Yoopies

Work experience report

A product owner intern's guide

Written by Jurgen PALSMA

April - August 2017

jurgen.palsma@epitech.eu

2/07/2017

YOOPIES

Abstract

This document, written by me, Jurgen PALSMA, a 3rd year student at EPITECH Paris, is a report of my experience as a Product Owner and Growth Hacker intern at Yoopies, which I carried out throughout the summer of 2017 (April to August).

In the first part of this document, I will present the role of the product owner, throughout my work experience at Yoopies, in order to introduce and guide a future employee to continue the work I have done.

In the second part, I will try to convince my internship supervisor (and internship tutor) Olivier W. to integrate me in a new project at Yoopies.

Table of contents

Written Report

Part 1 - Company Context

Dear reader,

Before we start explaining what your role is and what your tasks & responsibilities are as a fresh product owner, let me introduce you to the context of your new company, Yoopies[1]:

A. Yoopies: origins

In 2011, seeing how his sister had trouble finding a babysitter for her child in Paris, Benjamin Suchar, an entrepreneur (you might have heard of his app, CheckMyMetro[2]), founded Yoopies with his acolyte, Jessica Cymerman, a journalist known for her blog SerialMother[3].

Yoopies' concept is simple: it is a platform which links babysitters and parents. By bypassing the traditional french babysitting agencies, Yoopies cuts the cost of hiring a babysitter without affecting the sitter's salary.

After being incubated at *La Pépinière 27* in Paris, Yoopies knew a certain growth which permitted its' launch, starting its service in Paris and its suburbs.

B. Yoopies Today

Today, Yoopies employs more than 40 people, and has more than 150k monthly users worldwide. On top of being present in multiple countries across Europe (via the web platforms), Yoopies provides an app for more than 2,500 daily users.

Yoopies has also extended its services. Not only can you find a reliable babysitter on Yoopies, you can also find a housekeeper, a pet sitter, a caregiver for the elderly, or even private lessons and tutoring.

Yoopies' users can be segmented in two categories: employers (who request a service) and applicants (who perform services). To provide cash flows, Yoopies is a freemium service: to enjoy all the features of the platform (notably: being able to book regular reservations through which a considerable amount of state benefits can be acquired, by contracting a sitter), employers must register as "Premium" users. In addition, a symbolic fee of 1 euro (depending of your country's currency) is added to the hourly price of your service. Applicants also guarantee the company's cash flow, but in a lesser way : they also have access to a Premium service (to accelerate their job searches) and can request a paid certification (which is often asked by the employers).

Yoopies also proposes a B2B[4] service, which gives access to a Premium account and additional services (such as dry cleaning) for a company's employees.

---

[1] yoopies.fr - *Yoopies' french website*
[2] checkmymetro.com - *B. Suchar's first project*
[3] serialmother.yoopies.fr - *J. Cymerman's main media, a blog for young mothers*
[4] *Business to Business*

C. Startups

As you can see, Yoopies developed itself from a niche market (parisian parents) to a multi-service platform, in both B2B and B2C[5] services.

You should, by now, consider the complexity of the rapid growth of Yoopies. In the context of a extremely competitive market, between the old-school babysitting agencies[6], other babysitting apps[7] or platforms[8], you can deduce that an aggressive and effective development strategy is crucial.

In addition, the costs of development for french startups, especially parisian startups are high. Due to high salaries and rent prices, it is extremely difficult to expand the physical aspects of such a company. Yet, out of the hundreds of startups that are started or incubated each year, a few of them, like Yoopies, manage to stand out and flourish.

Thus, you might ask yourself:

How can a small company guarantee both users and cashflows?

Yoopies' approach to this problem is straight to the point: by continuously developing its platform. Once its proof of concept (babysitting in Paris) was deemed a success, Yoopies opened itself to different features and customer segments.

