ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Data Visualization (COM-480)

Process book :

Interactive Trip Planner

ZOUEIN Germain
BEN OTHMAN Firas
ELLEUCH Khalil

Introduction

This is a process book that covers the project of the fall semester of 2018 at the course of Visualization (COM-480) at EPFL.

The main objective is to guide the reader through the evolution of this trip planner from the very beginning to the implementation and final visualization.
As Ben Shneiderman said : *"The purpose of visualization is insight, not pictures."*

Overview

This project should enable a user to visually plan his/her train trip on a limited budget and be able to see the possible travel destinations with a reasonable journey time (we picked a maximum of 8 hours). It is aimed at any travelers who find themselves bored and looking for new destinations to visit, without emptying their wallet. The plan is, the final project should show a map of France with different cities as potential destinations, and the path to take to each one of them from a user-specified starting location. Originally, we wanted to work on the SBB datasets but found they were lacking (no available prices), so we looked for another European trains operator and found SCNF.

Motivation

What if you don't have the budget to go to the usual touristic destinations such as Paris, Amsterdam , London... ? Will you let that stop you from travelling and discovering new cities ?
France is full of historically and culturally rich cities. So why not use the small budget you have to travel by train to one of these cities from a given starting point.
In that case, our visualization will be very useful. The user is able to choose a departure station and a maximum budget on the train ticket and then look at different possible destinations summaries and touristic/historical attractions.

Target audience

Our target audience is avid backpackers, especially those with a restricted budget. A student who wants to spend a weekend far from his city would use our visualization. Since our dataset was restricted to SCNF trains, the target audience will have to be interested in travelling inside of France.

Inspiration

We have been captivated by a slide presented in the first course of Data Visualization presenting the Facebook network. When seeing this map, we wanted to work on something that shows a map with edges relating different nodes. Inspired by Sky Scanner, we first

wanted to make a trip planner on a budget with flights internationally instead of journeys with trains . That was not feasible due to the change in the flight prices depending on demand and time of the year and the unavailability of a dataset with average/approximate prices.

FIGURE 1 – Our inspiration : Facebook network

Concept

With previously stated inspiration and goal, we will present in this section the conceptual ideas we have come up with during the development process . We will present an overview of the dataset used in this project, how it was preprocessed and how the visualization was designed.

Datasets

As explained in the Inspiration section, our first idea was to make a trip planner with international flights on a limited budget to multiple international destinations.
Due to the infeasibility of the original idea, we decided to move on to trips by train which are more stable in terms of prices and planning.
First, we looked for datasets of train trips within Switzerland. Unfortunately, the available SBB datasets do not make the trip prices available. As a result, we looked for other European train datasets and we found that SNCF provide datasets of train timetables as well as their prices.
We took different datasets from SNCF open data. The datasets are as follows :

— For the **stations dataset** (station IDs/names/coordinates), we pre-process the dataset to obtain a cleaned dataset with the columns we care about and easy-to-use names (the original datasets are often in French and use some internal abbreviations).
Notebook: "preprocessing_1_stations.ipynb")
Dataset link : https://data.sncf.com/explore/dataset/liste-des-gares/table/

— For **TER and intercity datasets**, we pre-process the datasets to get well-separated and cleaned origin/destination names and prices (the naming convention is different from the stations dataset, so we had to fix that).
Notebook: "preprocessing_3_intercity.ipynb")
Dataset link : https://data.sncf.com/explore/dataset/tarifs-intercites-100-eco/table/?sort=origine

— We combine the above datasets to get the durations of the trips between two consecutive stops and its price (for both intercity+TER).
The mapping between the station names of the stations dataset and the prices dataset proved to be difficult due to the changes in names of the stations. For instance, some stations were named after the city in one data set, but used the actual station name in another. Also, we found small variations in city names and stop names which we had to fix automatically in some cases, and manually in others (we took the manual approach to determine a city's most popular stop in some cases, as that's not always obvious just by checking the name or coordinates ; we resorted to googling for these cases). Example : 'STRASBOURG' : 'Strasbourg-Ville' Notebook: "compute_durations_intercity_and_ter.ipynb"

— The resulting dataset was used to obtain the timetable for each station, which we use to build a graph (using NetworkX) from which we can easily extract paths from one station to another, trip durations/prices and arrival/departure times (routing). Optimizing this part was also crucial. Initially, we wanted to compute all possible destinations from a departure point, and prune the list later. However, this was not feasible : (1) One should not consider taking a 12 hour trip (2) it is far too computationally expensive. We therefore decided to filter out all trips that don't fall within a given timeframe of trip duration (e.g. 8 hours). We then did some further optimizations that make sense for our visualization, bringing the output's side from around 300MB to 15MB. For example, since we're only interested in showing routes to important destinations (more on how we picked those in a bit), one way we pruned our routing data was to remove all routes that lead to non-important destinations.
Notebook: "preprocessing_4_build_graph.ipynb" and "routing.ipynb"

— We obtained the lane/track lines (with curves) as a GeoJSON file from SCNF. Unfortunately, it's a list of disconnected points (not lines) so we had to pre-process this too to get a list of curves we can draw from a departure station to a destination. We were planning to use this to draw exact paths from one station to another instead of straight lines, but ended up abandoning the idea after finding out the vast majority of paths were not present in this data set.
Notebook: "preprocessing_lane_lines_from_geojson.ipynb"

— There are also pre-processed TGV stations/prices from one station to another. We couldn't find some other info we need (e.g. timetables to compute durations), unfortunately, so we didn't use this dataset in the end. Our preliminary work can be found in :
`Notebook:` "preprocessing_2_tgv.ipynb".

— To determine "important" cities, we leveraged Wikipedia articles about historic cities in France and extracted the list of (1) Villes et Pays d'art et d'histoire (link 1) (2) liste des monuments historiques par commune française (link 2 ; we extracted both the list of cities with monuments + the monuments themselves). We also use the list of monuments in our visualization to display enriched information when the user clicks on a city's name.
`Notebook:` : "get_wikipedia_data.ipynb"
`Links:`
Link 1 : https://fr.wikipedia.org/wiki/Villes_et_Pays_d'art_et_d'histoire
Link 2 : https://fr.wikipedia.org/w/?title=Liste_des_monuments_historiques_par_commune_fran%C3%A7aise

Design

The first idea was to draw a map with routes coloured in a way to reflect the prices of trips. To do so, the user selects a budget and departure location as shown on the top. To the right, we wanted to show a sidebar containing the information shown about a city when it is clicked on. (refer to FIGURE 2)

