INTERNSHIP REPORT

Software Engineer Intern

[Tech 5]
Joseph PERENIGUEZ
pereni_j
Company: Hublo
Internship period: 2020-06-01 / 2020-11-27

Table of content

General Information

Student information

Name: Joseph PERENIGUEZ
Login: pereni_j
Promotion: 2020

School information

Name: Epitech Paris
Location: 24 Rue Pasteur, 94270 Le Kremlin Bicêtre

Company information

Name: Hublo
Location: 4 Rue René Villermé, 75011 Paris
SIRET: 82227698600046
APE Code: 6311Z
CEO Antoine Loron
Co-founder: Adrien Beata
CTO Chris Rydahl

Internship report [Part 1]

A part intended for a newcomer in the company who will take over / continue the project in which you have been involved. It is therefore necessary to describe the context of the company, the project, its general architecture, the context and the organisation of the work team, all in a synthetic way. You should also provide details of any difficulties encountered in the project.

Acronyms

- **API**: Application programming interface
- **CDD**: Fixed-term contract
- **CEO**: Chief Executive Officer
- **COO** : Chief Operating Officer
- **CSS**: Cascading Style Sheets
- **CTO**: Chief Technical Officer
- **CV**: Curriculum vitae
- **DOM**: Document Object Model
- **DPAE**: Pre-recruitment Declaration
- **ECMA**: European Computer Manufacturers Association
- **EHPAD**: Residential institution for dependent elderly persons
- **ES**: ECMAScript
- **GTA**: Time and Activity Management
- **HTML** : Hypertext Markup Language
- **FDI** : Integrated Development Environment
- **JS** : JavaScript
- **JSX**: JavaScript XML
- **MVC**: Model View Controller
- **Mn**: Minutes
- **ORM**: Object Relational lapping
- **PM**: Product Manager
- **POO** : Object-oriented programming
- **RDBMS**: Relational DataBase Management System
- **HRIS**: Human Resources Management Information System
- **SMS**: Software as a Service
- **SQL** : Structured Query Language
- **SaaS**: Short Message Service
- **TS**: TypeScript
- **UML** : Unified Modeling Language
- **URSSAF**:Union for the Collection of Social Security Contributions and Family Allowances
- **XML** : Extensible Markup Language

1. General presentation

1.1. Presentation of the host organisation

At a time when an unprecedented economic crisis is affecting French companies, the start-ups **Whoog** and **medGo**, two major players in the management of replacements, are merging their activities under the same entity: Hublo. The objective of this merger: to join forces to serve a common mission: to improve the daily lives of healthcare professionals and meet the HR challenges of tomorrow's hospitals.

Whoog is an innovative start-up located in Sophia Antipolis, which has developed the 1st digital solution for managing replacements based on staff volunteering, in partnership with Montpellier University Hospital. Whoog has subsequently established partnerships with 19 CHU hospitals. Whoog won the Innovation Trophy at the Salon de la Santé et de l'Autonomie for the management of healthcare in the region in 2014 and received the Entrepreneur of the Year Award in the health category in 2017.

medGo is a solution for managing staff absences and replacements, available since February 2017. The start-up has convinced more than 1300 hospitals, clinics and EHPADs. Its customers include the Hospices Civils de Lyon, more than 20 Groupements Hospitaliers de Territoire (GHT), the private groups ELSAN, Ramsay Santé, Vivalto, Korian... At the end of 2017, the company raised funds from BPI France, Xavier Niel and several managers of healthcare institutions and digital companies.

Although this strategic alliance had been discussed before the Covid-19 crisis, the latter accelerated this rapprochement to work together to improve the quality of work of healthcare professionals. During the crisis, Whoog and medgo made it possible to mobilise more than 60,000 volunteer professionals in 2 months. Better HR management is one of the pillars of tomorrow's hospital. Hublo responds both to the problems of the establishments (absenteeism, leave, etc.) and to the needs of healthcare professionals (better management of their hours, right to disconnection, etc.).

For the past 5 years, the Whoog and medGo teams have been working towards the same mission: that of changing the daily lives of millions of healthcare professionals, with simple digital tools to enable them to refocus on their main mission, care.

Today, they are joining forces to pursue their common mission. By merging to become "Hublo", the two start-ups are investing their research and technical development efforts in the service of the same technology to offer a platform that is twice as innovative. They are also combining the expertise they have acquired in the private and public sectors to support health and medico-social institutions as closely as possible to their needs.

Another challenge of this rapprochement is to solidify and innovate in the long term. The economic context resulting from the health crisis has highlighted the interest for Whoog and medGo to merge in order to financially strengthen the new entity and accelerate the international deployment of the solution. "We have been thinking about this merger for several months now, and the current crisis affecting all French companies has reinforced our conviction that it was more than ever time to join forces and look to the future with confidence. "explains Antoine Loron, co-founder of medGo and president of Hublo.

Today, the Hublo solution is determined to meet the needs of tomorrow's hospital, and the expectations of new generations of carers: reactivity, mobility, quality of life at work…

1.2. Hublo's team

In order to be able to offer its services to its clients, Hublo has 60 employees; engineers and specialists divided into several teams specialised in particular activities:

Technical Team: This is the team responsible for defining the roadmap of Hublo's products based on customer feedback and statistics provided by the Data team. It is also responsible for preparing mock-ups of new features or those to be modified and for preparing test protocols for high-risk cards.

Product Team : This is the team responsible for defining the product roadmap by transforming customer requirements, customer feedback or proposed product

improvements into specific tasks which will then be estimated, prioritised and finally assigned to specialist teams.

Data Team: This is the team responsible for the generation of the various reports either to Hublo or to its customers, which will then be used to define or modify the product roadmap.

Sales Team: This is the team responsible for finding new establishments and convincing them to join Hublo by giving demonstrations or providing test access. This team is also responsible for invoicing our existing customers.

Marketing Team: This is the team responsible for Hublo's marketing campaign and also manages Hublo's presence on social networks. It also manages its participation in events, especially in the medical field.

Customer Success Team: This is the team responsible for saving and restoring the direct contact with customers after delivering the solution to them. This team is also responsible for making appointments for follow-up or product parameterisation and for providing training on how to use the product and how to make better use of it.

Support Team: This is the team responsible for helping healthcare institutions and replacements to use the Hublo application in case of a specific request or in case of a problem with the product.

Administrative Team: This is the team responsible for managing the Hublo team, office life and all necessary administrative procedures.