This approach to the market can be seen in a lot of start-ups. The key point of this approach is to be scalable and adaptable. Scalable, in order to guarantee your growth, and adaptable, in order to open yourself to different customer segments, which have different needs.

These needs can be technical (for example, different customer segments use products on different devices/mediums), monetary (budgets vary between segments, for example between babysitters and B2B partners), or functional (different customers want different features).

With that in mind:

How can you be simultaneously scalable and adaptable on different projects?

[5] *Business to client*
[6] *Like Yoopala.com, KangourouKids, or FamilySphere*
[7] *Like UrbanSitter or Bsit*
[8] *See Holidog - a petsitting service*

Part 2 - Work environment

A. SCRUM in a nutshell

With this context of scalability and adaptability, older development and management frameworks crumble to dust. Indeed, you cannot be adaptable if your processes rely on long term development and hierarchical practices. In the late 80s and early 90s, a new idea came to the software development world which revolutionized a lot of software (and lots of other) companies' approach to their products: SCRUM.

In a nutshell, the idea of SCRUM is to deliver your product in a continuous and modular flow. SCRUM bases itself on an iterative and incremental agile framework. This means that instead of delivering your product in a classical, long term process, you deliver it in small batches, one functionality after another.

The advantage of doing so, is to avoid cost of your unavoidable product design errors or diverse miscalculations, and enables you to reach your customers both continuously and on an early stage, with limited resources.

For example, with classical development frameworks, a critical bug on your launching day would ruin your product: your anticipated communication and development investments would be flushed down the toilet as your users will not be able to use your product, whilst degrading your company's image.

With SCRUM, the same scenario would be different: your users will still be able to use your product as you can retract the crashing feature and delay its launch by a few weeks.

In order to work in a SCRUM fashion, you will need to segment your product into independent functionalities that can be easily developed and attached to your existing product, and you will need to deploy these functionalities as frequently as possible to get continuous user feedback.

With that in mind, you are ready to discover Yoopies' organizational and technological architecture:

B. Organisational architecture

At Yoopies, the team constitutes 3 major divisions:

-The Marketing team, which handles marketing campaigns, b2b contracts, and legal affairs.
-The Support/CRM[9] team, which handles customer support and community management.
-The Development team, where you will be.

We will bring our focus to the development team as it is the one you will join.

[9] *Customer Relationship Management*

The development team consists of:

-A CTO, or chief technical officer, is focused on the team's technological issues.
-One or more Data Analyst(s), whose primary focus is the analysis of the data the platforms generate.
-One or more Product Owner(s), who is responsible for maximizing the value of the product and the work of the Development Team.
-Web developers, which, at Yoopies, are full-stack developers. The web developers compose two teams: the "Acquisition" developers, who are focused on the features which provide Yoopies' acquisition (for example, the site's SEO features, or registration to the website), and the "Conversion" developers, who are focused on conversion features (such as the reservation process).
-iOS developers, who handle the iOS apps
-Android developers, who handle the Android apps.

Now that you have a gross view of Yoopies' human organization, let's get back to SCRUM, and, more specifically, the "SCRUM Timeline".

C. SCRUM Timeline

Development through SCRUM is segmented in "Sprints". A Sprint is a segment of time in which a definite set of features is implemented in your product.
In most cases, a sprint will last for about 1 to 3 weeks.
The features implemented throughout the sprint are "definite", they are selected by the Product Owner upstream and are not supposed to be modified during the sprint.
A Sprint is rhythmed by several meetings[10]:

The Sprint Planning

The goal of the sprint planning is to define the work and effort necessary to meet their *sprint commitment* - the features that will be developed during the sprint. In order to do so, the team (developers, p.o. and CTO) vote the effort of each feature in order to evaluate the what can be developed during the sprint.
Web sprint plannings and Mobile sprint plannings occur every other week.

The Daily Scrums

The goal of the daily scrum is to be concise and to follow each developer's progress in the sprint. In order to do so, each team member must answer these three questions:

What did you do yesterday?
What will you do today?
Are there any impediments in your way?