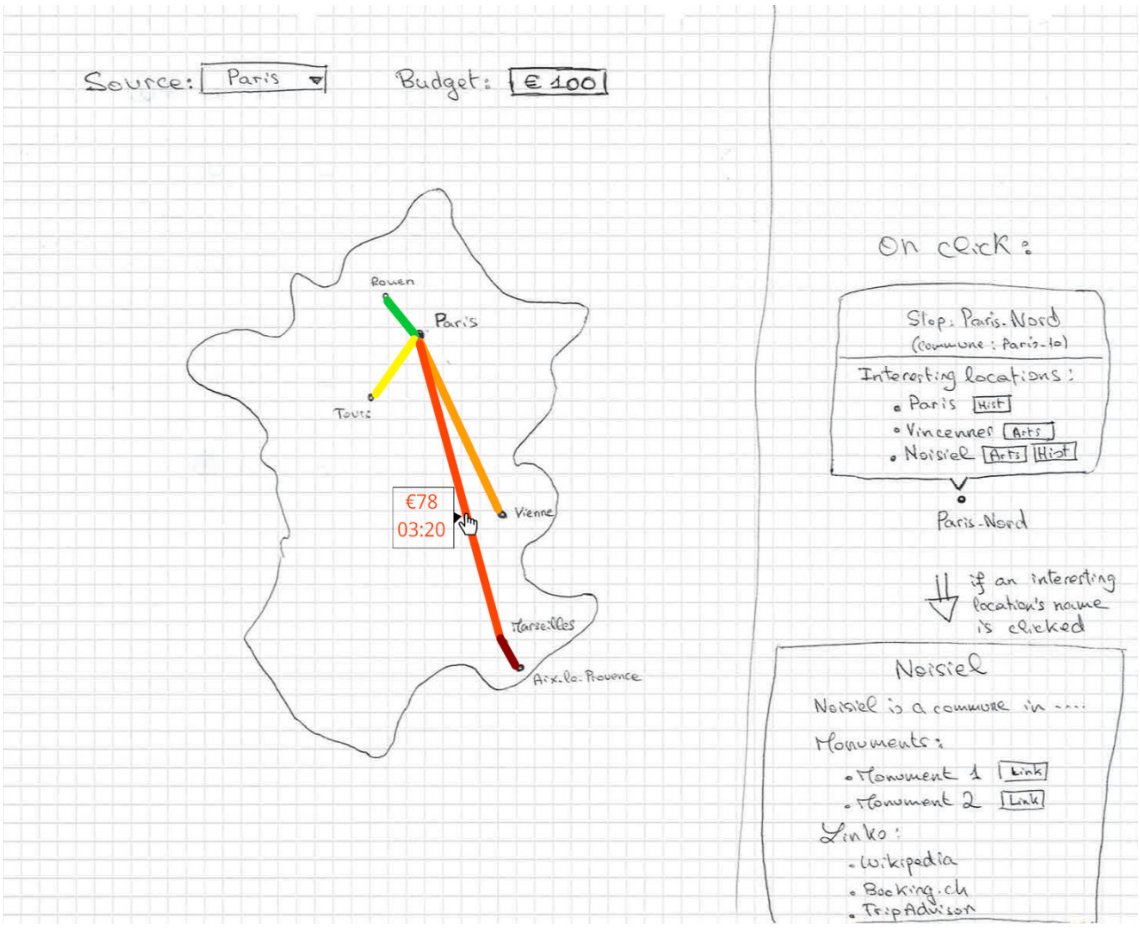

FIGURE 2 – Blueprint

Additionally, when a route is selected, we wanted to show a detailed list of stops along the way as well as the duration and the price from one stop to the next. This is to allow the user to get more information about the trip (FIGURE 3).

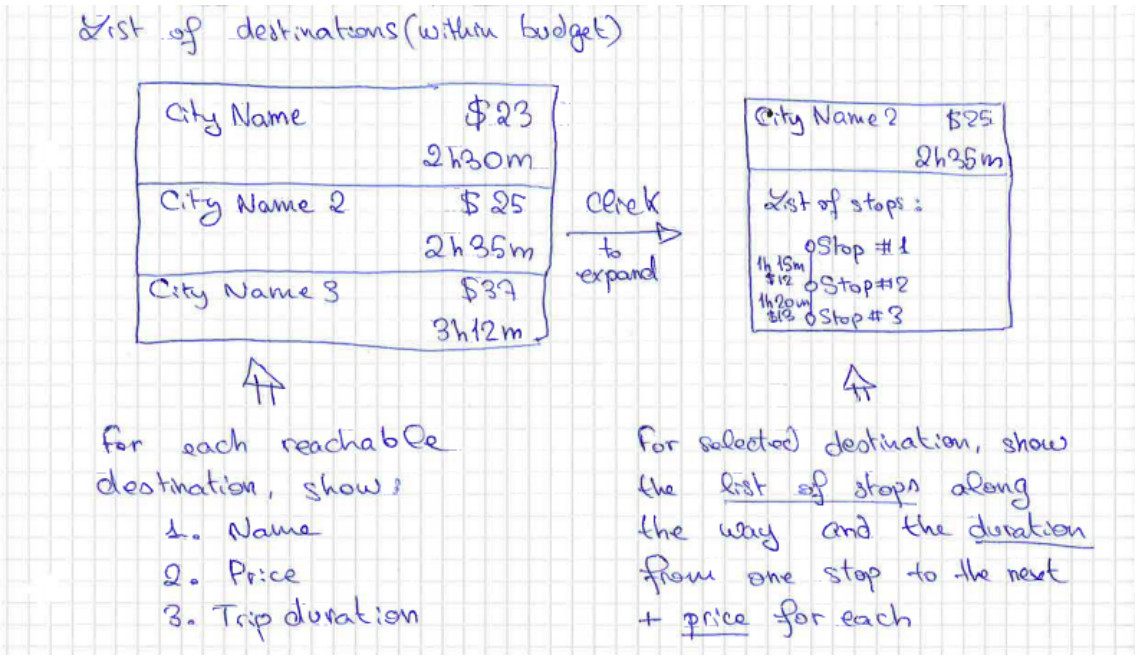

FIGURE 3 – Route list blueprint

Train stops are shown as dots on the map. However, this would overwhelm the user (there are several thousands stations) so we decided to be more smart about it and show a restricted set of cities when the view is zoomed out. In that case, we only show "important" stations (e.g. historic cities). When the user zooms in, less important stations are shown as well. (FIGURE 4 and 5)

To differentiate between important stops and the rest, we decided to use different colors that leverage visual popout. As such, we show important stops in blue and other stops in gray. Important stops' circles are also a bit bigger. This ensures that important stops are easily distinguishable at any zoom level. (Refer to FIGURE 5)

