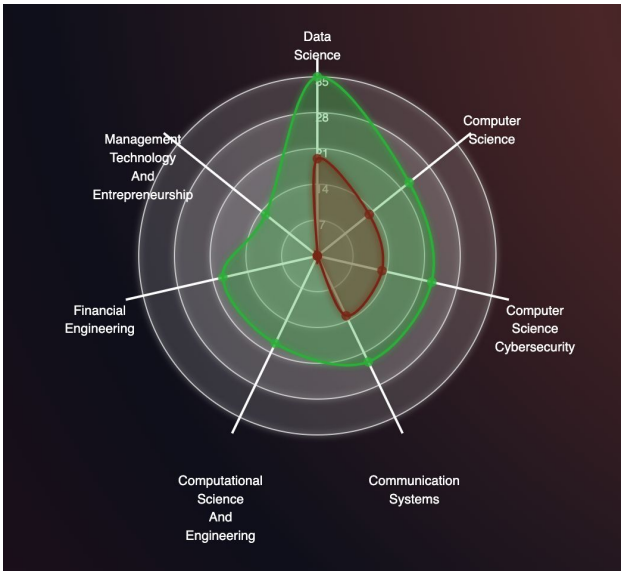


Figure 2: Skills plot

Semester planning: Since we also wanted to introduce more useful interactivity to our website, we made a decision to create a feature which would allow users to plan their studies for the next semesters. To achieve this, we decided to add a search bar for all courses, which would allow users to search for a particular course, see if it is core or optional for his/her programm and add it to the flower-like plot. On top of this, we wanted to create a button that would automatically add courses to fulfill all the requirements for the given master program.

Technical implementation: D3.js era

After deciding on all the details, we started building a prototype of our website. First of all, we implemented a pdf parser to get the necessary data from the transcript that a user uploads to our website to see the visualizations. After accomplishing this and testing on our transcripts to make sure our pdf parser works properly, we implemented some basic version of the flower-like plot. Initially, it did not contain any animations, however, we step-by-step enhanced the plot achieving more complete and beautiful versions. The latest version is shown in Figure 3.

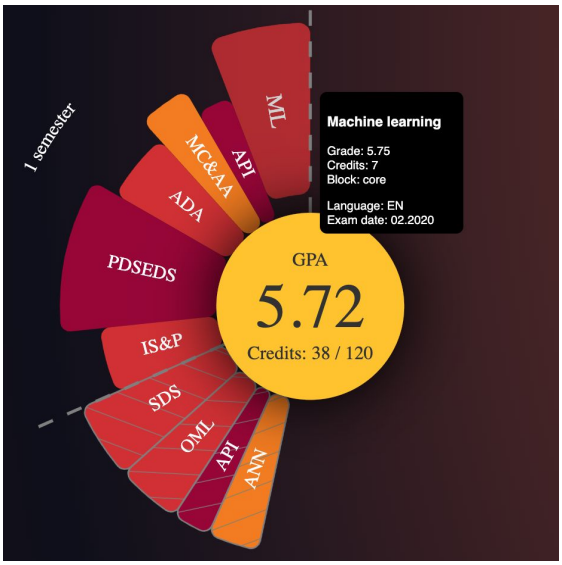
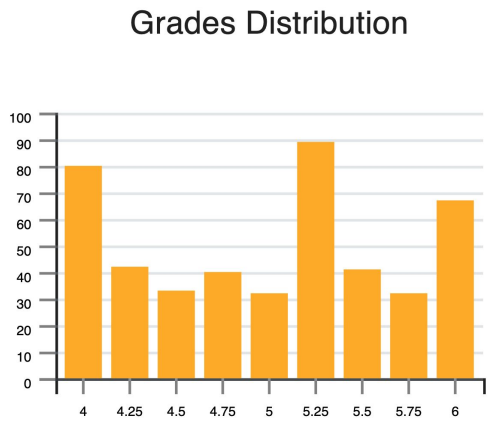


Figure 3: Flower-like plot

The library D3.js was really helpful in bringing our ideas to life. For example, the arc generator function allowed us to create the desired petal geometry in just a few lines. D3.js also helped us enormously in making the spider/radar chart for skills visualization. Our first version of the skills plot is based on the example from bl.ocks.org for which we would like to thank Matthieu Viry - the author of the plot used.