This way, everyone present at the daily scrum knows the daily achievements and failures of each team member. Problems can be discussed in order for everyone to be able to continue his work, ensuring your team mates' continuous skill development and fulfillment.

[10] *More detailed information about the Sprint Planning, Daily Scrums and Sprint Retrospectives can be found at: https://www.mountaingoatsoftware.com/ (links can be found in the document's appendix)*

The Sprint Retrospective

The sprint retrospective concludes the sprint. In this meeting, the team discusses the successes and the failures it has encountered during the sprint in order to solve problems and encourage good practices.

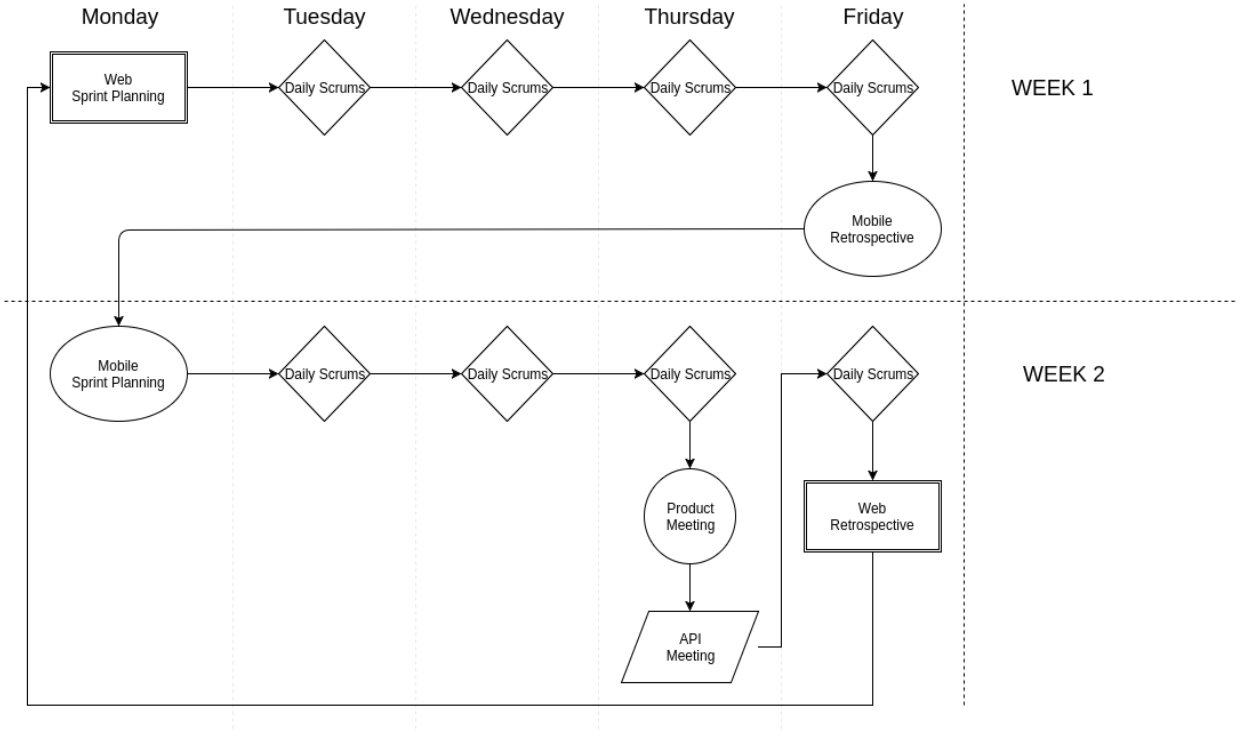
In addition to these meetings, you will have to organize and/or attend to the following: (which are not officially part of a classical sprint, but are incorporated at Yoopies in order to go with the sprint's flow and assure its development and preparation)

API meetings

These meetings take place every two weeks, to anticipate the api needs for the next sprint. For example, the mobile developers might need to implement a new API call[11] in order to communicate between a new feature and the server. During this meeting, the P.O. takes note of each team's requirements, and incorporates it in the sprint backlog (which we will talk about later on[12])

Product meetings

These meetings are similar to the API meetings, but are destined for the product team, such as business developers and data analysts. In these meetings, tasks are created upon requests or ideas concerning the product, such as user tracking.