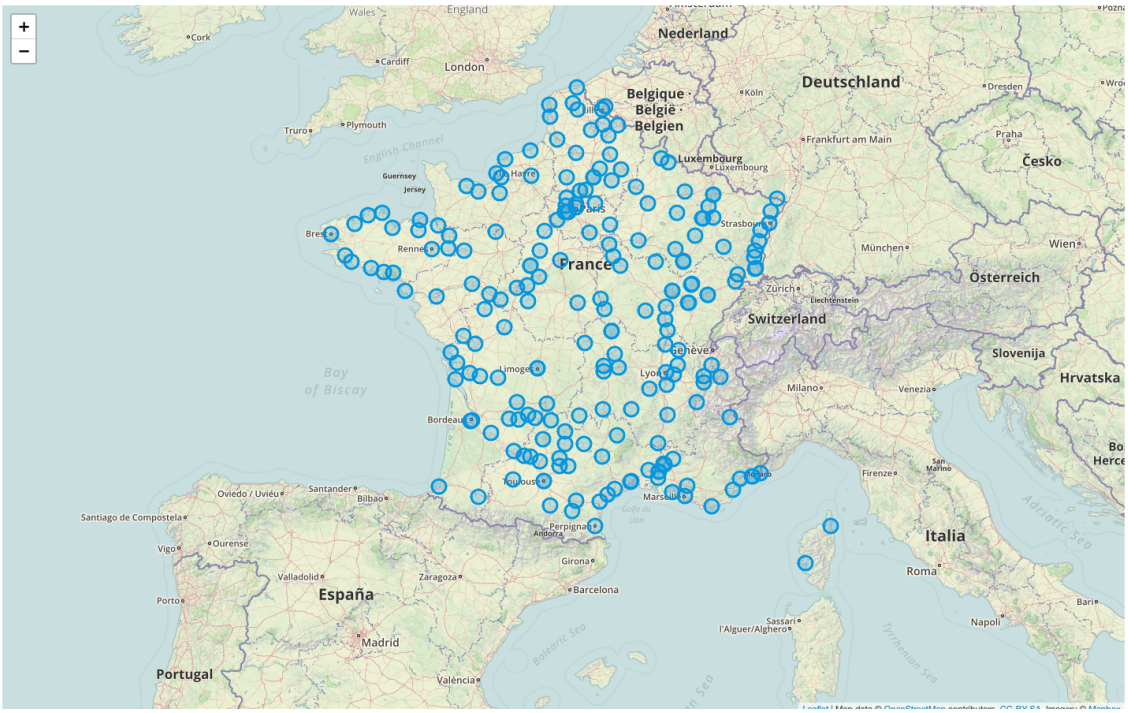

FIGURE 4 – Important cities in blue

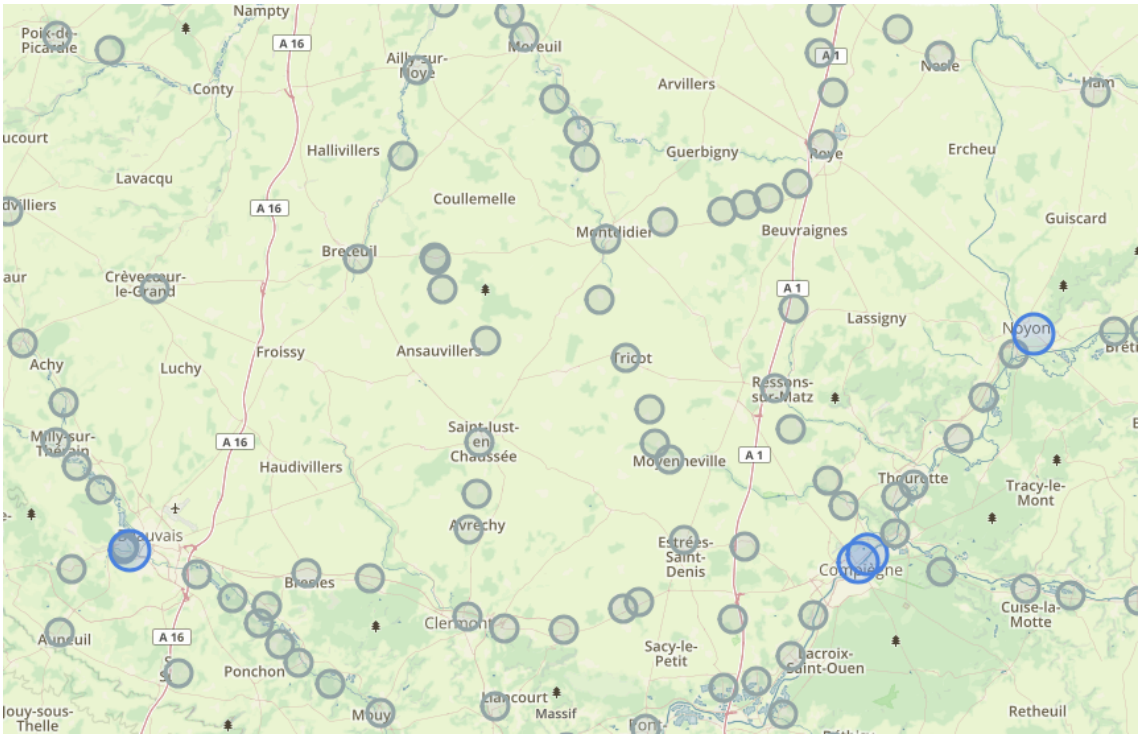

FIGURE 5 – Other cities in grey when zooming in

When the user clicks on a station, he gets the name of the city of the station and whether it is a historical or art historical city (Figure 6). Then, we added the summary of the city in question using the API of Wikipedia, so that the user can get information to help him decide whether or not he's interested in it. (Figure 7).