1.3. Hublo's products

1.3.1. The Hublo solution

As seen in the general introduction, the management of replacements is currently a time-consuming process and is done in a classic way: the manager calls all the qualified people he knows who are likely to take the assignment until he finds someone. Otherwise, he will go through a temporary employment agency which provides a replacement from its network of health professionals and bills the institution directly.

To solve these problems, the startup Hublo created the **Hublo solution**, which is a 100% digital solution (a website, an Android mobile application and another iOS). Indeed, Hublo is a SaaS platform allowing each health care institution to manage and solicit its network of replacements such as nurses, care assistants..., in all simplicity and which aims to save a considerable amount of time for these institutions.

With the arrival of Hublo, the replacement process has become much simpler:

- **Step 1**: The establishment invites its agents and its network of volunteer contract workers to register on the Hublo platform with the establishment code.
- **Step 2**: If necessary, a health executive posts a mission on the platform. A mission is simply a need for replacement.
- **Step 3**: Available replacements are instantly notified by SMS, email and push notification and can apply either by SMS or via the Hublo application.
- **Step 4**: The assignment is either automatically assigned to the first available replacement if the assignment is urgent, or assigned by the executive to one of the replacements who has applied, and the other candidates will be notified that their application has been unsuccessful.

This simple but efficient process allows Hublo to fill 76% of assignments in less than two hours.

This new process has enabled institutions to:

- To reduce the time required to manage replacements for the various players (executives, human resources managers and on-call administrators) by enabling them to save 43mn/day on average (90mn/day maximum).
- To filter the requests to the replacements by offering them only the missions that are relevant to them (based on their skills and availability).
- To reduce the use of temporary workers, thus increasing the money earned and reducing the risks of replacements who do not have the necessary skills.

In addition to the replacement management functionality, Hublo offers healthcare institutions additional modules such as:

- Organization of the planning of replacements by directly assigning them assignments based on their skills, availability and number of hours worked per

week/month in order to avoid breaking the law with overtime beyond the permitted thresholds.
- The automatic generation of fixed-term contracts/vacations according to the specific contract frames at the establishment.
- Automation of the DPAE at URSSAF.
- Interfacing and integration with ATM and HRIS software.
- Interfacing with temporary employment agencies to facilitate communication between health care institutions and their agencies.

1.3.2. The PlanBlanc solution

The PlanBlanc solution is a digital solution intended for health establishments for an immediate mobilisation of their network of agents and to organise the reception and care of a huge flow of people in case of a natural disaster, an epidemic or a huge accident. This solution enables precise management of communication in these crisis situations while maintaining continuity of care for patients already hospitalised.

It is also used to train staff in these emergencies by giving them false white plans to improve their reactivity and organisation.

The use of the PlanBlanc is too simple:

- Step 1: Import the database of all staff on a secure server, and train the different dissemination groups according to the needs of the institution.
- Step 2: Alert all staff to the crisis situation by preparing an alert message in a few seconds and select the relevant broadcast group(s). With the possibility of dialoguing with the people alerted throughout the white plan.
- Step 3: Monitor the progress of the mobilisation from a dashboard with the possibility of communicating any necessary adjustments.

The PlanBlanc solution is currently in beta version and has been deployed in a few establishments in order to test it before starting to market the product and to see the modifications that can be made to improve the product in order to meet customers' expectations.

1.4. Hublo customers

Today, Hublo has more than 1,000 customers, including public hospitals, private clinics, EHPADs and university hospitals.

Some of Hublo's largest clients include :

- The French Croix-Rouge.
- The Ramsay Générale de Santée group.
- The HCL group.
- The GHEF group.
- The ELSAN group.

1.5. Overview of the project

At Hublo, the technical team is not only present to quickly develop what the team produces their request to do. But it always tries to improve the project on the technical side, either by proposing new architectures, adopting new technologies, half granting to others or by setting up new rules/conventions that serve to properly structure the project and organise the work.

2. Theoretical background, state of the art and issues

2.1. Introduction

The aim of this chapter is to present some key concepts necessary to understand our project, then we will describe the Hublo technology stack and a study of the existing solution, its issues and end with the proposed solution.

2.2. Theoretical background

As promised, this section will present some of the theoretical concepts needed to understand the project.

2.2.1. JavaScript

JavaScript, often abbreviated to JS, is an interpreted programming language, which follows the ECMAScript specification. Together with HTML and CSS, JavaScript is one of the basic technologies of the World Wide Web since it allows web pages to be interactive. It is used to add animations to a web page, add dynamic behaviour, store information and manage requests and responses on a website.

Although it is best known as a scripting language for web pages, many environments other than browsers also use it:

- Node.js as a backend server
- Apache CouchDB as a database server

Historically, Javascript was created when Brendan Eich was tasked with developing a language that resembled Java for the Web for the Netscape browser. Eich, however, decided that Java was too complicated with all its rules and therefore decided to create a simpler language that even a beginner could code.

Once the language was finished, the Netscape marketing team asked Sun Microsystems (the creator of Java) to allow them to name JavaScript as a marketing stunt since Java was too well known at the time. That's why most people who have never used Ja- vaScript think it's related to Java.

After JavaScript was published in the browser, Microsoft and Adobe took the language and started creating their own implementations such as JScript and ActionScript respectively.

Before Netscape disappeared, due to the fact that Internet Explorer dominated the web browser market, they decided to create a standard that would guide the path of JavaScript, called ECMAScript, which will be detailed in the following paragraph.

2.2.2. ECMAScript

As mentioned earlier, Netscape submitted JavaScript to ECMA International for standardisation (ECMA is a European Association for the Standardisation of Education and Communication Systems), and Microsoft agreed with ECMA International and this resulted in a new language standard called ECMAScript. This name was a compromise in part between Netscape and Microsoft.

It is worth mentioning that ECMAScript is only a part of JavaScript that specifies its main syntax, types, objects, etc. It is also a part of the JavaScript that specifies its main syntax. (for example, it does not normalize DOM Api). Today JavaScript is the best implementation of this standard compared to JScript and ActionScript.

The first edition of ECMAScript appeared in June 1997. Several editions appeared afterwards including new features.

The initial editions of ECMAScript are named numerically, increasing by 1: ES1, ES2, ES3, ES4, ES5, while the new editions (from 2015 onwards) will be named ES, followed by year of publication: ES2015, ES2016, ES2017, ES2018.

In the following section, we will present the latest editions of ECMAScript as well as some of the functionalities brought by them and their implementations in JavaScript, and we will end with a comparison table of browser support for these editions.

