MAKINA CORPUS

{EPITECH} LEARN DIFFERENT*

04/03/2024 - 03/09/2024

INTERNSHIP REPORT

Backend and cartography: setting up a routing system

Submitted by:
Justine Fricou
justine.fricou@epitech.eu

Table of contents

Acknowledgements

First and foremost, my sincere thanks go to my internship supervisors Célia Prat, Jean-Etienne Castagnede and Joaquim Nallar. Their availability, guidance, and genuine commitment to my professional growth have been invaluable and I am truly appreciative of their efforts.

I would also like to express my appreciation to Lise Benazeth for ensuring that my internship went smoothly by encouraging open communication. Her support helped me make the most out of these 6 months.

Finally, I am grateful to everyone at Makina Corpus for their kindness and genuine interest in my work. This welcoming environment made my time here truly enjoyable.

SECTION I

Project onboarding guide

Foreword

The goal of this section is to address a fictional new employee, providing them with a manual to assist them in taking over the project that has been underway during this internship. This “handbook” must feature an introduction to the company, an overview of the team organization and processes, as well as the project’s objectives, overall architecture, and challenges.

Welcome to the GeoTrek team

This onboarding guide aims to provide you with essential information on the team and the project you will be taking over in order to ensure a smooth start. You will find a presentation of Makina Corpus and of the GeoTrek team and its operations, along with an overview of the purpose and global architecture of the GeoTrek project and its routing system.

Please do not hesitate to reach out to your supervisors if you require further information about the project or the day-to-day operations of the team.

Table of contents

Makina Corpus

We are a digital services company specialized in development of web and mobile applications. We are an SME (Small and Midsize Enterprise) present in Nantes and Toulouse and count between 44 and 46 employees, depending on the time of year.

We have core values that are inherently part of our operations and strategy. We believe innovation should benefit the end user and are committed to respecting people and the environment through our work. We are engaged in collaborative projects with research institutions such as IJA (Institut des Jeunes Aveugles, institute for young blind people) or solagro (organization dedicated to promoting sustainable agriculture and renewable energy). We value open-source software and open data and exclusively use free software for developing our products.

We offer a diverse range of services. We develop web and mobile systems, specializing in mapping solutions and producing open-source software. In addition, we offer training sessions on the many technologies in which we have expertise. Lastly, we conduct audits and provide assistance in project planning and management to ensure successful project execution.
These activities are arranged into different divisions:

- The services divisions, south (Toulouse) and west (Nantes), which handle services for projects awarded to us through call for tenders, for products that were not originally conceived by Makina Corpus;
- The product division, which also revolves around services, as we do not sell licenses: it offers services related to products that were initiated by us and in which we are an integral part of the community;
- The exploration division, which focuses on research and development;
- The training division.

Makina Corpus is dedicated to quality and innovation, and strives to foster a supportive environment where our employees can thrive. As you join us, we look forward to your contributions so we can help shape the future of technology in a sustainable and collaborative way.

GeoTrek

The project

The GeoTrek team, which you are going to join, is part of the product division and works on its namesake project. It was initiated in 2012, when several national parks and Makina Corpus co-designed this open-source solution for managing and promoting outdoor activities and natural sites.

Since its creation, GeoTrek has evolved to offer a variety of tools, and its modular architecture enables its users to tailor the system to their specific needs. It is composed of four interconnected components.

The first one, GeoTrek-admin, allows territories (natural parks, tourism offices, etc) to input and organize various cartographic data such as hiking routes and signs, points of interest, construction work, etc. Through this tool, administrators can manage geographical information and add descriptions and multimedia contents for public-facing GeoTrek interfaces.

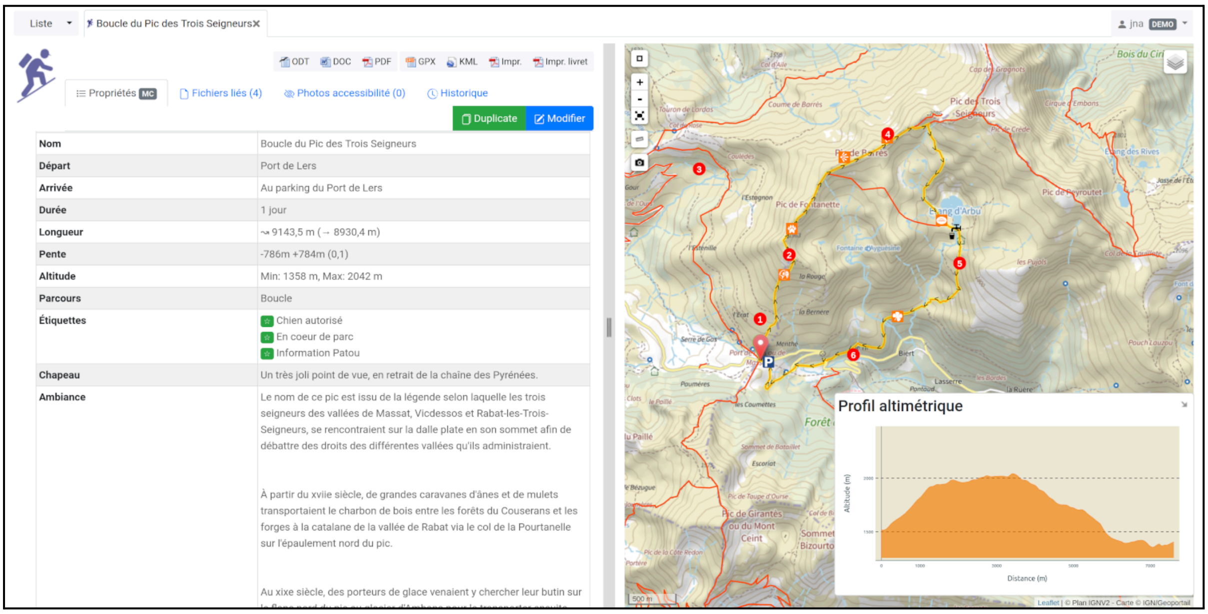


Figure 1: information page for a hiking route on GeoTrek-admin (source: geotrek.fr)

GeoTrek-rando is the main public-facing component of the GeoTrek suite, aimed at publishing and showcasing the information entered in GeoTrek-admin on a customized website. Visitors have access to interactive maps, detailed descriptions for trails and their points of interest, and practical information such as difficulty levels and estimated duration.