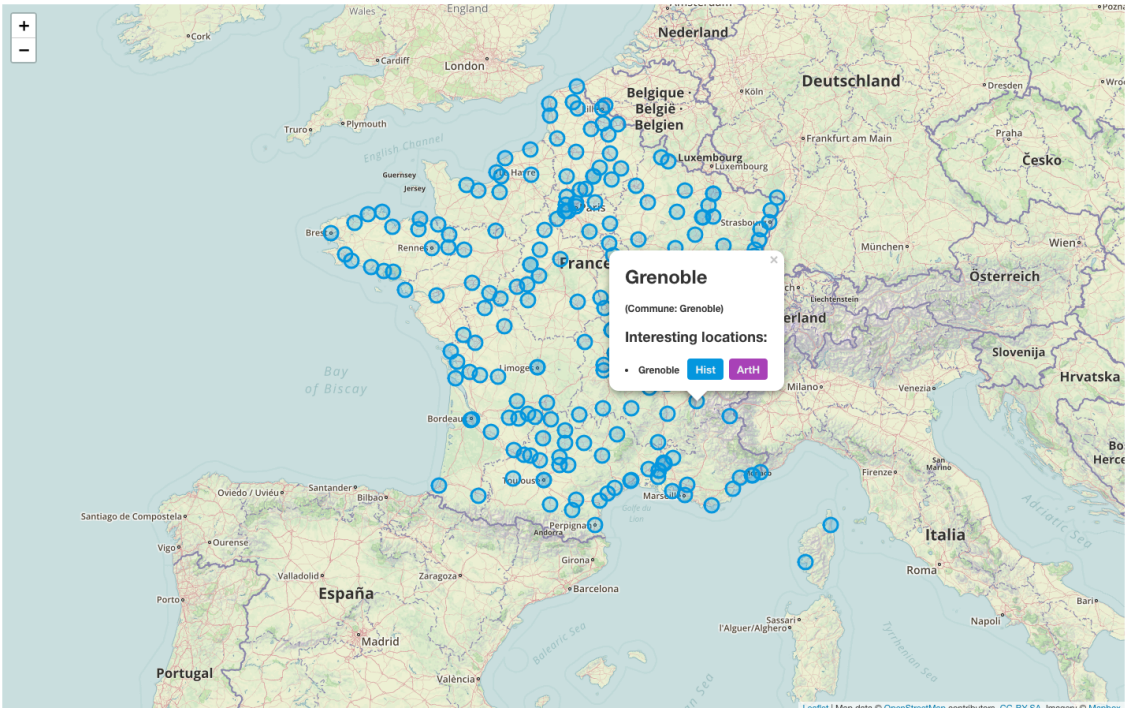

FIGURE 6 – Popup on Grenoble

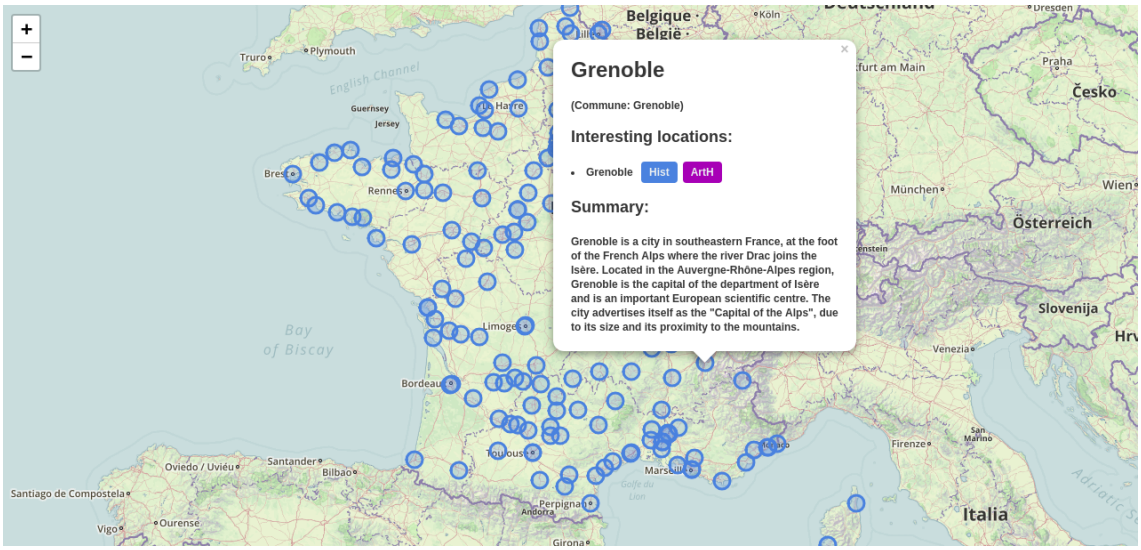

FIGURE 7 – Popup on Grenoble with Wikipedia summary

A search bar that allows the user to search by department (e.g. Isère) and zoom to it. The idea behind this is to let the user zoom in to the department he lives in. For the same purpose, we drew the department's borders in blue (FIGURE 8 and 9).

In order to simplify the map and make it less cluttered, we removed the department lines and added a translucent dark mask on the map to dark all countries except France, therefore putting the user's attention and focus on France which contains all our used stations (FIGURE 10).

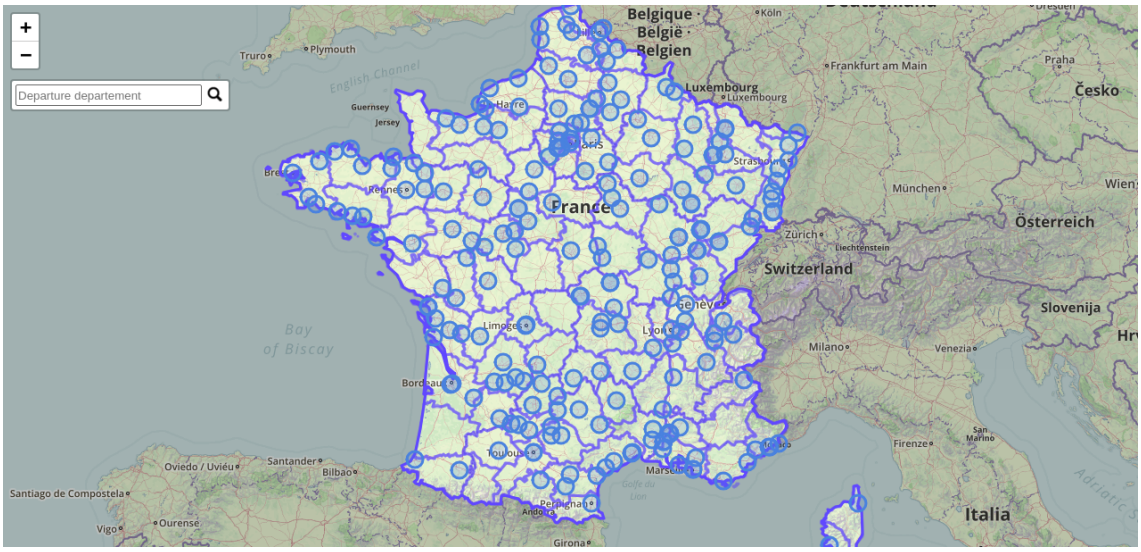

FIGURE 8 – Search by departement

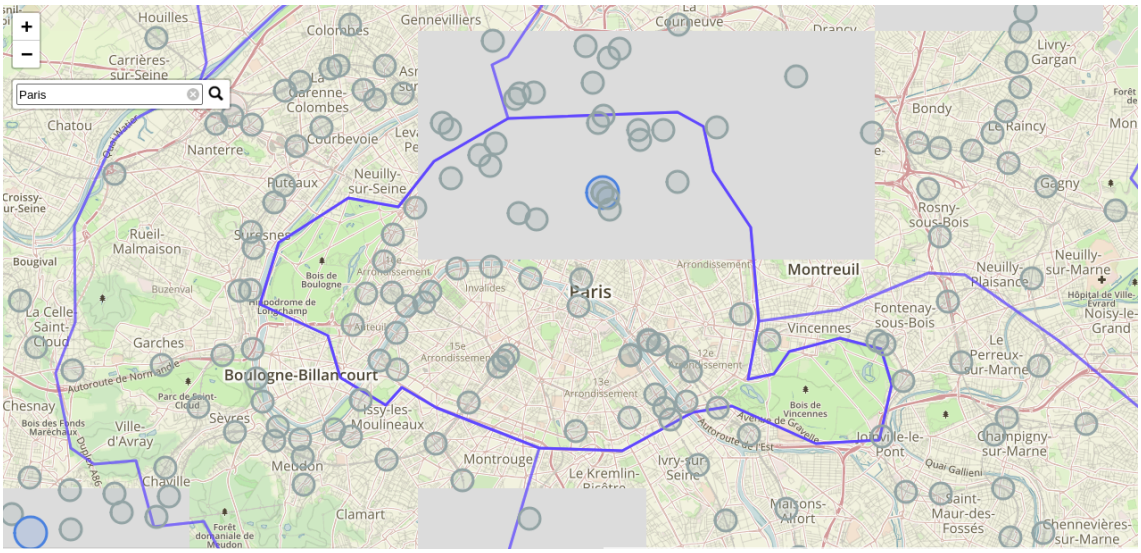

FIGURE 9 – Zoom into departement

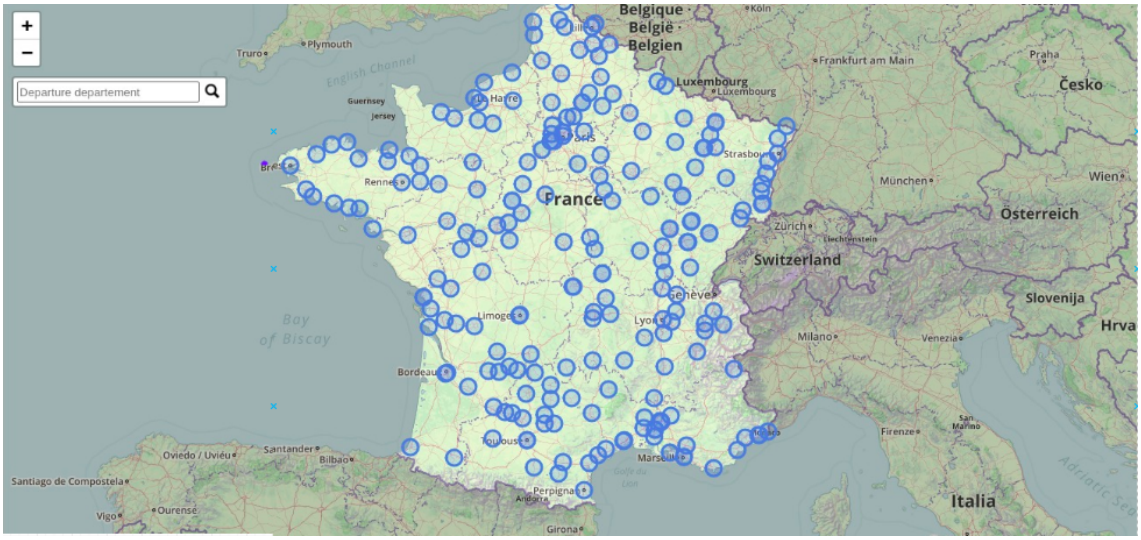

FIGURE 10 – Hide department separation

Then, we added a limit budget and a search bar for station names in the upper left corner (FIGURE 11).