2.2.2.1. ES5

- "strict mode" added

Strict mode" is a new feature of ES5 that allows us to place a programme or function in a "strict" operating context. This strict context prevents certain actions from being executed and generates more exceptions.

For example, in JavaScript without "strict mode", a typing error in the name of a variable creates a new global variable. Whereas with "strict mode", this will generate an error making it impossible to accidentally create a global variable.

```
"use strict";

mistypedVariableName = 5; // Uncaught ReferenceError: mistypedVariableName is not defined
```

ES5: Strict mode: Error when trying to assign a value to a variable when entering its name incorrectly

In JavaScript without "strict mode", a developer will not receive any error returns assigning values to properties that are not writable, whereas in "strict mode" he will receive an error.

```
"use strict";

var obj1 = {};
Object.defineProperty(obj1, 'x', { value: 42, writable: false });
obj1.x = 9; // TypeError: Cannot assign to read only property 'x'
```

ES5: Strict mode: Error when trying to assign a value to a variable when typing its name incorrectly

In JavaScript without "strict mode", a function can have several parameters with identical names while "strict mode" does not allow this.

```
"use strict";

function sum(a, a, c) { // Uncaught SyntaxError: Duplicate parameter name not allowed in this context
  return a + a + c;
}
```

ES5: Strict mode: An error when trying to use parameter names for a function that are not unique.

- String.trim() added

The trim() method removes spaces on both sides of a string and returns a new variable with the result.

- Array.isArray() added

The Array.isArray() method determines whether the passed value is an array or not.

- Array.prototype.map() added

The map() method creates a new table with the results of calling an element of the called table.

```
var fahrenheit = [0, 32, 45, 50, 75, 80, 99, 120];

var celcius = fahrenheit.map(function(elem) {
    return Math.round((elem - 32) * 5 / 9);
});

console.log(celcius); // [-18, 0, 7, 10, 24, 27, 37, 49]
```

ES5 : Array.prototype.map()

- Array.prototype.reduce() added

The reduce() method uses each element of the array to generate a single output value.

```
var fahrenheit = [0, 32, 45, 32, 75, 0, 99, 120];

var sum = fahrenheit.reduce(function(prevVal, elem) {
    return prevVal + elem;
}, 0);

console.log(sum); // 403
```

ES5: Array.prototype.reduce()

- Array.prototype.filter() added

The filter() method creates a new array with all the elements that pass the test implemented by the provided function.

```
var fahrenheit = [0, 32, 45, 32, 75, 0, 99, 120];

var uniqueArray = fahrenheit.filter(function(elem, index, array) {
        return array.indexOf(elem) === index;
    }
);

console.log(uniqueArray); // [0, 32, 45, 75, 99, 120]
```

ES5 : Array.prototype.filter()

2.2.2.2. ES2015

- Array.find() added

The find() method returns the value of the first element of the array that satisfies the test function provided. Otherwise, 'undefined' is returned.

- Array.findIndex() added

The findIndex() method returns the index of the first element of the array that satisfies the provided test function. Otherwise, it returns -1, indicating that no element has passed the test.

- let and const added

let' indicates that the variable can be reassigned (like a counter in a loop or a value exchange in an algorithm). It also indicates that the variable will only be used in the block in which it is defined, which is not always the entire function containing it, as opposed to 'var' which indicates that its use is independent of the block in which it is defined.

const' indicates that the variable will no longer be reassignable.

- Default parameter values added

The default function settings allow settings to be initialized with default values if no value or "undefined" is transmitted.

```
1  function multiply(a, b = 1) {
2    return a * b;
3  }
4
5  console.log(multiply(5, 2)); // Résultat attendu: 10
6
7  console.log(multiply(5)); // Résultat attendu: 5
```

ES2015 : Default parameter values

- Spread property added

The Spread property allows you to expand an iterable such as an array, object or string at locations where zero or more arguments are expected.

```
1  function sum(x, y, z) {
2    return x + y + z;
3  }
4
5  const numbers = [1, 2, 3];
6
7  console.log(sum(...numbers)); // Résultat attendue: 6
```

ES2015 : Spread property

- Syntax Rest added

The syntax of the parameter rest allows us to represent an indefinite number of arguments in the form of an array.

```
1  function sum(...theArgs) {
2    return theArgs.reduce((previous, current) => {
3      return previous + current;
4    });
5  }
6
7  console.log(sum(1, 2, 3)); // Résultat attendu: 6
```

ES2015: Syntax Rest

- The Classes

A class in terms of OOP is a model for the creation of objects. A class encapsulates the data for the object.

```
class Polygon {
    constructor(height, width) {
        this.h = height;
        this.w = width;
    }
    print() {
        console.log("The height of the polygon is", this.h, "while it's width is", this.w)
    }
}

var polyObj = new Polygon(10,20);
polyObj.print(); // Résultat attendue: The height of the polygon is 10 while it's width is 20
```

ES2015 : The classes

- Deconstruction

The decompression assignment syntax is an expression that allows you to decompress table values, or object properties, into separate variables.

```
let [a, b, ...rest] = [10, 20, 30, 40, 50];

console.log(a); // Résultat attendue: 10
console.log(b); // Résultat attendue: 20
console.log(rest); // Résultat attendue: [30, 40, 50]
```

ES2015: Deconstruction

- The arrow functions

Arrow functions make our code more concise and simplify the por- tion of functions and the keyword this. Usually these are one-line mini-functions that work a lot like Lambdas in other languages such as C# or Python.

```
// ES5
var multiplyES5 = function(x, y) {
  return x * y;
};

// ES6
const multiplyES6 = (x, y) => { return x * y };
```

ES2015 : The arrow functions

2.2.2.3. ES2016

- Exponentiation operator added (**)

** is an exponentiation operator.

```
1  console.log(Math.pow(2, 3)); // 9
2
3  console.log(3 ** 2); // 9
```

ES2016 : Exponentiation operator

- Array.prototype.includes added

The includes() method determines whether an array includes a certain value among its entries, returning true or false as appropriate.