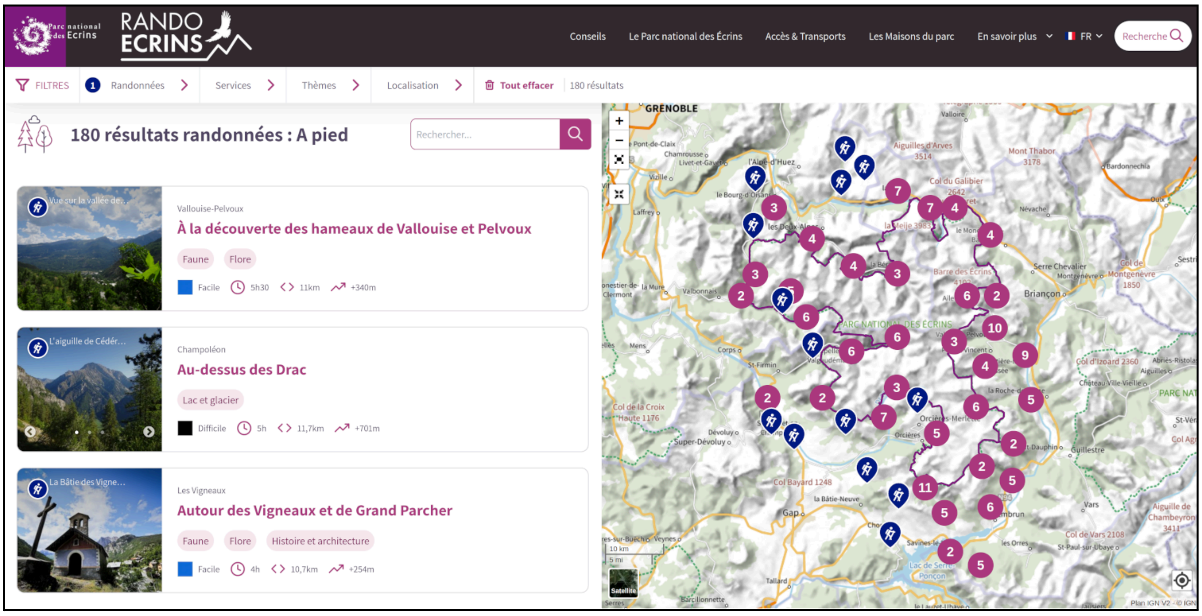


Figure 2: available hiking routes displayed on GeoTrek-rando (source: geotrek.fr)

GeoTrek-widget allows for the easy integration of GeoTrek features into existing websites. This widget can be embedded into a territory's website to display interactive maps, hiking route information, or specific points of interest from a GeoTrek database. It is a customizable solution for organizations wanting to showcase GeoTrek content without implementing the GeoTrek-rando interface.

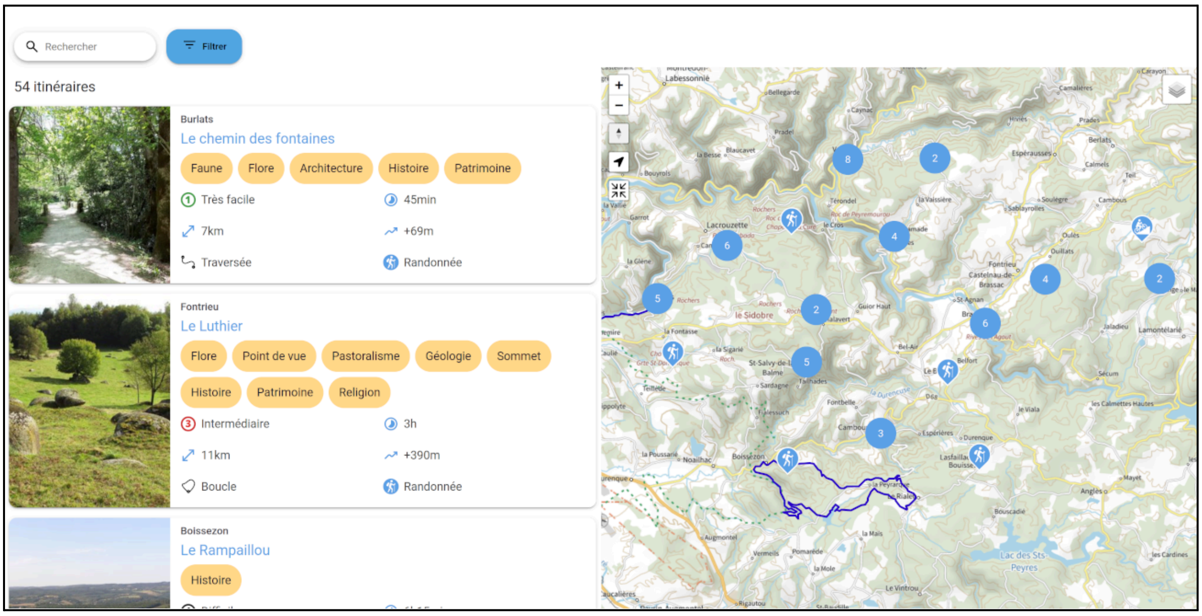


Figure 3: available hiking routes displayed on GeoTrek-widget (source: geotrek.fr)

GeoTrek-mobile gives hikers access to GeoTrek content from their smartphone. It allows them to browse and view detailed information about hiking routes, points of interest and outdoor activities without the need of an internet connection.

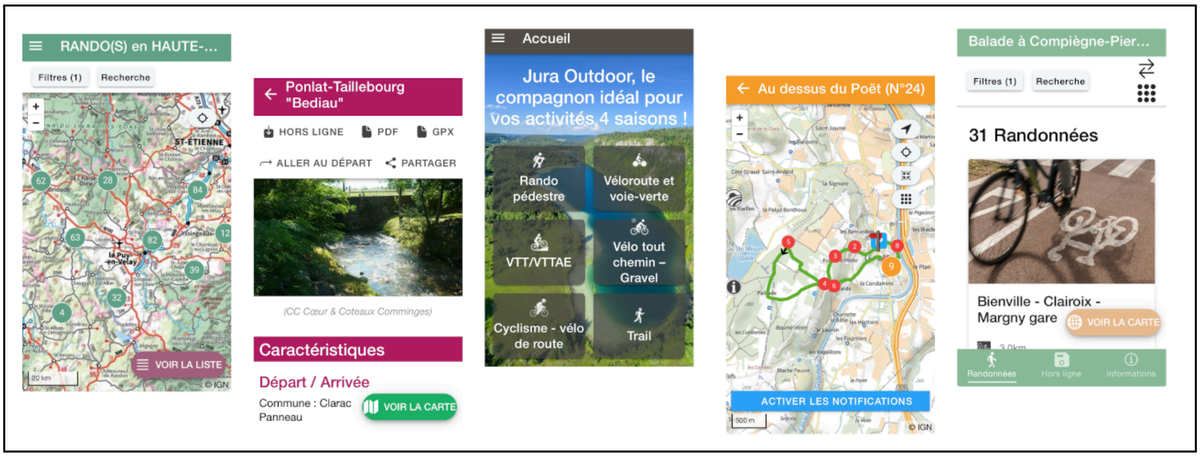


Figure 4: various information on hiking routes displayed on GeoTrek-mobile (source: geotrek.fr)

GeoTrek being an open-source project, any territory can use the GeoTrek solution freely. However, many of them request our assistance for a variety of services. This includes initial setup and configuration, to ensure the system is correctly installed and customized to meet our client's specific needs. We also offer maintenance of the system to guarantee that it remains up-to-date and functional. To finish, many clients commission us to implement new features and improvements to meet their new needs. This enables us to keep contributing to the project and maintain a central role in its evolution.

