A picture containing diagram

Description automatically generated

Diagram, text

Description automatically generated

“I verify that I am the sole author of this report, except where explicitly stated to the contrary.”

*16th October 2020*

***Amantini Alessandro***

# Abstract

The placement was hosted by an IT company that hired me as a Junior developer. Although the job mentioned only backend technologies, I spent the first 6 months (July to January) working both on backend and frontend code, developing PWAs1 (Progressive Web Apps). For the remaining 6 months (which, effectively, turned out to be only 3 due to Coronavirus), I only worked on the backend, in a team whose main aim was to test the performance of the company software and to spot possible bottlenecks. Besides the normal work tasks, I also joined two projects with the other placement students: the first one was a Java project for comparing images, the second one was more about investigating and testing the capabilities of low code technologies.

The company backend was primarily written in Java, so the surrounding stack, as expected, consisted of several Java frameworks such as Spring Boot, JAXB, JMS, etc.; the frontend, instead, was written in Typescript, SCSS, and HTML, and relied on two main frameworks: Angular and Ionic.

Thanks to this experience, my IT background received an incredible boost:

* I improved my knowledge, my mindset, and my skills in web technologies and reactive programming;
* I learnt several good programming practices and how to “refine” my code;
* I got familiar with CI/CD and cloud computing; and
* I became more proficient with Agile techniques.

Furthermore, being my first job ever, I had a taste of what working for a company really feels like, what are my duties and what my privileges, how to behave in a professional context, and what it means being an actual programmer.

Table of Contents

[1. The Company 5](#_Toc53704684)

[About Flooid 5](#_Toc53704685)

[Teams structure 5](#_Toc53704686)

[The Job Research 5](#_Toc53704687)

[Expectations 6](#_Toc53704688)

[2. The Experience 7](#_Toc53704689)

[First 6 Months 7](#_Toc53704690)

[Training 7](#_Toc53704691)

[Frontend 8](#_Toc53704692)

[First Placement Project (Frontend) 9](#_Toc53704693)

[Analyses 9](#_Toc53704694)

[Last 6 Months 11](#_Toc53704695)

[Backend 11](#_Toc53704696)

[Second Placement Project (Backend) 12](#_Toc53704697)

[Furlough 12](#_Toc53704698)

[Analyses 12](#_Toc53704699)

[3. Learning Outcomes 13](#_Toc53704700)

[Reality vs. Expectations 13](#_Toc53704701)

[The Types of Learning 13](#_Toc53704702)

[Training Period 13](#_Toc53704703)

[Frontend Period 14](#_Toc53704704)

[Backend Period 14](#_Toc53704705)

[4. Reflections 15](#_Toc53704706)

[Technical Learning 15](#_Toc53704707)

[5. Conclusion 16](#_Toc53704708)

[Appendix 17](#_Toc53704709)

# . The Company

## About Flooid

The company I worked for is based in Coventry, UK, and creates software for companies operating in the retail sector. They store information about these companies' products and customers, create software to manage and track products sold both in-store and online, and, of course, develop apps and GUIs for POSs to interact with this data. The company was formerly known as PCMS but was renamed at the beginning of 2020 to Flooid, which is also the name I will use throughout the paper.

## Teams structure

The IT Department is mainly split into customer and product teams, respectively, those allocated to specific clients2 and those working on software used across all the teams (e.g. database functions). All the IT teams follow agile methodologies and are, generally, composed by:

* 1 Project Manager;
* 1 Lead Dev;
* 1 BA, although often shared across many teams;
* 1 QA; and
* 1 or more software developers, including, at least, 1 Senior Dev.

## The Job Research

When I was looking for a company, one of the main things I paid attention to was the tech stack: I had no interest at all in frontend development and I wanted a company to mainly program in **Java** because, at the time, I considered it one the best languages out there and the one I was most proficient with.