- String padding added

The padStart() and padEnd() methods supplement the current chain with another chain (several times, if necessary) until the resulting chain reaches the specified length. The padding is applied from the beginning (left) or end (right) of the current chain, respectively.

```
1  console.log('5581'.padStart(10, '*')); // ******5581
2
3  console.log('5581'.padEnd(10, '*')); // 5581******
```

ES2017: String padding

- Async/Await added

The keyword async defines an asynchronous function. An asynchronous function is a function that operates asynchronously via the event loop, using an implicit promise to return its result. But the syntax and structure of the code using asynchronous functions are much more like synchronous standard functions. To wait for the result of an async function the keyword await must be used.

```
function resolveAfter2Seconds() {
  return new Promise(resolve => {
    setTimeout(() => {
      resolve('resolved');
    }, 2000);
  });
}

async function asyncCall() {
  var result = await resolveAfter2Seconds();
  console.log(result); // Résultat attendu: 'resolved'
}

asyncCall();
```

ES2017: Async/Await

2.2.2.4. ES2017

- Promise.finally() added

The finally() method provides a way for the code to be executed if the promise has been completed either successfully or unsuccessfully by writing it twice in the try and catch blocks.

- Addition to RegExp

In JavaScript, the successful matching of a regular expression with a string of characters returns a result object. Putting a fragment of the regular expression in brackets transforms this fragment into a capture group: the part of the string to which it corresponds is stored in result.

Before this proposal, all capture groups were accessible by number, which is its index in the string.

```
let re = /(?<year>\d{4})-(?<month>\d{2})-(?<day>\d{2})
let result = re.exec('2015-01-02');

console.log(result[0]); // Résultat attendue: '2015'
console.log(result[1]); // Résultat attendue: '01'
console.log(result[2]); // Résultat attendue: '02'
```

ES2018 : Regular expressions: Numbered groups

These numbered catch groups have several drawbacks :

- Finding the number of a capture group is an arduous task: You have to count the paren- thesis.
- You have to see the regular expression if you want to understand what the groups are for.
- If you change the order of the capture groups, you should also change the corresponding code.

To remedy all these problems, ES2018 has introduced the named capture groups.

```
let re = /(?<year>\d{4})-(?<month>\d{2})-(?<day>\d{2})/;
let result = re.exec('2015-01-02');

console.log(result.groups.year); // Résultat attendue: '2015'
console.log(result.groups.month); // Résultat attendue: '01'
console.log(result.groups.day); // Résultat attendue: '02'
```

ES2018 : Regular expressions: The named groups

Now groups are named, regular expression is more understandable and access to these groups is more efficient. In addition, the named groups are accessible in an ex- regular pressure, which helps us not to rewrite the same model if we want to match the same group again.

```
let duplicate = /^(?<half>.*).\k<half>$/u;

console.log(duplicate.test('a*b')); // Résultat attendue: false
console.log(duplicate.test('a*a')); // Résultat attendue: true
```

ES2018 : Regular expressions: Back references

2.2.2.5. ES2018

- Asynchronous Iteration

This is an extremely useful feature. Basically, it allows us to easily create asynchronous code loops.

This feature adds a new "for-await-of" loop that allows us to call asynchronous functions that return promises (or arrays containing many promises) in a loop. The good thing is that the loop waits for each promise to be resolved before moving on to the next loop.

```
const promises = [
    new Promise((resolve, reject) => setTimeout(() => resolve(1), 2000)),
    new Promise((resolve, reject) => setTimeout(() => resolve(2), 1000)),
    new Promise((resolve, reject) => setTimeout(() => resolve(3), 500))
];

async function test() {
    for await (const promise of promises) {
        console.log(promise);
    }
}

test(); // Résultat attendu aprés 3.5 secondes: 1, 2, 3
```

ES2018: Asynchronous Iteration

2.2.2.6. Browser support

As each browser implements its own JavaScript execution engine (for example V8 for Google Chrome or SpiderMonkey for Mozilla Firefox or others), a JavaScript execution engine is a program that converts JavaScript code into machine code, the support of the different ECMAScript standards remains in the hands of the owners of these engines which leads to functionalities that are implemented in browsers and not in others.

2.2.3. TypeScript

TypeScript is a strict syntactic superset of JavaScript, which adds fa- cultive static typing to the language and provides expected functionality of future JavaScript editions to current JavaScript engines.

Since types are completely optional, any .js file can be renamed to a .ts file and TypeScript will still give us a valid .js file equivalent to the original Ja- vaScript file. And this is in order to facilitate the migration of JavaScript code to TypeScript. Even in case of a compilation error like this piece of code

```
1  var foo = 123;
2  foo = '456'; // Error: cannot assign a `string` to a `number`
```

TypeScript code

TypeScript to generate errors but did not prevent the generation of a valid corresponding JavaScript code.

```
1  var foo = 123;
2  foo = '456';
```

Generated JavaScript code

2.2.4. Redux

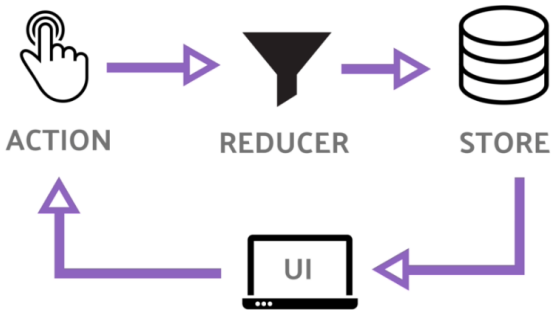
Redux is a state manager. Although it is mainly used with React, it can be used with any JavaScript framework or library.

In React, each component has its own state (the state is an object containing private variables controlled only by the component itself). For other components to be able to use this state one should either inherit it from the component or pass it as properties to the child components until it is needed. Let's imagine the case where several components need the state of another one including or if the brother component needs the state of its brother, it would be very difficult to manage.

With Redux, the state of the component can be stored in a "store" and each component that needs this state can access it via this "store". Each component will therefore be able to subscribe to changes in a store to receive data that has been updated by anyone.

In order for the "blind" to be modified, a component must call an "action" which allows a reducer to change the state in the "blind" and finally return a new state which will be transmitted to all components that are subscribed to this "blind". [22]

The figure below shows what has just been said in a clear diagram.



Operation of Redux

2.3. The Hublo technology stack

In this section we will present the technology stack used at Hublo.

2.3.1. Programming language

The programming language used at Hublo is JavaScript following the ECMAScript ES5 specification. JavaScript has a big advantage for Hublo as developers do not need to know 2 programming languages to work on the backend and frontend.