The team

The GeoTrek team includes professionals with a diverse range of roles and expertise. One project manager oversees the planning and progress of the team members, and handles prospecting in collaboration with a development coordinator.

Two members of the team are frontend developers, responsible for creating the visible part of the solution with which users can interact directly, such as the mobile application or the content displayed in the web browser. While they develop the generic, open-source versions of GeoTrek-mobile, GeoTrek-rando and GeoTrek-widget, one person is responsible for customizing the clients' instances of these products and handles customer support.

The frontend developers work hand in hand with four backend developers who build the server-side logic and the API, an interface allowing the frontend and the backend to communicate. Among them is the lead developer of the project, who is responsible for guiding and overseeing the development efforts.

The team uses a GitLab board containing all future, current and completed tasks, allowing to keep track of progress and ensuring that anyone can know who is working on which task at any given time.

The GeoTrek team has regular meetings which help enhance productivity and cohesion among team members by maintaining interactions despite their diverse and independent tasks.

During the weekly session led by the project manager every Monday, all completed tasks from the previous week are reviewed, and new tasks are assigned to each member for the upcoming week. This ensures a balanced workload and keeps everyone informed about project evolutions and progress on contracts.

Additionally, a short daily meeting allows team members to share updates on their progress and ask for assistance when needed. This creates a supportive environment in which developers can continuously improve by sharing knowledge and skills.

Most meetings are held online, therefore it is essential to be proficient in remote teamwork, as Makina Corpus practices hybrid work in a flexible manner. Each employee chooses to work in the office or remotely on the days they prefer. Most of them work from home 2 to 3 days per week, while some choose to work on-site at all times. Others, who live far from Toulouse and Nantes offices, must come on-site 4 days per month, either every Monday or 4 consecutive days in a single week.

The GeoTrek project is hosted on the GitHub website and is open-source, meaning that anyone can contribute to it through GitHub. However, once a change to the code has been developed and before it is merged into the project, it has to be approved by a member of the GeoTrek development community, which is composed of all developers who contribute to the project. This reviewer ensures that the proposed modification meets quality standards such as code cleanliness, performance optimization, and thorough testing. Therefore, the GeoTrek team members in Makina Corpus adhere to this process which allows to maintain the robustness of the software and to facilitate future development. An additional benefit of this approach is that it helps all team members to continuously enhance their skills by exchanging feedback and best practices.

GeoTrek-admin: backend routing system

This section will introduce you to the subject you will be working on at Makina Corpus. You will be contributing to both the browser-side and the server-side of GeoTrek-admin's codebase in order to improve the user experience by decreasing wait times during the creation of hiking routes.

Global architecture of the project

As any web-based interface, GeoTrek-admin can be divided into two parts: the frontend and the backend. The frontend is the part of a website or application that users see and interact with directly. This includes everything that appears on the users' screens: buttons, images, interactive elements, etc. The backend corresponds to the server-side operations which are hidden from the user. It involves the management of servers, databases, and application logic, which allows it to process data and user requests.

The frontend and the backend are able to communicate via an API (Application Programming Interface). Through the API, the frontend sends user requests to the backend, which then replies with the appropriate response so the frontend can display it.

The backend of GeoTrek-admin is developed using Django, a web framework which also plays a significant role in generating parts of the frontend, as it supports the creation of dynamic web pages and user interfaces.

To enhance user interactions, the frontend is further developed with JavaScript, a widely used technology for creating interactive elements on web pages. For the mapping features, GeoTrek-admin uses Leaflet, a collection of pre-written JavaScript code which simplifies the process of displaying and interacting with maps on a website.

The project relies on PostgreSQL for the management of databases, and on PostGIS which is an extension of PostgreSQL that allows to store and handle geographic data.

This management of geospatial data in GeoTrek-admin relies on a topological network corresponding to existing paths on a territory (see Fig. 5). The location of all other types of data (hiking trails, signboards, points of interest, sensitive areas, touristic events, maintenance work, etc) is stored in relation to these paths. For instance, a hiking route consists of a succession of paths (see Fig. 6), and a tourist spot's location is stored as a position relative to a path (more specifically: a distance to the closest point on the closest path). It is important to note that paths and routes are different concepts and should not be confused. A path (displayed in red in Fig. 6) represents a trail, while a route (displayed in yellow in Fig. 6) represents an itinerary within the paths network.

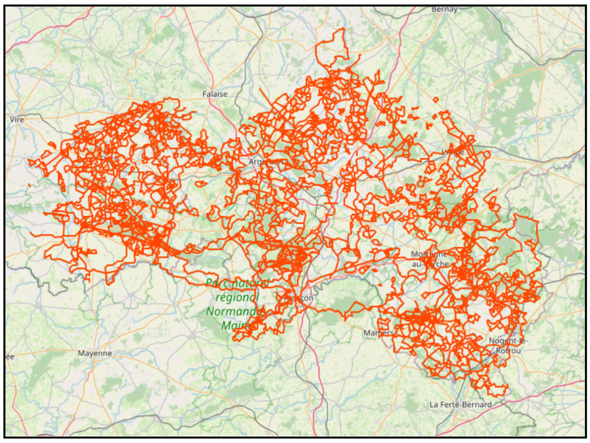


Figure 5: Topological network (paths) of the Orne department, northwest France

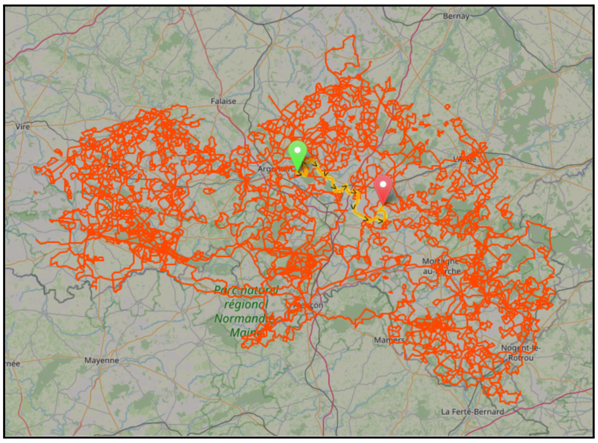


Figure 6: Hiking route based on the topological network (paths) of the Orne department

Routing system

In the case of data types defined as a series of paths, such as hiking routes, a routing system is used during creation. The user selects a starting point as well as an ending point, and GeoTrek-admin computes and displays the shortest route. The user can then add via-points if needed, to divert the proposed route.