To find the right company, I entrusted myself to a couple of websites primarily: [**RateMyPlacement**](https://www.ratemyplacement.co.uk/)and [**Gradcracker**](https://www.gradcracker.com/). As I was navigating through the latter, I bumped into Flooid insertion and I noticed immediately that they claimed to use Java, Spring Boot, MySQL, XML, JAXB, and other Java libraries. My first thought was that such a stack could perfectly fit my interests. Also, Gradcracker had a ranking for the best-rated companies3 and Flooid was in the top 5.

## Expectations

Given all the aforementioned “qualities” of the company, I expected to learn many strategies and technologies not taught at university and to get out of there being a way better programmer. Overall, I believed I would have become incredibly proficient in Java, able to create large Java applications that could rely on industry software such as Spring Boot, JAXB, JMS, etc.

Another thing I figured I would have gained from this experience was the idea of what coding feels like when you work for a company, that is:

1. How different are the pieces of code people write for a company compared to what I wrote so far for individual and group projects?
2. What does it mean coding every day, in a team of people with far more experience than me? How and how much can I learn from them?
3. What does it feel like to work in the industry and to create software someone out there will actually use?
4. What does it mean to work for someone who tells you what to do? And if I make a mistake?

I knew that the answers were different depending on the company, the team, possible deadlines, and so on, but, broadly speaking, this was what I had envisioned.

# 2. The Experience

The time I spent at Flooid can be divided into 4 phases, all about 3 months long, and very different from each other:

|  |
| --- |
| * Training; * Frontend; * Backend; and * Furlough. |

## First 6 Months

### Training

The very first week of work, every placement student was assigned to a team, a different one for each of us, and all working on backend stuff only...but me. My team was the only one in the company dealing with mobile apps, hence, their competence had to extend to frontend technologies, namely Angular (which also meant Typescript, SCSS, and HTML) and Ionic.

Although I had the luck to work on a React group project as part of a university course (Software Engineering Group), I had no clue how to use Angular, and the knowledge of Typescript, SCSS and HTML was not good enough to have my code passing code reviews. The company was aware that placement students were not ready to actually do any work yet, so, each team designated a senior/lead dev as a **mentor** to the respective student. Unfortunately, my mentor never fulfilled his role of “teacher”, mainly because he only dealt with database stuff and had no idea how to use Angular and Ionic himself.

Because of this, and because the team had not many tasks I could pick up, my manager had a talk with my mentor, and, together, they decided I should have spent as much time as I needed **training** myself on whatever could have been useful for the job. Hence, I dedicated this first 3 months to studying Angular and Ionic, mainly from their official documentation and from Lynda.com, and to practicing with Typescript and SCSS; I also tried to figure out how the company frontend code was structured and, every now and then, I fixed some buggy code.

I really wish my university had taught me more about JavaScript and reactive programming, not just to make my life easier during my placement, rather, because it is incredibly powerful and used across almost every IT branch: web technologies, desktop and native apps, REST and (recently introduced) GraphQL APIs, etc.

### Frontend

At the end of September, my mentor left the company; my lead dev became my new mentor, and I finally started to get assigned to some real tasks.

Although I could create and start Angular and Ionic applications, my methods were “rude”; it was thanks to all the **code reviews** I did not pass (when it came to reviewing code, my teammates were not lenient) and the hours spent **pair programming** (which in my case, pretty often, consisted of me observing the other person coding) that I had the chance to improve my techniques. I never realised how helpful it can be shadowing someone more experienced than you, or, vice versa, coding with them having you reflecting on your strategies and correcting you.

My tasks were generally easy, like fixing broken code after code merges, creating some JavaScript functions to add new features or to test existing ones, etc. A common one was a NaN showing up in a webview. I remember how much I struggled to find these kinds of errors at the beginning and I had to ask for help to either a senior dev or my mentor. It was mostly in these cases (searching for a bug) that I ended up pair programming and learning how to use Chrome DevTools, something I still find amazing and incredibly useful for my personal web projects.