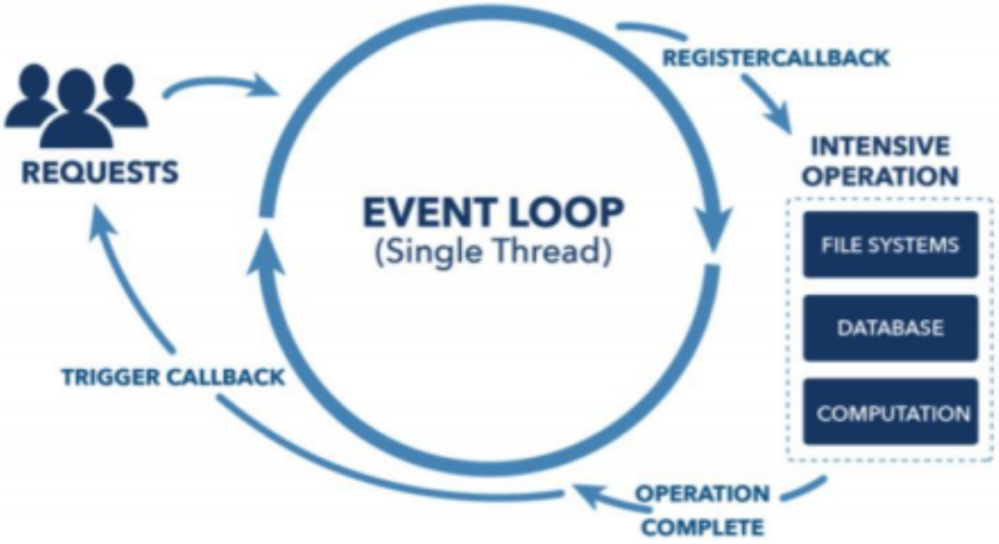
2.3.2. Backend

2.3.2.1. Node.js

Node.js is a cross-platform runtime environment that allows us to run JavaScript outside the browser. Node.js is built on the V8 JavaScript engine, the runtime engine of Google Chrome, which allows it to be very powerful.

A Node.js application is executed in a single process, without creating a new thread for each request. When Node.js needs to perform an input-output operation, such as reading from the bucket, accessing a database or the file system, instead of blocking the thread and wasting waiting processor cycles, Node.js resumes operations when the response is returned.

This is called Event Loop and is shown in the figure below.



Node.js Event Loop

2.3.2.2. Sails.js

Sails.js is an MVC Node.js framework. It is based on the popular Ruby on Rails web framework and allows developers to quickly create REST APIs and real-time applications. In addition, it offers code generators which allows developers to build their application with less code writing.

Sails.js has many interesting features such as :

- Its construction on Express.js.
- Its real-time support with WebSockets.
- its powerful code generation, thanks to Blueprints.
- Its integrated ORM, called Waterline.
- It is well documented.

2.3.3. Frontend

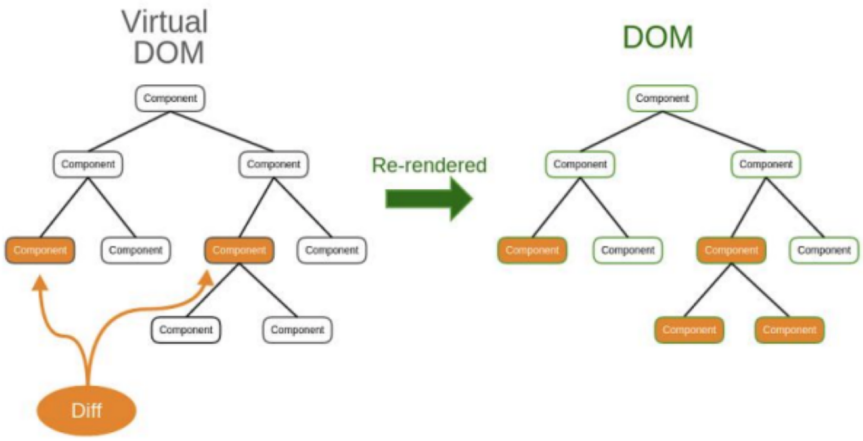
2.3.3.1. React.js

React.js or React is a JavaScript library, developed by Facebook, allowing to create user interfaces.

With its architecture, React facilitates the creation of interactive interfaces. All we need to do is to create independent components that manage their own state, and then compose them to create complex user interfaces. In this way, React makes our code reusable, simpler to understand and easier to debug.

React is based on a copy of the DOM which represents the HTML page called Virtual DOM. React allows the developer to declare the display of the application view according to the different variables used by the application. Based on this instruction, React compares the Virtual DOM with the browser's DOM and then uses a reconciliation algorithm to update the browser's DOM with the new values of the Virtual DOM.

The following figure shows an example of a change from the Virtual DOM transferred to the DOM.



Virtual DOM transferred to the DOM

2.3.3.2. JSX

JSX is a syntax extension to JavaScript. JSX simplifies the creation of React components by encapsulating the React.createElement(component, props, ...children) method.

For example, this React code is written without JSX.

```
class Hello extends React.Component {
  render() {
    return React.createElement('div', null, `Hello ${this.props.toWhat}`);
  }
}

ReactDOM.render(
  React.createElement(Hello, {toWhat: 'World'}, null),
  document.getElementById('root')
);
```

React without JSX

React without JSX Can be simplified with JSX to get the one

```
class Hello extends React.Component {
  render() {
    return <div>Hello {this.props.toWhat}</div>;
  }
}

ReactDOM.render(
  <Hello toWhat="World" />,
  document.getElementById('root')
);
```

React with JSX

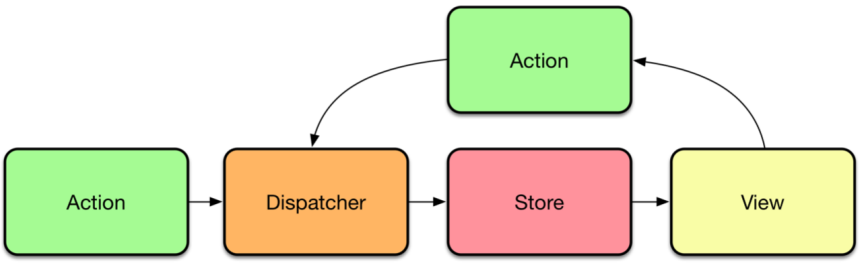
2.3.3.3. Flux

Flux is an architecture that Facebook uses internally when working with React. It is not a framework or library. It is simply a new type of architecture that complements React and the concept of unidirectional data flow.

Flux allows the transfer of data from one component to another without the need for a re- lation between the two.

The first React component calls an "action", this action passes data to the dispatcher who receives "actions" and the data and then asks the stores to perform specific processing on the data. And finally the second component will receive the new status of the blind.