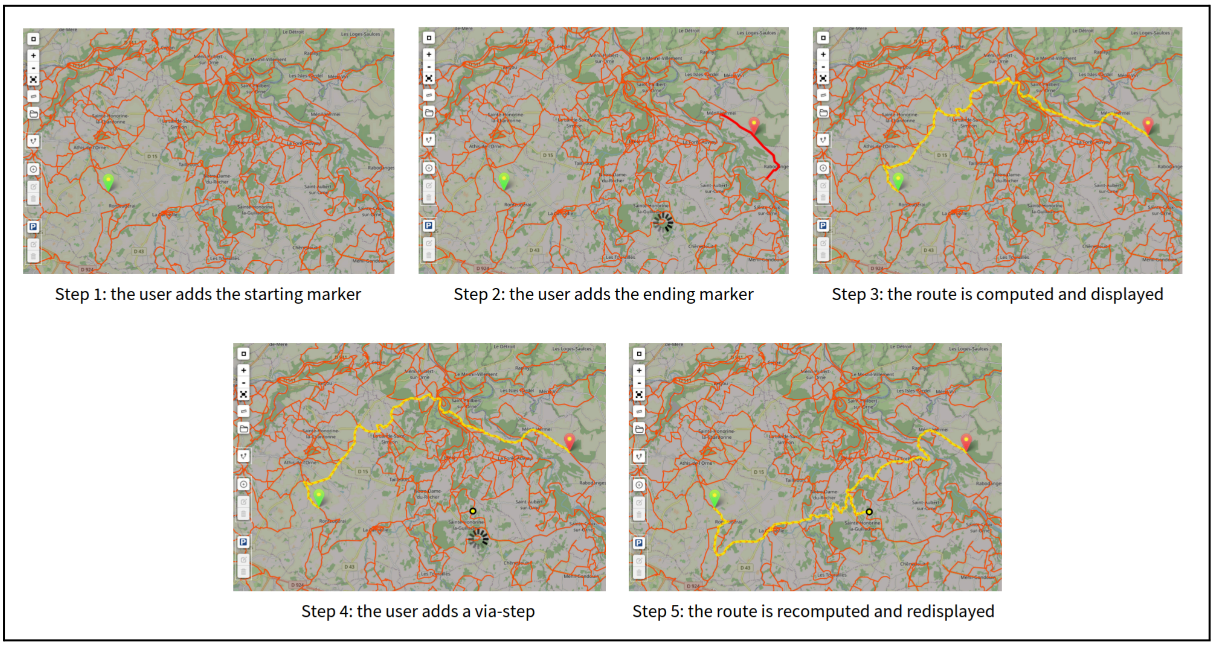


Figure 7: Creation of a hiking route on GeoTrek-admin

The existing routing system, having been developed more than a decade ago, is now outdated and better solutions have since emerged. However, the route is calculated in the user's browser, leading to several drawbacks.

First, the inability to deploy a more recent and efficient routing system (e.g. pgRouting, which will be discussed below), as such new solutions must be implemented server-side rather than browser-side.

Second, an increased load on the browser, whose performances depend on the user's device: a user with a low-performance device will experience significantly long wait times during route calculation. If the route were to be calculated on the server, every user would experience the same, shorter wait times.

Third, decreased test coverage. Thorough code testing verifies that any new contribution is functional and does not introduce bugs to the existing code. Therefore, these tests must cover as much code as possible, to ensure the project's robustness throughout the development process. The GeoTrek testing system relies on server-side code testing, as opposed to browser-side code testing. Thus, the route calculation being executed on the browser, it is not covered by tests.

In order to solve the above issues, Pull Requests have been opened. A Pull Request is a request opened on GitHub to submit code changes to be merged into a project. The first one moves the route calculation from the browser to the server, while the second one builds on this advancement and implements a more efficient server-side routing system. This is the subject you will be working on, as these pull requests have areas for improvement and optimization before they are merged into GeoTrek-admin.

Moving route calculation from frontend to backend

Pull Request #4070 moves routing to the backend, maintaining the current route calculation system (Dijkstra's algorithm). This update paves the way for future improvements that will enable faster routing and the handling of a larger topological network. Your mission for this first Pull Request will be to further optimize the code.

Work accomplished

This contribution focuses solely on relocating the routing system to the backend and assessing the performance differences related to this calculation's location. To this end, the routing system that has been implemented server-side in this Pull Request had to be the same as for the browser-side one. It consists of Dijkstra's algorithm, written in JavaScript for the browser-side routing, and in Python for the server-side routing. Dijkstra's algorithm is a method which allows to find the shortest route between two points by evaluating all possible routes and selecting the one with the least distance or cost.

A new endpoint has been created, which is a specific address on the internet where data or a service can be accessed. In this case, the frontend can provide it with geographical locations (which steps the route should go through) to request a route, and receive a response from the backend. This new endpoint is called 'route_geometry' and a request must provide a list of steps, each of them containing the following data:

- Its latitude and longitude;
- The path it is located on (as the user can only create steps on paths).

The response it provides contains:

- The route geometry: a geometric object that represents a series of connected points, used to display the route as a line on the map;
- The route topology: which portions of paths the route goes through, used by the browser to later save the route.

To compute this response, the backend processes as follows:

1. For each step, a point is created (using its latitude and longitude coordinates) and located on the path. This allows to determine the exact fraction of the path where the step is situated;
2. With this data are computed the sub-routes, which are the segments of the route between each pair of steps. For instance, a route with no waypoint contains only 1 sub-route (see Fig. 8), while another itinerary with 2 waypoints will contain 3 (see Fig. 9);

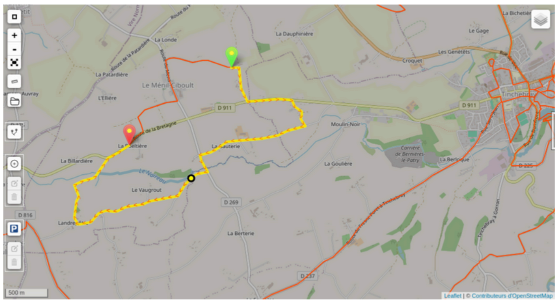
Figure 8: Route with no waypoint (1 sub-route)

Figure 9: Route with 2 waypoints (3 sub-routes)

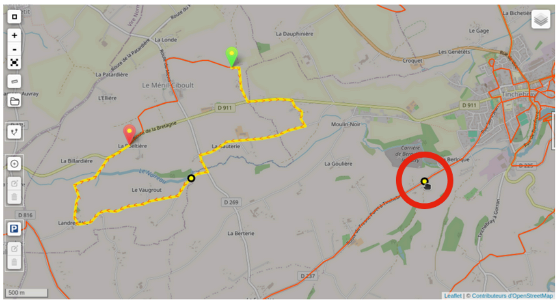
3. The geometry and the topology of each sub-route are determined;
4. To finish, all the sub-routes are compiled into a collection of geometries and a corresponding list of topologies, which provides a complete representation of the route while still allowing the frontend to manage different sub-routes independently.