Besides working, during these 3 months, I also continued to learn a lot. This was partly due to my manager, who allowed me to do some 3 hours of individual training every week, but mainly, thanks to my new mentor who taught me many things, the most significant were probably:

* the importance of writing more maintainable and consistent code;
* to avoid using JavaScript when CSS snippets could do the work;
* to rely on Chrome debugging tools to spot bugs and on a CI\CD software to be more time efficient; and
* to prefer pre-existing frameworks functionalities instead of writing my own ones.

I personally found those 3 hours of training very useful, both to understand more about patterns and software I did not understand throughout that week and to feed my curiosity about how some of them work behind the scene.

### First Placement Project (Frontend)

Starting from the beginning of November, besides the normal tasks, all the placement students were assigned to a common project: the goal was to create a program to simplify some QA tasks by analysing and comparing images. Everyone was assigned to a specific role and, of course, I was the only one taking care of the GUI.

At first, I thought giving us students the possibility to man our own project would have been cool and satisfying, but I easily changed my mind after our first meeting as a team. Despite being the only one touching the frontend, the other students forbid me to use Angular and I had to use Thymeleaf, a Java template engine. What happened there was an example of a team run by inexperienced people: they did not want me to use any JavaScript frameworks because they had never used one before and thought it was hard to handle...I had to use it, not them!! And besides, no one knew how to use Thymeleaf either, so, eventually, they did not even review my code.

Anyways, I did my job and I created the GUI, but the rest of the project was far from being completed. In fact, the time allocated to this activity was only 1 hour per week, which was too little for a good backend. As such, the project was never completed and, at the beginning of the next January, the company itself decided to terminate it to assign us to a new one in February.

Even though at that stage of my placement I had already spent roughly 5 months between studying and working on JavaScript, it was only while working on that project that I started to realise I was not longer interested in Java, and that I was a JavaScript enthusiast: even the simplest task in Java required so much boilerplate and it was lame to me, whereas doing the same thing in JavaScript was easy and funny.

### Analyses

The placement project was a fiasco, but it taught me how important some people are for a team, especially:

* those capable of analysing a problem, fragmenting it into smaller tasks and assigning the right amount of work to each team member; and
* those who can create the architecture and/or coordinate the other members to make everything compatible and fully working.

This is probably what this team was missing the most, but I guess no one can be blamed for that as we were all too inexperienced.

From another perspective, the team was overall fun to work with: we were all students who made jokes and, in spite of some childish behaviour every now and then, I had a good time programming in their company.

Compared to this one, my actual team (the one with my mentor, to be clear) was the right opposite. On one hand, probably due to the many deadlines that the company kept adding with very short notice, everyone was always stressed, never really exchanging jokes nor spending too much time interacting with each other. I personally felt that the social part was missing completely, and working in such an environment, I have to admit, was not very pleasant. On the other hand, the team was very productive, full of competent people who helped me grow a lot.

Which one did I prefer? Well, if that was to be a lifetime job, a team full of people I enjoy the time with is definitely my first choice, also, I know I made significant contributions code wise. However, as this was just a placement and learning was my first priority, I am glad I got to spend more time on the actual team rather than the one made of students, regardless of the dissatisfaction of little and marginal contributions.

## Last 6 Months

### Backend

This period started in January, when I moved to a new team whose main purpose, as already mentioned, was to time and test the efficiency of some pieces of software (e.g. database calls). The main language was Java, however, a couple of members of the team also happened to program in Scala; as they were not very good at it, I took the opportunity to refresh my memory about this language and I helped them from time to time. Having previously learnt both these languages at university made my life easier, nonetheless, I soon realised that the academic knowledge was not enough, for example, I did not know much about design patterns, which, in spite of not being part of the language technically speaking, are still very important and used quite often.

My role in this team was not strongly defined, I had to primarily write Java code, but I also picked up other tasks:

* Like mentioned before, I helped other members with Scala;
* (with a lot of supervising) I maintained some Prometheus and Grafana code running on Docker;
* I pair programmed with my lead dev to implement new features on the GCP environment; and
* I wrote some reports for the fortnightly showcases the team had.