The figure below shows what has just been said in a clearer diagram.



Flux Operation

2.3.4. Database server

2.3.4.1. PostgreSQL

PostgreSQL or Postgres is an open source RDBMS. Its source code is available free of charge, developed by a worldwide team of volunteers and is not controlled by any company or other private entity.

It supports a large part of the SQL standard and offers many modern functionalities:

- Complex queries
- Foreign keys
- Triggers

- Modifiable views
- Transactional integrity
- Control of competition

2.3.5. Limitations of the current solution

After two years of work on the project, the project code base has become almost impossible to maintain for several reasons:

- The size of the project has evolved very quickly by adding new features or by making improvements as the project progressed.
- The size of the project files became very large (more than 2000 lines of source code for some files).
- There were no conventions to follow when coding.
- Documentation not present.

Concerning testing, each developer had to test his work manually on his machine. On the day of pre-production on the dedicated server, the product manager was responsible for carrying out a global test on the platform by carrying out the necessary actions that the sales team does during a demonstration with a new customer to make sure that the modifications made did not affect the quality of the product impacted by other parties on the site.

These tests present a great workload for everyone as there are many of them and a whole day is dedicated to this stage to make sure that the production is going to be perfect.

As presented before, Hublo offers a module for interfacing with the GTA software. What is done with these interfaces is mainly to push the missions created on the Hublo platform and provided to its ATM software to keep track of what has happened in the establishment and to help managers not to reinsert them manually into their software.

Today, Hublo is not up to date with the latest API offered by Chronos, which is one of the largest ATMs used in France. As a result, we have to update it on our side as well. Furthermore, the interface with the Octime software lacks today the transparency on the status of the missions sent to the ATMs and whether they were posted correctly or not.

2.4. Analysis and specification of needs

2.4.1. Choice of Methodology

Successful projects are well managed. To manage a project effectively, the manager must consider many different software development methodologies in order to choose the one that will work best for the project. All methodologies have different strengths and weaknesses.

In the next section, we will present the one adopted at Hublo and explain why.

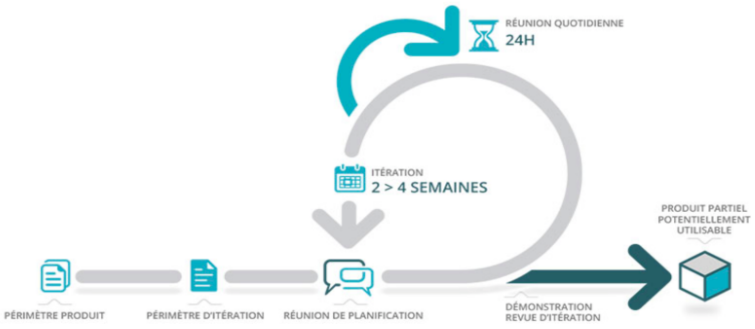
2.4.2. The Scrum methodology

Scrum is the most used agile methodology according to the 12th annual State of Agility Report as shown in the following graph.

Each Sprint starts with a planning meeting called "Sprint planning" in which three actors will be present :

- Product Owner: This is the person who conveys the overall mission and vision of the product to the development team to build it.
- Scrum Master: This is the person who guides the team in the application of the Scrum methodological framework.
- The developer team

It is at the end of this meeting that the team discovers the objectives of the Sprint and the tasks linked to each objective. During a Sprint, daily meetings called "Daily meetings" lasting less than 15 minutes allow each team member to speak about what he or she did yesterday, what he or she plans to do today and any difficulties encountered. During this Sprint, the team works on all the needs of this sprint by analysing, designing, developing, testing and integrating functionalities. At the end of each sprint, the delivered work must be ready to be delivered to the customers.



2.4.3. Scrum at Hublo

As for Hublo, the duration of the sprint is set to one week. This duration has been chosen in order to be super agile, to respond to customer needs as quickly as possible and possibly to pass bug fixes if any.

We start the week with the weekly meeting where we distribute our regular tasks and then we start the sprint.

Tuesday is the pre-production day where we test the functionalities developed during the last sprint. We also check if everything is going well, which will allow us to go into production. If not, we'll make the necessary corrections and redo the tests.

Wednesday is a normal sprint day where we continue our sprint normally.

Thursday is bug day where we try to fix as many bugs as possible that our customers have reported or that our team has found.

Friday is the end of the sprint where we continue our normal development.

Every day, each developer dedicates 1 hour of his day to review code written by another developer. So, the developer who does the review is called "catcher" and can ask for corrections or modifications on the written code.

2.4.4. Design methodology

To design the application, UML was chosen for several reasons. Among these reasons is the fact that UML is standardized. As a result, any developer familiar with UML can work directly on the same project. In addition, UML unifies object-oriented notations and concepts. In addition, UML offers a multitude of diagrams that make it possible to design independently of languages and environments.

In short, UML is a bridge that connects the architect to the client. Thanks to this, the architect has a clear vision of what the product should deliver. At the same time, the client can express what he thinks without having to know all the notions related to this unified language.

3. Specification of requirements

In this section, the expected functional and non-functional needs will be presented.

3.1. Background

Following the merger of the startups Whoog and MedGo, the founders decided to keep the Medgo solution as the basis of the Hublo application.

Customers with Whoog contracts should expect to benefit from the same service when switching to the Hublo application. Therefore, some features of the Whoog solution should be quickly added to the old Medgo solution to facilitate a merger of the two former startups and to be able to quickly bring the Hublo solution to market.

3.2. Functional requirements

Functional needs (or primary needs) represent what the system must offer users as services to meet their needs.