The frontend code has been adjusted to use this new endpoint rather than computing the route itself. This involved the removal of all code that is no longer necessary and a refactoring of the remaining code to work with this new system, including the creation, saving, and modification of the route. This restructuring has led a few to bonus enhancements.

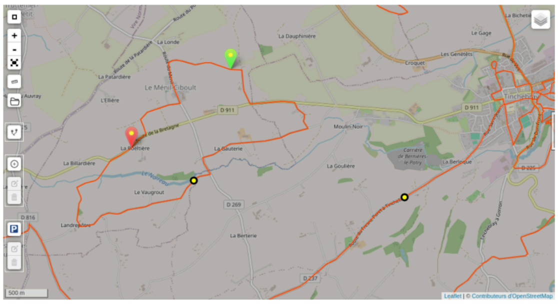
First, the user experience has been improved. When a user creates steps that lead to an impossible route, the GeoTrek-admin interface now assists them in correcting these steps. This can happen when markers are placed on paths that are not linked (see Fig. 10), as well as when they try to place a marker outside of a path (see Fig. 11).



A route on an isolated path, containing one via-step



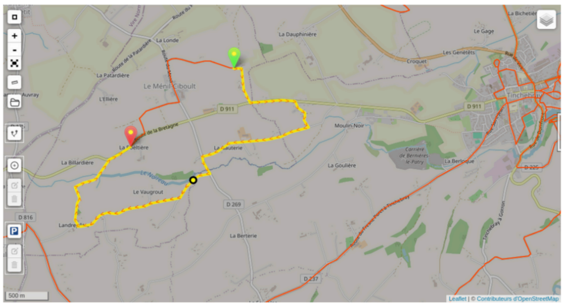
The user is adding a second via-step on a path that cannot be connected to the route



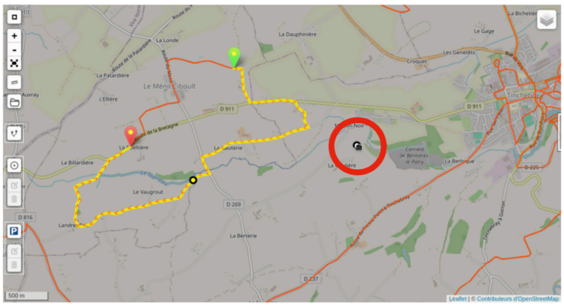
Original routing system: the valid portion of route is no longer visible and all markers remain editable, making it more difficult for the user to notice and correct the error

New routing system: the interface issues a warning and only the incorrect marker can be edited for as long as the route is invalid

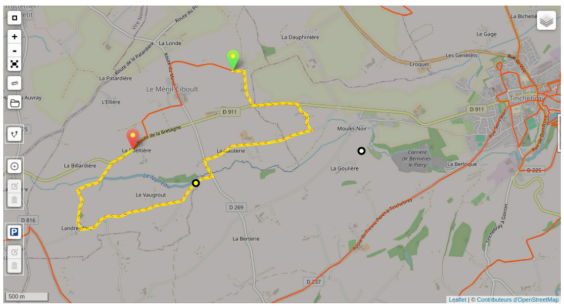
Figure 10: GeoTrek-admin's behavior when adding a step on an unreachable path



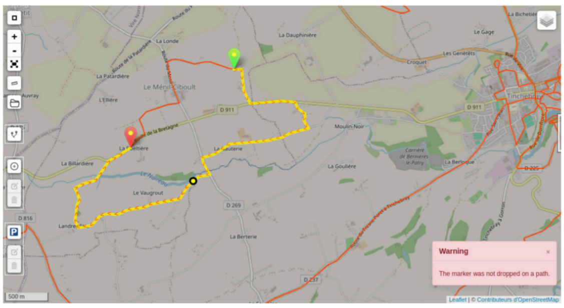
A route on an isolated path, containing one via-step



The user is adding a second via-step outside of a path



Original routing system: the incorrect marker persists, requiring the user to notice the error and correct it

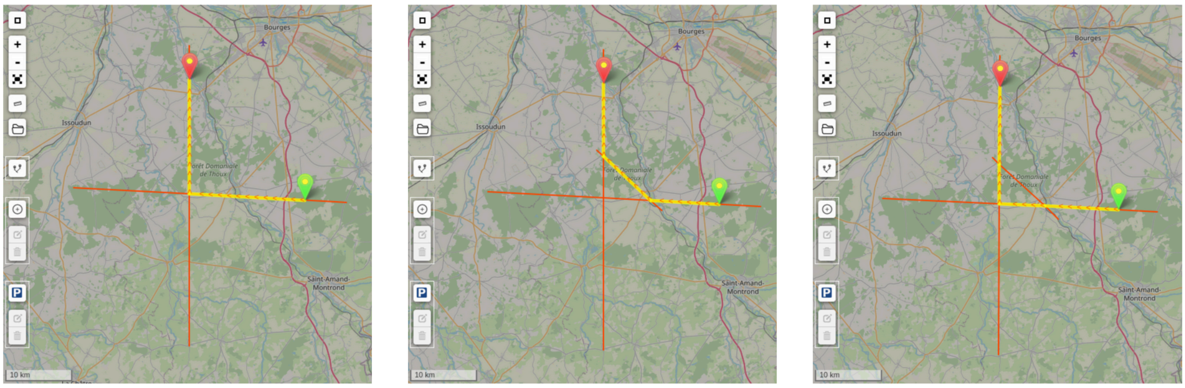


New routing system: the interface issues a warning and removes the new, incorrect marker

Figure 11: GeoTrek-admin's behavior when adding a step outside of a path

Second, the implementation of this new system has led to the modification of the route edit process, which solved a historic bug. When a user needs to modify an existing itinerary (i.e. an itinerary that has previously been created and saved to the database), the interface has to display this route along with the editable markers. In the original system, the browser displays these markers and recomputes the route which is then displayed. This leads to an issue: if the path network has changed since the creation of the itinerary, it is possible for the old and the new route to be different (see Fig. 12). This is unwanted behavior: the route should not be modified without user input.

