Epitech 3rd year

Internship report

By Romain Rhyn

Year 2017/2018

In order to fulfil my third year at the computer science school of Epitech, I had to find an internship in an IT company.

The one I choose lasted from 03/04/2018 to 24/08/2018, which means that when I will end to write this report, my internship will not be completely fulfilled. My mission was to create a back-office website for the company LiveStep. This document aims to explain all the details of this internship.

The company: LiveStep

LiveStep is a start-up created the January 11th, 2018 at Rennes. Its goal is to design, produce and sell a solution to monitor and solve the problem of falls of weaker people, like elders and convalescent people.

Indeed, the statistics are quite revealing:

- For the people older than 65, falls are the first cause of death. This concerns 18% of the French population, which correspond to around 12 millions of people.
- Every year in France, around 400.000 elderly people falls by accident, and nearly 12.000 of them dies from it.
- Falling once multiplies the risk of falling again in the same year by 12.
- Less than half of the falls are reported to the doctors.

When you add to all that the fact that when you fall you often feel weak and old, and you're afraid to fall again, which causes a limitation of activities, a lower quality of life, social isolation and sometimes even depression, a quite clear conclusion appears: falls are a major health issue.

To help dealing with this problem, LiveStep offers a solution that takes the form of a connected insole which gather information all the time and send an alert in case of trouble to a related website. It is invisible to the patient, it can be used in any shoe or slipper and it is useful for both the relatives of the patient and the medical team in charge of him.

The team

For all the duration of my internship, I worked with the two founders of the society: Romain Berrada and Franck Le Dortz.

Mr Berrada (on the right) is responsible of all the electronic part of the society: he manipulates the insoles prototypes and manages the code related to it. He also was my internship supervisor and helped me to understand how to use the tools the company already had when I arrived.

Mr Le Dortz (on the left) is responsible of the company designs: for the websites, for the insoles, for the ad campaigns and so on. He also helped me to design the visuals and the UI/UX of the website.

The project

When I arrived in the company, the tests on the insole prototype were about to start. They already had a first version of a front-office website that could be used by their clients, and they needed a back-office platform to gather information about the insoles themselves and the technical data.

The front-office website shows only the medical data, and only the data of the clients inside the medical facility which the logged user is related to. Indeed, some of the data are considered sensitive, so it's important to be careful with their security.

We decided to make, like in the front-office version, a strict separation between the server side and the client side. It's been decided so because of the legislation concerning medical data, and we had to make sure that those data are as secured as possible.

For the most part, they let me the choice of the technology I want to use as long as the result is working and can be understood and modified by other people in case of need, like when an internal algorithm of the insoles changes or if a new information need to be caught.

However, they suggested me some useful technologies I could use to start with, along which were the ones used for the front-office website.

The result

First of all, I had to use NodeJS as basis of the website, which was the technology already in use for the front-office website.

NodeJS is an open-source and cross-platform JavaScript run-time environment that is used to create server-side JavaScript code. To manage the dependencies of the project, we're using npm, which is the official package manager for NodeJS and offers ways to properly deal with dependencies. The dependencies we used for project will be described later in this document.

After about a week of benchmarking on some technologies we decided which ones to use: Angular, along with Kendo UI.

Angular is a free and open-source framework that can be used on every platform to develop websites and applications. It's using the MVC logic (Model-View-Controller), which implies a very segmented code, and gives a large and easy to use panel of way to test every part of a project. Most of the code uses, apart from HTML and CSS, a language named TypeScript, which is a superset of JavaScript that adds optional static typing to the language. It also was the framework used for the front-office website.

Kendo UI is a UI toolkit for web development. It can be used with a lot of different frameworks, including Angular. However, the Angular version is quite new, so using it can be a bit complicated, since in case of an unlisted problem, there isn't always that much of help you can get from the company who created it.

To manage the versioning of the project, we used Git and Bitbucket, which were the tools already used in the company.

Git is a version control system for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for source code management in software development, but it can be used to keep track of changes in any set of files.