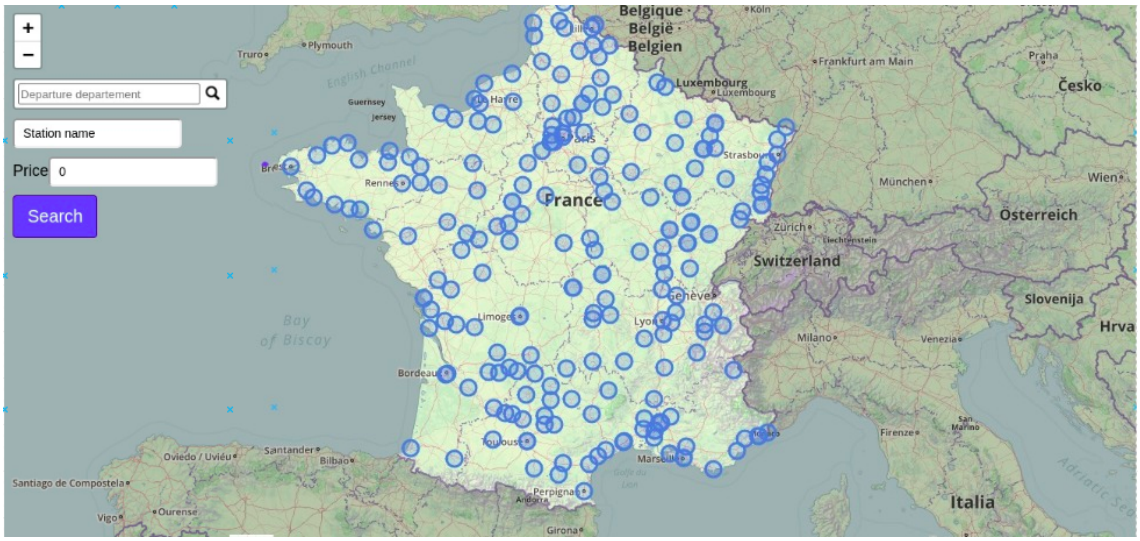

FIGURE 11 – Budget and starting station inputs added

After testing, we noticed that the popup implementation does not always work ideally when it is close to a page edge. (In fact, it disappears right away in that case and does not stay open unless the clicked marker is moved away from the edge of the page.) Additionally, the popup was becoming a bit cluttered as we added more info to it.
For these reasons, we decided to replace the popup with a sidebar, which provides a more consistent experience and more breathable space. It is also much more aesthetically pleasing this way .
Since some users might want to explore the map and pick their location instead of using the search bar, we also added the option to select a starting position from the sidebar (after a station is clicked). This is done via the "Select as starting location" button. (FIGURE 12)

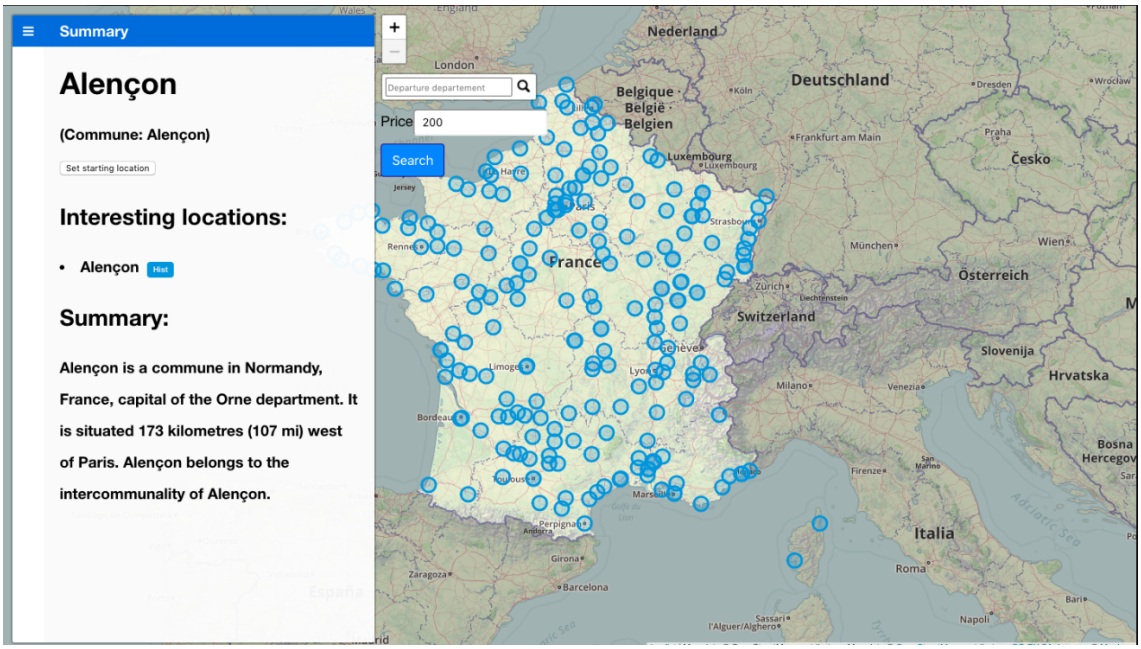

FIGURE 12 – Sidebar addition

To make our visualization better, we moved the search by department search box to the top right corner. We then designed the starting station search box and budget selector in a way that's more consistent with the map's components, placing them to the top left corner above the sidebar. In addition, we changed the budget input into a scroller. We also moved the zoom controller has to the bottom right corner, as that's more consistent with most tools that use maps.
Finally, we limited the zoom levels (minimum and maximum zoom) to values that make sense for our application : for example, we do not allow the user to zoom out too much.
When it comes to the color map we used for the routes, we were first using the RGB color space. As this was confusing due to human color perception, we then moved to use the LAB color space which modifies the lightness of the colors (something humans pick up much more easily) as well as the value. We also simplified the color scheme from five colors (light green, green, yellow, orange and red) to just two (green and red). The reason we picked these two colors is that green is commonly associated with something positive, while red is often associated with negatives : the greener a path is, the cheaper (more positive !) it is.
The result of these changes can be seen in FIGURE 13 : after choosing the departure station and the budget, all possible routes to potential destinations are drawn on the map.