In the case of server-side routing, the computation of the route no longer being done in the frontend, the display process of an existing route had to change. Instead of the browser displaying the markers and recomputing the route, it displays it thanks to its saved topology. The topology of the original itinerary being composed of portions of paths, the route follows its original course.



Itinerary saved to the database, before creating a new path | Original routing system: the itinerary has been rerouted on the new path | New routing system: the itinerary is unchanged despite the new path having been added

Figure 12: GeoTrek-admin's behavior when accessing an itinerary's modification page after adding a new path which would shorten the route

As per the GeoTrek project workflow rules, and to ensure that the contribution works as expected while not introducing any bugs to the existing code base, all changes to the code have been tested. This means that unit tests have been written and included into the Pull Request. They are small segments of code which allow to check whether a project's source code works properly. A Pull Request's code can be merged into the project if all its tests are successful.

To finish, a benchmark comparing the original and the new routing systems has been conducted. It assesses the gains and losses in performance if the project were to switch to the new calculation system. The benchmark results:

- Brought concrete evidence that the new calculation system should be adopted;
- Help justifying this choice to the GeoTrek community;
- Allow to pinpoint which areas of the code can be optimized.

The data used for comparison is the elapsed time between the user's action (clicking on a path to add a step) and the completion of the new route display. This execution time can have two components that were measured separately: Python (backend) and JavaScript (frontend). It should be noted that the reported JavaScript time measurements also include, when applicable, the network time related to the communication between the frontend and the backend.

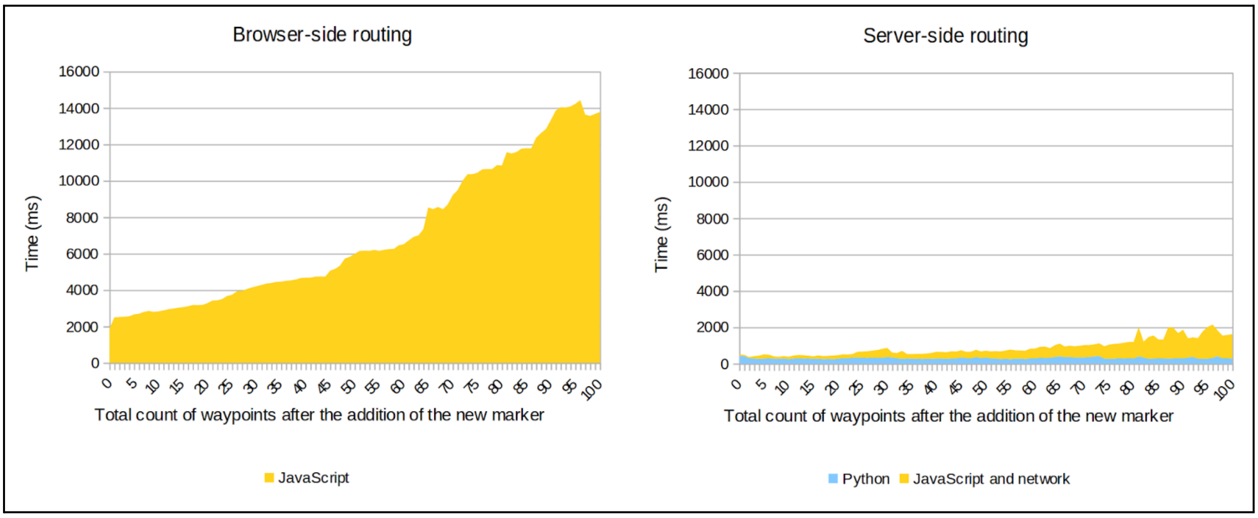


Figure 13: Wait time when adding a marker as a function of the total number of waypoints

The graphs seen on Fig. 13 represent the evolution of wait time between the user's action and the update of the display of the route, throughout the plotting of a 100-waypoints itinerary.

For instance, at the beginning of the itinerary creation, the user places the start marker and the end marker: this corresponds to the abscissa 0 (i.e. 0 waypoint between the start and end markers). Between the addition of the end marker and the display of the route, approximately 2,000 milliseconds elapsed in the case of frontend routing, and about 500 milliseconds for backend routing. The user then adds waypoints, and the same measurement is performed for each marker.

Note that the elapsed times are not cumulative between added markers. For instance, in the case of frontend routing, at the abscissa 30, approximately 4,000 milliseconds elapsed from placing the 30th waypoint to displaying the updated route. This is not the cumulative time for all markers up to that point.

For both routing systems, in order to collect reliable data, the time measurements were automated using Cypress, which enables simulating user interactions, such as clicking buttons and filling out forms. This approach allowed to:

- Use the same list of steps for all measures;
- Use the same database containing an average number of paths for a typical user;
- Perform all measurements 15 times, to compute the average duration for each.

This automation, which facilitates the execution of numerous time-consuming measurements, has been made generic to allow for future reusability. During optimization efforts, this benchmarking system will help future developers compare the pre- and post-optimization performances to ensure that a modification of the code base is effective. This is a topic on which you will be focusing your efforts.

Remaining work

As can be seen thanks to the benchmark report, work remains to be done on optimization: most of the waiting time during the creation of a route is due to operations in the frontend. The progressive increase of this JavaScript execution time is explained by the growing number of sub-routes.

When adding a new waypoint, only the sub-route(s) modified by the new waypoint is/are recalculated, but the entire route must then be redisplayed, sub-route by sub-route. This process should be rethought to keep the display of sub-routes not affected by the modification.

This is only one possible optimization. After it has been implemented, a detailed analysis of the new execution times and of the frontend code will be necessary to identify additional potential improvements.

For each new substantial modification of the frontend code you produce, you will be required to prove that it effectively reduces wait times before it can be merged into the project. To achieve this, you can use the aforementioned benchmarking system which has been developed for this purpose.

During this reworking of the frontend source code, please be aware of a few challenges presented by this Pull Request:

- It is a complex codebase which supports varied functionalities across several modules, only one of which you will be working on. When altering the code, it is crucial to ensure that you are not disrupting a feature in a different module. Make sure you fully understand the purpose of any code you edit or delete, and verify that other modules are still operating correctly following your changes;
- Due to technical constraints, GeoTrek-admin uses a nearly decade-old version of Leaflet. This prevents from being able to use more advanced features, and complicates the search for assistance. When editing the frontend code, make sure to check information on Leaflet for the correct version.

For backend code optimization, the second Pull Request should be prioritized. Indeed, while the first Pull Request simply relocates routing from the frontend to the backend, the second one builds upon it and aims to replace the Python routing system with a more efficient one by using pgRouting.