My new lead dev was a very innovative person, in the sense that he used to always look for new tools and software we could use for our purpose. Thanks to him, the team was using so many different tools such as Docker, Grafana, Prometheus, Jaeger Tracing, Gatling (a Scala framework which explains why some members were using this language), GCP (Google Cloud Platform), etc. Although I did not learn all of them, I had the chance to understand how some of them work, especially Docker and GCP.

Something worth mentioning is that my new manager did not care too much about the technical training (I was not assigned a new mentor and she removed that weekly self-training time, allocated to me by my previous manager, saying I would have learnt doing), but she paid a lot of attention to the professional one. In fact, about every 3 weeks, the two of us had a one-to-one meeting; during this meeting, she used to tell me all my negative as well as the positive sides as reported to her by my teammates, mainly related to my behaviour rather than my code. The aim was to make me aware of my “behavioural weaknesses.”

### Second Placement Project (Backend)

The second placement project began in mid-February. The goal was to investigate how low-code works and to use it to replicate a couple of the apps I worked on during my frontend period. As all the students were put on furlough at the end of March, this project, again, was never terminated.

### Furlough

March was my last month of work: on the morning of the 31st of March, a member of the HR invited all the placement students to a meeting to communicate us that, starting from the 1st of April, we would have been furloughed, we would have no longer been allowed to work on any code belonging to the company, and our permission to access private resources internal to the company would have been removed.

Not having any work left, but still wanting to code, I practiced what I learnt so far:

* I created a few, small PWAs, with and without Ionic;
* I made a couple of websites both using vanilla JS and Angular; and
* I kept looking for new JavaScript libraries and CSS frameworks to use in my future projects.

### Analyses

Here I did not learn as much as in the other team, but I managed to contribute way more, not only by coding. In fact, my lead dev involved me in the process of finding solutions to some coding problems. For the first time since the beginning of the placement, I felt satisfied with my contributions and, overall, mentally stimulated, something I really missed.

Although I was lacking some knowledge on things like design patterns, the expertise I gained from my university about Java, Scala, and the data structures, turned out to be very useful in this team. The languages helped me work on the majority of the tasks, understand was going on (except for some frameworks), and the data structures were useful when solving some of the aforementioned problems with my mentor.

What I appreciated the most about this team was the people: everyone was talkative, people went for coffee and tea together, and I had a good relationship with all of them. I am aware I learnt fewer coding skills, but I have grown up as a person thanks to the one-to-one I had with my manager and the stay was enjoyable.

# 3. Learning Outcomes

## Reality vs. Expectations

As almost always happens, expectations and reality did not match very well, but not necessarily in a bad way. I had an answer to most of the questions I had before joining the company (as mentioned in **chapter 1 - expectations**):

1. The code I used to write for personal projects is completely different from the industry one. I had the first taste of this during the training phase and the frontend phase confirmed it.
2. Coding every day, for people more experienced than me, helped my skills grow impressively. Unfortunately, right because I was so inexperienced, I almost never took part in those meetings where senior devs decided how to tackle problems, depriving me of the mental challenge and of the feeling of coming up with a solution.
3. Something I love about programming is to see my software running, to see that it works, no matter how long it may take. Working in the industry, it felt like I lost my individuality in a sense: hardly anything I did resulted in a tangible difference in the software I was working on.
4. Knowing exactly what the team should work on makes everything easier as you can pick up what better suits you. I thought messing up would have been terrible, but I never really had the stress of deadlines on me, so all my mistakes were reported to

me and all I had to do was to fix them.

## The Types of Learning

During all the first 3 phases (Training, Frontend & Backend) I learnt a lot, although, very different things in very different manners:

### Training Period

For the first 3 months, I studied on my own; having so much “free time”, that’s the period wherein I learnt the most. I have always enjoyed studying on my own as I can take breaks from the main subject to elaborate on minor details, I can spend more time on something I find harder, and I can productively manage my time. I was very happy with the outcome, however, the main downside with this approach was that the deriving coding skills were raw, the code I could write was not ready for company software production and not easy to maintain.