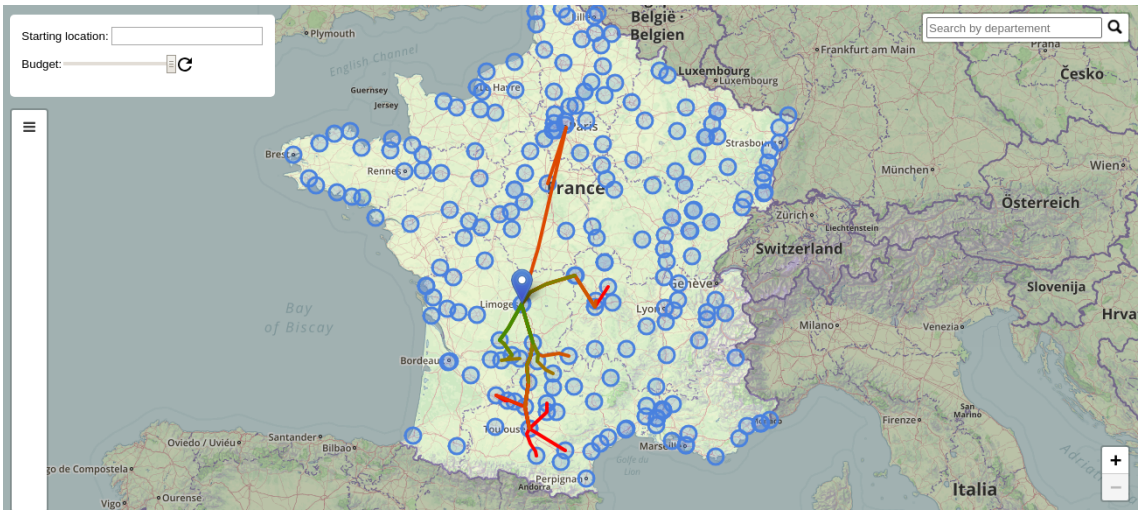

FIGURE 13 – Routing

Some general improvements for the design were added in order to further better the visualization. FIGURE 14 shows the general new design changes we made ; details of each part will be explained below. You'll notice that a sidebar was added to the right, which holds the details of each trip's prices and durations. Also, the search by department box was removed as we found that the search by stop name feature was more intuitive, and zooming in to a specific department actually hurts our visualization as it results in hiding most paths we draw !

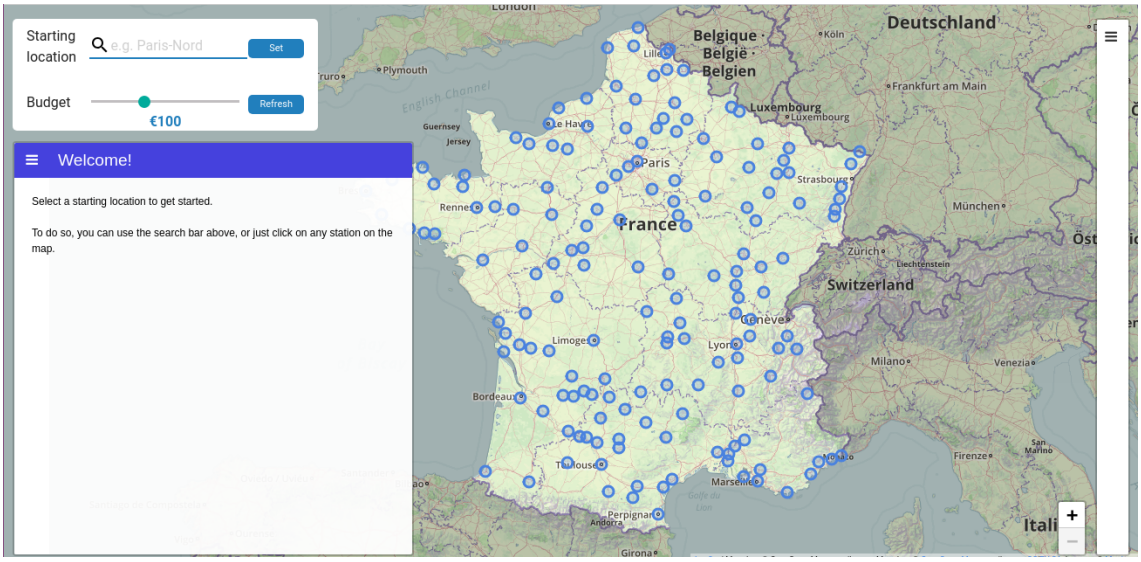

FIGURE 14 – General view

As per FIGURE 15, it is now possible to see all the important reachable cities by looking at the content of the sidebar to the right, which provides a sorted expandable list

of destinations. You can also hover the mouse on an edge that's drawn on the map to see some quick info about it.
All reachable cities are shown in the right sidebar, with the price of the trip and the duration. When a list item representing a destination is clicked, we show the detailed list of stops along the way, along with the duration and the price from one stop to the next. The destination stops are colored in the same way as the routes, giving a quick visual overview of each. Additionally, we made it so that clicking on an edge in the map directly opens the right sidebar with the respective trip expanded and in focus. An example will be shown below for a better understanding of the idea : when selecting the line going from Paris-Nord to Valenciennes, you should get this specific view (FIGURE 15) :

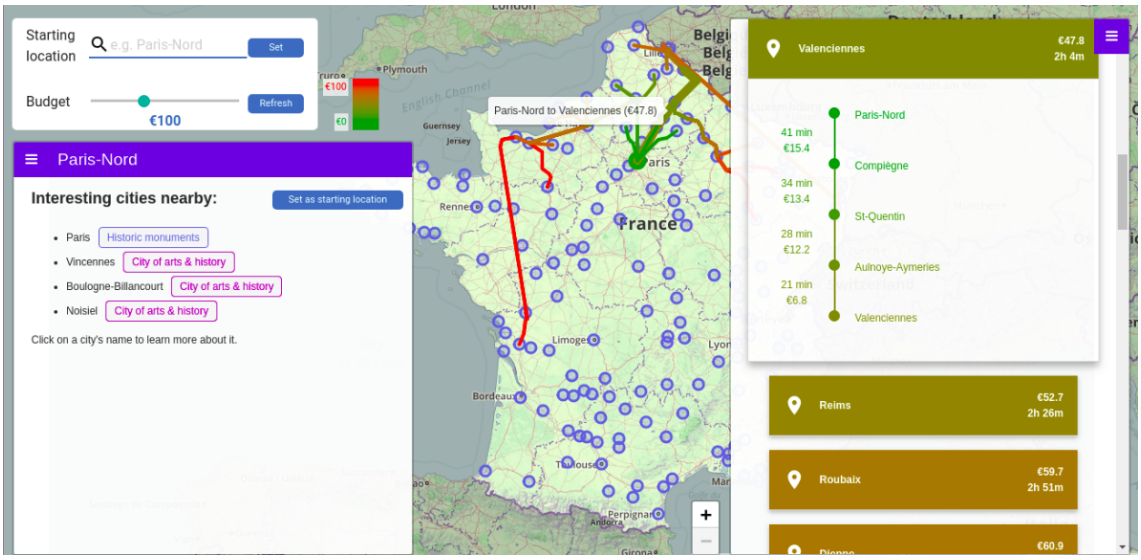

FIGURE 15 – Right sidebar when clicking on Valenciennes line

For the left sidebar, some interesting improvements were added as well. When clicking on important stops, the sidebar is opened showing important cities near the selected station (i.e. this stop is the closest to all these cities). For each city, when clicked on, a Wikipedia summary is shown, as well as an representative image (also provided by Wikipedia) and the list of monuments in that city. In this manner, the user can get complementary information about the cities reachable from a selected destination (FIGURE 16 + 17).