3.2.1. Part 1: Payroll Export

Scenario 1: As CSM, I'm adding a payroll code

- **Given:** The user is a CSM
- **And:** The CSM navigates to the export page
- **When**: The CSM selects add a paid code
- **And:** The CSM completes the different fields
- **And**: The CSM selects different options
- **And**: The CSM validates
- **Then:** The system saves the new paid code settings
- **And**: The system closes the modal
- **And**: The system sends a feedback to the user

Scenario 2: As a CSM, I must complete all required fields

- **Given**: The CSM is adding a payroll code
- **When**: The CSM does not complete the name,
- **And**: The CSM does not complete the start time,
- **And**: The CSM does not complete the end time,
- **And**: The CSM does not complete at least the intern or extern code
- **Then**: The system does not allow the user to save the payroll code

Scenario 3: As a CSM, I edit a payroll code

- **Given**: The user is a CSM
- **And**: The CSM has already add a payroll code
- **When**: The CSM selects edit on a payroll code
- **Then:** The system shows a modal to edit the payroll code
- **And:** The system has pre-completed the actual value in the field
- **When**: The CSM confirms his modification
- **Then**: The system save the new value
- **And**: The system closes the modal
- **And**: The systems sends a feedback to the user

Scenario 4: As a CSM, I delete a payroll code

- **Given**: The user is a CSM
- **And**: The CSM has already add a payroll code
- **When**: The CSM selects delete a payroll code
- **Then**: The system shows a modal to confirm the deletion
- **When**: The user confirms the deletion
- **Then**: The system deletes the payroll code
- **And**: The system closes the modal
- **And**: The system sends a feedback to the user

Scenario 5: As an admin, I am not able to edit or delete anything

- **Given**: The user is an admin
- **When**: The admin is on the export page
- **Then**: The admin is not able to edit or delete anything

3.2.2. Part 2: Refusal of mission

User Story:

- As a worker, I can refuse a job offer from the Details section in order to display only the offers that interest me.

Functional Specs:

- As a worker, I have a "Decline" button next to the "Apply" button in the Job Details section.
    - Staffing only for missions in the "to be filled" section
    - Creation of a second button: "Refuse".
    - Background colour: #D0021B

- As a worker, when I click on the "Refuse" button, the available mission disappears from my available missions.

User Story:

- As a worker who has declined an offer of assignment, I would like to be able to apply for a declined offer of assignment so that I can change my mind if I am available now.

Functional Specs:

- As a worker, I would like to have an Apply button in the details of a refused job offer
    - Blue button "Apply"
- As a worker applying for a job offer that has been refused, I would like to be reminded of the procedure for confirming compliance with working time regulations.
- As a worker applying for an unsuccessful assignment offer, I would like the assignment to be included in my future assignments (as a standard application).

User Story:

- As a worker, I would like to be notified when I refuse an assignment.

Functional Specs:

- When the worker clicks on the "Refuse" button :

If we come from the details of a mission => bring to the Page To provide + display the banner "Mission {Mission ID} refused".

- the banner "Mission {ID Mission} refused" must appear for a total of 1.5s value to be refined after test by PM
- For the time being, the refusal of an assignment does not affect the sending of notifications of the offer of assignment => will come in time 2

User Story:

- As a worker, I would like to be able to find the future missions that I have refused in order to apply again I have changed my mind

Functional Specs:

- As a worker, I have a "Refused" section in the "Missions" page.
    - Have a 4th section "Refused" to the right of "Passed".
    - If small screen (ex: Iphone 5c) =>
        - Either have the 4 tabs "To be provided"/"To come"/"Passed"/"Refused" on the same line (preferred solution)
        - Either have a side scroll if possible to have these 4 tabs "To be provided"/"To come"/"Past"/"Rejected" on the same line
- As a worker, I would like to find in the "Rejected" section all the future assignments that I have refused that have not been cancelled or started.
    - Refused" section = "To be filled" section but for refused missions
    - Refused assignments must not be cancelled.
    - The missions of the Refused section must not be passed or started.
    - If there are no future non-cancelled refused assignments, have the posting provided for this purpose
- As a worker, I would like the rejected missions to have the same visual appearance as the missions to be filled in order to make me understand that they are still to be filled.
- As a worker who has withdrawn my application, I would like the assignment to be automatically considered as rejected in order to save time.
    - If a worker withdraws his application for an assignment, the assignment will be refused by the replacement. This assignment will be available from the "Rejected" section of the Assignments page.

3.2.3. Part 3: Splitting a long mission into several short missions

User Story:

- As an admin needing a replacement over several days, I want to split a long shift in several distinct short shifts so that I could gain time and share my shift between several candidates.

Acceptance criteria:

Shift creation ->

- In “Post a new shift” :
    - For short missions => No change
    - For long missions (i.e. when an admin creates a shift for more than two days) => add two radio buttons below the "Ajouter un jour" button
    - The two radio buttons are :
        - "Créer une mission de plusieurs jours" (see model)
        - "Créer plusieurs missions d'un jour" (see model)
    - Have radio button "Créer une mission de plusieurs jours" selected by default
    - Modification of the locums according to the chosen option:
        - If "Créer une mission de plusieurs jours"
        - No change compared to today
        - If "Créer plusieurs missions d'un jour"
            - locums available => locums available for at least 1 day
            - locums available as last resort => locums not "available" every day of the mission and available as a last resort for at least 1 day
    - In the modal "Aperçu de votre mission avant envoi"
        - For short missions => No change
        - For long missions => add below the dates the chosen mission option:
            - "Mission de plusieurs jours : le remplaçant devra accepter la totalité de la mission"
            - "Plusieurs missions d'un jour : la mission pourra être partagée entre plusieurs remplaçants".
    - When the admin creates the mission :

        - Don't display the modal "Attention : mission de plusieurs jours" anymore
        - if the admin chose "Mission de plusieurs jours" => same wording as usual (V4_Model_Creation_Mission_Success_1) => creation of a single mission lasting several days
        - if the admin chose "Plusieurs missions d'un jour". => creation of X number of one-day missions=> have the wording taking into account the X missions.

Request creation ->

- In Post a request or Modify a request :
    - For short missions => No change
    - For long missions (i.e. when an admin creates a shift for more than two days) => add two radio buttons below the "Ajouter un jour" button
    - The two radio buttons are :
        - "Créer une mission de plusieurs jours" (see model)
        - "Créer plusieurs missions d'un jour" (see model)
    - Have radio button "Créer une mission de plusieurs jours" selected by default
- In the modal "Aperçu de votre besoin avant envoi"
    - For short missions => No change
    - For long missions => add below the dates the chosen mission option:
        - "Mission de plusieurs jours : le remplaçant devra accepter la totalité de la mission"
        - "Plusieurs missions d'un jour : la mission pourra être partagée entre plusieurs remplaçants".
- When validating a request, in the modal "Aperçu de votre mission avant envoi"
    - For short missions => No change
    - For long missions => add below the dates the chosen mission option:
        - "Mission de plusieurs jours : le remplaçant devra accepter la totalité de la mission"
        - "Plusieurs missions d'un jour : la mission pourra être partagée entre plusieurs remplaçants".
    - When the admin validates the request :
        - Don't display the modal "Attention : mission de plusieurs jours" anymore

- if the admin chose "Mission de plusieurs jours" => same wording as usual (V4_Model_Creation_Mission_Success_1)=> creation of a single mission lasting several days
- if the admin chose "Plusieurs missions d'un jour". => creation of X number of one-day missions=> have the wording taking into account the X missions

Extern communication:

- Preparing a communication on UserPilot
    - Ask PM to talk about it with Team CS and Antoine Bally (about 2 days)

Functional specs added (after Review with Tech Diver):

- Case of a long shift split into several short shifts : The first 200 profiles will be displayed ⇒ Some workers will be alerted but will not be displayed because it is technically too heavy.
- Cases of partial error for the creation of several shifts from a long shift :
    - Display the first error message
    - Do not create any shifts of all the shifts to be created
    - Do not remove shift credit for short shifts created and then deleted if this is the case

3.3. Realization

3.3.1. Introduction

After presenting the design of our application, we will now describe our hardware and software environment.