Your typical sprint schedule

[11] *You can see and test API calls in Postman*
[12] *See section 5 of part 3*

D. Technical Architecture

Yoopies' technical architecture consist of multiple components.
The essential component is the web API. The API links the different databases (each locale and environment has its own database) with the different services: the apps, or the websites.

Apps
There is one app per service at yoopies: at the time of writing, you can find apps in production for Babysitting, and an app for Petsitting in development. All apps are available on iOS and Android (with some differences depending on the platform).

Websites
You can access Yoopies on different websites. The url architecture resembles this pattern:

environment-b2benvironment.yoopies.country.

For example, if you are in the preproduction environment for the econocom b2b partnership on the french country, you would have:

preprod-econocom.yoopies.fr

As you can see, Yoopies has a complex architecture and organisation. With different services, platforms, and strategies, there a vast amount of projects in development, with multiple teams, which might be independent or interconnected. As so, you might ask yourself:

With all these multiple teams and projects, how can you guarantee the quality and consistency of a product?

Part 3 The PO - YOU

The PO is responsible for the quality of his product. The term "quality" contains the timing of the product (when), the usefulness of the product (what) and the experience of the product (how).
The essential part of the work is to segment workloads into small independent features. Although this seems a simple and futile work, it requires a big set of skills. Typically, a task is segmented throughout a process that we can divide in 10 different steps, each of these steps requiring a special skillset.

1. Research a problem

As Yoopies is already launched, you will not start from zero. Instead, you will have to find new projects and improvements that fit your existing product.
It is thus important to be able to recognize weaknesses and problems present on your product. However, this might not be an easy task. To research and define a problem, there are a few methods and tools that you will have to use:

Testing

The first step of you have to take as a Product Owner is to use your own product. Put yourself in your user or customer's skin, and see if your product answers its' promises.

Benchmarking

An essential part of knowing your product and customers is knowing your concurrent products. By testing your product, you can analyse their strategic choices, and how they answer their promises. You can also identify bad choices, such as unusable features or blocking processes. Don't hesitate to explore product that are different to yours: a lot can be learned from others!

For example, one of the features I implemented on Yoopies' website during my internship was inspired from *ChooseYourBoss*[13], a recruitment platform. As they have a similar job search feature (a job "dashboard"), you can compare their functionalities with your website's ones.

Benchmarking requires some time and analysis skills. However, there are some product comparative services you can use to gather information, such as *SimilarWeb*[14].
Make sure you are up to date with the ever-growing tech community and services. Everyday, new apps and services see the day with incredible ideas and features that can be implemented in your product.

Experience

Use your own knowledge! Think about the previous projects you have worked on; what features have you implemented in the past that can be relevant to your current project?
You can also analyse your own use of similar products; which product has fulfilled one of your needs recently? How did it achieve this? Can you inspire yourself from its strategies?

[13] https://www.chooseyourboss.com/
[14] https://www.similarweb.com/

Data and user Analysis

A plethora of tools are at your disposal to analyse your product and its users. First of all, make sure that you know your product's data. In order to do so, most of Yoopies' functionalities are tracked. You can visualise this tracking through third party services, which we will explore later on, at the 9th step of this process.

2. Research a solution

Once you have identified a problem, you have to find a solution to counter it. You might want to use the tools and techniques found in Step 1. By this point, you should have a couple of solutions, or at least ideas. It is good practice to have a small list of solutions in case some of them prove to be irrelevant during the next steps.