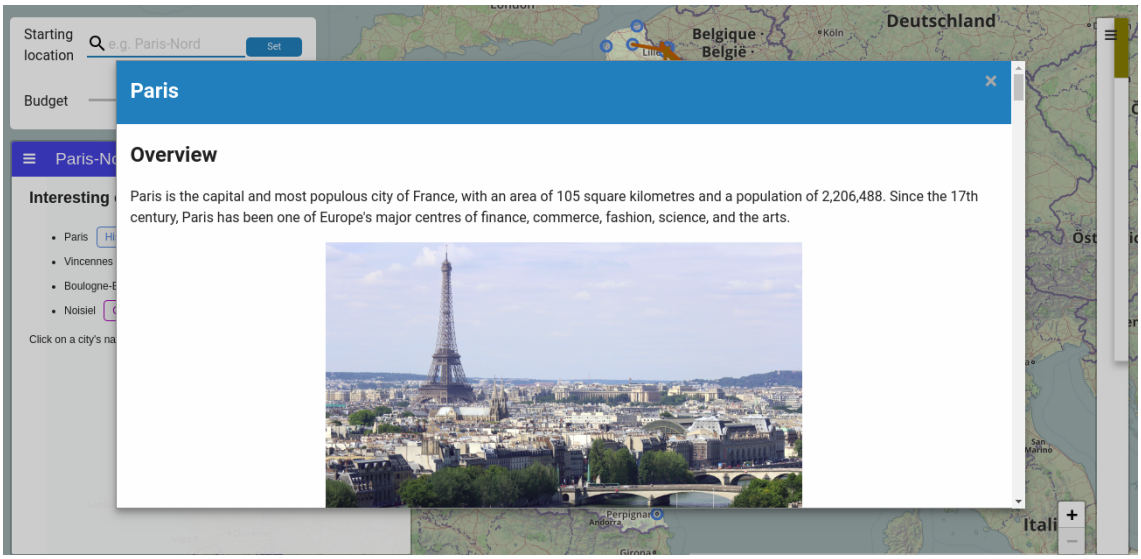

FIGURE 16 – Left sidebar : showing the summary + image

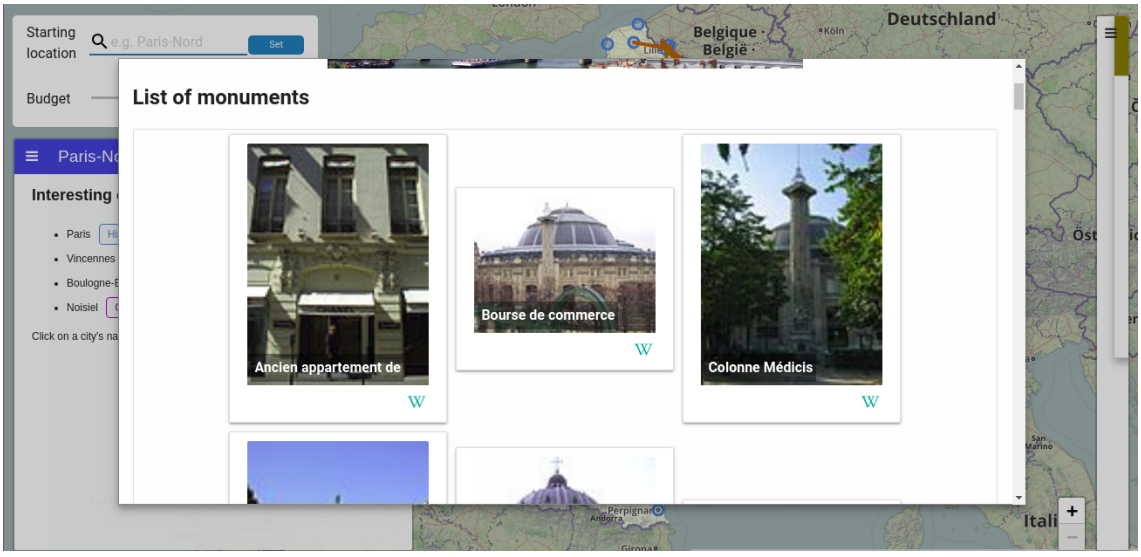

FIGURE 17 – Left sidebar : showing the monuments in the city

Other improvements were added for a better user experience. We mark the starting location with a green circle that's slightly bigger than the other markers, in order to make it easily visible on the map. This is both consistent with our color scale (green being the start) and still easily distinguishable on the map (since no other green circles exist on the map). We also noticed that showing all paths at once was confusing as the user would be overwhelmed at once and potentially forget where they start from. For this reason, we decided to animate paths construction. Each path starts drawing from the starting location, and progressively grows until it reaches the destination. This happens for all the

paths, providing a clear and intuitive animation for all the paths. In short, it makes it easier for the user to see where the routing begins and how it evolves.
In addition, hovering over a stop marker now shows the name and makes the circle a bit bigger, giving visual feedback for whichever stop the cursor is placed on. (FIGURE 18)

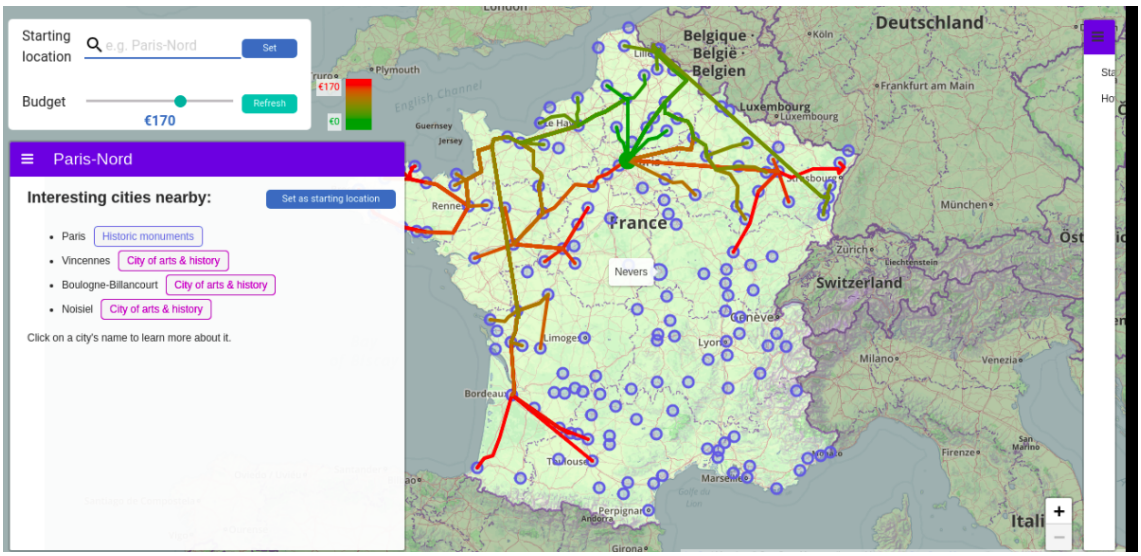

FIGURE 18 – Highlighted starting location (Paris-Nord) + bigger city marker (Nevers)

When the selected budget is not sufficient to travel from the selected departure, we show a helpful error message that lets the user know the minimal budget required to reach potential destinations (FIGURE 19).