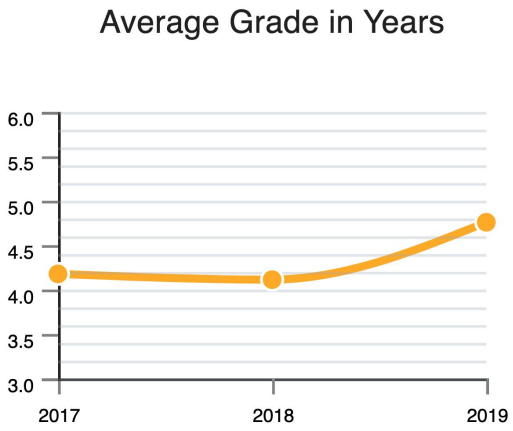


Figure 4: Grades distribution

Figure 5: Average grade in years

In addition, we extensively used D3.js to create an interactive histogram and line chart plots which are drawn once the user clicks on a petal to see the relevant course information. Both of the plots were made on fake data as discussed previously. The histogram (see Figure 4) shows the grade distribution of last year's students taking the course and the line chart plot (see Figure 5) illustrates how the mean average GPA of students taking the course has changed over the last 3 years.

After the first attempts, we quickly realized that we needed a tool that would allow us to synchronize dependencies, create the production build and deploy it on the Internet. To achieve it, we integrated webpack into our project to manage dependencies and used Netlify to host obtained builds online. Continuous Integration available on Netlify allows auto deployment of a new version just on a commit to the master branch!

Although our visualizations worked well in general, as we were moving forward, we wanted to make our website UI more interactive and build encapsulated components that could manage their own states. Firstly, we tried to do this with D3.js, but we faced a few problems:

1. The elements of the flower-like plot should interact with other elements on the page (e.g. the course info) and their animations should be synchronized.
2. User can interact with the flower-like plot in several ways: he can click on the petal to see the relevant course information, move the mouse over the petal to increase the size of it and see the tooltip, etc., and animations for all these interactions should be consistent with other other non-d3 animations. Using D3.js one could only animate attributes of a tag, and it's hard to synchronize such animations. For example, during the rotation animation, the petal that mouse pointed to behaved differently from other petals.
3. D3.js is highly imperative and doesn't allow to work with distinct elements rather than selections of groups of elements. In our case, it doesn't allow to structure code properly pure D3 does not allow to make the code scalable.

Due to the reasons described above, we decided to change our framework from D3.js to React.

Technical implementation: React era

In order to switch to React as smoothly as possible, we continued using the great functionality offered by D3.js for scaling, arc generations, etc, and used React only for DOM interactions. We thought it to be a good solution for our case allowing us to take the best from both libraries.

React vastly helped us in making our website UI more interactive and structuring our code logics in a better way. Furthermore, it helped efficiently update and render just the right components when the data changed. For example, unlike previously with React we extracted each petal with the course name on top of it to a distinct React component and it allowed us to treat each petal independently.

For the animations we used react-spring library that allowed us to connect related animations in a simple way and automatically removed many bugs with inconsistent animations that we suffered from while using D3.js for DOM modification.