3.3.2. Software environment

In this section, we will present our software environment.

3.3.2.1. Visual Studio Code

Visual Studio Code is a light but healthy cross-platform development environment. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages.

3.3.2.2. pgAdmin

pgAdmin is PostgreSQL's most popular and feature-rich open source administration and development platform.

3.3.2.3. Monday

Monday.com is a tool that simplifies the way teams work together: workload management, project monitoring, work progress, communication with others. It is mainly used in the technical team to define the sub-tasks of each need with a very personalised dashboard.

3.3.2.4. Github

GitHub is an American web-based hosting service for source code version management. It also allows several developers to work on the same project simultaneously without worrying about how everyone's work is put together.

3.3.3. Test automation

3.3.3.1. Unit tests

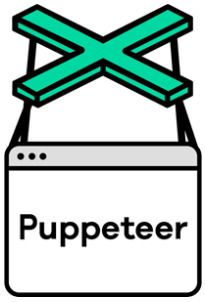
At Hublo today, the technical team follows the "Test Driven Development" approach. This is a software development technique that advocates writing unit tests before writing the source code of a piece of software.

As for us, to be as rigorous, we have also created unit tests for the most important existing functions to try to cover all possible cases. This was done with Mocha.

Mocha is a JavaScript test framework that allows you to test the return result of functions. Basically for each function we want to test, we write a test suite with past inputs and expected outputs and Mocha will check that the right result is generated.

3.3.3.2. Graphical tests

The third step was to automate the graphical tests. These tests are used to automate what the product manager does before each product is put into production. Our future vision is to create tests for all parts of the site. For these tests we use the Puppeteer library.



Puppeteer is a Node.js library developed by Google, allowing to control a Chrome na- vigor, either with a user interface or in "headless" mode (without graphical interface). It is possible to perform most of the actions that are done manually on a browser.

The graphical mode is very useful during the development phase of the tests, because it allows you to follow in "live" the different steps described. While the "headless" mode will allow the tests to be run in a continuous integration tool, this is the default behaviour of Puppeteer.

The functionalities covered for the moment are :

- Posting a mission (and similar mission) - Declaring a mission
- Post a need
- Organising my schedule
- Substitute Registration - Substitute Invitation

3.3.3.3. Continuous integration

To run these tests, a command has been added to our project which is "npm run test". This command will launch in order : unit tests, API tests and finally graphical tests.

Unfortunately, we found ourselves several times with problems due to forgetting a development, afraid to launch the tests before declaring to finish the task. To ensure that this would never happen again, CircleCi is used today.

CircleCi is our continuous integration tool that allows you to perform several tasks when "pushing" to a branch on Github as it integrates with it.

Internship report [Part 2]

A part designed to convince a line manager to take over full responsibility for the project you worked on during your internship. You will put forward your assets for such an assignment by justifying your capacities.

During my internship, I worked on many micro-services based on the company's main product.
You will find below the reasons to give me the full responsibility of the micro-service "serviceapi".

Since I joined Hublo's tech team, I've been very interested in the tech team's processes when working on micro-services, especially "serviceapi". That's how I managed to be quickly autonomous regarding the tasks assigned to me on this micro-service.
I spent my first sprint on the documentation of the tools used on this work environment and reading the code in order to understand how it works and how it interacts with the other micro-services, almost all of which depend on "serviceapi".
By my second sprint, I could already chain the tasks assigned to me on this environment. This means that I learn quickly.
After three months of development on this micro-service I know that my vision on how to improve it and make it more robust would be beneficial for the tech team.

Since I've been working on this micro-service since the switch from version v1 to v2, and I've done most of it, I know everything about its code. This means that I will be faster than any other developer to know which files to work in in order to add or modify some routes of the api.
Giving me the role of service owner to continue its development would be the most efficient choice for fast and immediate results.

As I said before, I've done most of the transition from the v1 to the v2 version of "serviceapi".
I always gave ideas to improve the micro-service. By entrusting me with more responsibilities, you are sure to get a better service.

In conclusion, I know perfectly well the "serviceapi" micro-service and the expected needs. I took care of the transition from V1 to V2 and proposed improvements for this micro-service. This is the proof that I am conscious, professional, proactive and versatile.
To continue its development, entrusting me with its responsibility would be the best choice for fast results and a high-performance service.

Internship report [Part 3]

A part intended to convince a superior to entrust you with the responsibility of the project of your dreams. First you will describe the project, then you will use your qualities and assets to convince.

During my internship, I worked on the backend of Hublo's product. This gave me more experience in programming. Moreover, this internship showed me how things work in a startup which is moreover a startup resulting from a merger. Not only in the IT fields but also in the knowledge and implementation of the company's processes.

I have always been passionate about development. So programming is not a boring job for me. Moreover, I like theoretical challenges, I never hesitate to look for answers by myself. I quickly deliver a job well done.

I quickly become familiar with the tools used in a project. For example, before the start of the internship, I had never worked on a micro-service project like the one from Hublo. But in just one week, I learned to master this work environment. So, at one point, the lead-developer didn't need to check my work. My willpower is the key to my success in every task I am assigned. When I am focused on a task, I never consider it boring work. Therefore, if there is some tasks to be done, I just do them, no matter how hard are they.

To improve the productivity of the tech team, I sometimes work in other squads (the tech team is divided into several squads spread over different micro-services) depending on the needs of the company.

In conclusion I have the technical skills and competencies required to be a good developer. I am also autonomous, and I learn quickly. This way, I will never lose the productivity time of other team members. Moreover, I have already been part of several developer squads, so I know how to work on the other parts of the Hublo product.

These elements are the reasons why I am asking for more responsibility in the technical team and more responsibility related to the Hublo product.