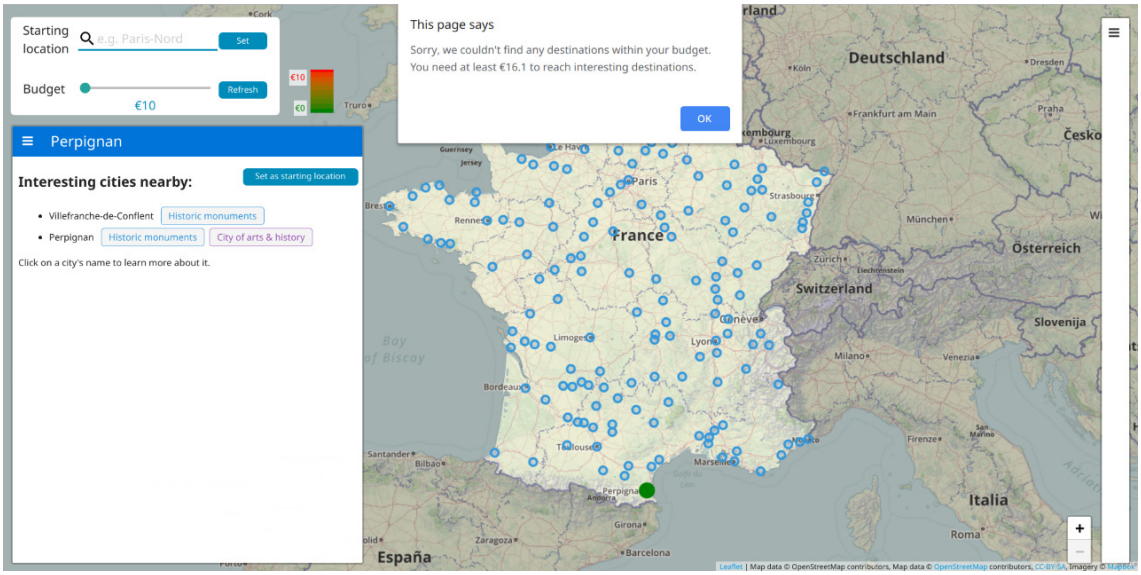

FIGURE 19 – Error message

Last but not least, we wrapped up our data visualization in a short blog post that introduces the aim of the tool and explains how to use it. We also added a fullscreen button to the map, allowing the user to expand the data viz if he wants, once he's done reading the blog post.

What is different from the initial proposal

— We were planning to develop this visualization for Switzerland in our initial proposal. However, the prices in the SBB datasets are not made available. We therefore searched for a different dataset, and found the SNCF open data.

— Adding booking prices information has proved to be impossible as we couldn't obtain API keys despite requesting them by filling the necessary forms multiple times. (TripAdvisor API, FourSquare API, Booking API). Scraping the data was also not an option due to the design of these websites (a few quick trials had us deal with Javascript-driven pages that eventually led us to reCAPTCHAs challenges). It also turned out that the Google Maps API is aimed at tools that directly use Google Maps, so we couldn't leverage it in our tool.

— TGV trains don't have a fixed timetable in the data. Due to this, we did not include them in our routing. Unfortunately, this means the visualization is missing some routes that would've made more destinations reachable for cheaper ; still, it does not affect the visualization itself or its usage for this project.

— Branching out to other countries in Europe hasn't been done yet, since the most difficult part of the project is dealing with the datasets. However, we'd love to do that and it shouldn't be too hard if we're provided with a sane and consistent dataset.

Technical details

The implementation of our project has been done using a server client architecture, as loading the whole raw data at once is not a good solution : it's around 20MB of data and involves a lot of processing that shouldn't be handled by the user's computer (after all, our purpose is not to turn their computers into expensive heaters).

Notebooks (pre-processing)

The pre-processing and routing implementations have been done using Jupyter notebooks, mainly using the Pandas/Numpy and NetworkX packages. This was a crucial step, as the raw data is unusable, and included the following tasks and challenges :

— There were separate tables containing prices, station names and positions and trip times.

— There were different tables for different kinds of trains.

— The different tables had several inconsistencies, including missing stops or different naming conventions / spelling variations.

— Building the routing data and saving it so that it can be directly used by the server.

— Obtaining the list of historic cities from Wikipedia, as well as the list of monuments for these cities.

Server

The server is implemented using Python/Flask. It is responsible for the following operations, and communicates with the client through HTTP GET requests and JSON responses :

— Returning the list of stations, used for the search by stop name feature.

— Returning the list of visible stations (including their coordinates), given the bounds of the current map configuration (zoom/pan). This was done as a measure to reduce lag, as drawing too many points caused some performance issues with Leaflet. This includes the logic behind filtering important cities depending on zoom level.

— For a given station, returning the list of nearby important cities and their type (i.e. Historic City or City of Arts and History).

— Given a starting stop, computing and returning the list of paths to each reachable important destination. This uses NetworkX with the saved routing data from pre-processing (so that we do not recompute all this info for every session ; this is crucial since pre-processing takes minutes !). This includes filtering according to budget constraints.

— Returning the list of monuments for a given city.

Client

The client is everything the user sees. While it communicates with the server to obtain the necessary data, it is still the core of our data visualization.

The client is responsible for all the tasks we mentioned in the Design section. We'll focus on specific implementation details here instead :

— We use Leaflet for everything related to the map, as well as some of its plugins for added functionality. These are detailed below.

— sidebar-v2 is used for the left and right sidebars.

— In order to darken the area outside of France, we make use of Leaflet-snogylop to invert a GeoJSON polygon representing the shape of France.

— Leaflet.EasyButton is used to add custom buttons to the map, which we needed to add a fullscreen button.

— Leaflet.Polyline.SnakeAnim is responsible for animating the paths.

— Chroma.js is used to generate the color scale for the paths and legend.

— We make minimal use of jQuery, mainly to handle input change for the slider. Since pure JS solutions can be quite complex and prone to error to handle all possible events (including changes caused by mouse clicks, mouse dragging/dropping and keyboard arrows), we used a reliable functionality for this provided by jQuery instead.

— Wikipedia.JS is used to fetch the summary text and image shown for important cities.

Website design :

— Bootstrap along with the Clean Blog template were used for the blog design.

— MaterializeCSS was used for the design of some of the data viz-related elements (e.g. expandable list in the right sidebar).

Additional notes :

— We made heavy use of HTML, CSS and JS to customize the map to our liking and improve the UX as much as possible. Some examples of this include modifying the sidebars to support a single pane without padding and adding sticky headers to them ; overlaying our inputs on top of the map in a way that's consistent with the design ; implementing the fullscreen functionality.

— We tried to follow JS best practices when it comes to coding conventions, naming, documentation, etc. MDN was of great use, and we avoided w3schools as much as we could. :)

— We made use of async operations whenever it made sense to avoid some performance penalties.

Evaluation

Data Insights

We learned that some cities can't be reached unless you go through Paris (or other big cities) and we confirmed the validity of this by going to oui.scnf . In some cases, this can not be visible on our map since we restricted the trip durations to a maximum of 8 hours.(For instance, going from Antun to Paris is impossible in under 8 hours.)

With a 200€ budget and starting from any of the biggest cities (e.g Lyon, Paris, Marseille...), you can get to most destinations.

It is very difficult to get from a smaller city to another without changing through one of the biggest cities.

Unlike in Switzerland, it is very difficult to go from one city to another without TGV given that they are sufficiently far from each other. This makes sense, since, France is quite big compared to Switzerland and its central departments have a lot of rural areas.

Future Work

The main improvement to our current project is including booking prices to the sidebar along with the Wikipedia summary and the monuments photos to further help the user to decide on which destinations are best for his budget.

Another potential but difficult improvement is including many other countries in the map. The main challenge in this case is finding and properly pre-process datasets.

It is also possible to branch out to bus trips as well as flights, boats ... and ultimately have an international trip planner if proper datasets are available.

Peer Assessment

The pre-processing was done in parallel on different tables by all three teammates. And then, different aspects of the visualization were added separately.
At the end, the tasks were splitted into implementation, process book and screencast preparation.
All the members contributed in offering different perspectives and feedbacks. That helped avoid tunnel vision and always had us keep an open mind and constantly look for improvements.
There was an environment that encouraged discussion and respect which helped everyone get their perspective across.