Even though you might have a lot of experience and good benchmarking references, this does not always suffice. Indeed, not all problems have been resolved in the past, and it is up to you and your innovative capacities to find a good solution.

3. Find a doable technical approach to the solution

Once you have found your improvements or ideas, you must challenge them to some technical constraints. You might want to ask yourself some of these questions:
-Has your idea any link or similarity to another feature found in the product?
-Could you cut development time by copying some aspects of this other feature?
-Does the development team have the resources necessary to develop your idea?
-If you can grossly evaluate how much time and resources are needed to develop your idea, are these costs worth the value of your idea?

In order to answer these questions, you should learn to know both your product and your team. If they have time, you can also ask your colleagues their opinion on the subject.

4. Find a doable design approach to the solution

If you have a plausible technical approach to the solution, you can address the design team to make a graphical mock-up of your feature. A graphical mock-up is essential, as it is needed for the development team to integrate your feature, and is useful to present your ideas to the rest of the team.

5. Timing

Once you have both a technical and design approach to your idea, you might have a clearer opinion of the value of your idea.
You now have to ask yourself: "How much of a priority is this idea compared to the others that are planned ?" Indeed, there are a lot of features and ideas developed simultaneously in both your web & app products. You should assess the importance of your task, coupled to its complexity and resources it might need.

For example, if there are a lot of big features planned for the next sprint or two to come, and your idea needs a lot of resources to be developed, you might want to assign your task to “The Backlog”.

-The Backlog is a list of feasible features. As a P.O., it is your responsibility to feed the backlog of features and demands of the other teams. These features are sorted by priority, in order to link them to future sprints, when they are selected for development.

On the contrary, if your idea needs a small amount of resources to be developed, and its value is high, or needs to be delivered quickly, you can assign it to the next sprint to come.

6. Organise and distribute

At this point, you have all you need to launch the development of your idea. In order to do so, you will need to create “tasks”. A task represents an objective, a feature, that is to be realized by the developers. Depending on the team that will realize the task, tasks are defined in various ways.

To create a task, you will need to use Jira, an online agile project management tool, which gives you the ability to create and follow your tasks. As task creation evolves constantly, I will only present to you the most important aspects of their creation:

In the web team, tasks are defined as “User Stories”. The user story describes the type of user, what they want and why. A user story helps to create a simplified description of a requirement. It is formatted like this: “As *[a user]*, I would like to *[do an action]* in order to *[reach a finality]*”.

An example task would be:

-On the online store, create a button which adds an article to the user’s basket.

Your user story would be : “As *a shopper*, I would like to *add an article in my basket* in order to *purchase multiple articles*”.

As you can see, the task, formulated this way, does not represent any technical approach to how it should be developed. Indeed, as a Product Owner, you should be concerned only by the product aspect of a task: all of the technical details are the responsibility of the developers.

However, if you have any information that you deem relevant to the developers, you can add it in the story’s description. You could, for example, add a link to the mock-ups the designers made for you, or links to code/urls where a similar functionality has been developed, or any special recommendations or details that do not fit in the user story.

You might find that on your bigger or more complex ideas, multiple user stories are needed. Indeed, a good user story only concern one functionality. In our previous example, you can see how the user story only concerns the button, and not the whole process of checking out an article.

Once your task is ready, you will have to present it in the sprint planning. You can always edit your task during the sprint planning if some of its information is incorrect or missing. You

will evaluate the story points of your task together with the participants of the sprint planning, “et voila!”, your idea is ready for development.

7. Follow development

During the course of the sprint, your tasks will be assigned by a developer to himself. From there on, a task will navigate through a set of states, which you can see on your JIRA dashboard.
The task statuses are as follows:

- To do
- Doing
- To verify in pre-production // developed in next part
- To verify in production
- Done

A task’s progress is not linear. Indeed, it can skip or come back to states during and after development. For example, as you report bugs or minor changes on a task which is “to verify”, you might observe that you will need several iterations of development & testing processes to arrive at a finality.