Once we were done with the flower-like plot, we went to enhance the “Class page”, i.e. the course info page that appears after clicking on the petal. At first, we tried to use pure HTML, however failing to get satisfying results, soon we realized that we needed some sort of solution that would allow us to organize the information about the course in a beautiful grid form. After careful exploration, we ended up with using the Material-UI library of React. With the help of it, we made an adaptive grid that looked consistent and nicely included all the necessary information, namely course name, professor names teaching the course, course section and course description (all of these were scraped from the EPFL website). Afterwards, we added the histogram and the line chart which were contained in independent svgs built by D3.js. Adding them to the grid while making the style of the plots consistent with the rest info page was a bit challenging, but we made it.

The last but not least feature that we wanted to implement initially was the semester planning feature. We used Material-UI to create a search bar widget that allows the user to search for all courses. We then made it possible for the user to add a particular course to the main flower-like plot by just clicking on it. The courses that are manually added have a colour with lower opacity to be distinguishable from the completed courses. The fact that we used React instead of D3.js

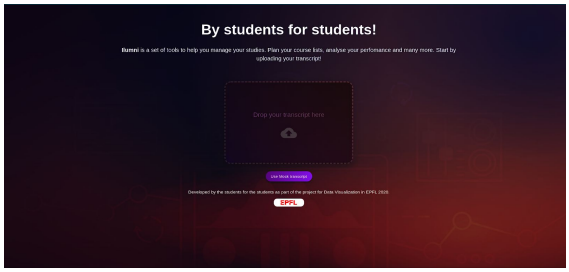
allowed us to introduce another set of petals with different styles and nature, but with the same behaviour in a short and simple way.

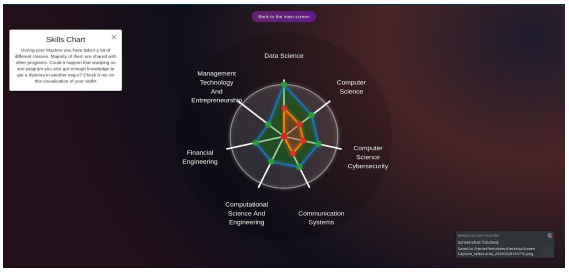
Assuming that manual addition of courses from the search bar can be boring sometimes, we also implemented a simple greedy algorithm that automatically selects future courses for the given user until all the program requirements are fulfilled. The greedy algorithm firstly adds semester project (if required), then core courses, then optional courses and finally the master thesis.

Website design

After the main set of features were implemented, we decided to put time and effort on designing our website. We chose Figma for this purpose which is a handy collaborative interface design tool. Firstly, we started with the design of the landing page, where the user is prompted to upload his/her transcript to see the corresponding visualizations. Afterwards, we moved on to the design elements of other pages. We used Figma along with the [Material-UI] which helped us enormously in choosing the color combinations to be used throughout the website and making our website design beautiful and coherent. Our design sketches produced with the Figma can be found here [link].

After many arguments and iterations, we agreed on the following design:





Limitations

We tried to make our solution as general as possible, but faced a few limitations. First of all, we only had transcripts from Data Science and Computer Science Master programs, so for now the full functionality of our website works only with these programs. Secondly, we tested our website only in Google Chrome (because, for example, Safari is only available on Mac) and can guarantee its proper functioning only in this browser. It should also work good in general in Safari but with small visual discrepancies. And finally, we had only recent transcripts, so if you will upload some older transcript, probably it will contain some courses that aren't in our database. Our website will still work properly, but you won't be able to see course descriptions.

Individual contributions

Artem Lukoianov - Created the initial/ version of the flower-like plot on the main page, pdf parsing of transcripts, made skills plot interactive and animated, implemented progress bars of the program requirements and hints system, deployed website on netlify.

Fedor Moiseev - Created the first version of the skills plot, made the flower plot animated and interactive, integrated webpack and React to the website, refactored a lot of code, implemented semester planning and course suggestion feature

Davit Martirosyan - Worked on data scraping, class info page along with histogram and line-chart plots on the class page, landing page, design with Figma and integration of the design to the website

Despite the roles described above, we worked as a whole team and everyone made a crucial contribution to the project. We believe that our contributions are equal.