### Frontend Period

During this phase, I found the perfect balance between professional engagement and self-learning that helped me refine my techniques. My mentor was a very experienced developer who had me critically analysing most of my choices; he taught me why many of the things I did were bad and how to improve them. The self-training was the icing on the cake, thanks to that I could deepen the topics I struggled with during the week, those I did not fully understand, and those I was just curious about.

### Backend Period

My mentor, expecting me to learn everything just by doing, removed the self-training time, but I was touching different frameworks every week and most of them were completely new to me. In fact, ay too often I totally depended on the help received from more experienced developers on the team. Due to this, I did not learn much in practical terms, but I understood the theory behind cloud servers, Docker, AOP, and other concepts. What I did gain from this team, instead, was how to be more professional, how to expose my ideas and to criticise other’s ones in a formal and not rude way, and how to properly behave when dealing with problems, even not strictly related to programming.

|  |
| --- |
|  |

# 4. Reflections

## Technical Learning

Looking at my current technical knowledge, I feel confident saying that I am a far better programmer than I was before.

The major improvement is definitely on the **frontend** side. Before starting to work at Flooid, I could barely take care of a webpage; now, I can create one in no-time, making it look good, professional, and with a solid underlaying structure.

All the JavaScript (and the surrounding stack) knowledge I gained really means a lot to me; not only because now I know how to make smartphone and desktop apps (thanks to the **PWA** power), but also because the future for this language seems incredibly bright, and having it on my CV gives me a lot confidence.

For what concerns **backend**, I got to practise a lot, getting a better insight of some Java working flows, and applying several design patterns, which, despite what I thought before the placement, are extremely important on big projects.

Conversely to what I mentioned about the frontend, the academic backend teaching was pretty exhaustive, and I felt it even more when I confronted myself with my “placement colleagues”. None of them knew how to do functional programming, a few knew about pointers and memory allocation, etc. Furthermore, the SEG course included some lectures about clean code, good programming techniques, and other things which were very close to a real working environment. I blame myself for having underestimated the importance of these lectures, as pretty much everything was useful for the job.

**Agile methodologies** are worth mentioning as well. Daily explaining to my team what I was planning to do for that day and having a fortnightly recap, turned out to be a great way to discuss on what should one spend more time on and how to better collaborate to make the project work. Nowadays, pretty much every tech company applies them daily and, like for JavaScript, experience in the matter is a nice-to-have on the CV.

Another way this experience affected me is the way I will program from now on: if before I used to rush more, starting to code before coming up with a concrete structure, now I will first design a concept and spend time analysing it to make a more maintainable and more efficient project, reducing the hassle of refactoring whenever I add a new feature.

# 5. Conclusion

Even though it was just 9 months long rather than 12, I have learnt a lot from this placement, both technically and professionally. Now I know what it feels like working for a company and this helped me have a clearer idea of what I want from my future career. All of this will not just help with my degree final year, but also for future interviews and jobs.

In the overall, I was happy with the preparation I had before starting this experience: the basic concepts of programming were very good; I also knew how to do functional programming and how to write tests.

I was a bit less satisfied with my university choice of teaching PHP rather than JavaScript which, thanks to Node.js, is way more powerful, more requested and can completely replace the opponent; however, this was not a real problem and I do not think King’s College is guilty in any ways about it.

One thing I want to blame my university for was not to have taught me anything about internet protocols, ports, REST and so on. I know there was a module for that, but as I do Computer Science **with Management**, I did not have it as part of my courses, and this was very penalising throughout the placement.

To conclude, I am happy with the programme I have chosen: taking the year in industry was a great idea and I think universities should do more to convince their students to take it.

# Appendix

1. The difference between a normal webpage and a PWA is that the latter tries to mimic the behaviour of a native app both for the capabilities (i.e. accessing the camera) and the UI (components and interactions); also, thanks to apps like Cordova, they can be compiled to native code (.ipa for iOS and .apk for Android) and be uploaded to the app stores.

2. It may be meaningful to keep in mind that customers are intended as big companies only, such as Waitrose and Arcadia.

3. The ranking offered by Gradcracker was based on former placement students’ reviews.