Optimization of route calculation with pgRouting

Moving the routing system to the backend enables us to replace it with a more efficient solution. To this end, we are exploring various existing solutions in order to identify the most effective one, as they are specifically designed for routing and provide proven and efficient systems.

pgRouting is the first of these solutions to be tested in this process. Pull Request #4254 replaces the Python-based routing system with a database-driven one, aiming to further improve the performances in terms of both speed and data volume capacity. By performing the routing directly in the database where the paths are stored, we eliminate the need to fetch and process them with Python, which can slow down calculation.

Work accomplished

The endpoint created in the previous Pull Request has been retained: this new contribution does not alter the communication process between the frontend and the backend. Instead, it uses the same information sent by the frontend and computes the route, its geometry and its topology using SQL, which is a programming language used to manage databases.

Similarly to the first Pull Request, the use of the Dijkstra algorithm has been retained. This allows for detecting performance differences solely related to the transition from Python to pgRouting. The route generation process is similar to the Python implementation:

1. Localization of each step on the path network;
2. Calculation of each sub-route, now using the Dijkstra implementation by pgRouting;
3. Compilation of all sub-routes' geometries and topologies.

A performance benchmark comparing this new Pull Request to the first one has been completed as well. Unlike the previous benchmark, it only measures execution times at the backend level, as this new Pull Request does not modify the frontend code. This new benchmarking system is similar to the previous one in the following ways:

- We compare the two Pull Requests using the same databases and the same steps to include in the route;
- We conduct the same measurements 15 times to calculate the average execution times.

Focusing solely on the backend times offers several advantages. In the previous benchmarking system, Cypress was used to simulate user interactions with the frontend, which triggered requests to the backend. This approach, although required for the first Pull Request, is slow and resource-intensive. The new system sends requests directly to the backend, bypassing the frontend. This makes measurements significantly faster and allows for benchmarking on a considerably larger database, which the Cypress version could not handle.

Fig. 14 shows the difference in performance between the Python and pgRouting implementations. As can be seen, the Python implementation proves to be more efficient, both on a medium-sized paths network (8,390 paths) and on a very large one (195,095 paths). The time difference appears to increase with the number of paths, as the pgRouting implementation takes 1 second longer than the Python implementation on the medium-sized database, while this time doubles on the large database.

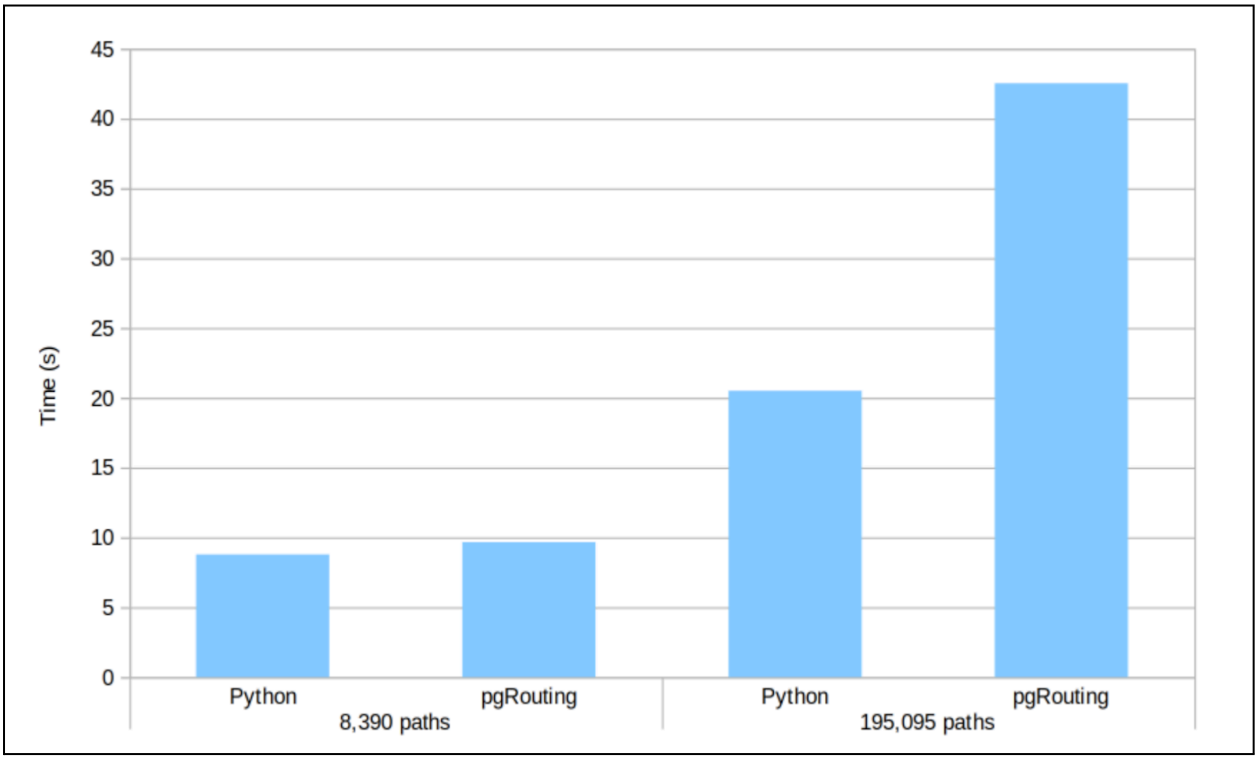


Figure 14: Route calculation times based on the routing system and the number of paths in the database

These results indicate that a basic Dijkstra implementation using pgRouting is not sufficient. Further research is needed to determine if it can be made more efficient through finer tuning.

Remaining work

The upcoming optimization efforts will focus on two fronts: attempting to improve pgRouting's performance and evaluating other routing solutions.

The first strategy to explore in order to shorten execution times is to replace the use of Dijkstra's algorithm with the A* algorithm. It is an extension of Dijkstra's algorithm which prioritizes which paths to explore. While Dijkstra's algorithm explores all possible routes in order to find the shortest one, A* estimates the cost from the current point to the goal. This allows to focus on the most promising paths first, which speeds up the search process. Once A* is implemented thanks to pgRouting, it should also be implemented in Python in order to compare the two versions.

The second approach is to restructure the SQL code that integrates pgRouting by eliminating potential duplicate calculations and simplifying the algorithm.

The last method is to experiment with PostgreSQL settings, which control how efficiently the database handles queries and processes data. The objective is to find an optimized configuration for this task, while ensuring that existing functionalities are not slowed down.

If, following these improvements, the Python version remains more efficient, pgRouting will not be selected to replace it.

Following this, other existing routing solutions should be tested and compared with the Python and/or pgRouting implementations. For all of them, the same process as for pgRouting should be followed: open a new Pull Request, implement the solution, work on optimizing it, and measure execution times using the aforementioned benchmarking system.

Once all the results are gathered, a comprehensive comparison of all these solutions will be available and the best option will be adopted. Since modifying the routing system is a significant shift, the benchmark results will also provide concrete evidence to demonstrate to the GeoTrek community that this change is for the better.