The main difficulty was to learn how to use these technologies, which were new to me, except for the versioning tools. Indeed, the installations of the dependencies, and most of all the structure of the code were not that easy to understand. Luckily, most of them are quite well documented, and with some effort and some time, things became more and more easy. Plus, as I already said, Mr Berrada helped me to get through this part, which made it easier to understand even for the parts that weren't that well documented.

It took some time to put the first pieces of code in the website, but now it is way easier to add new ones to it. Indeed, Kendo and Angular offer a very simple way to create new components and automatically adds it to the already existing project, and when you get the fundamental logic of them, using those parts becomes quite easy.

All the completed aspects of the website include testing modules, which permit to verify whether a change made in the code is working the way we want it or not without the server or the website themselves, giving a quick, clear and adaptable answer.

The server part is fully made and secured. Part of it is made based on the front-side website server. It's running non-stop on a Linux system and is only dealing with data, so the website can communicate with the database at any time.

The front part is partly made, some of the pages are functional but the visuals are not completed yet. All the website is secured by a login module, which allows only the qualified members of LiveStep to access the concerning data. The monitoring data are available and can be seen on the website, with the possibility to filter and sort all data. The other pages of the website are using the same structure, only the shown data are different.

To make the website I used the following dependencies:

- Chai, Mocha, Karma, Jasmine, Protractor and Supertest, which are used for the automated testing: Chai and Mocha for JavaScript, Supertest for HTTP.

- Express, used for building the API.
- JsonWebToken, MySql and Uuid, three dependencies that generates the named elements and allowed to create the website and the separation between server and front.
- TSLint, an extensible static analysis tool that checks TypeScript code for readability, maintainability, and functionality errors.
- Codelyzer, a set of TSLint rules for static code analysis.

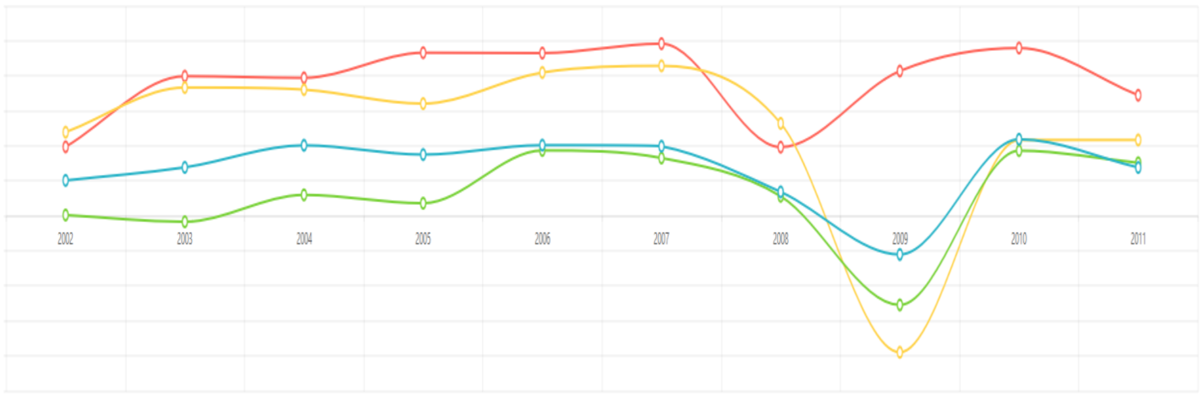
Proposition for a new internship

Even though the website is functional, there are some updates that can be make. Here are the ones I propose.

Adding new things to the database can be made only through the database itself. It could be interesting and more convenient to create a way to add them through the website itself. We could even imagine a way to make modifications to them, once again through the website related pages. Kendo UI offers tools that can help us to implement those functionalities, even though we'll have to adapt them to make it work with our website.

I'll had to that another option: a graphic representation of a product evolution, which could make even clearer a quite useful information. There are a lot of tools to make those graphs, and it could permit the company to be more reactive about all those concerned data, which are crucial to the company since they are needed to assure a good customer service. Plus, Kendo and Angular allows to deal with the data dynamically.

This graph could look something like that:



All this can be added and changed with the Kendo UI tools I've put on the project, and since I already know how to use them I could make this work without any problem.