Although you can track a specific task through your dashboard, you might have additional questions concerning the progress of your task. In this case, you can address your questions during the daily scrums, in order to disturb your developers as less as possible during development. It is indeed a good practice to attend the daily scrums of your team(s), as problems linked to development are pointed out during these meetings.

8. Validate the work

One of the crucial part of your role as a product owner is to verify developed tasks. Even though it might seem trivial, testing can be quite a puzzle.

You must keep in mind that your idea, on paper, or as a graphical mock-up, is rarely the same as it is developed. Indeed, time and technological constraints may incur, and the final result of your idea may differ of what you expected. You might, for example, discover a “bug”, or another functionality which comes unexpectedly with your task/idea.

In order to test a task correctly, you can approach the problem in 3 way:

The dummy test

Test your feature erratically. Try to trigger unexpected reactions. Use your feature in contrary ways than you would expect someone to use it (for example, use a negative value somewhere you expect positive ones)

The user test

Test your feature as a user would. Make sure you can easily perform a couple of ideal scenarios, and, most importantly, if you can complete your user story.

The product test

Test your feature from a product perspective; is your user guided correctly through the feature? Can the user's work flow be improved? Does your feature lead to other, logical features?

Don't forget to test on different devices and/or platforms. The *Chrome development tools* can help you simulate different screen resolutions, and provide powerful debugging tools. Some devices are also at your disposition to test the apps or the website on mobile devices.

Ideally, try and make other people test your feature. If it has a big impact on the product or the users, you can ask your co-workers what they think about it and how it could eventually be improved.

9. Analyse the customer reaction

Once your idea has been developed and thoroughly tested, it can be sent to production. To limit any failure of the integration of your idea, a good idea is to release your feature progressively.

A good tool for this is "A/B testing". This process allows you to release your feature to a specific segment of your users, and to compare statistically the success of your feature. Through A/B testing, you will typically submit two versions ("*variations")* of a feature, usually your new idea, and the current version of the feature. By setting up conversion funnels* or measurable objectives (KPIs*), you can measure the effectiveness of your new feature.

To get back to our example, we can imagine that before the "add to cart" button, a user would have to purchase his articles individually. Our A/B test would consist of having X number of users, with 50% of them having access to the new "add to cart" button, and the remaining 50% using the classical individual purchase.

As defined in our user story, the "add to cart" button's finality is to generate *multiple purchases* from single users. Our KPIs would resemble something like:

-The number of users buying multiple articles in one session
-The number of articles bought by sessions
-The number of articles added to a cart in a session

Data Analysis

With data visualization services such as *Amplitude or Tableau, y*ou can compare your variations' KPIs to decide whether the integration of your feature is a success or not. These services are extremely handy as understanding raw data can be complicated.

User Analysis

If your data proves you wrong, or your feature doesn't fulfill its goals, you can view *user recordings* through third party software such as *Hotjar* or *Uxcam*. These tools allow you to view directly what the user is doing, offering you a first hand view of how your feature is used.

You might also want to check any feedback or complaints from the users. You can follow the community managers' software such as *Salesforce* or *Intercom* to survey users, through their messages. For example, you might find that the day after releasing your feature, a lot of users complain about not being able to access it, or are having trouble completing some steps in a specific workflow.

10. Reiterate

With theses User and Data analysis tools, you can assess the performance of your feature. Have you reached your goal(s) without any negative impact on the rest of the product? If so, congratulations! You can reiterate through this process with another feature. Make sure to test your feature from time to time (with product, data, and user analysis) if your feature has a big impact or objectives, as its usage might vary over time or when new features are integrated in your product.