Note that this Pull Request has posed challenges that you will also have to overcome. First, implementing routing within the database requires understanding and modifying significant portions of SQL, while aiming for best possible efficiency. This can be challenging when not familiar with SQL. Second, configuring PostgreSQL for optimal performance involves having a solid understanding of its settings. Extensive research and learning are necessary when starting out with this tool. Lastly, every option that will be benchmarked has drawbacks, so it will be crucial to consider their benefits and downsides when deciding which one to adopt.

Final words

As you embark on this project, your work will play a critical role in enhancing the user experience of GeoTrek-admin. We are confident that your efforts will drive us closer to our goals, and we look forward to your valuable contributions.

SECTION II

Personal strengths and areas for improvement

Foreword

This section addresses one's internship supervisor. It describes one's main strengths and how they helped in contributing to the project, as well as a few areas for improvement that need to be worked on.

Personal strengths and areas for improvement

Joaquim,

My internship is coming to an end, and it is time to reflect on my performance during my time at Makina Corpus. I believe a few of my strengths have truly helped me make a difference, while there are areas in which I wish to improve to become an even better developer. I would like to share all of this with you.

Throughout this internship, I have strived to put my key strengths to use so as to deliver work of the highest quality I could.

In my opinion, the most significant one has been my effort to be proactive. This has led to a better user experience, even though my initial task was simply to move an existing background calculation from the user's browser to the server. Indeed, I researched and suggested efficient solutions and improvements, such as ways of reducing wait times and adding error management with user guidance when given incorrect inputs. When time came to make a benchmark to determine which improvements or losses are introduced by my work, I pushed for implementing an automated benchmarking system, which will be reusable when further optimizations are carried out and which I hope will continue to facilitate the search for the most efficient tools and implementations.

Additionally, I believe my ability to quickly learn new technologies and frameworks has been a notable advantage. Indeed, many of the technologies I used at Makina Corpus were new to me, and being able to understand them quickly gave me time to implement more than planned for the routing system, as I mentioned above.

To finish on this topic, I would like to highlight my analytical skills which have enabled me to work more effectively and efficiently. I was able to decipher and navigate a complex codebase that was developed more than a decade ago with tools I was still learning. Understanding the intricacies of this codebase ensured that I found optimal solutions while not interfering with unrelated functionalities, which is a crucial aspect of development.

While I am proud of my achievements, I also recognize areas where I can continue to grow in order to make my contributions to projects even more significant.

The first that comes to mind is my confidence in communication. Throughout my internship, I pushed myself to share ideas and engage in discussions and I am aware of the need to pursue my efforts in building this self-assurance. This improvement is crucial for my professional development and future interactions with recruiters, colleagues and clients. Furthermore, building confidence in my ideas and opinions will help collaborative efforts by improving my ability to contribute to brainstorming sessions and strategic decisions.

Another aspect I intend to improve is technology monitoring, as staying informed about the latest innovations and best practices is essential to becoming a top-tier developer. While I do make an effort to read industry news and announcements about new software features and changes whenever I encounter them, I need to proactively seek out this information to ensure I am up to date with development trends and advancements. A broader understanding of the technological landscape would help me make better strategic decisions and ultimately enhance the quality and robustness of projects I contribute to.

The last area of improvement I would like to mention is my knowledge of system administration. Gaining a deeper understanding of how to set up and maintain the computer systems I use daily would be invaluable in increasing my productivity and the quality of my work. While I am steadily making progress in this field through daily development, I aim to take the lead and expand my knowledge in order to enhance my expertise. Indeed, I would improve at troubleshooting issues by getting a better grasp of how the system environment affects the execution of the code I produce. Additionally, I would be better equipped to prevent these issues by making more informed decisions regarding technology choices and implementation. To finish, I would be able to configure tools more effectively to tailor them to my development habits and preferences.

To conclude, I would like to thank you once again for the time and effort you have invested in helping me develop my skills. With these strengths and my dedication to improving in the areas I mentioned above, I am confident that I am well-prepared to begin my career on the right path.

SECTION III

To the CEO

Foreword

In this section, one addresses their company's CEO in an effort to be put in charge of one's dream project. This argument should include a description of the project, a demonstration of one's ability to plan and manage it, and an explanation of how it would benefit the company.

To the CEO

<u>From:</u> justine.fricou@makina-corpus.com

<u>To:</u> jean-pierre.oliva@makina-corpus.com

<u>Subject:</u> New project idea for Makina Corpus

Dear Mr. Oliva,

I am writing to propose a project I believe would significantly benefit Makina Corpus and the community. I think you should consider this opportunity and I would be honored to be given the responsibility of initiating and leading this project.

The product would be a website and mobile application connecting consumers with local farmers and producers.

The platform would help users locate selling points such as farm stands, cooperative markets, and farmer's markets. It would provide essential information such as the location and operating hours, as well as what type of produce is available at the moment. The sellers would have the option to allow customers to order online before coming to pick up produce.

Makina Corpus has strong social and environmental values, and we strive to make a positive impact through our work. This project is perfectly aligned with them, as we could promote sustainable agriculture, support small-scale farmers and producers, and facilitate access to local and fresh products.

Additionally, such a platform would allow us to reach a new user base and increase the visibility of our company and its commitments to the community.

Makina Corpus specializes in developing cartography-based projects: we have the skills and the tools to build an efficient and attractive platform. In fact, we are currently working on similar projects, namely GeoTrek and GeoRivière, which offer such map-based interfaces. We could go as far as implementing a gateway to the GeoTrek project to allow territories to promote agritourism and enhance tourists' experiences through the discovery of local specialties.

Having grown up in a rural area surrounded by farmers, this cause is particularly important to me. Such experience and investment are invaluable, which is why I believe I should lead this project.

Over the past 5 years of my training at Epitech, I have been continuously working on development projects, building both technical and organizational skills. This culminated into my end-of-studies project for which I assumed the role of a manager for a team of 11 students. During the 2.5-year development phase, I have handled tasks ranging from sprint planning and running agile rituals, to project presentations, to framing and monitoring a beta test phase. Through this experience and numerous project-leading courses, I have developed project coordination and interpersonal skills essential to orchestrating a team.

Furthermore, prior experience as a developer is a significant advantage when leading a team of developers. This is particularly relevant given that at the end of my internship, I will have spent six months working on the GeoTrek project, which is similar to the product I am proposing.

I am confident that by addressing this unmet need of the market, we would strengthen our company's standing and bring a positive social outcome. I look forward to discussing, at your earliest convenience, how we can bring this idea to reality together.

Sincerely,

Justine Fricou
Web development intern
Makina Corpus
06.06.06.06.06