If you have not reached your goals, you can either:

- Abandon and/or recycle

In the case the resources necessary to fix/upgrade the problems linked to your feature are too high, you might want to consider abandoning the project, or recycling it into another feature.

- Fix and/or upgrade

On the contrary, if you have the resources at your disposal, you can reiterate the process we have described, by creating other tickets that would fix or upgrade your feature in order to reach its goal.

Conclusion

You are now ready to be a product owner! If at first these 10 steps might seem simple, keep in mind that this process has to be applied to an incalculable number of features, on different projects, with multiple teams.

If you wish to go into the subjects I approached in this document in further detail, you can find links in the appendix of this document.

If you have any questions or remarks concerning this document, you can contact me at: jurgen.palsma@gmail.com

I wish you the best in your internship.

Application Letter & Appendix

Jurgen Palsma
24 Rue Pasteur,
94270 Le Kremlin-Bicêtre
France

Olivier Wojdyniak
26 Rue Richer,
75009 Paris
France

Dear Olivier,

Since the departure of Armael, our Product Owner for the mobile development projects, we have seen multiple problems arise from these teams. In the absence of a Product Owner, we have had to do with what we have onboard, which is why Thibaut - our lead iOS developer - and Bryan - our lead android developer - have seen their development time cut in more than half because of all the product related tasks necessary for their teams.

To parry this problem, I undertook some of Armael's responsibilities, and have acted, as much as my timetable permitted it, as a temporary Product Owner for the iOS team. I have worked in collaboration with Thibaut in order to supervise each iOS release, and have accompanied the team in the realisation of the "Yoopies - Petsitting" app we plan to launch during this summer.

As my internship as a product Owner comes to an end, I would like to thank you for the opportunities you have given me during the past few months. I have learned a considerable amount, throughout the responsibilities and projects you entrusted to me, and I hope I have fulfilled them as thoroughly as they have fulfilled me. In consequence, I would like to apply to the position of mobile Product owner at Yoopies.

During my internship, I have become acquainted with Yoopies' projects, concurrency, and ambitions. As a Product Owner at Yoopies, I have learnt and contributed to the development processes, as well for the web departments as for the mobile ones. Considering the success of the projects you have trusted me with (the web's dashboard and the web & mobile notification abstraction project, to name a few), I believe that you will agree with me that I have integrated Yoopies successfully.

As you know, I have both a technical and product oriented background, and have worked on some scholar and personal mobile projects (some of them being actually used by android users). As a result, I could combine the product skills I have acquired during my internship and other projects and my technical skills in order to propose and execute relevant features and improvements in Yoopies' mobile applications.

In addition, I will obtain a master's degree in the course of the next year at the renowned University of Kent, as part of my year abroad in my Epitech cursus. I will focus my master on Data Analysis, and Data Science, which is, as you know, an great asset in the product world: I will thus be able to support my product and technical experience with data, and will be able to assert and prove the effectiveness of each improvement I will implement in Yoopies' future projects.

In conclusion, combining my technical, product, and data knowledge and experience coupled with my undertaking of the product tasks and responsibilities of the iOS team and projects, I believe that I could be a potential asset to your team, in the continuity of my work at Yoopies. Considering Yoopies' exponential growth and ambition, I would enthusiastically like to contribute to its' success, on my return from my year abroad.

Sincerely,
Jurgen PALSMA

Appendix

In this section, you can find complimentary readings and links to go into depth on the subjects approached throughout this document.

More about scrum:

- Sprint plannings
- Daily Scrums
- Sprint retrospectives
- A short video resuming the PO's responsibilities
- An "official" guide to Scrum
- A short article about User Stories

Tools:

- Jira documentation
- Data visualisation - Amplitude and Tableau, combine with Google tag manager
- User recordings - Hotjar (web only) and UxCam (apps only)
- Customer management tools - Intercom , Mailchimp
- Testing tools - Chrome development console

For additional content or questions, you can contact me via Linkedin or mail